

Installation and Run Manual

For the

MYSTRAN General Purpose Finite Element Structural Analysis Computer Program

(Nov 2011)

Table of Contents

1	QUICK START GUIDE	1
1.1	Files for the Windows 32 bit version of MYSTRAN,	1
1.2	Running the example problem in the Windows version	2
1.3	Files for the Linux 64 bit version of MYSTRAN.....	2
1.4	Running the example problem in the Linux version	3
2	COMMAND WINDOW MESSAGES.....	4
3	TRIAL VS LICENSED EDITIONS.....	6
3.1	Windows version.....	6
3.2	Linux version	6
4	MYSTRAN FILES	7
4.1	F06 text output file	7
4.2	F04 text output file	7
4.3	ERR text output file.....	7
4.4	BUG text output file.....	7
4.5	SEQ text output file.....	7
4.6	SPC text output file	7
4.7	ANS text output file.....	8
4.8	L1A text output file.....	8
4.9	NEU text output file.....	8
4.10	Lij binary output files	8
4.11	F2j binary output files	8
5	MODIFYING THE QUICK START SETUP	9
5.1	Changing directories where MYSTRAN executable resides	9
5.1.1	Windows version.....	9

5.1.1.1	Method 1 (easiest but Method 2 preferred).....	9
5.1.1.2	Method 2 (preferred)	9
5.1.2	Linux version	10
5.2	The MYSTRAN initialization file	10
5.2.1	Changing directories where the input and output files reside	11
5.2.2	Changing the default extension for input files	11
5.2.3	Setting the level of detail for the log file	11
5.2.4	Saving the <i>Lij</i> files	12
5.2.4.1	Saving all <i>Lij</i> files.....	12
5.2.4.2	Saving an individual <i>Lij</i> file	12
6	MYSTRAN MESSAGES	13
6.1	Error messages	13
6.2	Warning messages.....	13
6.3	Information messages	13
7	APPENDIX A: DESCRIPTION OF LIJ AND F2J FILES.....	14
7.1	Lij Files	15
7.1.1	L1B: Grid Data	15
7.1.2	L1C: Degree of freedom tables.....	17
7.1.3	L1E: G-set loads	20
7.1.4	L1L: G-set stiffness matrix	20
7.1.5	L1R: G-set mass matrix	21
7.1.6	L2A: GMN constraint matrix.....	21
7.1.7	L2H: A-set loads.....	21
7.1.8	L2G: A-set stiffness matrix.....	22
7.1.9	L2I: A-set mass matrix	22
7.1.10	L2A: HMN constraint matrix	22
7.1.11	L3A: UA displacement matrix.....	23
7.1.12	L5A: UG displacement matrix	23
7.2	F2j Files	24
7.2.1	F21: Element mass matrices	24
7.2.2	F22: Element thermal and pressure load matrices	24
7.2.3	F23: Element stiffness matrices.....	25
7.2.4	F24: Element stress recovery matrices	25
7.2.5	F25: Element displacement, UE, and total load, PE, matrices	26

1 Quick start guide

1.1 Files for the Windows 32 bit version of MYSTRAN,

The files that were installed when you ran the MYSTRAN setup.exe file should have included:

- README.TXT: a text file containing basic information on how to quickly install and get up and running on the MYSTRAN finite element computer program
- MYSTRAN.EXE: the executable that runs the MYSTRAN finite element program under the 32 bit Windows operating system
- MYSTRAN.BAT: a batch file that can be used to start MYSTRAN.EXE, if desired
- MYSTRAN.INI: a MYSTRAN initialization file (discussed herein)
- EXAMPLE1.DAT: the input data deck for the example problem explained in the MYSTRAN Users Reference Manual Appendix A
- EXAMPLE1.F06-archive: the output for the example problem explained in the MYSTRAN Users Reference Manual Appendix A
- MYSTRAN-Pricing.pdf: the cost for options of the unlimited problem size edition of MYSTRAN
- MYSTRAN-Users-Manual.pdf: the MYSTRAN User's Reference Manual
- MYSTRAN-Install-Manual.pdf: this detailed Installation and Run Manual
- MYSTRAN-Demo-Problem-Manual.pdf
- Errors-corrected.pdf: A list of the errors corrected in the current version
- New-features.pdf: New features that have been added to MYSTRAN
- A subdirectory with several MYSTRAN sample problem runs

These files plus a few others should have been installed in the directory:

C:\Program Files\MYSTRAN

if you accepted the defaults given during installation. If you did not accept the above for installation, you must read the section on the MYSTRAN initialization file.

In addition, a MYSTRAN icon should have been placed on your desktop and a MYSTRAN entry placed in your Start All Programs menu.

1.2 Running the example problem in the Windows version

The files with an extension of PDF are in Adobe PDF format and can be read, and printed, using Adobe Acrobat Reader; a free download available from the Adobe internet site: www.adobe.com.

After you have installed MYSTRAN, the quickest way to get started is to try running the example problem:

- Double click on the MYSTRAN icon on your desktop. A DOS command window should open
- At the prompt, enter the name of the example problem (EXAMPLE1). Note the extension (DAT) need not be entered since that is the default extension for input data decks.
- You should see the example problem execute in the DOS window with messages indicating the progress as MYSTRAN solves the problem. When you are finished reviewing the messages press any key to close the command window.

The output file (EXAMPLE1.F06) for the example problem will be found in the RUNS subdirectory of the directory in which the files were put during installation (C:\Program Files\MYSTRAN if not changed during installation).

In addition to the F06 file there may be several other files created during the execution depending on the settings in the INI and DAT files.

If you want to have your input files in a different folder than the RUNS subfolder, you should read the remainder of this Installation and Run Manual. However, an easy change is to edit the MYSTRAN.INI file and replace the line:

```
DEF DIR C:\Program Files\MYSTRAN\Runs
```

with the default directory where your input files exist. As mentioned, this is the easiest change but if you want to provide the most flexible way for MYSTRAN to run you need to create a Windows environment variable called MYSTRAN_directory and give it a value of the location where the MYSTRAN executable resides (and also delete the line DEF DIR entry in MYSTRAN.INI). See this Installation and Run Manual for details.

1.3 Files for the Linux 64 bit version of MYSTRAN

For the Linux version of MYSTRAN, the following files are tar zipped into MYSTRAN_Linux_files.tar:

- README.TXT: a text file containing basic information on how to quickly install and get up and running on the MYSTRAN finite element computer program
- mystran.exe: the executable that runs the MYSTRAN finite element program under a 64 bit Linux operating system
- MYSTRAN.INI: a MYSTRAN initialization file (discussed herein)
- Script file "set_MYSTRAN_directory.sh" (use is explained later)
- EXAMPLE1.DAT: the input data deck for the example problem explained in the MYSTRAN Users Reference Manual Appendix A

- EXAMPLE1.F06-archive: the output for the example problem explained in the MYSTRAN Users Reference Manual Appendix A
- MYSTRAN-Pricing.pdf: the cost for options of the unlimited problem size edition of MYSTRAN
- MYSTRAN-Users-Manual.pdf: the MYSTRAN User's Reference Manual
- MYSTRAN-Install-Manual.pdf: this detailed Installation and Run Manual
- MYSTRAN-Demo-Problem-Manual.pdf
- Errors-corrected.pdf: A list of the errors corrected in the current version
- New-features.pdf: New features that have been added to MYSTRAN
- A subdirectory with several MYSTRAN sample problem runs

1.4 Running the example problem in the Linux version

After unzipping the tar file, and making sure that mystran.exe, mystran.ini and the example problem DAT file are all in the same folder, the program can be run in a terminal by giving the command:

```
mystran.exe EXAMPLE1
```

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:

❖ LINK 0:

- Read input data deck and check for errors and possible restart
- Process grid and coordinate system input data
- Process Case Control output requests
- Forms degree of freedom (DOF) tables
- Process concentrated mass input data
- Calculates rigid body mass properties (Grid Point Weight Generator)
- Process temperature and pressure load input data into arrays needed for element load calculations

❖ LINK 1:

- Process MPC's and rigid elements into sparse array RMG
- Process all applied forces (including grid forces and moments, gravity, pressure, thermal, centrifugal, scalar) into sparse load array PG
- Formulate the G-set¹ sparse stiffness and mass arrays KGG and MGG
- Formulate the G-set sparse differential stiffness array KGGD

❖ LINK 2:

- 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)

- ❖ LINK 3 (for statics problems only):
 - Solve for the L-set displacements
- ❖ LINK 4 (for eigenvalue problems only):
 - Solve for the eigenvalues and the L-set eigenvectors
- ❖ LINK 5:
 - Build the A-set displacements back up to the G-set through use of the constraint matrices
- ❖ LINK 6 (for Craig-Bampton model generation only):
 - Builds a Craig-Bampton model from the input physical model
- ❖ LINK 9:
 - Use the G-set displacements from LINK5 and the constraint matrices 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 terminated normally  
The output file is:  
filename.F06
```

where *filename* is the name, including drive and path of the input file (e.g. EXAMPLE1)

3 Trial vs licensed editions

Prior to purchasing a license for MYSTRAN the user has the ability to run a trial edition of the program. This trial edition is different for the Windows and Linux versions, as explained below. Purchasing a license is easy and is explained on the “Download MYSTRAN” page of the MYSTRAN website:

<http://www.MYSTRAN.com>

3.1 *Windows version*

The trial edition of the Windows version is unlimited in the size problems that can be run but is limited to a trial period of 30 days. After the 30 days is up a message will appear that reminds users that, if they wish to continue using the program, they will have to purchase a license.

3.2 *Linux version*

The Linux version trial edition is limited to a maximum problem size of 100 grid points but does not expire (i.e. there is no trial duration after which the program will cease to run)². Once a license is purchased the program will no longer be limited in the size problem it can run.

² The reason that the trial edition limitation is different in the Linux than in Windows lies in the fact that the author does not have a “software protection” suite that works under Linux like the one that allows unlimited sixe trial period under Windows

4 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.

4.1 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).

4.2 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.

4.3 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.

4.4 BUG text output file

If element debug information is requested in Case Control via the ELDDATA 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 ELDDATA command discussion.

4.5 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).

4.6 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.

4.7 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).

4.8 L1A text output file

The L1A text file, as well as the *Lij* binary files (discussed below), are used to communicate data between the LINK's. The L1A file contains data needed to read the binary *Lij* 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

4.9 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.

4.10 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.

4.11 F2j 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 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.

5.1 Changing directories where MYSTRAN executable resides

This section should be read in its entirety prior to making any changes to the directory where the MYSTRAN.BAT or MYSTRAN.EXE files exist as well as where the input data (DAT) files exist. If you want to leave the BAT and EXE files where they were installed and have your input and output files be in the RUNS subdirectory, this section may be skipped (if you accepted the default installation location).

5.1.1 Windows version

5.1.1.1 Method 1 (easiest but Method 2 preferred)

The MYSTRAN.BAT file you received contains the three commands:

```
CD C:\Program Files\MYSTRAN\bin
MYSTRAN.EXE
PAUSE
```

The first line changes directories to the one where the MYSTRAN.EXE file is located (based on options selected in setup). If you wish to change this to some other directory, edit the first line to reflect that change and put MYSTRAN into that directory. The second line starts the MYSTRAN executable which will ask you for the name of the input file for this execution (the INI file, described later tells MYSTRAN what directory the DAT file is in if not in the current directory). The third line is included so that the command window opened when you double clicked on the MYSTRAN.BAT icon will remain open until you decide to close it. Otherwise, the window may close automatically when MYSTRAN completes its execution. This is not a problem, but you may want to keep it open to verify the ending status of your job's execution.

If you want MYSTRAN.EXE to reside in a different directory than the one in which you put it above and you still want to execute MYSTRAN in this manner, you can edit the MYSTRAN.BAT file. Say, for example, you want the executable to be in directory C:\PROGRAMS\MYSTRAN. You would edit the original MYSTRAN.BAT file to be:

```
CD C:\PROGRAMS\MYSTRAN
MYSTRAN.EXE
PAUSE
```

5.1.1.2 Method 2 (preferred)

The MYSTRAN executable can figure out what folder it is in through use of a Windows environment variable. In Windows XP Home Edition, go into the Windows Control Panel, select "System", then "Advanced" and click on "Environment Variables" 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 (DAT) 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. This method requires modifying the MYSTRAN.INI file, described below, by removing the line which begins with DEF DIR. It is important to note that the DEF DIR is only intended to be used if you do not use Method 2. If you use Method 1 and want the input (and output) files in a different directory than the ones in which they were installed then you need to modify MYSTRAN.INI

5.1.2 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
```

5.2 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 *Lij* files. The default is that no *Lij* 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.

5.2.1 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.

5.2.2 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).

5.2.3 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).

5.2.4 Saving the *Lij* 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.

5.2.4.1 Saving all *Lij* files

To save all of the *Lij* 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 .

5.2.4.2 Saving an individual *Lij* file

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

```
Lij KEEP
```

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

6 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.

6.1 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).

6.2 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
```

6.3 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.

7 Appendix A: Description of Lij and F2j files

7.1 Lij Files

Many of the Lij 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 below³. 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.Lij* where filename is the input data deck name for the problem MYSTRAN has run and Lij is the extension for the unformatted files discussed below. All integer numbers are 4 byte and all real numbers are 8 bytes.

7.1.1 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 basic 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:

³ 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. This 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.

7.1.2 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 set⁴
- A for a degree of freedom that is to belong to the analysis set⁵

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
- Col 07 :N-set degree of freedom number
- Col 08 :N-set degree of freedom number

⁴ See Bulk Data entries OMIT and OMIT1 and for a description of how degrees of freedom are placed in the O set

⁵ See Bulk Data entries ASET and ASET1 and for a description of how degrees of freedom are placed in the A set

- 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) 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

7.1.3 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) 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.

7.1.4 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) 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.

7.1.5 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) 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.

7.1.6 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) NTERM_GMN
DO I=1,NTERM_GMN
  READ(L1F) 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.

7.1.7 L2H: A-set loads

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

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

```

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 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.

7.1.8 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 LEG file is:

```

READ(L1L) NTERM_KAA
DO I=1,NTERM_KAA
  READ(L1L) 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.

7.1.9 L2I: A-set mass matrix

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

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

```

READ(L1R) NTERM_MAA
DO I=1,NTERM_MAA
  READ(L1F) I,J,MAA_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.

7.1.10 L2A: 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 L2A file is:

```

READ(L2A) NTERM_HMN
DO I=1,NTERM_HMN
  READ(L1F) 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).

7.1.11 L3A: UA displacement matrix

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

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

```
DO I=1,NDOFA
  READ(L3A) UA(I)
ENDDO
```

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

7.1.12 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:

```
DO I=1,NDOFG
  READ(L3A) 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)

7.2 F2j 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 fortran 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 F2j file.. As with the Lij files, all integer numbers are 4 byte and all real numbers are 8 bytes.

7.2.1 F21: Element mass matrices

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

```
READ(F21) F21_MSG
READ(F21) EID
READ(F21) TYPE
READ(F21) ELDOF
DO I=1,ELDOF
  DO J=I,ELDOF
    READ(F21) ME(I,J)
  ENDDO
ENDDO
```

F21_MSG is a 132 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 type and ME(I,J) is the real IJ-th element of the mass matrix for element EID.

7.2.2 F22: Element thermal and pressure load matrices

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

```
READ(F22) F22_MSG
READ(F22) EID
READ(F22) TYPE
READ(F22) ELDOF
READ(F22) NTSUB
READ(F22) NSUB
DO I=1,NTSUB
  DO J=1,ELDOF
    READ(F22) PTE(J,I)
  ENDDO
ENDDO
DO I=1,NSUB
  DO J=1,ELDOF
    READ(F22) PPE(J,I)
  ENDDO
ENDDO
```

F22_MSG, EID, TYPE and ELDOF are as described for the F21 file. NTSUB is the integer number of subcases that have thermal load and NSUB is the integer number of subcases.

7.2.3 F23: Element stiffness matrices

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

```
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.

7.2.4 F24: Element stress recovery matrices

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

```
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..

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

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

```

READ(F25) 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 *Lij* files

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