N89-22941

IMPROVED PERFORMANCE IN NASTRAN®

by

Gordon C. Chan
UNISYS Corporation
Huntsville, Ala˘ama

SUMMARY

PART I, SPEED IMPROVEMENTS

Three areas of improvement in COSMIC/NASTRAN, 1989 release, were incorporated recently that make the analysis program run faster on large problems, particularly on the VAX computer.

The first improvement on Bulk Data input cards was presented in great detail in the last NASTRAN Users' Colloquium. This paper completes the previous presentation by compiling actual log files and actual timings on a few test samples that were run on IBM, CDC, VAX, and CRAY computers. The tabulated result shows a speed improvement in COSMIC/NASTRAN Link 1 from 3 to 4 times faster on the CDC computer to 50 to 60 times faster on the VAX. IBM and CRAY show 20 to 30 times faster. The speed improvement is proportional to the problem size and number of continuation cards.

The second improvement involves vectorizing certain operations in BANDIT, that makes BANDIT run twice as fast in some large problems using structural elements with many node points (such as 8-, 20-, and 32-node elements). BANDIT is a built-in NASTRAN processor that optimizes the structural matrix bandwidth.

The third improvement is on the VAX computer only. The VAX matrix packing routine BLDPK was modified so that it is now packing a column of a matrix 3 to 9 times faster. The companion unpack routine INTPK is also running 3 to 9 times faster than before. The denser and bigger the matrix, the greater is the speed improvement. This improvement makes a host of routines and modules that involve matrix operation, such as MPYAD, DECOMP/SDCOMP, EMG, REIG/CEIG, PARTN/MERGE, ELIM, SSG2B/C, FACTOR, TRNSP, SDR2, etc., run significantly faster. This improvement also saves disc space for dense matrices; up to two-thirds of the hardware storage space could be saved.

PART II, IMPROVEMENTS, NEW ENHANCEMENTS, AND NEW VERSION

1. A new UNIX version, converted from 1988 COSMIC NASTRAN, was tested successfully on a Silicon Graphics computer using the UNIX V Operating System, with Berkeley 4.3 Extensions. This version with small changes was also known to run successfully on the SUN computer and Apollo workstations.

2. The Utility Modules INPUTT5 and OUTPUT5, developed in 1988, were expanded to handle table data, as well as matrices. INPUTT5 and OUTPUT5 are

general input/output modules that read and write FORTRAN files with or without format.

3. More user informative messages are echoed from PARAMR, PARAMD, and SCALAR modules to ensure proper data values and data types being handled. The ADD module was expanded to handle both single and double precision scale factors.

4. Two new Utility Modules, GINOFILE and DATABASE, were written for the 1989 release. They can be requested via DMAP Alter.

(a) GINOFILE captures any scratch file of the preceding DMAP module and makes it a legitimate GINO file.

(b) DATABASE copies (that is FORTRAN written) the grid point data, element connectivity data, displacement, velocity, and acceleration vectors, loads, grid point forces, eigenvectors, element stresses, and element forces to a user tape (one of the UT1, UT2, INPT, INP1,...,INP9 tape), formatted or unformatted. The grid point data is in system basic coordinates. The displacement vectors can be in system basic or system global coordinates.

5. Seven new rigid elements are added to COSMIC NASTRAN. They are: CRROD, CRBAR, CRTRPLT, CRBE1, CRBE2, CRBE3, and CRSPLINE.

PART I, SPEED IMPROVEMENTS, 1989 NASTRAN

A. Link 1 Improvement

In the sixteenth NASTRAN Users' Colloquium, the author presented a paper entitled "On Bulk Data Cards Processing", in which a new method of processing the NASTRAN raw input data was discussed in great detail. The only thing that was missing in that paper was some actual numbers showing the timing improvement of the 1988 NASTRAN version. Since then, further NASTRAN software improvement, not directly involving the bulk data cards processing, also makes Link 1 run faster. Appendix A tabulates a series of test runs showing the actual timings on several computers, using NASTRAN 87, 88, and 88.5 (which is now 89) versions. Appendix A begins with a cantilever model used throughout all the tests. The model includes 10,000 grid points and 9999 QUAD2 elements, with the grid points and elements intentionally not in sorted order. The actual timings of the XCSA, IFP1, XSORT, and IFP modules were tabulated directly from the NASTRAN log files of various NASTRAN versions on different computers, the NASTRAN new sorter, XSORT2, and the old sorter, XSORT. The conclusion from these tests indicates generally that the new Link 1 is 20 to 30 times faster, with only one exception: CDC tests showed only 2 to 3 times faster. This CDC slowness is explained in the next two paragraphs. The Link 1 improvement on the VAX computer was actually more impressive; it showed 30 to 70 times faster. The new speed improvement of Link 1, however, must be discounted somewhat due to certain source code standardization implemented in the 1987 NASTRAN version. The 1987 version was 3 to 10 times slower then the previous versions.

The CDC computer (a non-virtual memory model) is intended for number crunching. It is, however, extremely slow in handling characters. The new XSORT2 routine, where the bulk data are handled and sorted, and its supporting subroutines are heavily character oriented (that makes this group of routines completely machine dependent). In the 88.5 version, additional modifications to this group of routines were made to avoid CDC's weakness. The actual amount of changes was not too extensive, and therefore will not be a big burden to the other non-CDC computers. The following tests illustrate the CDC deficiency.

A FORTRAN (FTN5) source code 'A=B' is 5 times slower if A and B are characters than if A and B are integers. Similarly, 'If (A .EQ. B) C=D' is 15 times slower if A, B, C, and D are characters than if they are integers. These time-tests were made on the Langley Research Center Y computer (CDC CYBER 185).

B. Vectorizing BANDIT Operation

The grid point connectivity increases exponentially when elements of many node points (8-, 20-, and 32-node elements) are used in a finite element model. Consequently, BANDIT, the NASTRAN built-in bandwidth processor, would require substantial computer CPU time for optimizing the connectivity of the structural model. By modifying the source code using the array vectorizing concept, the BANDIT timing on an actual huge problem was reduced by half.

C. Pack and Unpack on the VAX Computer

All matrix data in NASTRAN are packed; that is, all elements of zero values are squeezed out, when the matrix data are written out to a storage disc. Conversely, the matrix is unpacked into memory space when it is brought back from the disc; that is, all missing zero elements are put back into their correct positions in the matrix. For efficiency, the packing and unpacking routines in NASTRAN are written in the machine dependent assembly languages for IBM, CDC, and UNIVAC computers. However, FORTRAN is used in the VAX. The FORTRAN source code of the VAX, which is much easier to understand, has been studied thoroughly; and improvements were incorporated in key areas. The final result shows great improvement in speed and reduction of disc space, particularly in large and dense matrices. These improvements in matrix packing and unpacking make a host of other modules, such as EMG, MPYAD, SDR2, SMP1, PARTN/MERGE, ELIM, SSG2B/C, FACTOR, TRNSP, SDR2, DECOMP/SDCOMP, EIGR/EIGC, etc. run significantly faster. As indicated by the test runs in Appendix A, all VAX runs on the 88.5 version were at least 30 to 40 percent faster than the corresponding runs on the 88 version.

PART II, IMPROVEMENTS, NEW ENHANCEMENTS, AND NEW VERSION, 1989 NASTRAN

A. UNIX Version of COSMIC NASTRAN

The present trend in computer applications is towards the UNIX operating system. The 1988 COSMIC NASTRAN has been converted to run on a Silicon Graphics computer with UNIX V OS, Berkeley 4.3 Extensions. All the machine independent source code was converted without any major problems. (Only the symbol '4H/*(something)(something)' needed to be changed.) The VAX machine dependent source code (all written in FORTRAN) was used in the UNIX version. Approximately 30 percent of this group of routines required modification and special attention. The major problem encountered here was the usage of the open core. This problem appeared to be very much machine dependent even in the supposedly machine-independent UNIX operation environment. The UNIX versions were also tried on the SUN computer and Apollo workstations successfully.

B. INPUTT5 and OUTPUT5 Improved Capabilities

Utility modules INPUTT5 and OUTPUT5 were incorporated into COSMIC NASTRAN since 1987. OUTPUT5 creates user written FORTRAN files, formatted or unformatted, and INPUTT5 reads the FORTRAN files, also formatted or unformatted. The 1987 versions of INPUTT5 and OUTPUT5 actually handle only GINO (NASTRAN General Input/Output) matrix data blocks. Since matrix data are either real or complex, in single or double precision forms, it is relatively easy to read the GINO matrix data blocks and send them out (OUTPUT5) under FORTRAN control, formatted or unformatted. Similarly, it is quite easy to read (INPUTT5) from a FORTRAN file, formatted or unformatted, and re-create the GINO matrix data block. The 1987 INPUTT5 and OUTPUT5 modules do not handle GINO table data blocks, because a table array normally contains mixed types of data, integers, reals, and BCD data, in endless combinations. To write an array of mixed type data formatted is not an easy task.

The INPUTT5 and OUTPUT5 modules in the 1989 release are expanded to include the GINO table data blocks, as well as the matrix data blocks. Again, the data transfer between the GINO file and FORTRAN file can be formatted and unformatted. The formatted file, in particular, can be used across different computer manufacturers' processors. Appendices B and C provide detailed descriptions of the INPUTT5 and OUTPUT5 modules.

C. Improved and Expanded Data Handling in PARAMR, PARAMD, PARAML, SCALAR, and ADD Modules

The original PARAMR, PARAMD, and SCALAR modules are difficult to use. In most cases, the user has no idea whether the input data he specified is used correctly, and the output from these modules is correctly computed. Unless these modules are further checked by PRTPARM, the user is completely in the dark. Usually, a user has no control over these modules even when errors are found in the input or output parameters.

The PARAMR, PARAMD, and SCALAR modules in the 1989 version provide much more user information. The actual input and output parameters are echoed out. For example, if data are to be abstracted from matrix data blocks by PARAMR/D, the precise row and column positions of the data element are printed, and the user is also informed that the matrix type is real, complex, single precision, or double precision. Similar useful information is echoed out for the table data blocks. In PARAMR and PARAMD, the user is also given the ability to stop or to continue a NASTRAN job at the end of these modules, in the cases of error, or no error found. Appendices D and E provide detailed descriptions of the PARAMD and SCALAR functional modules. PARAML was expanded in the 1988 version. Its manual update is reprinted in Appendix F.

The ADD module, which adds two matrices, [X] = a[A] + b[B], has been expanded. The scale factors a and b can now be in single precision or in double precision. The manual pages for ADD are updated and presented in Appendix G.

D. Two New Utility Modules - GINOFILE and DATABASE

D.1 GINOFILE Module - to Capture A Scratch File of Preceding DMAP Module

NASTRAN's General Input and Output (GINO) utility processor provides three kinds of files for each NASTRAN DMAP operation - input files, output files, and scratch files. Each DMAP module specifies its input files and output files explicitly, and they are saved in the GINO system until an explicit PURGE command is given. The scratch files, however, are not saved, and are left in the computer system unprotected at the end of a DMAP module operation. At this point they are ready to be over-written by anything in the next module. There would be a tremendous amount of work in DMAP (or Rigid Format) programming, and FORTRAN source code changes, if one would like to save one or more of these scratch files by the regular GINO and DMAP rules. After consulting the NASTRAN Theoretical manual and Programmer manual, and possibly studying the NASTRAN FORTRAN source code carefully, some advanced NASTRAN users would like to salvage certain scratch file(s) in a particular DMAP module, so that they can do extraordinary work with those files and DMAP Alters. A new DMAP functional module, GINOFILE, is provided in the 1989 version to capture one scratch file of the preceding DMAP module, and give it a legitimate GINO data block name, and proper GINO data block formation. For example, the scratch file captured may not have a header record and GINO name; the new output file from GINOFILE will have a GINO name and proper header record. This new GINOFILE module will work properly if and only if the scratch file of the preceding module still exists in the computer system, and only the last data written to that file is available if looping is involved. Appendix H describes in detail the usage of GINOFILE module and its limitations.

D.2 DATABASE Module - to Copy Grid Point Data, Element Connectivity Data, and Displacement Vectors (Velocities, Accelerations, Loads, Grid Point Forces, Eigenvectors, Element Stresses, or Element Forces) to a User Tape

Many users have expressed the need to transfer NASTRAN basic data (grids, elements, and displacements, etc.) to an external FORTRAN tape, or file, so that they can use the data for other purposes. Typically, they would like to link up the NASTRAN data to another commercially available program for plotting interactively, or to user written software for data manipulation. Very commonly, the users would like the grid point data and the displacement data in the basic rectangular coordinate system, and all the grid point ID's in their external numbers. To do just that, the users must copy grid point data in the GEOM1 (Geometry 1), or BGPDT (Basic Grid Point Definition Table) file, the Coordinate System Transformation Matrices in CSTM file, element data in the GEOM2 (Geometry 2), or EST (Element Summary Table) file, element type data in the FORTRAN source code GPTABD BLOCK DATA, the EQuivalence EXternal grid point vs. INternal grid point number tables in EQEXIN, and one or two displacement OFP files (Output File Processor, a group of 20 to 25 files). Since all the NASTRAN files are GINO (NASTRAN General Input and Output file processor) written, the users most likely would use the OUTPUT2 module (OUTPUT5 is now available in 88 and 89 releases) for the data transfer. Only then, the users would have enough data to convert the grid points from NASTRAN global coordinates to the basic rectangular coordinate system, and from NASTRAN internal grid point numbering system to the external numbers. Normally, The users have to provide the conversion program either from a commercial source or from their own written programs. In the latter case, the users must be familiar with OUTPUT2 operations, and the contents of various files and their exact data arrangements; they also must have a good knowledge of coordinate transformation, and the User's and the Programmer's Manuals. A seemingly easy job could turn out to be a mammoth task. It is to this end that a new utility module, DATABASE, was written for the 1989 release.

The new DATABASE module copies the grid point data, the element connectivity data, the displacement vectors, and other data out to a user tape (UT1 or UT2 for CDC computer, INPT, INP1, INP2,... INP9 for other computers), formatted or unformatted. All grid points are in basic rectangular coordinates, and the displacement vectors (and others) are in basic rectangular coordinates or NASTRAN global coordinates. All grid point ID's are in their external numbering system. The displacement vectors (and other data) can be real or complex, 'SORT1' or 'SORT2' formats, and single case with one output record, or sub-cases with multiple records. The unformatted tape from the DATABASE module is more efficient. The formatted tape can be printed out for verification, or edited by the system editor. The formatted tape can also be generated on one computer, and used on another computer of a different manufacturer. Appendix I, the user manual update pages for the DATABASE module, shows in detail the records being generated on the output tape. Appendix J presents a NASTRAN example run using this new DATABASE module. The formatted INP1 file of this example run is also listed. Appendix J also includes a FORTRAN program which was used to check out the unformatted tape during the development stage. This FORTRAN program is very useful as a guide to read a typical unformatted FORTRAN tape.

E. New COSMIC Rigid Elements

In addition to the four rigid elements, CRIGD1, CRIGD2, CRIGD3, and CRIGDR, in NASTRAN, seven new rigid elements are now available to COSMIC NASTRAN users.

CRROD - a rigid pin-ended rod element (similar to CRIGDR)
CRBAR - a rigid bar element
CRTRPLT - a rigid triangular plate element
CRBE1 - a general rigid body connected to an arbitrary number of grid points (similar to CRIGD3)
CRBE2 - a rigid body with independent d.o.f. at a single grid point, and with dependent d.o.f. at an arbitrary number of grid points (similar to CRIGD2)
CRBE3 - a rigid body that defines the motion at "reference" grid

points as the weighted average of the motions at a set of other grid points

CRSPLINE- a rigid element of multi-point constraints for interpolation of displacements at specified grid points

The implementation of this group of new elements is as follows.

The input data of CRROD, CRBE2, and CRBE1 are mapped into CRIGDR, CRIGD2, and CRIGD3 data formats respectively, and thus they are treated as a CRIGDR, a CRIGD2, or a CRIGD3 element.

The input data of CRBAR and CRTRPLT are mapped into the rigid general element CRIGD3 data format, and they are each treated as a CRIGD3 element.

The CRBE3 and CRSPLINE have no COSMIC old rigid element equivalence. A special subroutine was written to handle these two elements.

Appendix K provides the Users' Manual update pages for this new group of rigid elements.

F. Other Improvements

Other improvements in the 1989 release include:

(a) A new 'ECHO = NONO' option that provides absolutely no input card (and restart) echo.

(b) DIAG 38 will list each element being processed by the EMG module. Thus it may pinpoint which element that might have an input error.

(c) Complex stresses and forces for QUAD4 elements.

(d) Inclusion of QUAD4 in random analysis.

APPENDIX A

NASTRAN LINK1 TIME TESTS

(A) NASTRAN TIMING TEST INPUT DECK

```
NASTRAN BULKDATA=-1, TITLEOPT=-1
ID      TEN THOUSAND GRID POINTS TEST PROBLEM, CM=250,000 (CDC)
$                                              50K  WORDS (VAX)
$                                              7.5M BYTES (IBM)
$                                              100K WORDS (UNIVAC)
$                                              OPEN CORE? (CRAY)
APP     DISP
SOL     1,0
$DIAG   42    $ TO USE OLD XSORT MODULE
TIME    20
CEND
TITLE    = DATA INTENTIONALLY GENERATED NOT IN SORTED ORDER
SUBTITLE = NO CONTINUATION CARDS IN DECK
DISP     = ALL
ECHO     = NONE   $ OR  ECHO = SORT
SPC      = 135
LOAD     = 246
BEGIN BULK
$
$              TEST MODEL - A LONG CANTILEVER BEAM
$    Y
$    |
$    |
$  5001  5002  5003  5004  5005  5006  5007 .. 9998  9999  10000 = GRID PTS
$    +-----+-----+-----+-----+-----+-----+--    --+-----+-----+
$    |     |     |     |     |     |     |        | (49 | (49 |  (QUAD2 ELEM.ID
$    | (1) | (2) | (3) | (4) | (5) | (6) |  ...   | 98) | 99) |   IN BRACKETS)
$    +-----+-----+-----+-----+-----+-----+--    --+-----+-----+ -------------- X
$    1     2     3     4     5     6     7       4998  4999   5000
$
GRID, 1,, 0.0,    0.0,  0.0
=(2499), *(2),, *(2.),==
GRID, 5000,, 4999.0,   0.0,  0.0
=,*(-2),,*(-2.),==
=(2498)
CQUAD2, 1, 10,  1,   2, 5002,  5001
=(2498),*(2),=,*(2),///
GRID, 5001,, 0.0,  1.0,  0.0
=,*(2),,*(2.),==
=(2499)
GRID, 10000,, 4999.0  1.0  0.0
=,*(-2),,*(-2.),==
=(2498)
PQUAD2, 10   20      0.02
MAT1,   20   30.+6,, 0.33
CQUAD2, 4998  10  4999  5000   10000  9999
=,*(-2),=, *(-2),*(-2),*(-2),*(-2)
=(2497)
SPC1, 135  123456  1  5001
FORCE,246, 5000,, 200., 0. 0. -1.
=,=,      10000,==
ENDDATA
```

(B) TO GENERATE A FIXED-FIELD INPUT DECK FOR THE MODEL IN (A) (NOT AVAILABLE ON IBM MACHINE)

Assume file ABC.DAT contains the above model.
Run LINKFF (an independent link included in NASTRAN delivery tape, NAST01).
Answer a new file name when asked, such as DEF.DAT.
Enter 'READFILE ABC.DAT'.
A fixed-field deck will be generated and saved in DEF.DAT (approx. 15020 cards).

(C) SUMMARY OF NASTRAN TIMINGS USING ABOVE CANTILEVER MODEL IN (A)

VAX TIMING, COSMIC VAX/780 VMS 4.7 -
(Runs were made before November 1988)

```
VAX NASTRAN 87 RELEASE TIMING:
17:21:49      3.0 ELAPSED SEC       1.0 CPU SEC   TTIO     | TIMING COMPUTATION
17:22:32     46.0 ELAPSED SEC      26.0 CPU SEC   TTLP     |
17:22:32     46.0 ELAPSED SEC      26.0 CPU SEC   XCSA     | 32.0-  26.= 6.0 SEC
17:22:42     56.0 ELAPSED SEC      32.0 CPU SEC   IFP1     | 34.0-  32.= 2.0 SEC
17:22:46     60.0 ELAPSED SEC      34.0 CPU SEC   XSOR     | 7495.-  34.=7461 SEC
19:36:17   8071.0 ELAPSED SEC    7495.0 CPU SEC   IFP  BEGN
20:04:48   9782.0 ELAPSED SEC    9190.0 CPU SEC   IFP  END| 9190.-7495.=1695 SEC
20:04:48   9782.0 ELAPSED SEC    9190.0 CPU SEC   XGPI     |

VAX NASTRAN 88 RELEASE (WITH DIAG 42 ON, USING OLD XSORT) TIMING:
14:28:00      2.0 ELAPSED SEC       1.0 CPU SEC   TTIO     |
14:28:00      2.0 ELAPSED SEC       1.0 CPU SEC   XCSA     |  7.0-   1.0= 6.0 SEC
14:28:10     12.0 ELAPSED SEC       7.0 CPU SEC   IFP1     |  9.0-   7.0= 2.0 SEC
14:28:14     16.0 ELAPSED SEC       9.0 CPU SEC   XSOR     | 8334.-  9.0=8325 SEC
17:00:16   9138.0 ELAPSED SEC    8334.0 CPU SEC   IFP  BEGN
17:32:40  11082.0 ELAPSED SEC    9893.0 CPU SEC   IFP  END| 9893.-8334.=1559 SEC
17:32:40  11082.0 ELAPSED SEC    9893.0 CPU SEC   XGP      |

VAX NASTRAN 88 RELEASE TIMING:
14:13:05      2.0 ELAPSED SEC       0.0 CPU SEC   TTIO     |
14:13:05      2.0 ELAPSED SEC       0.0 CPU SEC   XCSA     |   6.0- 0.0=   6.0 SEC
14:13:14     11.0 ELAPSED SEC       6.0 CPU SEC   IFP1     |   8.0- 6.0=   2.0 SEC
14:13:18     15.0 ELAPSED SEC       8.0 CPU SEC   XSOR     | 160.0- 8.0=154.0 SEC
14:16:50    227.0 ELAPSED SEC     160.0 CPU SEC   IFP  BEGN
14:20:09    426.0 ELAPSED SEC     337.0 CPU SEC   IFP  END| 337.0-160.=177.0 SEC
14:20:09    426.0 ELAPSED SEC     337.0 CPU SEC   XGPI     |

VAX NASTRAN 88.5 NEW VERSION TIMING:
13:43:36     40.0 ELAPSED SEC      26.0 CPU SEC   TTLP     |
13:43:37     41.0 ELAPSED SEC      26.0 CPU SEC   XCSA     |  32.0-26.0=  4.0 SEC
13:43:46     50.0 ELAPSED SEC      32.0 CPU SEC   IFP1     |  33.0-32.0=  1.0 SEC
13:43:49     53.0 ELAPSED SEC      33.0 CPU SEC   XSOR     | 149.0-33.0=116.0 SEC
13:46:52    236.0 ELAPSED SEC     149.0 CPU SEC   IFP  BEGN
13:48:42    346.0 ELAPSED SEC     237.0 CPU SEC   IFP  END| 237.0-149.= 88.0 SEC
13:48:42    346.0 ELAPSED SEC     237.0 CPU SEC   XGPI     |
```

VAX/780 TIMING SUMMARY TABLE:

VERSION	COSMIC 87	COSMIC 88	COSMIC 88	COSMIC 88.5
		OLD XSORT	NEW XSORT2	AVAILABLE IN 89
MODULE	CPU	CPU	CPU	CPU
XCSA	6.0	6.0	6.0	4.0
IFP1	2.0	2.0	2.0	1.0
XSORT	7461.0	8325.0	154.0	116.0
IFP	1695.0	1559.0	177.0	88.0

VAX TIMING, COSMIC MICRO-VAX 3600 VMS 5.0-2 -
(Runs were made after December 1988, 100K HICORE)

VAX NASTRAN 88.5 RELEASE (WITH DIAG 42 ON, USING OLD XSORT) TIMING:

```
14:13:42      1.0 ELAPSED SEC      0.0 CPU SEC   TTIO     | TIMING COMPUTATION
14:13:42      1.0 ELAPSED SEC      0.0 CPU SEC   XCSA     |  2.0- 0.0=  2.0 SEC
14:13:46      5.0 ELAPSED SEC      2.0 CPU SEC   IFP1     |  2.0- 2.0=  0.0 SEC
14:13:48      7.0 ELAPSED SEC      2.0 CPU SEC   XSOR     | 1273.- 2.0=1271. SEC
14:41:49   1688.0 ELAPSED SEC   1273.0 CPU SEC   IFP  BEGN
14:43:14   1773.0 ELAPSED SEC   1321.0 CPU SEC   IFP  END| 1321.-1273.=48.0 SEC
14:43:14   1773.0 ELAPSED SEC   1321.0 CPU SEC   XGPI     |
```

VAX NASTRAN 88 RELEASE TIMING:

```
09:30:48      1.0 ELAPSED SEC      1.0 CPU SEC   TTIO     |
09:30:48      1.0 ELAPSED SEC      1.0 CPU SEC   XCSA     |  3.0- 1.0=  2.0 SEC
09:30:52      5.0 ELAPSED SEC      3.0 CPU SEC   IFP1     |  4.0- 3.0=  1.0 SEC
09:30:54      7.0 ELAPSED SEC      4.0 CPU SEC   XSOR     | 62.0- 4.0= 58.0 SEC
09:32:48    121.0 ELAPSED SEC     62.0 CPU SEC   IFP  BEGN
09:34:11    204.0 ELAPSED SEC    129.0 CPU SEC   IFP  END| 129.0-62.0= 67.0 SEC
09:34:11    204.0 ELAPSED SEC    129.0 CPU SEC   XGPI     |
```

VAX NASTRAN 88.5 NEW VERSION TIMING:

```
09:37:22      1.0 ELAPSED SEC      1.0 CPU SEC   TTIO     |
09:37:22      1.0 ELAPSED SEC      1.0 CPU SEC   XCSA     |  3.0- 1.0=  2.0 SEC
09:37:26      5.0 ELAPSED SEC      3.0 CPU SEC   IFP1     |  3.0- 3.0=  0.0 SEC
09:37:28      7.0 ELAPSED SEC      3.0 CPU SEC   XSOR     | 48.0- 3.0= 45.0 SEC
09:39:06    105.0 ELAPSED SEC     48.0 CPU SEC   IFP  BEGN
09:39:53    152.0 ELAPSED SEC     79.0 CPU SEC   IFP  END|  79.0-48.0= 31.0 SEC
09:39:53    152.0 ELAPSED SEC     79.0 CPU SEC   XGPI     |
```

MICRO-VAX 3600 TIMING SUMMARY TABLE:

VERSION	COSMIC 88	COSMIC 89	COSMIC 88	COSMIC 88.5
	OLD XOSRT	OLD XSORT	NEW XSORT2	AVAILABLE IN 89
MODULE	CPU	CPU	CPU	CPU
XCSA		2.0	2.0	2.0
IFP1	N/A	0.0	2.0	0.0
XSORT		1271.0	58.0	45.0
IFP		48.0	67.0	31.0

IBM TIMINGS, MSFC IBM 3084 -

IBM NASTRAN 87 RELEASE TIMING:

```
•      0.3 ELAPSED    0.088 CPU     TTIO        |     TIMING COMPUTATION
•      0.4 ELAPSED    0.088 CPU     XCSA        |    0.241-0.088=  0.153 SEC
•      1.4 ELAPSED    0.241 CPU     IFP1        |    0.330-0.241=  0.092 SEC
•      2.4 ELAPSED    0.330 CPU     XSOR        |  222.975-0.330=222.645 SEC
•    274.1 ELAPSED  222.975 CPU     IFP   BEGN  |
•    407.4 ELAPSED  319.521 CPU     IFP   END   | 319.521-222.975= 96.546 SEC
•    407.4 ELAPSED  319.522 CPU     XGP         |
```

IBM NASTRAN 88 RELEASE TIMING (WITH DIAG 42 ON, USING OLD XSORT):

```
•      0.9 ELAPSED    0.093 CPU     TTIO        |
•      0.9 ELAPSED    0.094 CPU     XCSA        |    0.227-0.094=  0.133 SEC
•      2.2 ELAPSED    0.227 CPU     IFP1        |    0.306-0.227=  0.079 SEC
*      3.2 ELAPSED    0.306 CPU     XSOR        |  222.801-0.306=222.495 SEC
•    252.6 ELAPSED  222.801 CPU     IFP   BEGN  |
•    339.5 ELAPSED  297.420 CPU     IFP   END   | 297.420-222.801= 74.619 SEC
*    339.5 ELAPSED  297.421 CPU     XGP         |
```

IBM NASTRAN 88 RELEASE TIMING (FIRST RUN):

```
•      0.5 ELAPSED    0.097 CPU     TTIO        |
*      0.5 ELAPSED    0.097 CPU     XCSA        |    0.215-0.097=  0.118 SEC
•      2.0 ELAPSED    0.215 CPU     IFP1        |    0.274-0.215=  0.059 SEC
•      3.6 ELAPSED    0.274 CPU     XSOR        |    8.766-0.274=  8.492 SEC
*     36.9 ELAPSED    8.766 CPU     IFP   BEGN  |
•     58.6 ELAPSED   16.813 CPU     IFP   END   |   16.813-8.766=  8.047 SEC
•     58.6 ELAPSED   16.813 CPU     XGP         |
```

IBM NASTRAN 88 RELEASE TIMING (SECOND RUN):

```
•      0.9 ELAPSED    0.100 CPU     TTIO        |
•      0.9 ELAPSED    0.100 CPU     XCSA        |    0.221-0.100=  0.121 SEC
*      2.3 ELAPSED    0.221 CPU     IFP1        |    0.282-0.221=  0.061 SEC
•      3.3 ELAPSED    0.282 CPU     XSOR        |    8.819-0.282=  8.537 SEC
*     31.9 ELAPSED    8.819 CPU     IFP   BEGN  |
•     51.9 ELAPSED   16.984 CPU     IFP   END   |   16.984-8.819=  8.165 SEC
*     51.9 ELAPSED   16.984 CPU     XGPI        |
```

IBM NASTRAN 88.5 RELEASE TIMING:

```
*      1.4 ELAPSED    0.118 CPU     TTIO        |
•     10.2 ELAPSED    0.540 CPU     TTLP        |
*     10.2 ELAPSED    0.580 CPU     XCSA          |     0.703-0.580=  0.123 SEC
*     12.9 ELAPSED    0.703 CPU     IFP1          |     0.742-0.703=  0.039 SEC
*     14.4 ELAPSED    0.742 CPU     XSOR          |     8.135-0.742=  7.393 SEC
*     75.0 ELAPSED    8.135 CPU     IFP   BEGN    |
*    139.0 ELAPSED   14.040 CPU     IFP   END     |    14.040-8.135=  5.905 SEC
*    139.0 ELAPSED   14.040 CPU     XGPI          |
```

IBM TIMING SUMMARY TABLE:

VERSION	COSMIC 87	COSMIC 88 OLD XSORT	COSMIC 88 NEW XSORT2	COSMIC 88.5 AVAILABLE IN 89
MODULE	CPU	CPU	CPU*	CPU
XCSA	0.153	0.133	0.120	0.123
IFP1	0.092	0.079	0.060	0.039
XSORT	222.645	222.495	8.515	7.393
IFP	96.546	74.619	8.106	5.905

* AVERAGE OF TWO RUNS.

UNIVAC TIMING TEST NOT AVAILABLE

CRAY TIMINGS, MODEL X-MP (COS), COURTESY OF RPK CORPORATION -
(Log files slightly edited)

CRAY NASTRAN 88 RELEASE TIMING (WITH DIAG 42 ON, USING OLD XSORT):

```
WALL CLOCK       TOTAL        INCREMENTAL      MODULE
   TIME       CPU SECONDS     CPU SECONDS      & STATUS
-------------------------------------------------------
 10:17:29         0.018           0.000          TTIO      |   TIMING COMPUTATION
 10:17:31         0.349           0.331          TTLP      |
 10:17:31         0.439           0.090          XCSA      |   0.618-   0.439=  0.179
 ID    TEN THOUSAND GRID POINTS TEST PROBLEM, OPEN CORE? (CRAY)
 10:17:34         0.618           0.179          IFP1      |   0.644-   0.618=  0.026
 TITLE = DATA INTENTIONALLY GENERATED NOT IN SORTED ORDER
 10:17:34         0.644           0.027          XSOR      | 108.788-   0.644=108.144
 10:22:24       108.788         108.144          IFP  BEGN
 10:23:43       141.799          33.010          IFP  END  | 141.799-108.788= 33.011
 10:23:43       141.799           0.000          XGPI      |
```

CRAY NASTRAN 88 RELEASE TIMING:

```
WALL CLOCK       TOTAL        INCREMENTAL      MODULE
   TIME       CPU SECONDS     CPU SECONDS      & STATUS
-------------------------------------------------------
 10:05:16         0.018           0.000          TTIO      |
 10:05:19         0.346           0.328          TTLP      |
10:05:19          0.435          0.090       XCSA     |  0.609-0.435=0.174 SEC
ID    TEN THOUSAND GRID POINTS TEST PROBLEM, OPEN CORE? (CRAY)
10:05:23          0.609          0.174       IFP1     |  0.626-0.609=0.017 SEC
TITLE = DATA INTENTIONALLY GENERATED NOT IN SORTED ORDER
10:05:23          0.626          0.017       XSOR     |  5.539-0.626=4.913 SEC
10:05:41          5.539          4.913       IFP  BEGN
10:05:57         10.575          5.035       IFP  END |  10.575-5.539=5.036 SEC
10:05:57         10.575          0.000       XGPI     |
```

CRAY TIMING SUMMARY TABLE:

VERSION	COSMIC 87	COSMIC 88	COSMIC 88	COSMIC 88.5
		OLD XSORT	NEW XSORT2	AVAILABLE IN 89
MODULE	CPU	CPU	CPU	CPU
XCSA		0.179	0.174	
IFP1	N/A	0.026	0.017	N/A
XSORT		108.144	4.913	
IFP		33.011	5.036	

CDC TIMINGS, CYBER 855 AT LRC, COURTESY OF JOE WALZ -

CDC NASTRAN 88 RELEASE TIMING WITH DIAG 42 ON (OLD XSORT):

```
WALL       ELAPSED     CPU
CLOCK      SECONDS   SECONDS  MODULE
===================================
18.50.41.     3.0      1.245  TTIO      |    TIMING COMPUTATION
18.50.57.    19.0      5.698  TTLP      |
18.50.58.    20.0      6.289  XCSA      |    10.532-  6.289=   4.243 SEC
18.51.05.    27.0     10.532  IFP1      |    11.876- 10.532=   1.344 SEC
18.51.07.    29.0     11.876  XSOR      |   650.388- 11.876= 638.512 SEC
19.16.03.  1525.0    650.388  IFP  BEG  |
20.27.41.  5823.0   3083.137  IFP  END  |  3083.137-650.388=2432.749 SEC
20.27.41.  5823.0   3083.141  XPGI      |
```

CDC NASTRAN 88 RELEASE TIMING (NEW XSORT2):

```
08.26.25.     3.0      1.278  TTIO      |
08.26.40.    18.0      5.895  TTLP      |
08.26.41.    19.0      6.501  XCSA      |     9.730-  6.501=   3.229 SEC
08.26.45.    23.0      9.730  IFP1      |    10.422-  9.730=   0.692 SEC
08.26.47.    25.0     10.422  XSOR      |   695.797- 10.422= 685.375 SEC
08.45.31.  1149.0    695.797  IFP  BEGN |
09.01.24.  2102.0   1335.940  IFP  END  |  1335.940-695.797= 640.143 SEC
09.01.24.  2102.0   1335.944  XGPI      |
```

CDC NASTRAN 88.5 RELEASE TIMING (TEST RUN BY G.CHAN):

```
03.10.50.     1.0       .184  TTIO      |
03.10.55.     6.0      4.541  TTLP      |
03.10.56.     7.0      5.127  XCSA      |     8.341-  6.501=   1.840 SEC
```

```
03.10.58.     9.0      8.341  IFP1       |    8.615-  8.341=   0.274 SEC
03.10.59.    10.0      8.615  XSOR       |  408.704-  8.615= 400.029 SEC
03.15.55.   306.0    408.704  IFP  BEGN  |
03.19.24.   515.0    708.021  IFP  END   |  708.021-408.704= 299.317 SEC
03.19.24.   515.0    708.025  XGPI       |
```

CDC TIMING SUMMARY TABLE:

VERSION	COSMIC 88 OLD XSORT		COSMIC 88 NEW XSORT2		COSMIC 88.5 NEW XSORT2	
MODULE	ELAPS	CPU	ELAPS	CPU	ELAPS	CPU
XCSA	7	4.2	4	3.2	1	1.84
FP1	2	1.3	2	0.7	2	0.27
XSORT	1493	638.5	1124	685.4	296	400.03
IFP	4298	2432.7	953	640.1	209	299.32
TOT.CPU	3141.4		1337.8		713.9	
*SRU's	4259		1839		976	

* CDC COMPUTER CHARGE IS BASED ON SRU UNITS.

APPENDIX B

USERS' MANUAL UPDATE PAGES FOR INPUTT5 MODULE

VI. PARAMETERS:

1. The meanings of the first three parameter values (P1, P2, P3) are the same as those described for INPUTT2 Module, except (1) values -5 through -8 for P1 are not available, and a new P1=-9 to rewind input tape; and (2) the user file code and the FORTRAN file name are given below. (The default value for P2 is 16, or 12 for a CDC computer.)

FORTRAN LOGICAL UNIT, P2	USER FILE CODE
11	UT1 (CDC only)
12	UT2 (CDC only)
14	INPT (UNIVAC,VAX)
15	INP1 (All
16	INP2 machines
:	: except
23	INP9 CDC)
24	INPT (IBM only)

2. The fourth parameter (P4) for this module is used to specify whether the user tape was written with formats (P4=1), or binary tape (P4=0). Default is P4=0.

VII. METHODS:

Since INPUTT5 is intended to be a companion module to OUTPUT5, it is therefore suggested that the user should refer to the Methods and Remarks sections of the OUTPUT5 module for input tape structure.

Subroutine INPTT5 is the main driver for the INPUTT5 module. Its primary function is to read matrix data blocks from the user input tape. When a table data block is encountered, INPTT5 calls subroutine TABLEV to process the data. The user input tape always begins with a tape ID record which tells when the tape was generated, on what machine, tape identification, formatted or unformatted tape, and NASTRAN system buffer size. This tape ID record can be skipped, or read by the following FORTRAN code:

```
      INTEGER TAPEID(2),MACHIN(2),DATE(3),BUFSIZ,P4X
      READ (TAPE   ) TAPEID,MACHIN,DATE,BUFSIZ,P4X    or
      READ (TAPE,10) TAPEID,MACHIN,DATE,BUFSIZ,P4X
   10 FORMAT (2A4,2A4,3I8,I8,I8)
```

DIRECT MATRIX ABSTRACTION

I. NAME: INPUTT5 (Reads A User-Written FORTRAN File, Formatted or Unformatted)
(The companion module is OUTPUT5)

II. PURPOSE: Recovers up to five data blocks from a FORTRAN-written user file, formatted or unformatted. (The FORTRAN file may reside either on physical tape or on a mass storage device.) This file may be written either by a user-written FORTRAN program or by the companion module OUTPUT5. The Programmers' Manual describes the format of the user tape which must be written in order to be readable by INPUTT5. The unformatted binary tape can only be read by a computer of the same manufacturer as the one that created the tape. The formatted tape can be created and read by different computers (CDC, UNIVAC, IBM, and VAX). The data blocks to be recovered can be matrices, tables, or both.

III. DMAP CALLING SEQUENCE:

```
INPUTT5  /DB1,DB2,DB3,DB4,DB5/C,N,P1/C,N,P2/C,N,P3/C,N,P4 $
```

INPUTT5 is intended to have the same logical action as the FORTRAN User File module INPUTT2 and the GINO User File module INPUTT1 except for formatted tape. It is therefore suggested that the examples shown under modules INPUTT2 and OUTPUT1 be used for OUTPUT5 as well, excepting the addition of the P4 parameter.

IV. INPUT DATA BLOCKS:

Input data blocks are not used in this module call statement.

V. OUTPUT DATA BLOCKS:

DBi are data blocks which will be recovered from one of the NASTRAN tape files INP1, INP2 through INP9 (UT1, UT2 for CDC computer). Any or all of the output data blocks may be purged. Only non-purged data blocks will be taken from the user tape. The data blocks will be taken sequentially from the tape starting from a position determined by the value of the first parameter. Note that any purged output file will cause skipping of a corresponding file in the user input tape. The output data block sequence A,B,,, is not equivalent to ,A,,B, or ,,,A,B.

UNFORMATTED TAPE:

The rest of the unformatted tape can be read by the following FORTRAN code:

```
READ (TAPE) L,J,K,(ARRAY(I),I=J,K)
```

where L is a control word;

```
      L = 0, ARRAY contains matrix (or table) header record
        =+n, ARRAY contains data for the nth column of the matrix
        =-1, ARRAY contains end of matrix record.
      The ARRAY below J and above K are zeros.
```

The matrix header record and the table header record (L=0) differ only on the 5th and 6th words of ARRAY. If both words are zeros, it is a table header, and the entire table data can be read by:

```
READ (TAPE) L,(ARRAY(I),I=1,L)
```

where ARRAY may contain integers, BCD words, and real single and double precision numbers.

Table data ends with a (1,0.0) record.

FORMATTED TAPE:

For matrix data, the rest of the formatted tape can be read by:

```
    READ (TAPE,20) L,J,K,(ARRAY(I),I=J,K)
20  FORMAT (3I8,/,(10E13.6))     (for single precision data) or
20  FORMAT (3I8,/,(5D26.17))     (for double precision data)
```

where the control words L, J, and K are the same as in the unformatted case, and th data type, single or double precision, is determined already by the 4th word of the matrix trailer embedded in the matrix header record. (See Remark 5 of OUTPUT5 module)

For table data, the rest of the formatted tape can be read by:

```
    CHARACTER*5 ARRAY(500)
    READ (TAPE,30) J,(ARRAY(I),I=1,J)
30  FORMAT (I10,24A5,/,(26A5))
```

Notice the formatted record was written in the units of 5-byte character words, and the first byte of each unit indicates what data type follows. The following table summarizes the method to decode the character data in ARRAY.

FIRST BYTE	DATA TYPE OF ARRAY	UNITS USED	FORMAT
'/'	BCD word	1	A4
'I'	Integer	2	I9
'R'	Real, s.p.	3	E14.7
'D'	Real, d.p.	3	D14.7
'X'	Filler	1	4X

Table data ends with a (1,'0') record.

VIII. EXAMPLES:

```
$ COPY KJI AND KGG TO INP1 (UNIT 15), SEQUENTIAL FORMATTED TAPE
  OUTPUT5 KJI,KGG,,,//-1/15/*MYTAPE*/1  $
$ RECOVER THE 2 FILES FROM INP1 (UNIT 15) AND MAKE THEM NASTRAN GINO FILES
  INPUTT5 /OKJI,OKGG,,,/-1/15/*MYTAPE*/1  $
```

VIII. REMARKS:

1. Since open core is used to receive data from user input tape, INPUTT5 can handle all kinds and all sizes of data blocks.

2. The UNIVAC and VAX users should read the Important Note at the end of the description of the INPUTT2 module.

APPENDIX C

USERS' MANUAL UPDATE PAGES FOR OUTPUT5 MODULE

DIRECT MATRIX ABSTRACTION

I. NAME: OUTPUT5 (Creates A User-Written FORTRAN File, Formatted or Unformatted)
(The companion module is INPUTT5)

II. PURPOSE: Writes up to five NASTRAN GINO data blocks to a user FORTRAN file using a FORTRAN write, formatted or unformatted. (The FORTRAN file may reside either on physical tape or on a mass storage device.) If the data block contains matrix data, each matrix column is first unpacked, then written out to the user file in unpacked form. If the data block contains table data and formatted records are requested, a dynamic scheme is used to generate the appropriate format for the FORTRAN write. Coded symbols are also included in the formatted table data, so that they can be read back into the NASTRAN system by the INPUTT5 module, or by a user written FORTRAN program. Mixed matrix and table data blocks are allowed in one OUTPUT5 operation.

The unformatted (binary) user file is intended to be used later in the same computer, or a similar computer of the same manufacturer. The formatted file can be generated in one computer system and used later in another, with complete freedom in operating systems and computer manufacturers. The formatted file can be viewed and edited by the use of the system editor. The records contain 132 characters (or less) per line.

The parameters in OUTPUT5 are modeled after OUTPUT2. They can be used to direct which user output file (INP1, INP2, UT1 etc.) is to be used, to write formatted or unformatted records, to position the output file prior to writing, and to place an End-Of-File mark at the end of the tape. Multiple calls are allowed. The user is cautioned to be careful when positioning the user output file with OUTPUT5, since he may inadvertently destroy information through improper positioning. Even though no data blocks are written, an EOF will be written at the completion of each call, which has the effect of destroying anything on the tape forward of the current position.

III. DMAP CALLING SEQUENCE:

```
OUTPUT5  DB1,DB2,DB3,DB4,DB5//C,N,P1/C,N,P2/C,N,P3/C,N,P4/C,N,T1/C,N,T2/
         C,N,T3/...C,N,T10 $
```

OUTPUT5 is intended to have the same logical action as the FORTRAN User File module OUTPUT2 and the GINO User File module OUTPUT1, except for formatted tape. It is therefore suggested that the examples shown under modules OUTPUT2 and OUTPUT1 be used for OUTPUT5 as well, excepting the addition of the P4 parameter. All samples should be ended with a call to OUTPUT5 with P1=-9.

IV. INPUT DATA BLOCKS:

DBi - Any data block which the user desires to be written on one of the NASTRAN FORTRAN user files INPT, INP1, INP2,..., INP9. Any or all of the input data blocks may be purged. Only unpurged data blocks will be placed on the user file.

V. OUTPUT DATA BLOCKS: None.

VI. PARAMETERS:

1. The meanings of the first three parameter values (P1, P2, P3) are the same as those described for the OUTPUT2 Module, except the user file code and the FORTRAN file name are given below. (The default value for P2 is 15, or 11 for a CDC machine.)

FORTRAN LOGICAL UNIT, P2	USER FILE CODE
11	UT1 (CDC only)
12	UT2 (CDC only)
14	INPT (UNIVAC,VAX)
15	INP1 (All
16	INP2 machines
:	: except
23	INP9 CDC)
24	INPT (IBM only)

2. The fourth parameter (P4) for this module is used to specify whether the user output tape is to be written formatted (P4=1), or unformatted (P4=0, default). Unless the tape is to be used later by a different computer or a different operating system, the unformatted tape should be used.

3. The 10 Ti parameters (T1, T2, T3,..., T10) are used only for table data blocks. They are used only when a formatted output file is requested (P4=1), and the user wants to override the automatic format generation of the OUTPUT5 module. (Default - all Ti are zeros)

   The following rules are used to create user-directed output format:

   a. 9 digits must be specified on a Ti parameter. Zero fill if necessary.
   b. The digits are continued among the Ti parameters; therefore up to 90 digits are allowed. The digits are arranged from left to right. First digit specifies the

format of the first data word. Second, third, fourth, etc., specify the second, third, fourth data words, etc. (See exception below using digits 5 through 9)

c. The values of digits and their meanings are -
   0, format not specified; whatever format OUTPUT5 generated will be used,
   1, specifies integer format,
   2, specifies single precision real format,
   3, specifies BCD format,
   4, specifies double precision real format, and
   5-9, specify multiple format of the same type indicated by next digit, which must be 0 through 4.
   e.g. 061352000 is same as 0111111322222000

VII. METHODS:

The methods used to transfer data from NASTRAN GINO data blocks to the user output tape (or file) depend on whether

a. the data blocks are matrix or table,
b. formatted or unformatted output tape is requested, and
c. data contains single precision real numbers or double precision numbers, or both. (Table data block only)

The methods used must also guarantee continuity of mixed matrix and table types of block data on the user output tape. That is, the mixed data must be able to be read back into the NASTRAN system, or processed by a user's program, by a common switching mechanism.

OUTPUT5 treats any input data block as matrix if the 5th and the 6th words (maximum non-zero matrix column length and matrix density) are both non-zero. Otherwise, the data block is table. This method is, however, not perfect. Most table data blocks generated by LINK1, such as GEOM1, GEOM2, EPT, MPT, etc. may have non-zero 5th and 6th trailer words.

UNFORMATTED TAPE -

The data transfer from a GINO file to an unformatted tape is comparatively simple. The difference in processing matrix data and table data lies in a single key word of the length of each record.

MATRIX - A matrix header record that includes the original GINO trailer is written to user tape first. Thus the total number of records (equal number of columns) and the length of each record (equal number of rows) are known. Each column of the matrix is unpacked and copied out to the user tape, except that the leading and trailing zeros are not copied out. The data is either single precision or double precision real numbers. Each output record is also preceded by three control words. The following FORTRAN code can read one such column array (the ICOL matrix column):

```
     READ (TAPE) ICOL,JB,JE,(ARRAY(J),JB,JE)
```

TABLE - A table header record, with the 5th and 6th trailer words set to zeros, is also written out to indicate the following records are of table type. Records from the input GINO data block are read and transferred to user tape directly, except each output record is preceded by one additional word, which tells the total length of this current record. The following FORTRAN code can be used to read one such record:

```
     READ (TAPE) LENGTH,(ARRAY(J),J=1,LENGTH)
```

FORMATTED TAPE -

Most of the attributes of unformatted tape apply equally well to the formatted tape, except tapes are written with FORTRAN formats.

MATRIX - All integers are written in I8 format, BCD in A4 format, single precision real numbers in E13.6, and double precision numbers in D26.17. Only the matrix header record can have all four data types; the matrix column records contain only real numbers. The following FORTRAN code reads the header record and/or a matrix column:

```
       WRITE (TAPE,10) I,J,K,(A(L),L=J,K)
    10 FORMAT (3I8,/,(10E13.6 ))     (for single precision data) or
    10 FORMAT (3I8,/,( 5D26.17))     (for double precision data)
```

TABLE - All integers are written in ('I',I9) format, BCD in ('/',A4) format, single precision real numbers in ('R',E14.7) format, and double precision numbers in ('D',E14.7). Notice that 5 bytes are used for BCD, 10 bytes for integer, and 15 bytes for real numbers, single or double precision. NASTRAN table data blocks often contain integers, BCD, and single and double precision real numbers in a mixed fashion. Each table record may have a different table length. To write formatted NASTRAN tables and to read them back later present a real challenge in FORTRAN programming. The OUTPUT5 module calls subroutine TABLE5 to process table data, and the INPUTT5 module calls subroutine TABLEV to read them back.

TABLE5 generates dynamically a unit of format - ('I',I9), ('/',A4), etc. - to match each data type - integer, BCD, etc. When the synthesized format reaches 130 characters (or bytes), a line of data is written out. A table therefore may require multiple lines (each line physically is a record). In addition, the first word of the first line contains the total length of this table. The following FORTRAN code can be used to read back a table from the user tape into 5-character ARRAY:

```
      CHARACTER*5 ARRAY(500)
      READ (TAPE,20) LENGTH,(ARRAY(J),J=1,LENGTH)
   20 FORMAT (I10,24A5,/,(26A5))
```

The first byte of each 5-character ARRAY (which is I,/,R, or D) can be used to convert the 5-, 10-, or 15-character data back to BCD, integer, or real numbers (single or double precision). For more details, see INPUTT5 module and INPTT5 FORTRAN source subroutine.

TABLE5 calls subroutine NUMTYP to determine the data type, then issue the corresponding format for output. NUMTYP, however, is not one hundred percent foolproof. One in five or ten thousand times, NUMTYP may err in determining exactly the data type. Also, when TABLE5 passes a computer word to NUMTYP with no other information, NUMTYP cannot tell if it is part of a double precision word, or if it is a single precision word. (In this case, single precision word is assumed.) Finally, NUMTYP cannot distinguish between integer zero and real number zero. (A period may be important in the output format). TABLE5 therefore may generate the wrong format due to NUMTYP's internal limitations.

In case that TABLE5 does produce erroneous format, the user can override the automatic format generation by the Ti parameters which supply OUTPUT5 the exact format to use, in a condensed, coded form. 90 (or more if 5, 6, 7, 8, or 9 are used in the Ti specification) unit formats can be specified.

The following example illustrates the use of the Ti parameter.

```
Data on table:
3  4  3.4  5.0E-3  TESTING  .6D+7  9  G  3.2  8  0.  0  4 12 13  14  15  28  61   88
14   44 .7D+7

Ti specification:
T1=112233413, T2=212516140  or
T1=604000025, T2=060400000 (7th and 24th words are d.p.
                            and 12th word is real)
NOTE - 2 BCD words in 'TESTING',
       all others are 1 computer word per data entry.
       T2, the last Ti used here, must fill up with zeros to make up a 9-digit word.
```

When viewed with a system editor, the above example looks like this (first line):

```
37I       3I        4R 5.0000000E-3/TEST/ING D 6.0000000D+07 etc.
++----------+++++++++---------------++++++++++---------------
     1st       2nd        3rd          4th           5th data etc.
```

The first 37 indicates there are 37 5-byte words in this record.
the '++----' line and the '1st,2nd...' line are added here for video purposes.

Since the formatted data line may not end exactly at 130 bytes, one or two fillers of the form 'X' and four blanks may appear at the end of an output line.

The matrix data blocks are handled by the main routine OUTPT5. OUTPT5 calls TABLE5 only when the former encounters a table data block input.

VIII. EXAMPLES:

```
$ Copy KJI, KGG, and CASECC to INP2 (unit 16), sequential formatted tape
  OUTPUT5 KJI,KGG,CASECC,,//-1/16/*MYTAPE*/1  $
$ Recover the files from INP2 (unit 16) and make them NASTRAN GINO files
  INPUTT5 /OKJI,OKGG,OCASECC,,/-1/16/*MYTAPE*/1  $
```

IX. REMARKS:

1. Formatted tape (P4=1) takes a longer time and more space to write than the unformatted tape. Unless the tape is intended to be used later by a different computer, unformatted tape should be selected (P4=0).

2. The OUTPUT5 'records' are written to tape 'identically' with both formatted and unformatted FORTRAN write commands. The matrix header and the table header can be read 'identically' without prior knowledge of what type of data, matrix or table, is coming up next.

3. All matrix records are written to tape in a standard way, except the first matrix header record.

   All table records are written to tape in a standard way, including table header record and the last ending record.

4. The first tape header record is composed of 9 words as shown below:

RECORD	WORD	CONTENTS	P4=0	P4=1
0	1,2	Tapeid (=P2)	2*BCD	2A4
	3,4	Machine (CDC,UNIVAC,IBM,VAX)	2*BCD	2A4
	5-7	Date	3*INT	3I8
	8	System BUFFER SIZE	INT	I8
	9	P4 used in creating tape (0,1)	INT	I8

5. This remark and the next one deal only with matrix data blocks.

Three types of data records follow the header record, or the EOF record of a previous data block. They are:

a. Matrix header record
b. Matrix column data record
c. EOF record

These records are written to tape in a standard procedure. Three control words are written out first, followed by the actual data. Binary FORTRAN write is used in unformatted tape (P4=0), and each logical record holds a complete set of data. The following FORTRAN statement is used to write the entire data record:

```
     WRITE (TAPE) I,J,K,(A(L),L=J,K)
```

For formatted tape, multiple logical records are actually written for each complete set of data. The following FORTRAN statements are used to write the entire data record:

```
     WRITE (TAPE,30) I,J,K,(A(L),L=J,K)
  30 FORMAT (3I8,/,(10E13.6))      (for single precision data) or
  30 FORMAT (3I8,/,(5D26.17))      (for double precision data)
```

In the above WRITE statements, the value of I is used to indicate the type of record just read.

VALUE OF I	TYPE OF RECORD
0	Matrix header record
+n	Nth matrix column data
-1	End-of-matrix

The column data is written to tape from the first non-zero row position (J) to the last non-zero row position (K). The following table describes the contents of the data records written to tape by the OUTPUT5 module.

RECORD+	WORD	CONTENTS	P4=0	P4=1
1		Matrix header record -		
	1	0	INT	I8
	2,3	1,1	2*INT	2I8
	4	0.0	F.P.	E13.6 or D26.17
	5-10	Matrix trailer (Col,Row,Form,Type,Max,Density)	6*INT	6I8
	11,12	DMAP Name of DB1	2*BCD	2A4
2	1	1 (First matrix column)	INT	I8
	2	Row pos. of first non-zero elem.	INT	I8
	3	Row pos. of last non-zero elem.	INT	I8
	4-W	First banded column data (W=Word3-Word2)	6*INT	(**)
3	1	2 (Second matrix column)	INT	I8
	2	Row pos. of first non-zero elem.	INT	I8
	3	Row pos. of last non-zero elem.	INT	I8
	4-W	Second banded column data	6*INT	(**)
4	1	3 (Third matrix column)	INT	I8
	2	Row pos. of first non-zero elem.	INT	I8
	3	Row pos. of last non-zero elem.	INT	I8
	4-W	Third banded column data	6*INT	(**)
:	:	:		
L	1	L-1 (last matrix column)	INT	I8
	2	Row pos. of first non-zero elem.	INT	I8
	3	Row pos. of last non-zero elem.	INT	I8
	4-W	Last banded column data	6*INT	(**)
L+1	1	-1	INT	I8
	2,3	1,1	2*INT	2I8
	4	0.0	F.P.	D26.17

(Repeat records 1 through L+1 for next matrix data block.)

Where (**) is (10E13.6) or (5D26.17).

(+RECORD No. does not correspond one to one to the actual physical record No.)

6. A record of (n,1,1,0.0) is written out for a null Nth column.

7. This remark deals only with table data blocks.

   Three types of data record follow the header record, or an EOF record of previous data block. They are:

   a. Table header record
   b. Record(s) of a table (a table data block can have more than one table record)
   c. EOF record.

   The table header record has a general structure as in the standard procedure for the matrix records, except that the 5th and 6th words of the matrix trailer section are zeros.

   The table record was discussed in great detail in the METHOD section for both formatted and unformatted output tape. A table record is created for each table in the input data block, and no skipping forward or backward is allowed on the input file.

   If double precision data are encountered in a table record, the double precision data will be truncated to single precision, but the format of ('D',E14.7) will be used. (INPUTT5 will re-generate the data back to their double precision status.)

   An End-Of-File record in the form of '-1 1 1 0.0D+0' ends the table record output.

8. Since the formatted tape (P4=1) is intended to be used in different computers, the OUTPUT5 module appends no system control word(s) to the FORTRAN written formatted records. The output tape must be unlabeled, fixed block size with record size of 132 characters, and ANSI unpacked character data set. The specification of the tape is either internally specified (UNIVAC) by a FORTRAN open statement, or uses system default tape specification (IBM and VAX). The CDC user must specify the output tape externally by the appropriate FILE, LABEL, or REQUEST cards:

   For example:
   ```
   LABEL,TAPE,NT,D=1200,CV=AS,F=S,LB=KU,PO=W.
   FILE,TAPE,MRL=132,MBL=132,RT=F,BT=C.
   ```

9. Since open core is used in data processing, the OUTPUT5 module is capable of handling all kinds and all sizes of input data blocks.

APPENDIX D

USERS' MANUAL UPDATE PAGES FOR PARAMD MODULE

DIRECT MATRIX ABSTRACTION

I. NAME: PARAMD (Parameter Processor - Double Precision)

II. PURPOSE: To perform specified arithmetic, logical, and conversion operations on double precision real or double precision complex parameters.

III. DMAP CALLING SEQUENCE:

```
PARAMD  // C,N,OP / V,N,OUTD / V,N,IND1 / V,N,IND2 / V,N,OUTC / V,N,INC1 / V,N,INC2 /
                  V,N,FLAG $
```

IV. INPUT DATA BLOCKS: None.

V. OUTPUT DATA BLOCKS:

None.

VI. PARAMETERS:

OP - Input-BCD operation code from the table below - no default
OUTD - Output D.P.-default = 0.0D+0
IND1 - Input-D.P.-default = 0.0D+0
IND2 - Input-D.P.-default = 0.0D+0
OUTC - Output-D.P.-complex-default = (0.0D+0, 0.0D+0)
INC1 - Input -D.P.-complex-default = (0.0D+0, 0.0D+0)
INC2 - Input -D.P.-complex-default = (0.0D+0, 0.0D+0)
FLAG - Output/output-integer-default= 0 (See Remark 6)

The values of parameters are dependent upon OP as shown in the table described in PARAMR module (pages 5.5-40 and 41). In addition, a new OP operation code is added:

OP	OUTPUTS
ERR	If Flag is set to 0 (or by default), NASTRAN system NOGO flag (the 3rd word of /SYSTEM/) is set to integer zero unconditionally. If FLAG is set to non-zero by user, NASTRAN job will terminate if any preceding PARAMD (or PARAMR) has non-fatal error(s).

VII. REMARKS:

1. All parameters, except OP, must be "V" type. Default parameter values will be used in case of error. Error in input parameter(s) would cause output parameter(s) to pick up the original default value(s).

2. All input errors are non-fatal, with error messages printed.

3. PARAMD does its own SAVE; therefore, a SAVE is not needed following the module.

4. For OP = DIV or OP = DIVC, the output is zero if the denominator is zero, and FLAG is set to +1.

5. For OP = SIN, OP = COS or OP = TAN, the input must be expressed in radians.

6. The default value of FLAG is zero as stated in the Programmer's manual. All NASTRAN releases prior to 1989 actually used a +1 instead of 0. The case where FLAG = -1, was not affected.

7. Remarks 1, 2, and 6 also apply to the PARAMR module. The new ERR operation code is also available in PARAMR.

VIII. EXAMPLES:

```
PARAMR //*ERR*  $
PARAMR //*ADD*      /V,N,R1SP4 /V,N,R1    /V,N,SP4   $
PARAMR //*SUB*      /V,N,R1SP4 /V,N,R1    /V,N,SP4   $
PARAMR //*ABS*      /V,N,ABSR1 /V,N,R1               $
PARAMR //*SQRT*     /V,N,SQTR1 /V,N,ABSR1            $
PARAMR //*MPYC* ////V,N,CMPY   /V,N,SCPLX /V,N,CS1   $
PARAMR //*COMPLEX*//V,N,R1     /V,N,SP4   /V,N,OUTC  $
PARAMR //*LE*     //V,N,R1     /V,N,SP4////V,N,LEFLG $
PARAMD //*MPY*      /V,N,RDPDP /V,N,RDPX  /V,N,RDPX  $
PARAMD //*DIV*      /V,N,DP4X  /V,N,DP4   /V,N,RDPX  $
PARAMD //*EXP*      /V,N,EXPX  /V,N,DP4   /V,N,RDP   $
PARAMD //*CONJ* ////V,N,CONJX  /V,N,CDP4             $
PARAMD //*EQ*     //V,N,EXPX   /V,N,DP4////V,N,EQFLG $
PARAMD //*DIVC* ////V,N,DIVCX  /C,Y,DCPLX4/V,N,CDP4  $
PARAMD //*ERR*  ////  //       /C,N,1                $
PRTPARM // 0    $
```

APPENDIX E

USERS' MANUAL UPDATE PAGES FOR SCALAR MODULE

DIRECT MATRIX ABSTRACTION

I. <u>NAME</u>: SCALAR (Converts matrix element to parameter)

II. <u>PURPOSE</u>: To extract a specified element from a matrix for use as a parameter.

III. <u>DMAP CALLING SEQUENCE</u>:

```
SCALAR  DB // C,N,ROW/C,N,COL/V,N,RSP/V,N,RDP/V,N,CSX/V,N,CDX   $
```

IV. <u>INPUT DATA BLOCKS</u>:

DB - may be any type of matrix (single precision or double precision, real or complex)

V. <u>OUTPUT DATA BLOCKS</u>: None.

VI. <u>PARAMETERS</u>:

ROW - Row number of element to be extracted from [DB]. Integer input, default= 1

COL - Column identification of element. Integer input, default= 1

RSP - Output, value of element(ROW,COL) in single precision real, default= 0.0

RDP - Output, value of element(ROW,COL) in double precision real, default= 0.D+0

CSX - Output, value of element(ROW,COL) in single precision complex, default= (0.,0.)

CDX - Output, value of element(ROW,COL) in single precision complex, default= (0.D+0,0.D+0)

VII. <u>REMARKS</u>:

1. RSP, RDP, CSX and CDX will be set by the module whenever they are present and of the "V" type parameters. The parameters will be printed out in their respective formats according to their precision types. Warning message will be printed if type mismatch occurs or element specified is out of matrix range.

2. After execution, the parameter value will be delivered to NASTRAN's executive VPS table as a numerical value in the form specified by any of the parameters RSP, RDP, CSX, or CDX. The output parameters can also be printed by the PRTPRM module which carries normally more digits.

3. SCALAR does its own SAVE; therefore, a SAVE is not needed following the module. There is no save for any invalid parameter, and the default value remains unchanged.

4. If [DB] is purged, all parameter default values remain unchanged.

5. All the output parameters can be printed out by PRTPRM module.

6. See PARAML for similar capability.

VIII. EXAMPLES:

Obtain the value of the element in column 8 and row 2 of the matrix KLL.

```
SCALAR  KLL//C,N,2/C,N,8  /V,N,S1  $
SCALAR  KLL//C,N,2/C,N,8 //V,N,D1/V,N,S2/V,N,D2  $
```

The output parameters give the following results:

S1 = KLL(2,8), in single precision real,
D1 = KLL(2,8), in double precision real,
S2 = KLL(2,8), in single precision complex expression, and
D2 = KLL(2,8), in double precision complex expression.

APPENDIX F

USERS' MANUAL UPDATE PAGES FOR PARAML MODULE

DIRECT MATRIX ABSTRACTION

I. NAME: PARAML (Abstract parameters from a list)

II. PURPOSE: To convert an element from a GINO matrix or table data block, to a legitimate NASTRAN parameter, or parameters

III. DMAP CALLING SEQUENCE:

```
PARAML  DB // C,N,OP / V,N,P1 / V,N,P2 / V,N,RSP/ V,N,INT/ V,N,RDP/ V,N,BCD/ V,N,CSX/
        V,N,CDX   $
```

IV. INPUT DATA BLOCKS:

DB - Any GINO data block file (table or matrix, single precision or double precision, real or complex)

V. OUTPUT DATA BLOCKS: None

VI. PARAMETERS:

OP - One of the following key words, BCD input, no default. 'MATRIX', 'NULL', 'PRESENCE', 'TRAILER', 'TABLE1', 'TABLE2', or 'TABLE4'

P1,P2 - Input, see Remark 4 below, integer input, default= 1,1

P2 - Output, integer (only in OP=TRAILER)

RSP - Output, single precision real number, default= 0.0

INT - Output, integer number, default= 0

RDP - Output, double precision real number, default= 0.D+0

BCD - Output, two BCD words in 2A4 format, default= (VOID)

CSX - Output, single precision complex number, default= (0.,0.)

CDX - Output, double precision complex, default= (0.D+0,0.D+0)

VII. REMARKS:

1. RSP, INT, RDP, BCD, CSX and CDX will be set by the module whenever they are present and of the "V" type parameters. The parameters will be printed out in their respective formats according to their precision types. Warning message will be printed if type mismatch occurs or end-of-record is encountered.

2. After execution, the parameter value will be delivered to NASTRAN's executive VPS table as a numerical value in the form specified by any one or some of the parameters RSP, RDP, CSX, CDX, INT, or BCD (4 BCD characters per word, the rest of the word blank filled).

3. PARAML does its own SAVE; therefore, a SAVE is not needed following the module. Invalid parameter due to type mismatch or EOR encountered, is not saved and the default value remains.

4. P1 and P2 control the location in the data block of the element to be selected. The meaning of P1 and P2 depend on OP selection as explained in Remarks 5 through 9.

5. If OP = TABLEi (where i=1,2,OR 4), P1 is the record number and P2 is the word position of the target element in DB. Word position is based on computer word count (1 word per integer or s.p.real, 2 words per d.p.real or s.p.complex, and 4 words per d.p.complex). The table data from record P1 and word P2 (or word P2 plus more) will be delivered to the VPS table as a numerical value in the form specified.

   If OP = TABLE1, one data word from P2 word position, record P1, will be used to form the output parameter.

   If OP = TABLE2, two data words from P2 and P2+1, record P1, will be used.

   If OP = TABLE4, four words from P2, P2+1, P2+2, and P2+3, record P1, will be used.

   Since table data block DB can contain mixed types of data, the user must know ahead of time what the original data type is, and select TABLE1, TABLE2, or TABLE4 accordingly.

For example,

the data in P2, p2+1, P2+2, and P2+3 are a, b, c, d, and the output parameter request is d.p.complex CDX,
TABLE1 gives CDX = (a.D+0, 0.D+0)
TABLE2 gives CDX = (a.D+0, b.D+0)
TABLE4 gives CDX = (e.D+0, f.D+0)
where e is a d.p.real number formed by the union of a and b,
and f, by the union of c and d.

6. If OP = MATRIX, P1 is the row number and P2 is the column number of the matrix in [DB] to be read. The matrix element of (ROW,COL) will be delivered to VPS as a numerical value in the form specified by one or more of the parameters RSP, RDP, CSX, or CDX. Requests for CSX or CDX from a real matrix will assign the value of (ROW,COL) to the real part and zero to the imaginary part. The requested output parameter(s) are set to zero(s) and a warning message is issued if:
   (1) P1 and/or P2 exceed the matrix order,
   (2) requests for RSP and RDP from a complex matrix,
   (3) requests for INT and BCD from [DB],
   and the invalid output parameter(s) are not saved.

   (Notice that row first and column second is consistent with SCALAR module parameter input, and also with common practice in matrix element designation; (row,column)).

7. If OP = NULL and if [DB] is a matrix, INT is set to -1 if the sixth word of the matrix trailer, the matrix density, is zero.

8. If OP = PRESENCE, INT will be -1 if input data block is purged.

9. If OP = TRAILER, P2 is output as the value of ith word of the matrix trailer where i is set by P1 in accordance with the following table.

P1	TERM OF MATRIX TRAILER
1	Numbers of columns
2	Number of rows
3	Form of matrix
4	Precision of matrix
5	Maximum number of nonzero terms in any column of the matrix
6	Matrix density

10. One or more of the output parameters can be requested simultaneously.

11. After execution, a user information message prints out the parameter value in the format prescribed by the user. The output parameters can also be printed by the PRTPRM module which carries normally more digits. (PRTPRM may actually print integer zero in a real number format, 0.0)

12. See SCALAR module for similar capability.

VIII. EXAMPLES:

Obtain the value in column 1, row 4 of a real matrix, and record 2 word 5 of a table.

```
PARAML  KGG //*MATRIX*/C,N,4/C,N,1      /V,N,STERM  $
PARAML  KGG //*MATRIX*/C,N,4/C,N,1    ///V,N,DTERM  $
PARAML  KGG //*MATRIX*/C,N,4/C,N,1 /////V,N,CSTERM $
PARAML  KGG //*MATRIX*/C,N,4/C,N,1//////V,N,CDTERM $
PARAML  KGG //*MATRIX*/C,N,4/C,N,1/V,N,TERM1//V,N,TERM2//V,N,TERM3/V,N,TERM4 $
PARAML  CASECC //*TABLE1*/C,N,2/C,N,2  //V,N,ATERM $
PARAML  CASECC //*TABLE2*/C,N,2/C,N,5////V,N,BTERM $
```

The above output parameters yield the following results:

```
STERM ,TERM1 = KGG(4,1), in single precision,
DTERM ,TERM2 = KGG(4,1), in double precision,
CSTERM,TERM3 = KGG(4,1), in single precision complex expression,
CDTERM,TERM4 = KGG(4,1), in double precision complex expression
ATERM  = 2nd word of the 2nd record of CASECC, integer, and
BTERM  = 5th and 6th words of the 2nd record of CASECC, 2 BCD words.
```

APPENDIX G

USERS' MANUAL UPDATE PAGES FOR ADD MODULE

I. NAME: ADD (Matrix Add)

II. PURPOSE: To compute [X] = a[A] + b[B] where a and b are scale factors.

III. DMAP CALLING SEQUENCE:

```
ADD    A,B / X / C,Y, ALPHA=(1.0,2.0)     / C,Y, BETA=(3.0,4.0)
                / C,Y,DALPHA=(5.D+0,6.D-1) / C,Y,DBETA=(7.D+2,8.D-3)  $
```

IV. INPUT DATA BLOCKS:

A - Any GINO matrix

B - Any GINO matrix

V. OUTPUT DATA BLOCKS:

X - Matrix

VI. PARAMETERS:

ALPHA - Input-complex-single precision, default = (0.0, 0.0). This is a, the scalar multiplier for [A] if DALPHA and DBETA are zeros.

BETA - Input-complex-single precision, default = (0.0, 0.0). This is b, the scalar multiplier for [B] if DALPHA and DBETA are zeros.

DALPHA - Input-complex-double precision, default = (0.0D+0, 0.0D+0). This is a, the scalar multiplier for [A] if ALPHA and BETA are zeros.

DBETA - Input-complex-double precision, default = (0.0D+0, 0.0D+0). This is b, the scalar multiplier for [B] if ALPHA and BETA are zeros.

VII. SUBROUTINE: DADD

VIII. METHOD:

The parameters are checked. If [A] is not purged, the number of columns, rows, and form of [X] are set to those of [A]. Otherwise the [B] descriptors are used. The flags for the type of [X] (see Remark 2) and multiply-add operations are set before calling subroutine SADD, which performs the actual scalar multiplication and matrix addition.

VIII. REMARKS:

1. Matrix [A] and/or matrix [B] may be purged, in which case the corresponding term in the matrix sum will be assumed null. The input data blocks must be unique.

2. Matrix [X] cannot be purged. The type of [X] is maximum of the types of [A], [B], a, b. The size and shape of [X] are the size and shape of [A] if [A] is present. Otherwise they are those of [B].

3. The use of double precision parameters DALPHA and DBETA will force the matrix multiply-and-add operation to be performed in double precision unconditionally. The single precision ALPHA AND BETA may cause the multiply-and-add operation to be performed in single precision or in double precision depending on the matrix original precision types.

4. Either the DALPHA-DBETA pair or the ALPHA-BETA pair is used. They cannot be mixed; that is, DALPHA-BETA pair is illegal; so is DALPHA-ALPHA.

5. If Im(ALPHA or DALPHA) or Im(BETA or DBETA) is zero, the corresponding parameter will be considered real.

APPENDIX H

USERS' MANUAL UPDATE PAGES FOR THE GINOFILE MODULE

I. NAME: GINOFILE (Gino File Creation)

II. PURPOSE: To capture data from a scratch file of a preceding DMAP module and copy the data to a NASTRAN GINO file. Type of data can be table or matrix.

III. DMAP CALLING SEQUENCE:

```
GINOFILE  /FILE/C,N,P1/C,N,P2/C,N,P3   $
```

IV. INPUT DATA BLOCK: None.

V. OUTPUT DATA BLOCK:

FILE - Any GINO output file name

VI. PARAMETERS:

P1 - Any 300-series scratch File number (301,302,303,...), integer.

P2 - Additional records to be skipped on P1 file before data transfer from P1 to FILE, integer. GINOFILE will automatically skip over header record if a header record exists in P1, or it will not skip if it does not exist. (Default P2 = 0.) Data transfer starts from P2+1 record after header (or no header) record on scratch file.

P3 - Last record to be copied, or up to an EOF mark on P1 file. Total number of records copied is (P3 - P2), integer. (Default is to copy to EOF mark.)

VII. SUBROUTINE:

GINOFL - Subroutine in GINOFILE module

VIII. METHOD:

At the end of a NASTRAN executable module, all the input files, output files, and scratch files are closed. The input files are read only and they will remain untouched. The output files are saved, and their names are preserved. (The output file names are actually allocated before the beginning of the module execution). The scratch files are released without any mechanism of saving them. However, the data of the scratch files are still in the system disc space, and will remain there until they are over-written by another part (or another module) of the NASTRAN program. It is at this point that GINOFILE module grabs hold of a scratch file of the preceding module and copies the data to a GINO output file, without changing the scratch file data. Tables or matrices are copied the same way - as they exist in the original form on the scratch file.

A NASTRAN GINO file always has a header record and a 6 word trailer. However, the header record and the trailer are not required for a scratch file, and they may or may not exist. The GINOFILE module will first test the header record of the scratch file and skip over it, if it exists. A header record is always generated by GINOFILE for the new GINO file. The beginning record and the ending record where data are to be transferred are under user control. Finally, a trailer for the output file is generated and saved. An EOF record is written to the new GINO file at the completion of the module.

IX. DESIGN REQUIREMENT:

The GINOFILE module is mapped in all NASTRAN Links, except LINK1. The user can request this module through a regular NASTRAN DMAP Alter.

The user must request this module immediately following the DMAP module where the scratch file was used. It is the user's responsibility to see that the Executive Segment File Allocator, XSFA, does not come in between the preceding DMAP module and this GINOFILE module. If XSFA does intervene before GINOFILE execution, the FIAT/OSCAR table (see XSFA Module description in section 4.9) is rearranged, and the scratch files are no longer accessible.

If XSFA does intervene, the user can provoke the XSFA operation and FIAT/ OSCAR table rearrangement before the execution of preceding DMAP module so that XSFA will not come in between this preceding and GINOFILE modules. The technique here can involve a DMAP alter to PURGE some obsolete files, TABPT to print some files that have been generated some time ago, and currently are not on the FIAT/OSCAR table, or any other DMAP module that would disturb the NASTRAN filing system. The user could turn on DIAG 2 and observe the flow of the GINO files created or allocated by XSFA/FIAT/ OSCAR operation.

If the scratch file in the preceding DMAP module was used repeatedly such as being used in a loop, only the "last-time-used" set of data on the scratch file can be copied out by GINOFILE.

The user should turn on DIAG 8,15,-n (where n is the current LINK number) and see that the scratch file, FORTRAN unit number, and associated trailers are being processed correctly.

X. DIAGNOSTIC MESSAGES

Message numbers 3001, 3002, and 3008 may be issued by GINOFILE.

APPENDIX I

USERS' MANUAL UPDATE PAGES FOR THE NEW DATABASE MODULE

DIRECT MATRIX ABSTRACTION

I. NAME: DATABASE (To Save Grids, Elements, Displacements, Velocities, Accelerations, Loads, Grid Point Forces, Eigenvectors, Element Stresses, and Element Forces on User Tape)

II. PURPOSE: To save following data on user tape, formatted, or unformatted for user external use:
(1) Grid points - external numbers, and their x,y,z coordinates in basic rectangular coordinate system;
(2) Connecting elements - element names, GPTABD element types, NASTRAN symbols, property IDs (or material IDs if elements have no property IDs), number of grid points, connecting grid (external) numbers; and
(3) Displacement vectors* - real or complex data in basic rectangular coordinate system, or in NASTRAN global coordinate system, in SORT1 or SORT2 data format, single-case or subcases, displacement or mode shape data.
(*including velocity, acceleration vectors, loads, grid point forces, eigenvectors, element stresses, and element forces)

III. DMAP CALLING SEQUENCE:

```
DATABASE   EQEXIN,BGPDT,GEOM2,CSTM,OUGV//C,N,OUTTP/C,N,FORMAT/C,N,BASIC  $
```

IV. INPUT DATA BLOCKS:

EQEXIN - External-internal grid tables. Must be present.

BGPDT - Basic Grid Point Definition Table.
If purge, no grid point data sent to OUTTP output tape.
If BGPDT is purged, and OUGV is present, displacement vector will not be converted to basic coordinates.

GEOM2 - Geometry 2 Data Block.
If purge, no element connectivity data sent to OUTTP.

CSTM - Coordinate System Transformation Matrix Data Block.
If purge, displacement vectors remain in global coordinate system.

OUGV - Any output displacement (velocity, acceleration, load, grid point force, eigenvector, element stress, and element force) data block written for OFP module. If present, the displacement vectors are processed and results sent out to user OUTTP tape. OUGV must be one of the following files characterized by an 1, 2, 3, 7, 10, 11, 15, or 16 on the 2nd word, last 2 digits, of the first header record, and an 8 or a 14 on the 10th word:

OUDV1, OUDVC1, OUGV1, OUHV1, OUHVC1, OUPV1, OUPVC1,
OUDV2, OUDVC2, OUGV2, OUHV2, OUHVC2, OUPV2, OUPVC2,
OUBGV1, OPHID, OPHIG, OPHIH, OCPHIP,
OPG1, OPP1, OPPC1, OQG1, OQP1, OQPC1, OQBG1,
OPG2, OPP2, OPPC2, OQG2, OQP2, OQPC2, OBQG1,
OEF1, OEFC1, OES1, OESC1, OEFB1, OBEF1, OEF2,
OEFC2, OES2, OESC2, OESB1, OBES1

If purge, no data are sent out to OUTTP.

V. OUTPUT DATA BLOCK: No GINO output data block.

VI. PARAMETERS:

OUTTP - User output tape. Must be one of the UT1, UT2, INPT, INP1, ..., INP9 files; tape or disc file. (Default INP1, FORTRAN Unit 15)

FORTRAN LOGICAL UNIT, OUTTP	USER FILE CODE
11	UT1 (CDC only)
12	UT2 (CDC only)
14	INPT (UNIVAC,VAX)
15	INP1 (All
16	INP2 machines
:	: except
23	INP9 CDC)
24	INPT (IBM only)

FORMAT = 0, unformatted output to OUTTP tape (Default);
= 1, formatted.

BASIC = 0, displacement vectors in NASTRAN's global coordinate system (Default);
= 1, displacement vectors in basic rectangular coordinate system.

VII. EXAMPLE:

```
DATABASE   EQEXIN,BGPDT,GEOM2,, /C,N,15/C,N,+1    $
DATABASE   EQEXIN,BGPDT,,CSTM,OUGV/C,N,16         $
```

First example writes the grid points and element connectivity data out to INP1 tape, formatted. The second example writes the grid points and displacement vectors in NASTRAN global coordinates out to INP2 tape, unformatted.

VIII. SUBROUTINE:

DBASE - Subroutine for DATABASE Module.

IX. METHOD:

There are three independent sets of data to be copied out to user tape OUTTP - grids data, connecting elements data, and displacement vectors (velocities, accelerations, eigenvectors, stresses, and forces). If BGPDT file is purged (that is, is not present), the grid point data set is not generated. Similarly, if GEOM2 file is purged, the element connectivity data is not generated; and the same with the OUGV file and the displacement vectors. The exact contents in the output tape OUTTP depend therefore on the input file assignment.

In all cases, EQEXIN file is opened and the grid point external number vs. the internal number table is read. If BGPDT file is present, the basic grid point data is read, and each internal grid point number is converted to its external ID number. The grid point's x, y, z coordinates from BGPDT are already in the basic rectangular coordinate system. The grid points data are then sorted by their external grid IDs before they are written out to OUTTP tape, under FORTRAN control. The following table gives the precise contents of each record in the OUTTP tape.

For UNFORMATTED tape - grid point data in one long record:

RECORD	WORD	CONTENT (UNFORMATTED)
1	1-2	'GRID PTS--------', a 16-letter identification. (BCD)
2	1	No. of words (this first word not included) in this record. (Integer)
	2	External grid ID. (Sorted, integer)
	3	0 (Not used; reserved for future use)
	4,5,6	x,y,z coordinates in basic rect. coord. system. (single precision real)
	:	Repeat words 2 thru 6 as many times as there are grids

(Total number of grid points = (WORD 1 of record 2)/5)

To read the second record into array XYZ, one can use

```
READ (OUTTP) L,(XYZ(J),J=1,L)
```

For FORMATTED tape - grid point data in multiple short records:

RECORD	WORD	CONTENT	FORMAT
1	1,2	'GRID PTS--------' identification	4A4
2	1	Total number of grid points	I8
3	1	External grid ID (Sorted)	I8
	2	0 (Not used; Reserved for future use)	I8
	3,4,5	x,y,z coordinates in basic rect. coordinate system.	3E12.6
:	1-5	Repeat record 3 as many times as there are grids	

If GEOM2 file is present, the elements data will be generated next. An element identification record is written out first.

RECORD	WORD	CONTENT (FORMATTED or UNFORMATTED)	FORMAT
1	1-2	'ELEMENTS--------', identification. BCD	4A4

The element data in GEOM2 file will be written out to the OUTTP file almost in the same way, and same order as the original data. A header record is written out for each type of element, then followed by the element data. The element data will be written out in a long record if the OUTTP is unformatted, and in multiple short records, one for each element, if OUTTP is formatted. Notice that the element types are sorted according to the NASTRAN'S GPTABD data block order; and within each type, the elements are sorted by their element IDs.

DIRECT MATRIX ABSTRACTION

ELEMENT HEADER RECORD for the UNFORMATTED output tape:

RECORD	WORD	CONTENT (UNFORMATTED)
2	1-2	Element name. (BCD)
	3	Element type number, according to GPTABD order. (Integer)
	4	Element symbol. (2 letters)
	5	Number of grid points per element. (Integer)
	6	Total no. of elements of this current element type. (Integer)
	7	No. of words in next record = WORD5 + 2 (Integer)
	8	No. of 132-column lines needed in next record if OUTTP is written with a format. (Integer)

ELEMENT RECORDS, Repeat as many times as there are elements not of the same type (that is a record for each element type):

RECORD	WORD	CONTENT (UNFORMATTED)
3	1	Element ID. (Integer)
	2	Property ID. (Positive Integer); or 0 (Element has no property ID nor material ID); or Material ID. (Element has no property ID, but it has a material ID. (Negative Integer)
	3	0 (Not used; Reserve for future use, integer)
	4,5,...	Element connecting (external) grid points. (Integers)
	:	Repeat words 1,2,3,4... as many times as there are elements of this same tape. (See WORD 6 in header record)

For FORMATTED tape -

ELEMENT HEADER RECORD, in 8-column format:

RECORD	COLUMNS	CONTENT	FORMAT
2	1- 8	'ELEMENT '	8 letters
	9-16	Element name	2A4
	17-24	' TYPE ='	8 letters
	25-28	Elem. type no. according to GPTABD	I4
	29,30	Blank	2X
	31-32	Element symbol	A2
	33-40	' GRIDS ='	8 letters
	41-48	No. of grids per element	I8
	49-56	' TOTAL ='	8 letters
	57-64	Total no. of elements of this elem. type	I8
	65-72	' WDS/EL='	8 letters
	73-80	No. of words per element in next records	I8
	81-88	' LINES ='	8 letters
	89-96	No. of lines (records) needed on next record for this element type	I8

A printout of this header record may look like this:
(the ---+++ line is for video aid; it is not part of the record)

```
 --------++++++++--------++++++++--------++++++++--------++++++++--
'ELEMENT CBAR      TYPE =  34  BR GRIDS =       2 TOTAL =      54 etc.'
```

ELEMENT RECORDS (FORMATTED) -
There should be (TOTAL X LINES) records in each element type:

RECORD	WORD	CONTENT	FORMAT
3	1	Element ID.	I8
	2	Property ID. (Positive integer); or 0 (Element has no property nor material ID); or Material ID. (Element has no property ID, but it has a material ID)	I8
	3	0 (Not used; reserve for future use)	I8
	4-16	First 13 external connecting grid points	13I8
4		(IF NEEDED, and LINES in header record = 2)	
	1-15	Next 15 Grid points	8X,15I8
5		(IF NEEDED, and LINES in header record = 3)	
	1-15	More grid points	8X,15I8
:	:	Repeat element record 3 (and possible 4 and 5) as many times as there are elements of the same type.	

Repeat the header record and the element records as many times as there are different types of elements.

The end of element data records is signaled by an element ENDING record of the following form, 8 words:

Words 1 and 2 form the word ' -END-',
Word 4 holds the symbol '--',
and all other words are zeros

The ENDING ELEMENT RECORD of the FORMATTED tape looks like this:

```
--------++++++++--------++++++++--------++++++++--------++++++++---
'ELEMENT -END-    TYPE =   0  -- GRIDS =       0 TOTAL =       0 etc.'
```

If the OUGV file is present, the displacement vectors will be processed and the final results sent out to the OUTTP tape. (In this and the next few paragraphs, the word "displacement" implies also velocity, acceleration, load, grid point force, eigenvector, element stresses, and element forces.) The input OUGV file must be one of the GINO files described in the INPUT DATA BLOCKS section, which gives the displacements in the g-set or p-set, or the other data types. The output data are sorted by their external grid ID numbers. The displacement records in OUTTP also begin with an identification record:

RECORD	WORD	CONTENT (FORMATTED or UNFORMATTED)	FORMAT
1	1-2	'DISPLCNT--------' identification*. BCD	4A4
		(* or 'VELOCITY--------',	
		'ACCELERN--------',	
		'LOADINGS--------',	
		'G FORCES--------',	
		'EIGENVCR--------',	
		'E STRESS--------',	
		'E FORCES--------'')	

The original displacement data in NASTRAN are always in the global coordinate system. If the parameter BASIC is zero (default), the displacement vectors will be passed over to OUTTP without changes. However, if the parameter is set to +1, the displacement vectors will be converted to the basic rectangular coordinate system. In this latter case, the coordinate transformation matrices from CSTM will be brought into the computer, the grid point coordinate CID will be identified, and proper coordinate transformation will be applied to the displacements of each grid point. Again, the output OUTTP tape can be formatted or unformatted. In the unformatted tape, each grid point and its displacement values will form one logical record of 8 or 14 words (variable word length if element stresses or element forces). In the formatted tape, one logical record (8 words) is used if the displacement data is real, and an additional record (for data words 9 through 14) if the data is complex. In either case, a formatted record has 128-column of words. Similarly to the grid and element sets of data, a HEADER record is written out to OUTTP first before the grid point displacement vectors.

DISPLACEMENT HEADER RECORD for UNFORMATTED TAPE -

RECORD	WORD	CONTENT (UNFORMATTED)
2	1	Subcase or mode number. (Integer)
	2	Zero or frequency. (Real)
	3	Number of words per entry in next record.
	4-5	Original data file name, 2 BCD words
	6-7	' GLOBAL ' if BASIC=0, 2 BCD words
		' BASIC ' if BASIC=1
	8-13	CODE (See note below; 6 integers)
	14-45	Title, 32 BCD words
	46-77	Subtitle, 32 BCD words
	78-109	Label, 32 BCD words

Note - Each code word holds 8 digits. Therefore there are 48 digits, from CODE(1) through CODE(6), and from left to right, they describe the data type of the next displacement record:

1 for integer
2 for real, and
3 for BCD

The first digit points to the first data word; 2nd, 3rd, 4th, etc. point to 2nd, 3rd, 4th data words, etc.

DISPLACEMENT RECORDS in UNFORMATTED tape - in one long record:

RECORD	WORD	CONTENT (UNFORMATTED)
3	1	No. of words (excluding this first word) in this record. (Integer)
	2	External grid point number. (Integer)
	3	Point type (1=grid pt. 2=scalar pt. 3=extra pt. 4=modal pt., integer)
	4-9	Displacements. (Real parts, t1,t2,t3,r1,r2,r3, single precision real)
	10-15	(COMPLEX data only) Displacements. (Imaginary parts, t1,t2,t3,r1,r2,r3, single precision real)
	:	Repeat words 2 thru 9 (or 15) as many times as there are grid points in OUGV file
:	:	Repeat record 3 as many times as there are subcases or frequencies

DISPLACEMENT HEADER RECORD for FORMATTED tape -

RECORD	WORD	CONTENT (FORMATTED)	FORMAT
2	1-2	' CASE = ' or ' MODE = '	8 letters
	3	Subcase number	I8
	4	Zero or frequency	1PE12.5
	5-6	' WORDS ='	8 letters
	7	NWDS, number of words per entry in next record (=8 for REAL data, or =14 COMPLEX, for all displacement records)	I8
	8-9	' INPUT ='	8 letters
	10-11	Original GINO file name	2A4
	12-13	' COORD ='	8 letters
	14-15	' BASIC ' or 'GLOBAL '	2A4
	16-17	' CODE ='	8 letters
	18-22	Format code	5I8
		8 digits per word, 1 for INTEGER	
		2 for REAL	
		Ex. 13222200 3 for BCD	
		0 not applicable	
	23	NA4, number of words per entry in next record, in A4-word count	I8
3	1-32	Title, 32 BCD words	32A4
4	33-64	Subtitle, 32 BCD words	32A4
5	65-96	Label, 32 BCD words	32A4

DISPLACEMENT RECORDS in FORMATTED tape - in multiple short records:

RECORD	WORD	CONTENT	FORMAT
6	1	External grid point number. (Integer)	I8
	2	Point type (1=grid pt. 2=scalar pt. 3=extra pt. 4=modal pt., integer)	I8
	3-8	Displacements. (Real parts, t1,t2,t3,r1,r2,r3, single precision real)	6E12.6
7		(COMPLEX DATA only)	
	1-6	Displacements (Imaginary parts, t1,t2,t3,r1,r2,r3, single precision real)	16X,6E12.6
:	:	Repeat record 6 (records 6 and 7 if complex data) as many times as there are grid points displacement	

DIRECT MATRIX ABSTRACTION

At the end of each subcase, if the output tape OUTTP is formatted, a ZERO record (two records if data is complex) is written out to OUTTP tape. This ZERO record has the same format as a DISPLACEMENT record, and consists of 8 or 14 zeros (first two are integers, minus zeros). This ZERO record is not needed in the unformatted OUTTP output tape.

Repeat the HEADER record, the DISPLACEMENT records, and the ZERO record (formatted OUTTP tape only) as many times as there are subcases. At the end of the last subcase, or end of the input file OUGV, an ENDING record is written out. It has the same form as the HEADER record:

DISPLACEMENT ENDING RECORD -

RECORD	WORD	CONTENT (UNFORMATTED)
LAST	1	Zero. (Integer)
	2	Zero. (Real)
	3	Zero. (Integer)
	4-5	' -END-'. (BCD)
	6-101	96 Blank words. (BCD)

RECORD	WORD	CONTENT (FORMATTED)	FORMAT
LAST	1-2	' CASE = ' or ' MODE = '	8-LETTERS
	3	Minus 0 (Integer)	I8
	4	Zero	1PE12.5
	5-6	' WORDS ='	8-LETTERS
	7	Minus 0 (Integer)	I8
	8-11	' INPUT = -END- '	16-LETTERS
	12-17	Blanks	4A4
LAST+1	1-32	Blanks	32A4
LAST+2	1-32	BLANKS	32A4
LAST+3	1-32	Blanks	32A4

If OUGV is an element stress or an element force file, the stress or force data have variable length depending on the type of element. The stress or force records written to the OUTTAP tape are therefore different from those of the displacement records.

THE ELEMENT STRESS or FORCE RECORD HAS the following forms:

RECORD	WORD	CONTENT (UNFORMATTED)
3	1	Number of words, excluding this first word, in this record. (Integer)
	2-NWDS	Element ID, stress or force data (Variable data types are described in 'CODE')
	:	Repeat (2-NWDS) words as many times as there are elements.
:	:	Repeat record 3 as many times as there are subcases.

where NWDS is the number of computer words per entry, and CODE is the 6-word format code, as described in header record.

or

RECORD	WORD	CONTENT (FORMATTED)	FORMAT
6	1-NA4	Element ID, stress or force data (The data types are described in 'CODE'; all integers in 2A4, real numbers in 3A4, and BCD in A4)	33A4
:	:	(Maximum record length is 132 columns (33A4); continuation into next record(s) if necessary)	
:	:	Repeat above record(s) as many times as there are elements	

where NA4 is the number of words per entry in A4-word count, and CODE is 5-word format code

Notice that the DATABASE module does not copy out the external-internal grid points table in EQEXIN file, nor the coordinate transformation matrices in CSTM. The coordinate systems originally associated with the external grid points are never mentioned in the OUTTP tape.

If the user must copy the EQEXIN and CSTM files (both are in table forms), the new OUTPUT5 can be used.

X. DESIGN REQUIREMENT:

The DATABASE module is mapped in NASTRAN Links 2, 4 and 14. This module is accessible only through a NASTRAN DMAP Alter.

Minimum open core requirement = 10 x (total number of grid points) words.

The formatted outputs are flagged only by the parameter FORMAT. The formatted output records are designed not to exceed 132 columns in length and include printer carriage control. In most cases, I8-formats are used for integers and E12.6 for real data (no double precision words used); and BCD words are in multiples of 2A4. The entire OUTTP file can be printed, or it can be edited by a system editor. The formatted OUTTP file, if written on magnetic tape by a computer, can be used in another computer of a different manufacturer.

The unformatted OUTTP file is more efficient; and the integer and real data are more accurate. The grid point data and data of each connecting element type are written out unformatted in long records; that requires large working space in the computer system. On the other hand, only short records are written to the formatted OUTTP file, and the working space requirement is less critical.

XI. REMARKS:

1. Conversion of element stresses or forces to the basic coordinates is not allowed.

XII. DIAGNOSTIC MESSAGES:

Message numbers 3001, 3002, and 3008 may be issued by DATABASE.

APPENDIX J

(A) A NASTRAN EXAMPLE USING DATABASE MODULE (OUTPUT LISTING SHORTENED)

```
NASTRAN   TITLEOPT=-1, FILES=INP1


                             ****
                           *      *
                          *        *
                          * N A S T R A N
                          *        *
                           *      *
                             ****


      DEC VAX COMPUTER SYSTEMS          SYSTEM RELEASE -  1989 ED.
      FTN VERSION -  50K

                             DISTRIBUTED BY

      COMPUTER SOFTWARE MANAGEMENT AND INFORMATION CENTER (COSMIC)
                          UNIVERSITY OF GEORGIA
                          ATHENS, GEORGIA 30602
                          PHONE  (404) 542-3265


    N A S T R A N    E X E C U T I V E    C O N T R O L    D E C K    E C H O

ID        TEST, VAX MACHINE
SOL       1,0
APP       DISP
ALTER     106
DATABASE  EQEXIN,BGPDT,GEOM2,CSTM,OUGV1 //C,N,15/C,N,+1/C,N,+1  $
ENDALTER
TIME 10
CEND


                                  C A S E   C O N T R O L   D E C K   E C H O
             CARD
             COUNT
               1      TITLE    = TESTING DATABASE MODULE
               2      SUBTITLE = USING CYLINDRICAL COORDINATES
               3      LABEL    = GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED
               4      SPC   = 10
               5      DISP  = ALL
               6      OLOAD = ALL
               7      ECHO  = BOTH
               8      SUBCASE 123
               9      LOAD  = 1000
              10      SUBCASE 456
```

```
11       LOAD  = 2000
12       BEGIN BULK


                          I N P U T   B U L K   D A T A   D E C K   E C H O

             ---1--- +++2+++ ---3--- +++4+++ ---5--- +++6+++ ---7--- +++8+++ ---9--- +++10+++
             -FF-    CORD1C, 3  101  333 999
             -FF-    GRID,101,, 10.  0.  0.,0, 123456
             -FF-    GRID,111,, 20.  0.  0.,0, 123456
             -FF-    GRID,222,, 30.  0.  0.,0, 123456
             -FF-    GRID,333,, 40.  0.  0.,0, 123456
             -FF-    GRID,555,, 20.  0. -9.+9,0, 123456
             -FF-    GRID,999,, 10. 10. 10.,0, 123456
             -FF-    GRDSET, 7)3
             -FF-    GRID,  1,  3,  5.     0.  0.
             -FF-    =(6),*(1), =,  =, *(15.), ==
             -FF-    GRID, 11,  3,  5.,    0., 10.
             -FF-    =(6),*(1), =,  =, *(15.), ==
             -FF-    GRID, 21,  3,  5.,    0., 20.
             -FF-    =(6),*(1), =,  =, *(15.), ==
             -FF-    GRID, 32,  3,  5.,   15., 30.
             -FF-    =(4),*(1), =,  =, *(15.), ==
             -FF-    GRID, 31,  0, 40.   3.53553  3.53553  0
             -FF-    GRID, 37,  0, 40.  -3.53553  3.53553  0
             $
             -FF-    CBAR,  1,2, 1    2, 101
             -FF-    =(5),*(1),=, *(1),/,=
             -FF-    CBAR, 11,2, 11  12, 111
             -FF-    =(5),*(1),=, *(1),/,=
             -FF-    CBAR, 21,2, 21  22, 222
             -FF-    =(5),*(1),=, *(1),/,=
             -FF-    CBAR, 31,2, 31  32, 333
             -FF-    =(5),*(1),=, *(1),/,=
             -FF-    CBAR, 41,2,  1  11, 555
             -FF-    =(2),*(1),=, *(10),/,=
             -FF-    CBAR, 51,2,  7  17, 555
             -FF-    =(2),*(1),=, *(10),/,=
             $
             -FF-    CQUAD2,71, 7,  1  11  12   2
             -FF-    =(5),*(1), =,*(1),///
             -FF-    CQUAD2,81, 7, 11  21  22  12
             -FF-    =(5),*(1), =,*(1),///
             -FF-    CQUAD2,91, 7, 21  31  32  22
             -FF-    =(5),*(1), =,*(1),///
             $
             -FF-    PBAR,  2,100, .4  .5  .3  .3
             -FF-    PQUAD2,7,100, .05
             -FF-    MAT1,100  3.0E+7,, .3 1.0
             -FF-    SPC1,10,123456, 1 THRU 7
             -FF-    FORCE,1000,31,0,100.0,  0.0, 0., -1.0
             -FF-    FORCE,1000,37,0,100.0,  0.0, 0., -1.0
             -FF-    FORCE,2000,34,3,200.0, -1.0, 0.   0.0
             ENDDATA
         TOTAL COUNT=     46
```

```
                                 S O R T E D   B U L K   D A T A   E C H O
CARD
COUNT        ---1--- +++2+++ ---3--- +++4+++ ---5--- +++6+++ ---7--- +++8+++ ---9--- +++10+++
   1-        CBAR    1       2       1       2       101
   2-        CBAR    2       2       2       3       101
   :
   5-        CBAR    5       2       5       6       101
   6-        CBAR    6       2       6       7       101
   7-        CBAR    11      2       11      12      111
   8-        CBAR    12      2       12      13      111
   9-        CBAR    13      2       13      14      111
   :
  17-        CBAR    25      2       25      26      222
  18-        CBAR    26      2       26      27      222
  19-        CBAR    31      2       31      32      333
  20-        CBAR    32      2       32      33      333
   :
  24-        CBAR    36      2       36      37      333
  25-        CBAR    41      2       1       11      555
  26-        CBAR    42      2       11      21      555
  27-        CBAR    43      2       21      31      555
  28-        CBAR    51      2       7       17      555
  29-        CBAR    52      2       17      27      555
  30-        CBAR    53      2       27      37      555
  31-        CORD1C  3       101     333     999
  32-        CQUAD2  71      7       1       11      12      2
  33-        CQUAD2  72      7       2       12      13      3
   :
  37-        CQUAD2  76      7       6       16      17      7
  38-        CQUAD2  81      7       11      21      22      12
  39-        CQUAD2  82      7       12      22      23      13
   :
  43-        CQUAD2  86      7       16      26      27      17
  44-        CQUAD2  91      7       21      31      32      22
  45-        CQUAD2  92      7       22      32      33      23
   :
  48-        CQUAD2  95      7       25      35      36      26
  49-        CQUAD2  96      7       26      36      37      27
  50-        FORCE   1000    31      0       100.0   0.0     0.      -1.0
  51-        FORCE   1000    37      0       100.0   0.0     0.      -1.0
  52-        FORCE   2000    34      3       200.0   -1.0    0.      0.0
  53-        GRDSET                                          3
  54-        GRID    1       3       5.      0.      0.
  55-        GRID    2       3       5.      15.     0.
  56-        GRID    3       3       5.      30.     0.
  57-        GRID    4       3       5.      45.     0.
  58-        GRID    5       3       5.      60.     0.
  59-        GRID    6       3       5.      75.     0.
  60-        GRID    7       3       5.      90.     0.
  61-        GRID    11      3       5.      0.      10.
  62-        GRID    12      3       5.      15.     10.
  63-        GRID    13      3       5.      30.     10.
   :
  73-        GRID    26      3       5.      75.     20.
  74-        GRID    27      3       5.      90.     20.
```

```
75-         GRID    31      0       40.     3.53553 3.53553 0
76-         GRID    32      3       5.      15.     30.
77-         GRID    33      3       5.      30.     30.
78-         GRID    34      3       5.      45.     30.
79-         GRID    35      3       5.      60.     30.
80-         GRID    36      3       5.      75.     30.
8'          GRID    37      0       40.     -3.535533.53553 0
82-         GRID    101             10.     0.      0.      0       123456
83-         GRID    111             20.     0.      0.      0       123456
84-         GRID    222             30.     0.      0.      0       123456
 :
87-         GRID    999             10.     10.     10.     0       123456
88-         MAT1    100     3.0E+7          .3      1.0
89-         PBAR    2       100     .4      .5      .3      .3
90-         PQUAD2  7       100     .05
91-         SPC1    10      123456  1       THRU    7
            ENDDATA


*** USER INFORMATION MESSAGES FROM RESEQUENCING PROCESSOR - BANDIT     (CRI= 1,  MTH= 3,  MPC= 0,  DEP=-1,  PCH=-1)

                         BEFORE RESEQUENCING - - -
                                 BANDWIDTH            9
                                       :

                         AFTER RESEQUENCING BY GIBBS-POOLE-STOCKMEYER (GPS) ALGORITHM - - -
                                 BANDWIDTH            6
                                       :
                                 RMS WAVEFRONT    5.335


                                           *** BANDIT SUMMARY ***

                                                                   BEFORE     AFTER
                                 BANDWIDTH (B)                          9         6
                                 PROFILE (P)                          199       145
                                 MAXIMUM WAVEFRONT (C-MAX)              9         6
                                 AVERAGE WAVEFRONT (C-AVG)          7.107     5.179
                                 RMS WAVEFRONT (C-RMS)              7.500     5.335
                                 NUMBER OF GRID POINTS (N)                       34
                                 NUMBER OF ELEMENTS (NON-RIGID)                  48
                                 NUMBER OF RIGID ELEMENTS PROCESSED*              0
                                         :
                                 CRITERION*                          RMS WAVEFRONT
                                 METHOD USED*                                   GPS
                                 NO. OF NON-ACTIVE GRID POINTS                    6
                                 NO. OF SEQGP CARDS GENERATED                     9


                              S Y S T E M   G E N E R A T E D   S E Q G P   C A R D S

                      SEQGP         1       1       2       5       3       9       4      13
                      SEQGP         5      17       6      21       7      25      11       2
```

```
                         :
                      SEQGP        101       29      111       30      222       31      333       32
                      SEQGP        555       33      999       34
          **NO ERRORS FOUND - EXECUTE NASTRAN PROGRAM**


*** USER INFORMATION MESSAGE 3035
FOR SUBCASE NUMBER     1, EPSILON SUB E =  8.3046791E-13
*** USER INFORMATION MESSAGE 3035
FOR SUBCASE NUMBER     2, EPSILON SUB E = -2.6844558E-13


*** USER INFORMATION MESSAGE -
    DATABASE MODULE TRANSFERRED THE FOLLOWING  3 SETS OF DATA TO OUTPUT FILE  INP1   (FORTRAN UNIT 15),   FORMATTED

    1. GRID POINT DATA - EXTERNAL NUMBERS AND BASIC RECTANGULAR COORDINATES

    2. ELEMENT CONNECTIVITY DATA - ALL GRIDS POINTS ARE EXTERNAL NUMBERS

    3. DISPLCNT DATA FROM INPUT FILE OUGV1    , DATA CONVERTED TO BASIC RECT. COORDINATES,     2 SUBCASES


     TESTING DATABASE MODULE                                                  JANUARY  19, 1989    RELEASE  1989  VAX
     USING CYLINDRICAL COORDINATES
     GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED                                             SUBCASE 123


                                        D I S P L A C E M E N T   V E C T O R

     POINT ID.   TYPE          T1             T2             T3             R1             R2             R3
            1     G      0.0            0.0            0.0            0.0            0.0            0.0
            2     G      0.0            0.0            0.0            0.0            0.0            0.0
            :
            7     G      0.0            0.0            0.0            0.0            0.0            0.0
           11     G     -3.318057E-03  -3.292276E-03  -2.601234E-04   5.895234E-04  -6.161431E-04   4.702619E-06
           12     G     -4.063191E-03  -2.321371E-03   3.838165E-04   3.955249E-04  -7.332436E-04   4.287300E-06
           13     G     -4.531379E-03  -1.199775E-03   7.735121E-04   1.998269E-04  -7.991001E-04   2.540319E-06
            :
           17     G     -3.318057E-03   3.292276E-03  -2.601235E-04  -5.895234E-04  -6.161430E-04  -4.702721E-06
           21     G     -1.135364E-02  -1.133029E-02  -4.189800E-04   9.427494E-04  -9.638459E-04  -2.057090E-06
           22     G     -1.389689E-02  -8.006683E-03   6.217674E-04   6.450522E-04  -1.161740E-03  -1.690856E-06
            :
           36     G     -2.659377E-02   1.533314E-02   7.031241E-04  -7.324436E-04  -1.305977E-03   1.251280E-05
           37     G     -4.714102E-04   2.952591E-05  -3.071109E-02   1.414761E-05   1.510598E-03   1.724372E-05
          101     G      0.0            0.0            0.0            0.0            0.0            0.0
          111     G      0.0            0.0            0.0            0.0            0.0            0.0
            :
          999     G      0.0            0.0            0.0            0.0            0.0            0.0
```

```
TESTING DATABASE MODULE                                      JANUARY  19, 1989   RELEASE  1989  VAX
USING CYLINDRICAL COORDINATES
GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED                                SUBCASE 456

                                       D I S P L A C E M E N T   V E C T O R

POINT ID.   TYPE         T1             T2             T3             R1             R2             R3
        1     G      0.0            0.0            0.0            0.0            0.0            0.0
        2     G      0.0            0.0            0.0            0.0            0.0            0.0
        :
        7     G      0.0            0.0            0.0            0.0            0.0            0.0
       11     G     -3.318148E-03  -3.292200E-03  -2.601124E-04   5.894569E-04  -6.161953E-04   4.774461E-06
       12     G     -4.063299E-03  -2.321291E-03   3.838149E-04   3.955561E-04  -7.332786E-04   4.285285E-06
        :
       16     G     -4.063299E-03   2.321291E-03   3.838153E-04  -3.955560E-04  -7.332786E-04  -4.285384E-06
       17     G     -3.318147E-03   3.292200E-03  -2.601125E-04  -5.894569E-04  -6.161953E-04  -4.774563E-06
       21     G     -1.134697E-02  -1.132912E-02  -4.194220E-04   9.431769E-04  -9.614227E-04   1.676048E-06
       22     G     -1.389428E-02  -8.006698E-03   6.221989E-04   6.458150E-04  -1.160364E-03   9.538337E-07
        :
       27     G     -1.134696E-02   1.132912E-02  -4.194221E-04  -9.431769E-04  -9.614226E-04  -1.676340E-06
       31     G     -4.712943E-04   1.271940E-05  -3.068326E-02   2.592916E-05   1.508337E-03  -1.354597E-05
        :
       37     G     -4.712941E-04  -1.271908E-05  -3.068326E-02  -2.592964E-05   1.508337E-03   1.354600E-05
      101     G      0.0            0.0            0.0            0.0            0.0            0.0
        :
      555     G      0.0            0.0            0.0            0.0            0.0            0.0
      999     G      0.0            0.0            0.0            0.0            0.0            0.0


TESTING DATABASE MODULE                                      JANUARY  19, 1989   RELEASE  1989  VAX
USING CYLINDRICAL COORDINATES
GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED                                SUBCASE 123

                                             L O A D   V E C T O R

POINT ID.   TYPE         T1             T2             T3             R1             R2             R3
       31     G      0.0            0.0          -1.000000E+02  0.0            0.0            0.0
       37     G      0.0            0.0          -1.000000E+02  0.0            0.0            0.0


TESTING DATABASE MODULE                                      JANUARY  19, 1989   RELEASE  1989  VAX
USING CYLINDRICAL COORDINATES
GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED                                SUBCASE 456

                                             L O A D   V E C T O R

POINT ID.   TYPE         T1             T2             T3             R1             R2             R3
       34     G     -2.000000E+02  0.0            0.0            0.0            0.0            0.0


                                  * * * END OF JOB * * *
```

(B) FORMATTED INP1 FILE AS GENERATED FROM ABOVE NASTRAN RUN (LISTING SHORTENED)

```
GRID PTS--------
     34= TOTAL NUMBER OF GRID POINTS
      1        0 1.00000E+01 3.53553E+00 3.53553E+00
      2        0 1.00000E+01 2.50000E+00 4.33013E+00
      3        0 1.00000E+01 1.29410E+00 4.82963E+00
      4        0 1.00000E+01 0.00000E+00 5.00000E+00
      5        0 1.00000E+01-1.29410E+00 4.82963E+00
      6        0 1.00000E+01-2.50000E+00 4.33013E+00
      7        0 1.00000E+01-3.53553E+00 3.53553E+00
     11        0 2.00000E+01 3.53553E+00 3.53553E+00
     12        0 2.00000E+01 2.50000E+00 4.33013E+00
      :
     26        0 3.00000E+01-2.50000E+00 4.33013E+00
     27        0 3.00000E+01-3.53553E+00 3.53553E+00
     31        0 4.00000E+01 3.53553E+00 3.53553E+00
     32        0 4.00000E+01 2.50000E+00 4.33013E+00
      :
     37        0 4.00000E+01-3.53553E+00 3.53553E+00
    101        0 1.00000E+01 0.00000E+00 0.00000E+00
    111        0 2.00000E+01 0.00000E+00 0.00000E+00
      :
    555        0 2.00000E+01 0.00000E+00-9.00000E+09
    999        0 1.00000E+01 1.00000E+01 1.00000E+01
ELEMENTS--------
ELEMENT BAR       TYPE =  34  BR GRIDS =      2 TOTAL =      30 WDS/EL=      5 LINES =      1
      1       2       0       1       2
      2       2       0       2       3
      3       2       0       3       4
      4       2       0       4       5
      5       2       0       5       6
      6       2       0       6       7
     11       2       0      11      12
     12       2       0      12      13
      :
     25       2       0      25      26
     26       2       0      26      27
     31       2       0      31      32
     32       2       0      32      33
      :
     36       2       0      36      37
     41       2       0       1      11
     42       2       0      11      21
      :
     53       2       0      27      37
ELEMENT QUAD2     TYPE =  18  Q2 GRIDS =      4 TOTAL =      18 WDS/EL=      7 LINES =      1
     71       7       0       1      11      12       2
     72       7       0       2      12      13       3
     73       7       0       3      13      14       4
     74       7       0       4      14      15       5
     75       7       0       5      15      16       6
     76       7       0       6      16      17       7
     81       7       0      11      21      22      12
```

```
      82        7        0       12       22       23       13
       :
      95        7        0       25       35       36       26
      96        7        0       26       36       37       27
ELEMENT  -END-    TYPE =   0  -- GRIDS =       0 TOTAL =       0 WDS/EL=       0 LINES =       0
DISPLCNT--------
 CASE =      123 0.00000E+00 WORDS =       8 INPUT =OUGV1    COORD =  BASIC   CODE = 11222222
 TESTING DATABASE MODULE
 USING CYLINDRICAL COORDINATES
 GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED                                    SUBCASE 123
       1        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
       2        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
       :
       7        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
      11        1-2.60123E-04-1.82304E-05-4.67421E-03 4.70262E-06 8.52535E-04-1.88229E-05
      12        1 3.83816E-04-2.12288E-05-4.67951E-03 4.28730E-06 8.32770E-04-2.40871E-05
      13        1 7.73512E-04-1.39135E-05-4.68750E-03 2.54032E-06 8.23590E-04-1.38043E-05
       :
      17        1-2.60124E-04 1.82304E-05-4.67421E-03-4.70272E-06 8.52535E-04 1.88230E-05
      21        1-4.18980E-04-1.65105E-05-1.60400E-02-2.05709E-06 1.34817E-03-1.49174E-05
      22        1 6.21767E-04-1.44541E-05-1.60384E-02-1.69086E-06 1.32862E-03-2.22383E-05
       :
      36        1 7.03124E-04 1.79987E-05-3.06975E-02 1.25128E-05 1.49723E-03 1.86739E-05
      37        1-4.71410E-04 2.95259E-05-3.07111E-02 1.41476E-05 1.51060E-03 1.72437E-05
     101        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
     111        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
       :
     999        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
      -0       -0 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
CASE =      456 0.00000E+00 WORDS =       8 INPUT =OUGV1    COORD =  BASIC   CODE = 11222222
TESTING DATABASE MODULE
USING CYLINDRICAL COORDINATES
GRIDS, ELEMENTS, AND DISPLACEMENTS OUTPUT TO INP1, FORMATTED                                     SUBCASE 456
       1        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
       2        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
       :
       7        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
      11        1-2.60112E-04-1.83478E-05-4.67422E-03 4.77446E-06 8.52525E-04-1.89070E-05
      12        1 3.83815E-04-2.13520E-05-4.67957E-03 4.28529E-06 8.32816E-04-2.40777E-05
       :
      16        1 3.83815E-04 2.13524E-05-4.67957E-03-4.28538E-06 8.32816E-04 2.40777E-05
      17        1-2.60113E-04 1.83480E-05-4.67422E-03-4.77456E-06 8.52525E-04 1.89070E-05
      21        1-4.19422E-04-1.26166E-05-1.60344E-02 1.67605E-06 1.34676E-03-1.29017E-05
      22        1 6.22199E-04-1.31377E-05-1.60362E-02 9.53834E-07 1.32781E-03-2.08896E-05
       :
      27        1-4.19422E-04 1.26157E-05-1.60344E-02-1.67634E-06 1.34676E-03 1.29018E-05
      31        1-4.71294E-04 1.27194E-05-3.06833E-02 2.59292E-05 1.50834E-03-1.35460E-05
       :
      37        1-4.71294E-04-1.27191E-05-3.06833E-02-2.59296E-05 1.50834E-03 1.35460E-05
     101        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
       :
     555        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
     999        1 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
      -0       -0 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
CASE =        0 0.00000E+00 WORDS =       0 INPUT = -END-   COORD =           CODE =        0
```

(C) RDBASE - A FORTRAN PROGRAM TO READ UNFORMATTED OUTPUT FILE GENERATED BY DATABASE MODULE

```
      PROGRAM RDBASE
C
C     THIS FORTRAN PROGRAM READS THE UNFORMATTED OUTPUT FILE INP1
C     (FORTRAN UNIT 15) GENERATED BY DATABASE MODULE
C
C     (1) GRID POINTS DATA ARE READ AND SAVED IN GRID-ARRAY
C     (2) ELEMENTS DATA ARE READ AND SAVED IN ELM-ARRAY,
C         WITH ELEMENT NAMES AND POINTERS IN SAVE-ARRAY
C     (3) DISPLACEMENTS (VELOCITIES, ACCELERATIONS, LOADS, GRID-POINT
C         FORCE, OR EIGENVECTORS) DATA ARE READ AND SAVED IN DIS-ARRAY,
C         WITH SUBASES AND POINTERS IN SAVD-ARRAY
C
C     ANY OF ABOVE 3 SETS OF DATA NEED NOT EXIST IN ORIGINAL INP1 FILE
C
C     TO READ ELEMENT FORCES OR ELEMENT STRESSES, (3) ABOVE NEEDS SOME
C     CHANGES. PARTICULARLY WE NEED THE INFORMATION IN CODE TO GIVE US
C     THE TYPE OF EACH DATA WORD IN THE DATA LINE.
C     ASSUME CODE(1) = 11222222
C            CODE(2) = 31222000
C     THIS MEANS
C            THE 1ST, 2ND, AND 10TH DATA WORDS ARE INTEGERS;
C            9TH DATA WORD IS BCD; AND
C            3RD THRU 8TH, 11TH, 12TH AND 13TH WORDS ARE REAL NUMBERS
C
C
C     WRITTEN BY G.CHAN/UNISYS, JAN. 1989
C
      IMPLICIT INTEGER (A-Z)
      INTEGER          GRID(5,500),ELM(35,300),DIS(11200),SAVE(4,10),
     1                 SAVD(3,20),NAME(2),TITLE(32),SUBTTL(32),
     2                 LABL(32),CODE(6)
      REAL             GRIR(5,1),RIS(1),FREQ
      DOUBLE PRECISION GED,GD,EL,DS,ENDD,COORD
      EQUIVALENCE      (GRID(1),GRIR(1)),(DIS(1),RIS(1))
      DATA             INTAP, NOUT, MAXGRD, MAXELM, MAXDIS, MAXWDS    /
     1                 15,    6,    500,    300,    11200,  35        /
      DATA             GD,          EL,          DS,           END1   /
     1                 8HGRID PTS, 8HELEMENTS, 8HDISPLCNT,  4H -EN    /
C
      REWIND INTAP
C
C     READ DATA IDENTICATION RECORD
C
 70   READ (INTAP,END=400) GED
      IF (NOUT .EQ. 6) WRITE (NOUT,80) GED
 80   FORMAT (1X,A8,'--------')
      IF (GED .EQ. GD) GO TO 100
      IF (GED .EQ. EL) GO TO 200
      IF (GED .EQ. DS) GO TO 300
      STOP 'DATA TYPE UNKNOWN'
C
```

```
C     PROCESS GRID DATA
C     =================
C
C     READ GRID POINT DATA, ONE LONG RECORD OF MIXED INTEGERS AND REALS
C
  100 READ (INTAP,END=400) L,(GRID(J,1),J=1,L)
      IF (NOUT .NE. 6) GO TO 70
      NGRID = L/5
      IF (NGRID .GT. MAXGRD) STOP 'GRID DIMENSION TOO SMALL'
      WRITE (NOUT,120) NGRID
  120 FORMAT (1X,I8,'=TOTAL NO. OF GRID POINTS')
      DO 140 I=1,NGRID
      WRITE (NOUT,130) GRID(1,I),GRID(2,I),GRIR(3,I),GRIR(4,I),GRIR(5,I)
  130 FORMAT (1X,2I8,3(1PE12.5))
  140 CONTINUE
      GO TO 70
C
C     PROCESS ELEMENT DATA
C     ====================
C
  200 JS = 0
      JE = 0
C
C     READ ELEMENT HEADER RECORD, 8 WORDS
C
  210 READ (INTAP,END=400) NAME,TYPE,SYMBOL,GRIDS,TOTAL,WDS,LINE
      IF (NAME(1).EQ.END1 .AND. TYPE.EQ.0) GO TO 250
      IF (WDS .GT. MAXWDS) STOP 'ELM ROW DIMENSION TOO SMALL'
      IF (JE  .GT. MAXELM) STOP 'ELM COL DIMENSION TOO SMALL'
      JB = JE+1
      JE = JE+TOTAL
C
C     READ ELEMENT DATA, ONE LONG RECORD PER ELEMENT TYPE (ALL INTEGERS)
C
      READ (INTAP) ((ELM(I,J),I=1,WDS),J=JB,JE)
      JS = JS+1
      IF (JS .GE. 10) STOP 'SAVE DIMENSION TOO SMALL'
C
C     SAVE ELEMENT NAMES AND BEGINNING POINTERS IN SAVE-ARRAY
C     FOR EASY IDENTIFICATION
C
      SAVE(1,JS) = NAME(1)
      SAVE(2,JS) = NAME(2)
      SAVE(3,JS) = JB
      SAVE(4,JS) = WDS
      IF (NOUT .NE. 6) GO TO 210
      WRITE (NOUT,220) NAME,TYPE,SYMBOL,GRIDS,TOTAL,WDS,LINE
  220 FORMAT (1X,'ELEMNT =',2A4,'  TYPE =',I4,2X,A2,' GRIDS =',I8,
     1           ' TOTAL =',I8,' WDS/EL=',I8,      ' LINE  =',I8)
      DO 240 J=JB,JE
      WRITE (NOUT,230) (ELM(I,J),I=1,WDS)
  230 FORMAT (1X,3I8,13I8, /,(1X,8X,15I8))
  240 CONTINUE
      GO TO 210
C
```

```
C     WRAP UP SAVE-ARRAY
C
 250  JS = JS+1
      SAVE(1,JS) = END1
      SAVE(2,JS) = NAME(2)
      SAVE(3,JS) = JE+1
      SAVE(4,JS) = 0
      IF (NOUT .NE. 6) GO TO 70
      WRITE (NOUT,260)
      WRITE (NOUT,270) ((SAVE(I,J),I=1,4),J=1,JS)
 260  FORMAT (/30X,'THIS REFERENCE TABLE IS NOT PART OF INPUT FILE')
 270  FORMAT (40X,2A4,3H @ ,I4,',  WORDS=',I3)
      GO TO 70
C
C     PROCESS DISPLACEMENT DATA
C     ========================
C
C
 290  STOP 'ERROR IN READING DISPLACEMENT DATA'
C
 300  KB = 1
      KS = 0
C
C     READ DISPLACEMENT HEADER RECORD
C
 310  KS = KS+1
      IF (KS .GT. 20) STOP 'SAVD DEMINSION TOO SMALL'
      READ (INTAP,END=380) CASE,FREQ,NWDS,NAME,COORD,CODE,TITLE,SUBTTL,
     1                     LABEL
      IF (CASE+NWDS .EQ. 0) GO TO 380
      IF (NOUT .NE. 6) GO TO 330
      WRITE (NOUT,320) CASE,FREQ,NWDS,NAME,COORD,CODE(1),CODE(2),TITLE,
     1                 SUBTTL,LABEL
 320  FORMAT (' CASES =',I8,1PE12.5,' WORDS =',I8,' INPUT =',2A4,
     1        ' COORD =',A8,'  CODE = ',2I8, /,(1X,32A4))
C
C     DISPLACEMENT RECORS HAVE EITHER 8 OR 14 WORDS EACH DATA POINT
C     WITH CODE(1)=11222222, CODE(2) THRU (5) ARE ZEROS.
C
C     (IF THIS WERE TO READ ELEMENT STRESS RECORDS, THERE WOULD BE
C     NWDS DATA WORDS PER ELEMENT, AND THERE WOULD BE NWDS DIGITS IN
C     CODE INDICATING THE TYPE OF EACH DATA WORD. FIRST DIGIT (FROM
C     LEFT TO RIGHT) POINTS TO THE DATA TYPE OF FIRST DATA WORD,
C     2ND DIGIT TO 2ND DATA WORD, AND SO ON. SEE EXAMPLE AT THE
C     BEGINNING OF THIS SUBROUTINE)
C
 330  IF (NWDS.NE.8 .AND. NWDS.NE.14) STOP 'WORD COUNT ERROR'
      IF (CODE(1) .NE. 11222222) STOP 'FORMAT CODE ERROR'
C
C     SAVE SUBCASE NUMBER AND BEGINNING POINTERS IN SAVD-ARRAY
C     FOR EASY IDENTIFICATION
C
      KBM1 = KB-1
      SAVD(1,KS) = CASE
      SAVD(2,KS) = KB
      SAVD(3,KS) = NWDS
C
C     READ DISPLACEMENT RECORD, ONE LONG RECORD PER SUBCASE (OR FREQ.)
C     EACH GRID POINT DISPLACEMENT DATA IN EVERY 8 OR 14 WORDS,
C     2 INTEGERS + 6 (OR 12) REALS
C
 340  READ (INTAP,ERR=290) L,(DIS(I+KBM1),I=1,L)
      KE = L+KBM1
      DO 370 K=KB,KE,NWDS
      WRITE (NOUT,350)  DIS(K),DIS(K+1), (RIS(K+I),I=2, 7)
      IF (NWDS .EQ. 14) WRITE (NOUT,360) (RIS(K+I),I=8,13)
 350  FORMAT (1X,2I8,6(1PE12.5))
 360  FORMAT (1X,16X,6(1PE12.5))
 370  CONTINUE
      KB = KE+1
      GO TO 310
C
C     WRAP UP SAVD-ARRAY
C
 380  SAVD(1,KS) = 0
      SAVD(2,KS) = KE+1
      SAVD(3,KS) = 0
      IF (NOUT .NE. 6) GO TO 70
      WRITE (NOUT,260)
      WRITE (NOUT,390) (SAVD(1,K),SAVD(2,K),SAVD(3,K),K=1,KS)
 390  FORMAT (40X,'CASE',I8,3H @ ,I4,',  WORDS=',I4)
      GO TO 70
C
 400  REWIND INTAP
      END
```

APPENDIX K

USERS' MANUAL UPDATE PAGES FOR THE NEW RIGID ELEMENTS

Input Data Card CRROD Rigid Pin-Ended Rod

Description: Defines a pin-ended rod that is rigid in extension-compression.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRROD	EID	G1	G2	C1	C2	☒	☒	☒	
CRROD	14	1	2	2					

Field	Contents
EID	Element identification number (Integer > 0)
Gi	Identification numbers of connection grid points (Integers > 0)
Ci	Component number of one and only one dependent translational degree of freedom in the global coordinate system assigned to either G1 or G2. (Integer equals to 1, 2, or 3.) Either C1 or C2 must contain an integer and the other must be blank. See Remarks 2 and 3.

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. The grid point that associates with a blank Ci field, is designated as the reference independent grid point.

3. The dependent (that is constrained) degrees of freedom in a CRROD element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

4. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

5. Forces of constraint are not recovered.

6. Rigid elements are ignored in heat transfer problems.

7. The degree of freedom selected to be dependent must have a nonzero component along the axis of the rod.

8. Nastran actually converts the CRROD input card into the CRIGDR card format, and thus processes a CRROD card as if it were a CRIGDR card. The following table shows the conversion, in free-field format, of two possible cases:

Case	CRROD Card	Equivalent CRIGDR Card
1	CRROD, EID, G1, G2, C1,	CRIGDR, EID, G2, G1, C1
2	CRROD, EID, G1, G2, , C2	CRIGDR, EID, G1, G2, C2

9. See section 1.4.2.2 for a discussion of rigid elements.

BULK DATA DECK

Input Data Card CRBAR Rigid Bar

Description: Defines a rigid bar with six degrees of freedom at each end.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRBAR	EID	G1	G2	IC1	IC2	DC1	DC2	☒	
CRBAR	5	1	2	234	123				

Field	Contents
EID	Element identification number (Integer > 0)
Gi	Identification numbers of connection grid points (Integers > 0)
ICi	Independent degrees of freedom in the global coordinate system for the element at grid points Gi (any of the digits 1-6 with no imbedded blanks. Integers ⩾ 0 or blank.) See Remark 2.
DCi	Dependent degrees of freedom in the global coordinate system assigned by the element at grid points Gi (any of the digits 1-6 with no imbedded blanks. Integers ⩾ 0 or blank.) See Remarks 3 and 4.

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. The total number of degrees of freedom specified (IC1 and IC2) must equal six; for example, IC1 = 1236, IC2 = 34. Further, they should together be capable of representing any general rigid body motion of the element.

3. If both DC1 and DC2 are zero or blank, all of the degrees of freedom not in IC1 and IC2 will be made dependent.

4. The dependent (that is, constrained) degrees of freedom in a CRBAR element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

5. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

6. Forces of constraint are not recovered.

7. Rigid elements are ignored in heat transfer problems.

8. Nastran actually converts the CRBAR input card into the CRIGD3 card format, and thus processes a CRBAR card as if it were a CRIGD3 card. The following table shows the method of conversion, in free-field format:

```
CRBAR Card                    ===> Equivalent CRIGD3 Card
--------------------------------------------------------------
CRBAR, EID, G1, G2, IC1, IC2, DC1, DC2
                              ===> CRIGD3, EID, G1, IC1, G2, IC2
                                   ,"MSET", G1, DC1, G2, DC2
```

9. See Section 1.4.2.2 for a discussion of rigid elements.

Input Data Card CRTRPLT Rigid Triangular Plate

Description: Defines a rigid triangular plate.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRTRPLT	EID	G1	G2	G3	IC1	IC2	IC3	☒	abc
CRTRPLT	7	1	2	3	1236	3	3		ABC
+bc	DC1	DC2	DC3	☒	☒	☒	☒	☒	
+BC									

Field	Contents
EID	Element identification number (Integer > 0)
Gi	Identification numbers of the triangular plate grid points. (Integers > 0)
ICi	Independent degrees of freedom in the global coordinate system for the element at grid points Gi (any of the digits 1-6 with no imbedded blanks. Integers $\geq$ 0 or blank.) See Remark 2.
DCi	Dependent degrees of freedom in the global coordinate system (any of the digits 1-6 with no imbedded blanks. Integers $\geq$ 0 or blank.) See Remarks 3 and 4.

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. The total number of degrees of freedom specified for the reference grid points (IC1 IC2, and IC3) must be six; for example, IC1 = 1236, IC2 = 3, IC3 = 3. Further, they should together be capable of representing any general rigid body motion of the element.

3. If DC1, DC2, and DC3 are all zero or blank or if the continuation card is omitted, all of the degrees of freedom not in IC1, IC2, and IC3 will be made dependent.

4. The dependent (that is, constrained) degrees of freedom in a CRTRPLT element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly

implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

5. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

6. Forces of constraint are not recovered.

7. Rigid elements are ignored in heat transfer problems.

8. Nastran actually converts the CRTRPLT input card into the CRIGD3 card format, and thus processes a CRTRPLT card as if it were a CRIGD3 card. The following table shows the method of conversion, in free-field format:

```
CRTRPLT Card          ===> Equivalent CRIGD3 Card
-------------------------------------------------------------------
CRTRPLT, EID, G1,  G2, G3, IC1, IC2, IC3
,        DC1, DC2, DC3
                      ===> CRIGD3, EID, G1, IC1, G2, IC2, G3, IC3
                           ,"MSET", G1, DC1, G2, DC2, G3, DC3
```

9. See Section 1.4.2.2 for a discussion of rigid elements.

BULK DATA DECK

Input Data Card CRBE1 Rigid Body Element, Form 1

Description: Defines a rigid body connected to an arbitrary number of grid points.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRBE1	EID	IG1	IC1	IG2	IC2	IG3	IC3		abc
CRBE1	103	11	1	12	2	13	4		ABC
+bc		IG4	IC4	IG5	IC5	IG6	IC6		def
+BC		14	35	15	6				CDF
+ef	"UM"	DG1	DC1	DG2	DC2	DG3	DC3		ghi
+DF	UM	21	123	22	1	23	123456		EFI
+hi		DG4	DC4	DG5	DC5	etc.			
+HI		24	456	25	2				

Field Contents

EID Element identification number (Integer > 0)

IGi Identification numbers of the reference independent grid points (Integers > 0.)

ICi Independent degrees of freedom in the global coordinate system for the preceding reference grid point (any of the digits 1-6 with no imbedded blanks. Integer > 0.) See Remarks 2, 3, and 5.

"UM" BCD word that indicates the start of the data for dependent grid points.

DGi Identification numbers of the dependent grid points (Integer > 0).

DCi Dependent degrees of freedom in the global coordinate system for the preceding dependent grid point (any of the digits 1-6 with no imbedded blanks. Integer > 0.) See Remarks 4 and 5.

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. The total number of degrees of freedom specified for the reference grid points (IC1 through IC6) must be six; for example, IC1=1, IC2=2, IC3=4, IC4=3,5, IC5=6. Further, they should together be capable of representing any general rigid body motion of the element.

3. The first continuation card is not required if less than four reference independent grid points are specified.

4. The dependent (that is, constrained) degrees of freedom in a CRBE1 element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

5. A degree of freedom cannot be both independent and dependent for the same element. However, both independent and dependent components can exist at the same grid point.

6. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

7. Forces of constraint are not recovered.

8. Rigid elements are ignored in heat transfer problems.

9. Nastran actually converts the CRBE1 input card into the CRIGD3 card format by switching the "UM" BCD word to "MSET", and thus processes a CRBE1 card as if it were a CRIGD3 card.

```
CRBE1 Card           ===> Equivalent CRIGD3 Card
-----------------------------------------------------------------------
CRBE1, EID, IG1, IC1, IG2, IC2, IG3, IC3
,     "UM", DG1, DC1, IG2, DC2, etc.
                     ===> CRIGD3,  EID, IG1, IC1, IG2, IC2, IG3, IC3
                                ,    "MSET", DG1, DC1, DG2, DC2, etc.
```

10. See Section 1.4.2.2 for a discussion of rigid elements.

Input Data Card CRBE2 Rigid Body Element, Form 2

Description: Defines a rigid body whose independent degrees of freedom are specified at a single grid point and whose dependent degrees of freedom are specified at an arbitrary number of grid points.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRBE2	EID	IG	C	G1	G2	G3	G4	G5	abc
CRBE2	9	8	12	10	12	14	15	16	ABC
+bc	G6	G7	G8	etc.					
+BC	20								

Field Contents

EID Element identification number (Integer > 0)

IG Identification number of the reference grid point, to which all six independent degrees of freedom for the element are assigned (Integer > 0)

C The dependent degrees of freedom in the global coordinate system for all the dependent grid points Gi (any of the digits 1-6 with no imbedded blanks. Integer > 0.) See Remark 2.

Gi Identification numbers of the dependent grid points (Integers > 0)

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. The dependent (that is constrained) degrees of freedom in a CRBE2 element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

3. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

4. Forces of constraint are not recovered.

5. Rigid elements are ignored in heat transfer problems.

6. Nastran actually converts the CRBE2 input card into the CRIGD2 card format, and thus processes a CRBE2 card as if it were a CRIGD2 card. The following table shows the method of conversion, in free-field format:

```
CRBE2 Card           ===> Equivalent CRIGD2 Card
----------------------------------------------------------------------
CRBE2, EID, IG, C, G1, G2, G3, etc.
                      ===> CRIGD2, EID, IG, G1, C, G2, C, G3, C, etc.
```

7. See Section 1.4.2.2 for a discussion of rigid elements.

Input Data Card CRBE3 Rigid Body Element, Form 3

Description: Defines the motion at a "reference" grid point as the weighted average of the motions at a set of other grid points.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRBE3	EID	╳	IG	IC	W1	C1	G1,1	G1,2	abc
CRBE3	14		100	1234	1.0	123	1	3	ABC
+bc	G1,3	W2	C2	G2,1	G2,2	G2,3	W3	C3	def
+BC	5	4.7	1	2	4	6	5.2	2	DEF
+ef	G3,1	G3,2	G3,3	W4	C4	G4,1	G4,2	G4,3	ghi
+EF	7	8		5.1	1	15	16		GHI
+hi	"UM"	DG1	DC1	DG2	DC2	DG3	DC3	╳	jkl
+HI	UM	100	14	5	3	7	2		JKL
+kl	╳	DG4	DC4	DG5	DC5	DG6	DC6	╳	
+KL									

Field Contents

EID Element identification number (Integer > 0)

IG Reference grid point (Integer > 0)

IC Global components of motion whose values will be computed at the reference grid point (any of the digits 1-6 with no imbedded blanks. Integer > 0)

Wi Weighting factor for components of motion on the following card at grid points Gi,j (Real)

Ci Global components of motion which have weighting factor Wi at grid points Gi,j (any of the digits 1-6 with no imbedded blanks. Integers > 0)

Gi,j Grid point whose components Ci have weighting factor Wi in the averaging equations (Integers > 0)

"UM" BCD word that indicates the start of the data for the components of motion at grid points DGi (Optional). The default is that all of the component in IC at the referent grid point IG, and no others, are included in the dependent component set $\{u_m\}$

DGi Grid points with components DCi in $\{u_m\}$ (Integers > 0)

DCi Components of motion at grid point DGi (any of the digits 1-6 with no imbedded blanks, Integers > 0)

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. Blank spaces may be left at the end of a Gi,j sequence.

3. The default for UM should be used except in cases where the user wishes to include some or all IC components in displacement sets exclusive from the $\{u_m\}$ set. If the default is not used for UM:

   a. The total number of components in $\{u_m\}$ (that is, the total number of dependent degrees of freedom defined by the element) must be equal to the number of components in IC (four in the above example).

   b. The components in UM must be a subset of the components mentioned in IC and (Gi,j; Ci).

   c. The coefficient matrix $[R_m]$ in the constraints equation $[R_m]\{u_m\} + [R_n]\{u_n\} = 0$ must be nonsingular.

4. The dependent (that is constrained) degrees of freedom in a CRBE3 element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

5. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

6. Forces of constraint are not recovered.

7. Rigid elements are ignored in heat transfer problems.

8. Unlike the other rigid elements, this CRBE3 element and the CRSPLINE element cannot be converted into CRIGD2 or CRIGD3 elements. A Fortran subroutine (in single precision version and in double precision version) was written to handle these two special rigid elements.

NASTRAN DATA DECK

Input Data Card CRSPLINE

Description: Defines multipoint constraints for the interpolation of displacements at grid points

Format and Example:

1	2	3	4	5	6	7	8	9	10
CRSPLINE	EID	D/L	G1	G2	C2	G3	C3	G4	abc
CRSPLINE	73	.05	27	28	123456	29		30	ABC
+bc	C4	G5	C5	G6	etc.				
+BC	123	75	123	71					

Field Contents

EID Element identification number (Integer > 0)

D/L Ratio of the diameter of the elastic tube which the spline represents to the sum of the lengths of all segments. Default = 0.1 (Real > 0.)

Gi Identification number of the ith grid point (Integer > 0)

Ci Components to be constrained at the ith grid point (any of the digits 1-6 with no imbedded blanks, or blank) See Remark 3.

Remarks:

1. Element identification number must be unique with respect to all other element identification numbers.

2. Displacements are interpolated from the equations of an elastic beam passing through the grid points.

3. A blank entry in Ci indicates that all six degrees of freedom at Gi are independent. Since G1 must be independent, no field is provided for C1. Since the last grid point must also be independent, the last entry must be a Gi, not a Ci. For the example shown, G1, G3 and G6 are independent; G2 has six constrained degrees of freedom while G4 and G5 each have three.

4. The dependent (that is, constrained) degrees of freedom in a CRSPLINE element may not appear on OMIT, OMIT1, SPC, or SUPORT cards, nor may they be redundantly implied on ASET or ASET1 cards. They may not appear as dependent degrees of freedom in other rigid elements or on MPC cards. Degrees of freedom declared to be independent by a rigid element can be made dependent by another rigid element or by an MPC card.

5. Rigid elements, unlike MPC's, are not selected through the Case Control Deck.

6. Forces of constraint are not recovered.

7. Rigid elements are ignored in heat transfer problems.

8. Unlike the other rigid elements, this CRSPLINE element and the CRBE3 element cannot be converted into CRIGD2 or CRIGD3 elements. A Fortran subroutine (in single precision version and in double precision version) was written to handle these two special rigid elements.