Installation and Run Manual (1.0)

For the

MYSTRAN General Purpose Finite Element Structural Analysis Computer Program

(September 2022)

Table of Contents

1   Quick Start Guide

Listed below are 3 options for running MYSTRAN. The first is the easiest and quickest way to get MYSTRAN to run. The third option is the most flexible.

OPTION 1 – SIMPLEST

Conditions:
MYSTRAN.exe and the input file (*filename.ext*) exist in the same directory
Set a command prompt to the directory where MYSTRAN.exe and the input file exist.

Run:
Command line (general input files including BDF and DAT files):
mystran *filename.ext*

Command line (allowed if input file extension is DAT)
mystran *filename*

Output:
The output files will be in the directory where mystran.exe and input file are located.

OPTION 2 – A LITTLE MORE FLEXIBLE

Conditions:
MYSTRAN.exe need not be in the same directory as the input file. Say it is in generic path Anypath1.
The command prompt can be set to any directory.
The input file exists in a generic path such as "C:\Anypath2".

Run:
Command line (general input files including BDF and DAT files):
"C:\Anypath1\mystran.exe" C:\Anypath2*filename.ext*
If the input file extension is DAT. It need not be entered on the command line

This could also be run from a batch file, say MYSTRAN.bat, the 1 line of which is shown below:
"C:\Anypath1\mystran.exe" %1

Command line for running the MYSTRAN.bat file:
MYSTRAN *filename.ext*
Note: %1 is a wildcard in the bat file and is replaced by *filename.ext.* when the command is executed (if not entered MYSTRAN will prompt for it). If the batch file is executed from a folder different then where the input file exists then *filename.ext* must contain the relative path to the input file (ot the path must be in the bat file). Note: quote marks, above, are only needed if C:\Anypath1 contains blank spaces (e.g. C:\Anypath 1)

Output:
The output files will be in the same directory as the input.

OPTION 3 – MOST FLEXIBLE: ENVIRONMENT VARIABLES AND INI FILE

Section 4 has the details of this option but a few words about it here are in order. In Windows, the user can set up an environment variable called MYSTRAN_directory which gives the location in which MYSTRAN.exe is located. Then, using the MYSTRAN initialization file (MYSTRAN.INI), options such as changing the default extension (from the default dat) to something like bdf could be done. The INI file can also be used to accomplish several other tasks as described in Section 4..

Conditions:
MYSTRAN.exe and the input file can be anywhere. No bat file is needed.

Run:
Command line (file extension not needed if it is dat or is set to something else like bdf in the INI file):
MYSTRAN *filename*
Note that if the input is not in the same folder as the one in which the MYSTRAN command was executed then *filename* must contain the full path as well

Output:
The output will be in the same folder as the input.

2 Command window messages

When you ran the example problem in the command window (or terminal) you should have seen messages such as:

>> MYSTRAN BEGIN:

>> LINK i BEGIN
.
.
.
>> LINK i END

>> MYSTRAN END:

with several lines in between indicating the progress as MYSTRAN solves your problem. The MYSTRAN program is divided into several major subroutines (called LINK's) that solve the problem. Below is a list of these LINK's and what they do:

- ❖ LINK0:

  - o Read input data deck and check for errors and possible restart

  - o Process grid and coordinate system input data

  - o Process Case Control output requests

  - o Forms degree of freedom (DOF) tables

  - o Process concentrated mass input data

  - o Calculates rigid body mass properties (Grid Point Weight Generator)

  - o Process temperature and pressure load input data intp arrays needed for element load calculations

- ❖ LINK1:

  - o Process MPC's and rigid elements into sparse G-set[1] array RMG

  - o Process all applied forces (including grid forces and moments, gravity, pressure, thermal, centrifugal, scalar) into sparse G-set load array PG

  - o Formulate the G-set sparse stiffness and mass arrays KGG and MGG

  - o Formulate the G-set sparse differential stiffness array KGGD

- ❖ LINK2:

  - o Reduce the G-set stiffness, mass, load and constraint matrices to the L-set

---

[1] See Section 3.6 of the MYSTRAN Users Reference manual for a discussion of displacement set notation (which is the same as, or a subset of, that used in the NASTRAN program)

- ❖ LINK3 (for statics problems only):

    - o Solve for the L-set displacements

- ❖ LINK4 (for eigenvalue problems only):

    - o Solve for the L-set eigenvalues and eigenvectors

- ❖ LINK5:

    - o Build the A-set displacements back up to the G-set through use of the constraint matrices

- ❖ LINK6 (for Craig-Bampton model generation only):

    - o Builds a Craig-Bampton model from the input physical model

- ❖ LINK9:

    - o Use the G-set displacements from LINK5 to solve for the outputs requested in Case Control.

If the job executes without error the last message from MYSTRAN (after >>>>> LINK 9 END) is:

```
 >> MYSTRAN END  :  1/19/2006 at 15: 5: 3.8. The output file is:
    filename.F06
MYSTRAN terminated normally. Total CPU time =  1.56E-01 seconds
```

where *filename* is the name of the input file

3   MYSTRAN files

MYSTRAN produces several output files besides the F06 file (which contains the output of the answers to the problem). If *filename*.dat is the input file name, the output files are all *filename.ext* where *ext* is the extension of the output file. Each of the output files is discussed below under the heading of the extension for that file.

3.1   F06 text output file

The F06 output file has the answers for the problem submitted to MYSTRAN. It is a text file that can contain an echo of the input, messages that MYSTRAN outputs (warning, fatal error and information messages) as well as the outputs that were requested in Case Control. Appendix A of the MYSTRAN Users Reference Manual shows the F06 output file for example problem: 1 EXAMPLE1.F06).

3.2   F04 text output file

The F04, or log, output file can contain a list of the subroutines run along with the subroutine begin and end times. This text file is only produced if explicitly requested. That request must be made using the MYSTRAN initialization file discussed in a later section.

3.3   ERR text output file

A list of all warning and fatal error messages is printed in the F06 file as discussed above. However, since warning messages may be suppressed in the F06 file via the Bulk Data PARAM SUPMSG entry, a separate list of all warning and error messages is put into the ERR text file as well. This file only contains the warning and fatal error messages.

3.4   BUG text output file

If element debug information is requested in Case Control via the ELDATA command, a text file with extension BUG is created. The contents of that file are enumerated in the MYSTRAN Users Reference Manual under the Case Control ELDATA command discussion.

3.5   SEQ text output file

If Bulk data PARAM GRIDSEQ is set to BANDIT, then the SEQGP card images created by the automatic grid sequencer can be output to this file (see Bulk data PARAM entry for parameter GRIDSEQ).

3.6   SPC text output file

If Bulk data PARAM AUTOSPC = Y, then a file containing the SPC1 Bulk Data card images of the SPC's for the AUTOSPC'd DOF can be obtained by using a Bulk Data PARAM entry with a parameter name of PCHSPC1 whose value is Y.

3.7 ANS text output file

This file is generally only useful by the author in the checkout of test problem answers but it is described here for the sake of completeness. It contains all of the answers generated in LINK9, the same as the F06 text file, but none of the F06 info written prior to LINK9. It is only generated if a Bulk Data DEBUG entry is present (see User's Reference Manual for a list of all DEBUG entries).

3.8 PCH text output file

The PCH text file can contain some of the outputs in MYSTRAN (see section 6.3 of the MYSTRAN User's Reference Manual for details of which outputs are allowed)

3.9 L1A text output file

The L1A text file, as well as the L*ij* binary files (discussed below), are used to communicate data between the LINK's. The L1A file contains data needed to read the binary L*ij* files. Generally these files are deleted when MYSTRAN completes its execution, but they may be saved via the MYSTRAN initialization file, discussed in a later section

3.10 NEU text output file

The NEU file is created if there is a Bulk Data PARAM entry for parameter POST with a value of -1. This file is used by FEMAP for the post-processing of outputs (displacements, forces, stresses, etc) from MYSTRAN.

3.11 Lij binary output files

In addition to the above mentioned text files that contain output data, MYSTRAN also produces some files that are used to communicate data between the several LINK's of the program. These files all have an extension of three characters length beginning with L followed by a number and a letter (e.g. L2A). The contents of these files, and the details of the way data is written to them, are explained in Appendix A of this manual. Generally these files are deleted when MYSTRAN completes its execution, but they may be saved via the MYSTRAN initialization file, discussed in a later section.

3.12 F2*j* binary output files

The Case Control ELDATA entry can request output of several items to unformatted binary files in addition to printing of these items to the BUG file. See Appendix A to this manual for the fortran code to read these files.

5

4   Modifying the quick start setup

The quick start setup described in Section 1 was the easiest, but perhaps not the most convenient, way to run MYSTRAN. This section discusses moving the MYSTRAN.EXE program to some other directory. It also discusses the MYSTRAN initialization file which can be used to change from some of the default values built into MYSTRAN.

4.1   Changing directories where MYSTRAN executable resides

4.1.1   Windows version

The MYSTRAN executable can figure out what folder it is in through use of a Windows environment variable. In Windows 10 open Settings and, under "Find a setting" type "environment variables" and click on "Edit environment variables for your account" and create a new environment variable called "MYSTRAN_directory" and give it a value of the folder where you have put the MYSTRAN.EXE program. You can then run MYSTRAN from a command window by changing to the directory where your input file (e.g. dat file) is located and typing MYSTRAN followed by the input file name (e.g. EXAMPLE1). If you do not give the input file name, MYSTRAN will prompt you for the name.

4.1.2   Linux version

If it is desired to have more flexibility in choosing the directory where the executable resides (i.e. not in the same directory where the input dat file exists), the following must be done (let *mystran_folder* be the folder where you want the files mystran.exe and MYSTRAN.INI to be):

- In the user's ".bashrc" file add the following 4 lines (the file set_MYSTRAN_directory.sh, below, was one of the files in the tar file downloaded from the MYSTRAN web site. File set_MYSTRAN_directory.sh is to be put into *mystran_folder* along with mystran.exe and MYSTRAN.INI

:

> # Run script to set MYSTRAN_directory
> source *mystran_folder*/set_MYSTRAN_directory.sh
>
> # Environment variables
> MYSTRAN_directory=*mystran_folder*

- In the user's ".bash_profile" file add a line to include the directory where the mystran.exe and MYSTRAN.INI files reside (i.e. *mystran _folder*):

> PATH=$PATH:*mystran _folder*

4.2   The MYSTRAN initialization file

When MYSTRAN starts, it looks for an initialization file (MYSTRAN.INI) in the same directory that the MYSTRAN.EXE executable file is located. If this file does not exist, default values enumerated below will be used. The initialization file is a text file that is used to:

- Change the directory where MYSTRAN looks for the input files. The output files are put into this directory also. The default is that these files must be in the directory where the input dat file resides. Note, this is only needed if you did not create the environment variable "MYSTRAN directory" discussed in paragraph 4.1.2, and you are not in the directory of the dat file when you type MYSTRAN in the command window

- Change the default extension for input files. The default is dat

- Set the level of detail of subroutine begin/end times that is written to the log file. The default is no log file is produced.

- Save L*ij* files. The default is that no L*ij* files will be saved.

If any of these are to be changed from their default values, a MYSTRAN.INI file must be created. If there is no MYSTRAN.INI file, the default values discussed below will be used. MYSTRAN.INI is a text file containing commands, discussed below, that can change the default values.

Any line in the initialization file that begins with a $ or that is a blank line is ignored.

4.2.1 Changing directories where the input and output files reside

Note: this is only needed if you did not create the environment variable "MYSTRAN_directory" discussed in paragraph 4.1.2.

The default for where MYSTRAN looks for the input file and where it puts the output files is the directory where you currently are when you run MYSTRAN. This can be changed with the DEF DIR command in the MYSTRAN.INI file. Say, for example, you want the input and output files to be in C:\RUNS\\PROJECT1 but you do not want to have to change to that directory to run MYSTRAN. The MYSTRAN.INI text file in order to accomplish this would contain the line:

```
DEF DIR C:\RUNS\PROJECT1
```

where DEF DIR must be in columns 1 through 7 of the line in MYSTRAN.INI and the default directory, C:\RUNS\PROJECT1 (where you want the input and output files) must begin no sooner than column 9 of that line. With this default directory defined, you only need to enter, at the command prompt when you execute MYSTRAN, the file name itself (EXAMPLE1 for the example problem). MYSTRAN will go to the defined default directory and look for the input data file and will put the output files in that directory also.

4.2.2 Changing the default extension for input files

The default extension for input files is dat which can be changed with the command:

```
DEF EXT ext
```

where DEF EXT must be in columns 1 through 7 and *ext* is a character name that must be in column 9 through column 16 and can only consist of 1 to 3 characters and or numbers that are valid for file names. If the extension for a job is not the same as the default extension (either the built in one of dat or one defined with `DEF EXT`) it must be entered with the file name when MYSTRAN is executed. For example, suppose that you have changed the default extension to D and you want to run EXAMPLE1 that has an extension that is INP. You can either modify the default extension to be INP in the initialization file or enter it with the file name on the command line when MYSTRAN is run (EXAMPLE1.INP).

4.2.3   Setting the level of detail for the log file

The initialization file can also change the level of detail that the log file has. This file contains the start and end times of subroutines. The variable that controls this level of detail is called WRT_LOG and has a default value of 0, which results in no log file at all. Values of WRT_LOG greater than 0 will result in a log file being written. To obtain a log file showing subroutine begin and end times, put the following line into the MYSTRAN.INI file:

```
WRT_LOG  n
```

where n is an integer > 0.   WRT_LOG must begin in column 1 and n can be entered anywhere from column 9 thru column 16. The value n=1 will write begin/end times for the highest level of subroutine calls (LINK's). Successively larger values of n will print more detail. Almost all of the subroutines called in an execution can be shown in the log file by entering a high number for n. A value of n=11 gives about as much detail as can be obtained, but if you want to make sure you get the most detail, enter a higher number (e.g. 99).

4.2.4   Saving the L*ij* files

The section on MYSTRAN files, and Appendix A, discuss the files used to communicate data between LINK's. These files will be deleted upon normal termination of a MYSTRAN execution unless you specify that they be saved. In some circumstances, you may want to save these. For example, the stiffness matrix for the G and A-sets are written to two of these files. You can either specify that all of these files be saved, or you can specify individual ones be saved. The following two sections discuss how to do this.

4.2.4.1   Saving all L*ij* files

To save all of the L*ij* files, enter the following line in the MYSTRAN initialization file:

```
ALLFILES KEEP
```

where ALLFILES must begin in column 1 and KEEP must be in column 9-16 .

4.2.4.2   Saving an individual L*ij* file

To save one of the L*ij* files, enter the following line in the MYSTRAN initialization file:

```
Lij     KEEP
```

where L*ij* is the "name" of a file (e.g. L2B – see appendix A for a description of all Lij files). L*ij* must begin in column 1 and KEEP must be in column 17-24.

5 MYSTRAN messages

MYSTRAN writes error, warning and information messages to the F06 file. The error and warning messages are also written to the ERR file.

5.1 Error messages

There are approximately 600 occurrences of more than 300 different fatal error messages that MYSTRAN has. Eventually, these cause MYSTRAN to abort in an orderly fashion with the following written to the F06 and ERR files:

```
*ERROR  nnnn: error message
```

where nnnn is a 3 or 4 digit error number and *error message* is a description of what MYSTRAN found to be a fatal error. Sometimes, several fatal error messages will be written prior to MYSTRAN aborting the execution. A specific example of a fatal error message is:

```
*ERROR  1900: GRID xxxx ON ELEMENT yyyy TYPE type NOT  DEFINED
```

where xxxx is a grid point number and yyyy is an element number and *type* is the element type (for example BAR).

5.2 Warning messages

There are approximately 60 warning messages that MYSTRAN has. None of these are fatal. The format of these is:

```
*WARNING    : warning message
```

Warning messages have no message number. An example of the type of warning message that can be written is when MYSTRAN reads an entry from the input data deck but does not recognize the entry. A specific example of a warning message is:

```
*WARNING    : ERROR READING SET ID ON CASE CONTROL SET CARD. CARD IGNORED
```

5.3 Information messages

There are approximately 100 information messages that MYSTRAN has. The format of these is:

```
*INFORMATION: information message
```

Information messages are written to tell various items of interest as MYSTRAN is executing. A specific example of an information message is:

```
*INFORMATION: GRID POINT xxxx DOF(s)yyyy ARE SINGULAR
```

where xxxx is a grid point number and yyyy are the displacement components at the grid that are singular.

6 Appendix A: Description of Lij and F2j files

6.1 L*ij* Files

Many of the L*ij* files are of little use other than communicating data between the various LINK's of MYSTRAN. However, several contain data that may be of interest to the user. These particular files are explained below[2]. In each of the file explanations, the data sets are explained and the fortran code that can be used to read the file is given. In this manner, the user can create fortran programs to read the data from each file. Each of the files has a name of *filename*.L*ij* where filename is the input data deck name for the problem MYSTRAN has run and L*ij* is the extension for the unformatted files discussed below. All integer numbers are 4 byte and all real numbers are 8 bytes.

All Lij unformatted files have as its first record a time stamp, STIME, that is also the first record of all of the formatted output files (e.g. F06 file). STIME is only written once, at the very beginning of file.

6.1.1 L1B: Grid Data

This file contains grid, coordinate system, and grid sequence data. The grid data is written first, followed by the coordinate system data and then the grid sequence data.

The grid data is contained in arrays GRID and RGRID. GRID is an array of integer data that was input on the GRID Bulk Data entry for each grid point. RGRID is a real array of the <u>basic</u> coordinates of the grids (X, Y, Z). Arrays GRID and RGRID are explained below:

- GRID(I,1) is the number of grid point I

- GRID(I,2) is the number of the coordinate system in which grid I coordinates are defined

- GRID(I,3) is the number of the global coordinate system for grid I

- GRID(I,4) are the integers (1-6) that define the permanent single point constraints for grid I

- GRID(I,5) is the value of the number of line breaks in the F06 file to be placed after this grid. This is usually 0 but can be specified otherwise in field 10 of the GRID Bulk data entry)

- GRID(I,6) is 1 if this entry in array GRID is for an SPOINT or 6 if for an actual GRID

- RGRID(I,1) is the X coordinate of grid I in the basic coordinate system

- RGRID(I,2) is the Y coordinate of grid I in the basic coordinate system

- RGRID(I,3) is the Z coordinate of grid I in the basic coordinate system

The fortran code that can be used to read the grid data from the L1B file is:

---

[2] All integer data are 4 byte words and all real data are 8 byte words

```
DATA_SET_NAME = 'GRID, RGRID'
READ(L1B) DATA_SET_NAME
READ(L1B) NGRID
DO I=1,NGRID
   DO J=1,MGRID
      READ(L1B) GRID(I,J)
   ENDDO
   DO J=1,MRGRID
      READ(L1B) RGRID(I,J)
   ENDDO
ENDDO
```

where NGRID is the number of grids.   MGRID, MRGRID are defined in Table 6-1.

The coordinate system data is contained in arrays CORD and RCORD. CORD is an array of integer data for the coordinate systems describing the coordinate type, the coordinate system number (CID) and the reference coordinate system number (RID). When the coordinate system data was input on CORD2R, CORD2C and/or CORD2S Bulk Data entries, RID was the number of the coordinate system in which CID was being defined. However, when file L1B has been written, all coordinate systems have been transformed such that RID is 0 (basic).

RCORD is an array of real coordinate system data defining the origin of CID in basic coordinates and the 3 x 3 coordinate transformation matrix which will transform a vector in CID to a vector in the basic coordinate system. Arrays CORD and RCORD are explained below:

- CORD(I,1) is a description of the type of coordinate system
      11 for CORD1R
      12 for CORD1C
      13 for CORD1S
      21 for CORD2R
      22 for CORD2C
      23 for CORD2S

- CORD(I,2) is the number of the coordinate system I (CID)

- CORD(I,3) , CORD(I,4) and CORD(I,5) are all zero (the basic coordinate system number)

- RCORD(I,1) is the basic X coordinate of the defining rectangular system for CID

- RCORD(I,2) is the basic Y coordinate of the defining rectangular system for CID

- RCORD(I,3) is the basic Z coordinate of the defining rectangular system for CID

- RCORD(I,4) through RCORD(I,6) is the 1st row of the coordinate transformation described above

- RCORD(I,7) through RCORD(I,9) is the 2nd row of the coordinate transformation described above

- RCORD(I,10) through RCORD(I,12) is the 3rd row of the coordinate transformation described above

The fortran code that can be used to read this coordinate system data from the L1B file is:

```
DATA_SET_NAME = 'COORDINATE SYSTEM DATA'
READ(L1B) DATA_SET_NAME
READ(L1B) NCORD
DO I=1,NCORD
   DO J=1,MCORD
      READ(L1B) CORD(I,J)
   ENDDO
   DO J=1,MRCORD
      READ(L1B) RCORD(I,J)
   ENDDO
ENDDO
```

where NCORD is the number of coordinate systems. MCORD and MRCORD are defined in Table 6-1.

The grid sequence data is contained in integer array GRID_SEQ and real arrays SEQ1 and SEQ2. Thjis would generally not be of interest, but will be explained since it is data in file L1B. Integer array GRID_SEQ, and real arrays SEQ1 and SEQ2 are written to file L1B. GRID_SEQ(I) is the grid sequence number for grid I. SEQ1 and SEQ2 are the sequence values from the Bulk Data entries SEQGP. These arrays are explained below:

- GRID_SEQ(I) is the number sequence order for grid I. If there are no Bulk Data SEQGP entries then this will be either:

  - Grid input numerical order if Bulk Data entry PARAM SEQUENCE = 2

  - Grid input input order if Bulk Data entry PARAM SEQUENCE = 3

- SEQ1(I) are the grid ID's input on Bulk Data SEQGP entries (if any are input)

- SEQ2(I) are the grid sequence numbers for SEQ1(I) grids input on Bulk Data SEQGP entries

The fortran code that can be used to read this grid sequence data from the L1B file is:

```
DATA_SET_NAME = 'GRID_SEQ'
READ(L1B) DATA_SET_NAME
READ(L1B) NGRID
DO I=1,NGRID
   READ(L1B) GRID_SEQ(I)
ENDDO

DATA_SET_NAME = 'SEQ1, SEQ2'
READ(L1B) DATA_SET_NAME
READ(L1B) NSEQ
DO I=1,NSEQ
   READ(L1B) SEQ1(I),SEQ2(I)
ENDDO
```

Where NGRID is the number of grid points in the model and NSEQ is the number of Bulk Data SEQGP entries in the data deck.

6.1.2 L1C: Degree of freedom tables

This file contains the degree of freedom tables TSET, TDOFI and TDOF written to file L1C in that order. For a discussion of the major displacement sets, see the MYSTRAN Users Reference Manual.

Integer array TSET contains 1 row and 6 columns for each grid point. The array contains a 1 or 2 byte character representation of the degree of freedom set that each of the 6 degrees of freedom for the grid belongs. Only the mutually exclusive degree of freedom sets (M, S, O, A) are given in this table:

- M for a degree of freedom that is constrained via a rigid element or multi-point constraint

- S for a degree of freedom that is single point constrained. This is further subdivided into:

  - SZ for a degree of freedom single point constrained to zero displacement. This is further subdivided into

    - SA for a degree of freedom single point constrained by AUTOSPC

    - SG for a degree of freedom single point constrained via the PSPC field on a GRID Bulk Data entry

    - SB for a degree of freedom single point constrained to zero displacement on a SPC or SPC1 Bulk Data entry

  - SE for a degree of freedom that has nonzero enforced displacement via Bulk Data SPC entry

- O for a degree of freedom that is to belong to the omit set[3]

- A for a degree of freedom that is to belong to the analysis set[4]

Integer arrays TDOF and TDOFI have 6 rows and MTDOF columns for each grid point. See Table 6-1 for MTDOF. The arrays contain the actual degree of freedom numbers for all of the displacement sets (not just the mutually exclusive sets). The information in arrays TDOF and TDOFI is the same; however TDOF is sorted in grid numerical order while TDOFI is sorted in internal grid order. The 6 rows for each grid point represent the 6 displacement components (3 translations and 3 rotations) per grid. The MTDOF columns of the arrays are:

- Col 01: Actual grid point number

- Col 02 :Displacement component number (1, 2, 3, 4, 5 or 6) for the actual grid

- Col 03: Internal grid point number (consecutive integers from 1 to NGRID)

- Col 04 : Displacement component number (1, 2, 3, 4, 5 or 6) for the internal grid

- Col 05 :G-set degree of freedom number

- Col 06 :M-set degree of freedom number

---

[3] See Bulk Data entries OMIT and OMIT1 and for a description of how degrees of freedom are placed in the O set
[4] See Bulk Data entries ASET and ASET1 and for a description of how degrees of freedom are placed in the A set

- Col 07 :N-set degree of freedom number

- Col 08 :N-set degree of freedom number

- Col 09 :SA-set degree of freedom number

- Col 10 :SG-set degree of freedom number

- Col 11: SZ-set degree of freedom number

- Col 12: SE-set degree of freedom number

- Col 13: S-set degree of freedom number

- Col 14: F-set degree of freedom number

- Col 15: O-set degree of freedom number

- Col 16: A-set degree of freedom number

- Col 17: R-set degree of freedom number

- Col 18: L-set degree of freedom number

Note that all degrees of freedom belong to the G-set and that an A-set degree of freedom also belongs to the F and N-sets prior to the reduction to the A-set.

The fortran code that can be used to read these arrays data from the L1C file is:

```
      DATA_SET_NAME = 'TSET'
      READ(L1C) STIME
      READ(L1C) DATA_SET_NAME
      READ(L1C) NGRID
      DO I = 1,NGRID
         DO J = 1,6
            READ(L1C) TSET(I,J)
         ENDDO
      ENDDO
      DATA_SET_NAME = 'TDOFI'
      READ(L1C) DATA_SET_NAME
      READ(L1C) 6*NGRID
      READ(L1C) MTDOF
      DO I = 1,6*NGRID
         DO J = 1,MTDOF
            READ(L1C) TDOFI(I,J)
         ENDDO
      ENDDO
      DATA_SET_NAME = 'TDOF'
      READ(L1C) DATA_SET_NAME
      READ(L1C) 6*NGRID
      READ(L1C) MTDOF
      DO I = 1,6*NGRID
         DO J = 1,MTDOF
            READ(L1C) TDOF(I,J)
         ENDDO
      ENDDO
```

```
where MTDOF is defined in Table 6-1
```

6.1.3 L1E: G-set loads

This file contains the nonzero loads on the G-set for all subcases

The fortran code that can be used to read the G-set load data from the L1E file is:

```
READ(L1E) STIME
READ(L1E) NTERM_PG
DO I=1,NTERM_PG
    READ(L1E) I,J,PG_IJ
ENDDO
```

where NTERM_PG is the number of nonzero load components on the G-set for all subcases.

6.1.4 L1L: G-set stiffness matrix

This file contains the nonzero terms in the G-set stiffness matrix

The fortran code that can be used to read the G-set stiffness data from the L1L file is:

```
READ(L1L) STIME
READ(L1L) NTERM_KGG
DO I=1,NTERM_KGG
    READ(L1L) I,J,KGG_IJ
ENDDO
```

where NTERM_KGG is the number of nonzero terms in the G-set stiffness matrix and I, J, KGG_IJ are the row, column and stiffness value for a nonzero G-set stiffness.

6.1.5 L1R: G-set mass matrix

This file contains the nonzero terms in the G-set mass matrix

The fortran code that can be used to read the G-set mass data from the L1R file is:

```
READ(L1R) STIME
READ(L1R) NTERM_MGG
DO I=1,NTERM_MGG
    READ(L1F) I,J,MGG_IJ
ENDDO
```

where NTERM_MGG is the number of nonzero terms in the G-set mass matrix and I, J, MGG_IJ are the row, column and mass value for a nonzero G-set mass.

6.1.6 L2A: GMN constraint matrix

This file contains the nonzero terms in the GMN constraint matrix. This matrix is based on the constraints developed by the rigid elements and multi-point constraints (see MYSTRAN Users Reference Manual, Appendix B).

The fortran code that can be used to read the GMN constraint data from the L2A file is:

```
READ(L2A) STIME
READ(L2A) NTERM_GMN
DO I=1,NTERM_GMN
    READ(L2A) I,J,GMN_IJ
ENDDO
```

where NTERM_GMN is the number of nonzero terms in the GMN constraint matrix and I, J, GMN_IJ are the row, column and multi-point constraint coefficient.

6.1.7 L2H: L-set loads

This file contains the nonzero loads on the L-set for all subcases

The fortran code that can be used to read the L-set load data from the L2H file is:

```
READ(L2H) STIME
READ(L2H) NTERM_PG
DO I=1,NTERM_PG
    READ(L2H) I,J,PL_IJ
ENDDO
```

where NTERM_PG is the number of nonzero load components on the A-set for all subcases and I, J, PG_IJ are the row, internal subcase number and load value for a nonzero G-set load.

6.1.8 L2G: A-set stiffness matrix

This file contains the nonzero terms in the A-set stiffness matrix

The fortran code that can be used to read the A-set stiffness data from the L2G file is:

```
READ(L2G) STIME
READ(L2G) NTERM_KAA
DO I=1,NTERM_KAA
    READ(L2G) I,J,KAA_IJ
ENDDO
```

where NTERM_KAA is the number of nonzero terms in the A-set stiffness matrix and I, J, KAA_IJ are the row, column and stiffness value for a nonzero A-set stiffness.

6.1.9 L2I: L-set mass matrix

This file contains the nonzero terms in the L-set mass matrix

The fortran code that can be used to read the L-set mass data from the L2I file is:

```
        READ(L2I) STIME
        READ(L2I) NTERM_MLL
        DO I=1,NTERM_MLL
            READ(L2I) I,J,MLL_IJ
        ENDDO
```

where NTERM_MAA is the number of nonzero terms in the A-set mass matrix and I, J, MAA_IJ are the row, column and mass value for a nonzero A-set mass.

6.1.10 L2J: HMN constraint matrix

This file contains the nonzero terms in the HMN constraint matrix. This matrix is used to recover multi-point constraint forces.

The fortran code that can be used to read the HMN constraint data from the L2J file is:

```
        READ(L2J) STIME
        READ(L2J) NTERM_HMN
        DO I=1,NTERM_HMN
            READ(L2J) I,J,HMN_IJ
        ENDDO
```

where NTERM_HMN is the number of nonzero terms in the HMN constraint matrix and I, J, HMN_IJ are the row, column and constraint values (see MYSTRAN Users Reference Manual, Appendix B).

6.1.11 L3A: UL displacement matrix

In a statics problem, this file contains the displacements for all L-set degrees of freedom for all subcases (one subcase at a time). In an eigenvalue problem, it contains all L-set eigenvectors (one vector at a time)

The fortran code that can be used to read the UL data from the L3A file for each subcase or eigenvector is:

```
        READ(L3A) STIME
        DO I=1,NDOFL
            READ(L3A) UL(I)
        ENDDO
```

NDOFL is the number of degrees of freedom in the L-set and can be found from table TDOF or TDOFI (see file L1C)

6.1.12 L5A: UG displacement matrix

In a statics problem, this file contains the displacements for all G-set degrees of freedom for all subcases (one subcase at a time). In an eigenvalue problem, it contains the G-set eigenvectors (one vector at a time)

The fortran code that can be used to read the UG data from the L5A file for each subcase or eigenvector is:

```
READ(L5A) STIME
DO I=1,NDOFG
    READ(L5A) UG(I)
ENDDO
```

NDOFG is the number of degrees of freedom in the G-set and can be found from table TDOF or TDOFI (see file L1C)

6.2 **F2***j* **Files**

As explained in the MYSTRAN Users Reference Manual, the ELDATA Case Control entry can request output to unformatted fortran files information on element matrices. These include element mass, thermal and pressure load, stiffness matrices, stress recovery matrices and displacement and load matrices. Below is the fortean code that can be used to read these files. Note that the code shown must be executed once for each element that has been written to the F2*j* file.. As with the L*ij* files, all integer numbers are 4 byte and all real numbers are 8 bytes.

All Fij unformatted files have as its first record a time stamp, STIME, that is also the first record of all of the formatted output files (e.g. F06 file). STIME is only written once, at the very beginning of the file.

6.2.1 F21: Element thermal and pressure load matrices

Each of the element thermal, PTE, and pressure, PPE, load matrices can be read as follows

```
READ(F21) STIME
READ(F21) F21_MSG
READ(F21) EID
READ(F21) TYPE
READ(F21) ELDOF
READ(F21) NTSUB
READ(F21) NSUB
DO I=1,NTSUB
   DO J=1,ELDOF
      READ(F21) PTE(J,I)
   ENDDO
ENDDO
DO I=1,NSUB
   DO J=1,ELDOF
      READ(F21) PPE(J,I)
   ENDDO
ENDDO
```

F21_MSG is a 64 byte character message, EID is an integer element number, TYPE is an 8 byte character element description, ELDOF is the integer number of degrees of freedom for the element. NTSUB is an integer number of subcases that have thermal load and NSUB is the integer number of subcases.

6.2.2 F22: Element mass matrices

Each of the element mass matrices, ME, can be read as follows

```
READ(F22) STIME
READ(F22) F21_MSG
READ(F22) EID
READ(F22) TYPE
READ(F22) ELDOF
DO I=1,ELDOF
   DO J=I,ELDOF
      READ(F22) ME(I,J)
   ENDDO
ENDDO
```

. F22_MSG, EID, TYPE and ELDOF are as described for the F21 file. ME(I,J) is the real IJ-th term of the element mass matrix for element EID

6.2.3 F23: Element stiffness matrices

Each of the element stiffness matrices, KE, can be read as follows

```
READ(F23) STIME
READ(F23) F23_MSG
READ(F23) EID
READ(F23) TYPE
READ(F23) ELDOF
DO I=1,ELDOF
   DO J=I,ELDOF
      READ(F23) KE(I,J)
   ENDDO
ENDDO
```

F23_MSG, EID, TYPE and ELDOF are as described for the F21 file.

6.2.4 F24: Element stress recovery matrices

The element stress recovery matrices (SEi, STEi) are described in Appendix C to the MYSTRAN Users Reference Manual. They can be read as follows:

```
READ(F24) STIME
READ(F24) F24_MSG
READ(F24) EID
READ(F24) TYPE
READ(F24) ELDOF
READ(F24) NTSUB

DO I=1,3
   DO J=1,ELDOF
      READ(F24) SE1(I,J)
   ENDDO
ENDDO

DO I=1,3
   DO J=1,ELDOF
      READ(F24) SE2(I,J)
   ENDDO
ENDDO

DO I=1,3
   DO J=1,ELDOF
      READ(F24) SE3(I,J)
   ENDDO
ENDDO
```

```
      DO J=1,NTSUB
         DO I=1,3
            READ(F24) STE1(I,J)
         ENDDO
      ENDDO

      DO J=1,NTSUB
         DO I=1,3
            READ(F24) STE2(I,J)
         ENDDO
      ENDDO

      DO J=1,NTSUB
         DO I=1,3
            READ(F24) STE3(I,J)
         ENDDO
      ENDDO
```

F24_MSG, EID, TYPE and ELDOF are as described for the F21 file and NTSUB, NSUB as described for the F22 file..

6.2.5   F25: Element displacement, UE, and total load, PE, matrices

Each of the displacement and total load matrices can be read as follows

```
READ(F25) STIME
READ(F25) F25_MSG
READ(F25) MESSAG, ELFORCEN
READ(F25) EID
READ(F25) TYPE
READ(F25) ELDOF
READ(F25) JVEC
DO I=1,ELDOF
    READ(F25) UE(I),PE(I)
ENDDO
```

The stiffness, displacement, and load relationship for one element is: KE*UE = PE

F25_MSG, EID, TYPE and ELDOF are as described for the F21 file. MESSAG is:

```
MESSAG = 'Displs and forces are in coord system: '
```

ELFORCEN is an 8 byte character variable describing the coordinate system that the element forces and nodal loads are expressed in and is either 'LOCAL', 'BASIC' or 'GLOBAL' (see the MYSTRAN Users Reference Manual for discussion of coordinate systems). ELDOF are the number of degrees of freedom for the element. JVEC is an internal vector number (e.g. internal subcase number). UE and PE are the element displacements and nodal loads written 6 components per grid in the internal grid order for the element.

Table 6-1
Description of parameters used in reading L*ij* files

Parameter Name	Value
MGRID	6
MRGRID	3
MCORD	5
MRCORD	12
MTDOF	18