Methods for fatigue analysis of full body automotive structural systems.

Neil Bishop, Andrew Woodward and Mike Coleman

Date: 1 August 2001
Report Number: RLD020801
Methods for fatigue analysis of full body automotive structural systems.

Dr NWM Bishop, Andy Woodward and Mike Coleman

Introduction

Testing of full body systems to resist vibration loading is an expensive task. Any analysis procedure that can be used to reduce the number of tests, or supplement testing procedures, is likely to be extremely cost effective. However, traditionally the main form of analysis applicable to such problems, transient methods of analysis, is far too computationally expensive to be a realistic design approach. This document will set out 2 alternative methods of analysis. Firstly a frequency (PSD) based approach will be used and for the remainder of this document we will refer to this method as vibration fatigue analysis. The second method, a time based method, uses the modal participation factor information from a transient analysis. However, the stress combination work is done externally from the Nastran solver, within MSC.Fatigue. We will refer to this method as Modal Participation Factor (MPF) method. These 3 methods, transient fatigue, vibration fatigue and MPF fatigue will all be applied to a realistic automotive system and compared. It will be seen that both the vibration and MPF methods have good advantages when applied to large FE systems and therefore offer cost saving opportunities when used to supplement test based systems. Very little background theory will be presented. This is available in “Vibration Fatigue Analysis in the Finite Element Environment”, Dr NWM Bishop, An Invited Paper presented to the XVI ENCUENTRO DEL GRUPO ESPAÑOL DE FRACTURA, Torremolinos, Spain, 14-16 April 1999. This paper is available from the author of this case study.

General comments on using MSC.Fatigue

Generally, MSC.Fatigue routines can be run in one of three ways. The first 2 are both accessible from MSC.Patran (see below). The first of these runs modules from within the MSC.Fatigue main form. To access modules this way go to the Patran top menu bar and access Tools/FATIGUE. This will spawn the main MSC.Fatigue menu form and this is one area from where MSC.Fatigue modules can be run.

![Figure 1. Spawning MSC.Fatigue from the main Patran form](image1)

![Figure 2. The main MSC.Fatigue form](image2)
The second way, within *Patran*, uses the *Tools+* icon accessible from the *Patran* top menu bar (see below). This spawns a variety of modules that can be used to undertake fatigue related activities.

![ MSC.Fatigue tools](image)

**Figure 3. Using the Tools+ menu icon to access MSC.Fatigue tools**

The 3rd method that can be used involves typing the (usually) 4-letter module name from within a command window. For example, type `mATD` and this will run a program which converts asci time series data in to a binary form, which is then more easily used with other *MSC.Fatigue* modules. Four time series files will be used (accsignal.asc, pave_1.asc, pave_2.asc and Pave_3.asc). After running `mATD` a series of options are given from the following can be chosen,

![ mATD](image)

**Figure 4. Using mATD to convert asci time series data into a binary format**

This will then produce a binary file (in `.DAC` format) that can then be used with all the *MSC.Fatigue* programmes. Next, Pave_1.dac can be read in to a plotting program to graphically assess it. Either the Tools+ method or the command prompt method can be used to run `mQLD`. The x window should be set to full x and then the time series data shown in Figure 5 will result. Note: There are several programs for plotting data. `mQLD` for single time series, `mMFD` for multiple files, `mGED` for graphically editing data, `mTPD` for displaying non-equal x-y points, etc.

![ Pave_1.asc](image)

A lot of useful information is included on this graph, including the *root mean square (rms)* value as well as other interesting statistics. This data is, of course, only one sample from many possible samples and it is interesting to look at further samples measured under the same conditions. Pave2.asc and Pave_3.asc can be converted into binary form and then plotted alongside the first data set using `mMFD` (again accessible from Tools+ or command prompt). In this case it is important to set the y axis to be the same for all four plots. The plot in Figure 6 then results,
Sometimes it is useful to view the data as if it were being measured real time. In order to do this use the view icon on the mMFD top menu and set it to view / window x / 0 secs to 2 secs. Then pick view / scroll / right / 0.2 pause / 0.9 overlap, and this will play through the time signal at approximately the same speed that it was measured.

It is interesting to investigate the statistical variations within each file and a very useful tool for doing this is mRSTATS. As an example, use Pave_1.dac as an input file and then tag rms, mean, min and max and then press accept and use the options in Figure 7.
Once completed, use the *mMFD* programme to show these statistical quantities plotted against “time” (see below).

This clearly shows that the intensity of the time series, as shown by the *rms*, has low spots at the beginning, middle and end. This causes some difficulties when doing frequency based vibration fatigue calculations because it is normally necessary to assume the input loading is stationary.
However, we will show later that the tools being used are quite robust to small fluctuations like those shown here.

We will now look in detail at the first of our three methods of analysis.

**Method 1. Vibration fatigue (in the frequency domain)**

The basic input for a frequency based vibration fatigue analysis is not the time signal, but rather a *Power Spectral Density (PSD)*. Very often this will be the only source of loading available but for this case study we start with a time history and so this must be converted into a *PSD*. To do this we use the program *mASD* as shown in Figure 9, Figure 10 and Figure 11.
There is a trade off in the choices made in these forms. The choice of FFT buffer size, for instance 2048 pts, fixes the frequency interval in the resultant PSD. It also determines the number of buffers obtained from the whole time history, and this determines how much scatter is present in the result. Larger buffer sizes result in good statistical resolution (small $\delta f$) but few buffers with which to perform the averaging, and so larger scatter. Readers with more time may wish to assess this affect by varying the input parameters to this process.

Up to now we have only considered the input acceleration loading to our car structure. We may consider the overall, process in the following way

In our case the input loading is one acceleration time history (or PSD) but in general the loading can be made up from multiple inputs. In this case the correlation between each input also needs to be determined and there is a tool (mFRA) that can be used for doing this. The input then becomes a matrix of PSD’s and cross PSD’s in the following form.

We are now ready to consider the structural model, which in this case forms our system. Both a bdf file and an op2 file exist for this model. These can be used in the following way to view and analyse the model.

The first method is to load database Front_Car_FRA.db in to Patran. This database contains the model file and frequency response results from Nastran.
Alternatively, load *Front_Car_FRA_orig.bdf* into *Patran* (this contains FE model information which does not get put in the *op2* file) and then run *Nastran* using the same input file (*Front_Car_FRA.bdf*). For instance, *File / Import / Source = MSCNastran Input*, then use *Utilities / Groups / Group from Properties* to create a set of groups which can be used later. The resultant *op2* file contains all of the required stress information and should then be read in to *Patran* using the *Analysis* menu icon.

The acceleration loading was applied through a single MPC fixed to 4 locations under the front of the car. A force of 9.81E8 was applied to a mass of 10E8 resulting in an acceleration sine wave of 1G at a series of frequencies covering the frequency range of interest.

The choice of frequencies used is of vital significance to the accuracy of the final result and this topic will be returned to later. In the meantime it is sufficient to know that 61 frequencies at 1Hz intervals between 1Hz and 61Hz are used.

The main area of interest for this analysis concerns the battery tray and surrounding area shown in Figure 12, Figure 13 and Figure 14. The battery tray is held on the car through a bracket. This bracket is spot welded on to the car body and bolted on to the battery tray. This load transfer route offers the possibility for high stresses and hence low fatigue life and this will be investigated in detail below.

![Figure 12. The battery tray](image)
It is of interest to consider the form of the results which have been read from the OP2 file. These are now in to the *Patran* database and are available to us for processing. Go to the *Results* form from the *Patran* menu icon and then choose *Create / Quick Plot* and then pick the forced response result at frequency 5Hz and something similar to the plot in Figure 15 should be obtained.
This plot shows the maximum von-mises stress (in a sine wave of response) caused by a 1G sine wave of acceleration at the loading position shown earlier. It is of interest to plot the maximum (principal) stress response at the high stress location shown in Figure 15 as a function of frequency. To do this use results / Create / Graph, pick all 61 frequencies, set stress output to Maximum Principal, pick the node at the high stress location (probably node 642192) and be sure to set Complex Number as “magnitude” and also give the xy graph a plot title such as Transfer Function at Node 642192. Press apply and the plot in Figure 16 should result.

This plot is the linear transfer function between input acceleration, in this case 1G sine wave, and stress response at one critical node. It is interesting to note that, in this case, there appear to be 2 modes of response present at approximately 17Hz and 55Hz. The static response, well below 17Hz, is approximately equal to 100MPa (for a 1G input). The peak responses at the 2 modes of vibration are approximately 220MPa and 650MPa. The matrix of similar curves for all possible nodes of interest constitutes the system transfer function. This information was read in to the Patran database from the OP2 file earlier.

So we now have all of the information required in order to perform a (frequency based) vibration fatigue analysis on the battery tray and surrounding area. In order to do this start up MSC.Fatigue from the Tools menu button in the Patran menu bar. Select the options shown in the Figure 17, Figure 18, Figure 19, Figure 20 and Figure 21.
In Figure 17 we are defining general setup parameters, such as defining the analysis type as vibration fatigue with nodal results and group rather than global averaging, stress results in MPa.

Once these parameters have been defined then each of the three set up forms have to be defined.

In Figure 18 we are specifying how the vibration fatigue analysis will be done. For instance, Dirlik (one choice of PSD fatigue algorithm) is specified, as well as the choice of Maximum Absolute Principal stress (the largest of the 3 Principal stresses). Confidence levels (on material data) can be specified but the usual 50% has been specified here. Next, the materials form must be completed.
In Figure 19 we are defining the material S-N curve to use for the analysis and other material related parameters such as surface finish and treatment. The region for which the fatigue calculation has to be performed is defined by the region *fatigue*. Further to the right on this form are other material relevant options which can be used to, for instance, superimpose a $K_N$ term or adjust the S-N curve in some user defined way. Next the loading form must be specified.

In Figure 20 the Results Type must be set to *Transfer Function* since we are reading transfer functions from the Patran database. Next, the frequency response (transfer functions) must be specified. In order to do this it is necessary to go through an intermediate process. First, press *Get/Filter results*, select the results case, press *filter* and then *add* to move the relevant choices back to the main loading form. This intermediate form can then be closed and the loading form above will be left, from which the correct results case, stress type and, if relevant, surface Z1 or Z2, can be chosen.
In the materials form there is a *Materials Database Manager*. This is one way of accessing numerous useful programs for pre and post processing of the materials database. This is also accessible from the *PFMAT* option in the *Tools+* menu bar icon. By opening *PFMAT* the form in Figure 22 becomes available.

In order to view a particular material type the following steps must be followed

- Open the correct User database using *Preferences / Database Select / User* and then point to the appropriate database.
- Load dataset 1 and then the form in Figure 23 will become available,
• Load the appropriate material and then use *Graphical Display* which will spawn the form in Figure 24.

In this particular case we are doing a vibration fatigue analysis using an Stress-Life approach, so a Stress-Life data plot must be chosen as shown in Figure 25.
It is interesting to cross reference this plot against the Strain-Life plot shown in Figure 26.

To transform between plots the cyclic stress strain curve must be used as shown in Figure 27.
But it should be remembered that the life axis with an Stress-Life plot is cycles but with strain-Life plot it is reversals. Also, it is conventional to use stress range for the y axis of the Stress-Life plot but stress amplitude for the strain-Life plot. It should then be possible to switch backwards and forwards between the 2 plots.

Also, from inside the loading form it is possible to start the time history (in this case PSD) manager _PTIME_. This spawns the form given in Figure 28.

There are numerous useful tools inside this module, some of which we have already used. For instance, _x-y PSD_ entry allows simple _PSD_ plots to be entered by hand.

If the model being analysed has more than one loading case then the appropriate button in the loading form must be set from _Single_ to _Multiple_. Also, a _PSD_ matrix (of file names) corresponding to all of the _PSD_’s and cross _PSD_’s must be specified. If the loading inputs exist originally as time series of acceleration then the _PSD_’s and where relevant, _cross PSD_’s, must be computed. _mASD_ is a useful tool for calculating the _PSD_’s. _mFRA_ will calculate both the _PSD_’s and _cross PSD_’s.
Since we have now completed all of the necessary forms it is possible to run the analysis using *Job Control / Full Analysis*. At the beginning of this process the whole set up of the analysis is saved in the form of a `.FIN` file. Furthermore, since this file contains all of the data necessary to set the job up it is possible to start from scratch and set up the whole analysis simply by typing *Job Control / Read Saved Job*. The `.FIN` file is used to set up the job and perform a variety of pre processing. All of the FE data is read and all of this is then saved, along with the data from the `.FIN` file, in a `.FES` file. The fatigue analysis itself then starts. Firstly a `.FPP` file is produced which contains the rainflow counted stress information for all requested nodes. This is then processed in order to produce a `.FEF` file. This is a full results file containing all of the fatigue life information required. Once the initial pre processing has been completed (.FES file) then *Job Control / Monitor Job* can be used to find out about progress with the analysis. Once the job has completed the *Results / Read Results* button at the bottom of the MSC.Fatigue form can be used to read the results back in to the Patran database. They are then available for plotting with Patran in the normal way.

![Figure 29. Plot of fatigue damage showing critical location near to bolt hole](image1)

![Figure 30. Zoomed plot of critical location](image2)

Figure 29 and Figure 30 show a critical location near to the bolt hole with a fatigue damage of 10^10, equivalent to a life of 7817 second (2.2 hours).

A number of very useful tools can be accessed via the *Job Control / Interactive* menu buttons as shown in Figure 31.
First of all, by accessing \textit{Results Processing} a list of, for instance, lives at the 20 most critical nodes, or just one node, can be obtained from the \texttt{Jobname.FEF} file (see Figure 32).

\textit{Global vibration fatigue analysis} allows the complete fatigue analysis to be performed for one or more nodes.

\textit{Output power spectrum} allows the user to output the PSD of stress response at some critical location as shown in Figure 33.
We will use this plot later! However, it is first useful to convert this plot into a slightly different form because it is, by default, made up of unequal-x x-y data pairs and many of the programs we want to use prefer constant x data sets. The easiest way to achieve this is to use the digital to asci conversion process \textit{mDTA} and then reverse the process with \textit{mATD} as shown in Figure 34 and Figure 35. Notice that the conversion process assumed that the resultant PSD was a time history. This needs to be corrected with the \textit{mFILMNP} program as shown in Figure 36, Figure 37 and Figure 38.
Figure 35. The form obtained for a single channel mATD

Figure 36. Using FILMNP to change the axis on the graph from time to frequency

Figure 37. Using FILMNP / Header Manipulation
Be sure to save the changes before exiting the program. The output PSD obtained this way, at node 642192, is shown in Figure 39.

*Job Control / Interactive / Design Optimisation* reads the Jobname.FES file and spawns the form shown in Figure 40.
Input the required node (642192), and design life as shown in Figure 41, and press OK.

Pressing end will then finally spawn the very useful form given below with which a variety of different tasks may be accomplished.
For instance, a *global offset mean stress* may be included in the fatigue calculation (as shown in Figure 42) as long as the mean stress correction option was chosen earlier. Or a *sensitivity analysis* can be performed to assess the variation of fatigue life with, for instance, input acceleration intensity as shown in Figure 43.

Type recalculate in the main form and then the results in Figure 45 appear.
It is now useful to perform some general checks on the input–output process by plotting the input, transfer function and output PSD together. The transfer function for node 642192 was plotted earlier in *Patran*. Let us now output this file to an asci file using the process shown in Figure 46.

This asci data can then be read back in using *mATD* in the normal way. Remember to use *mFILMNP* to convert the x axis to frequency. Then use *mQLD* to plot it as shown in Figure 47.
You will remember that the transfer function required has to be in units of [MPa/Hz]^2 so we have to square the function using \textit{mART} before it can be used as a transfer function.

All three can then be plotted on one graph as shown in Figure 48,
This is an extremely important plot since it shows the dynamic behaviour of the structure (2 modes at approximately 18Hz and 56Hz) and also the way that the input loading and transfer function combine to produce the output PSD. Stresses at higher frequencies than might be expected result!

It is often very interesting to find out the fatigue damaging contribution that each part of a response plot makes to the overall fatigue damage. This is very easy to do using 2 modules in MSC.Fatigue. mGED can be used to graphically modify a particular PSD and then mFLF can be used to recalculate fatigue damage. So if we start with RANDOM4642192.PSD. Then we will recalculate the fatigue life directly from it using mFLF as shown in Figure 49.

![Figure 49. Using mFLF to calculate fatigue damage from a PSD of stress](image)

The various parameters needed are shown in the following figures.

![Figure 50. The mFLF analysis file and parameters form](image)
Figure 51. The advanced options form

Figure 52. Intermediate rainflow range options

Figure 53. The material form used for a vibration fatigue analysis

Figure 54. The number of rms levels of the rainflow range histogram to integrate
There are some interesting points to note. First of all, in this example we are again using the Dirlik method (generally recommended) but in this case we are also doing a mean stress correction with the Goodman method. Secondly, because we ticked the advanced options button we are told the frequency limit used to integrate the PSD (60Hz) and asked what stress range to integrate the rainflow range histogram up to (in this case 5rms levels). Finally the result is produced of 1.853 Hours. This is slightly lower than the previous result (2.2 Hours) even though the global offset stress was zero because the local stresses of each rainflow cycle can, in general be a little higher or lower than zero.

To finish this part of our case study we are going to show how mGED, a graphical editor, can be used in conjunction with mFLF to assess the fatigue damaging contribution that each part of the PSD can make. Figure 56 shows how to access mGED from the Tools+ menu. Figure 57 shows the mGED main form. Leave Reference Filename blank and the plot shown in Figure 58 will result. This form allows any part of the PSD to be rescaled or deleted. In this case the range 46-60Hz is reduced in magnitude by 50%.

The new PSD (shown in Figure 59) can then be processed using mFLF to get a new fatigue life (shown in Figure 60). The effect of reducing this part of the PSD to zero is shown in Figure 61.
Figure 57. The mGED main form

Figure 58. Rescaling parts of the PSD response

Figure 59. Reducing the 46-61Hz range by 50%
The effect of these changes to the higher frequency part of the PSD (50% reduction, 100% reduction) is to increase the fatigue life to 7.4 Hours and 61.7 Hours respectively. So a relatively small change to a high frequency part of the response can have a big affect on fatigue life.

Method 2. Transient Analysis

In order to perform either a transient analysis or a modal superposition analysis (see next section) a Nastran SOL112 analysis must be performed. This does both a modal analysis followed by the subsequent transient analysis. Because of the solution procedure used the memory requirement, hard disk requirement (for scratch space) and CPU requirement can all be extremely large. The method described in the next section (Modal Participation Factor method) has been developed in order to reduce these requirements, especially the hard disk requirement.

Once the Nastran job has been completed the results are read from the OP2 file in the normal way. However, in this case a time history of stress is read in to MSC.Fatigue for each node of interest on the structure. Because of the severe computer resource requirements only a small part of the input time history has been used as loading and the fatigue life is calculated on a small area around the critical region. The piece of time history used is shown in Figure 62.
In order to enter this in to a Nastran input deck (TABED1 entry) the easiest method is to use the Patran Fields option, which is obtained from the Utilities/Fields/Time Dependent Field button (see Figure 63, and Figure 64) to enter the data. The time step used was 0.00488 seconds, and 3936 points were used.
The fatigue life analysis is then set up in a very similar way to before. Figure 65, Figure 66, Figure 67 and Figure 68 show the various forms needed. The only difference is that in the loading form Results Type is set to transient and then in the lower part of the form the location of the stress time histories for all points of interest have to be specified. All results cases (time steps) have to be picked at this point. The output time history of stress at the critical node obtained from Nastran is shown in Figure 69. Using this approach the fatigue life could be calculated for all points of interest. The fatigue life at a critical node can also be calculated using the mSLF program and Figure 70 shows the results. The fatigue life is somewhat different 0.6 Hours than the frequency based approach which is to be expected because of the short sample length and also, perhaps, because of the possible existence of non-random components (spikes) in the input loading. In the next section (Modal Participation Factor method) we will further investigate this.
Figure 66. Choosing the parameters for a Stress-Life analysis

Figure 67. The materials form for a Stress-Life analysis
Figure 68. The loading form for a Stress-Life (transient) analysis

Figure 69. The time history of (Maximum Principal) stress output (using Nastran) at the critical node 642192
Method 3. Modal Superposition Analysis

The basis of the Modal Participation Factor (MPF) method is that each loading input (in this case one) can be split up into the contribution factors associated with each mode shape for the structure. In NASTRAN 2001 both the modal stresses and modal participation factors can be extracted from a single sol 112 analysis. The modal superposition is then calculated as follows:

\[ \sigma(t) = \sum_{l} \sigma_l \cdot \phi_l(t) \]

where \( \sigma(t) \) is the output stress tensor
\( \sigma_l \) is the stress tensor for mode \( l \)
\( \phi_l(t) \) is the modal participation factor for mode \( l \)

Nastran can be used to generate these MPF sets in punch file format. However, in order to input them in to MSC.Fatigue we need them in the form of .DAC files. In order to do this a DMAP alter and associated Cshell script (see appendix) have been written which enable this process to be automated. A typical set of results for MPF sets 1 to 8 are shown in Figure 71. The modal stresses are extracted from an initial subcase in the sol 112 analysis using the new option ANALYSIS=MODES and are written to the op2 file. In previous versions of NASTRAN this would have to been achieved with a separate modal analysis (sol 103). Once both sets of data are available it is a very simple task to set up a psuedo static type analysis with MSC.Fatigue as shown in Figure 72. Plots of fatigue damage over the bracket are shown in Figure 73, and Figure 74. However, unlike with the traditional transient method, the computing resources needed are relatively small so results over the whole bracket can be requested. Also, if required the output time history of stress at a critical node can be obtained (see Figure 75) and compared with that obtained using a traditional transient method (see Figure 69). If sufficient modes are included in the MPF method then identical results are obtained when compared with the transient method, but with dramatically less computing effort. Finally, a list of results for critical nodes is shown in Figure 76.
Figure 71. Modal Participation Factors for 1st 8 modes

Figure 72. Method used to perform a Modal Participation Factor (MPF) method fatigue analysis
Figure 73. Plot of fatigue damage for a Modal Participation Factor (MPF) analysis

Figure 74. Zoomed plot of fatigue damage
Conclusions

Three methods of dynamic fatigue analysis (frequency based vibration fatigue, transient and Modal Participation Factor (MPF) method) have been performed and compared. Generally similar results are obtained between the 3 methods. Identical results are obtained for the transient and MPF methods if the same mode shapes are used for both. The frequency domain method (vibration fatigue) is much more efficient computationally. The MPF method is significantly more efficient, computationally, than the traditional transient method.
Appendix

pchdac2.alter

$ include 'pchdac.alter'
compile semtran souin=mscsou,list,noref
alter 'call SUPER3 CASECC ,CASET' $
type parm,,i,y,random=1
$ Return code. 0 Successful.
type parm,,i,y,irtn
$ type parm,,i,y,punch =7
$
$ if (random > 0) then $
message //'' '/$
message //'' Running pch2dac...... '/$
message //'' '/$
$
$ Following ishell parameter pass the unit number of the
$ two file in position number 27 and 28. Physical file
$ names associated with these unit numbers will be passed automatically
$ at position numbers 31 and 32.
$
$ 1 2 3 4 5 6
ishell //pch2dac2'/irtn/ 0/ 0/ 0/ 1/ 1/
$ 7 8 9 10 11 12 13 14 15 16 17 18
/ / / / / / / / / / / / /
$ 23 24 25 26 27 28 29 30
/ / / /punch/ / / /

$ if (irtn <> 0) then $
message //'' '/$
message //''USR_DAC failed with return code '/ irtn $
message //'' '/$
else
message //'' '/$
message //''USR_DAC ran successfully.'/ $
message //'' '/$
endif $ endif for irtn
endif $ endif

pch2dac2.txt

#!/bin/ksh -f
############################################################################
#THIS SHELL CONVERTS SDISPLACEMENT DATA FROM THE PUNCH FILE TO asc FORMAT
#FOR USE IN THE FATIGUE MODAL SUPERPOSITION METHOD
############################################################################
#GET PUNCH FILE NAME
#
set -A argarray "$@" # load arguments into an array
set -a
FILE1=${argarray[30]}
# FILE1='s112.pch'
#
pch=$FILE1
#
#DETERMINE NUMBER OF MODES
nm=`cat $pch | grep 'POINT ID' | tail -1 | awk '{print $4}'`
#
#DETERMINE NUMBER OF LINES WHICH CONSTITUTE THE DATA FOR EACH MODE
nl=`cat $pch | grep 'POINT ID =           2' | awk '{print $5}'`
nl=`expr $nl - 7`
#
#SET LINE START (ls) AND LINE END (le) VARIABLES FOR FIRST LOOP PASS
ls=8
le=$nl
#
mc=1
while [ $mc -le $nm ]
do
#PRINT THE NUMBER OF LINES FOR THE FIRST MODE TO MODE FILE 1
echo ${ls},${le}p > sedcom
cat $pch | sed -nf sedcom > mode_${mc}.asc
#remove extraneous data
cat mode_${mc}.asc | sed '/-CONT-/d' | awk '{print $3}' > mode_${mc}.tmp
mv mode_${mc}.tmp mode_${mc}.asc
#CONVERT ASC DATA TO DAC
matd /mode=singlechannel /inp=mode_${mc}.asc /out=mode_${mc}.dac /ov=yes /sam=100 /head=0 /all=yes /lab=Magnitude /uni=SDISPLACEMENT /mode=exit
#ADVANCE THE COUNT
ls=`expr $le + 8`
le=`expr $le + $nl`
mc=`expr $mc + 1`
done
#CLEAN UP FILE NAMES
mv mode_1.asc  mode_01.asc
mv mode_2.asc  mode_02.asc
mv mode_3.asc  mode_03.asc
mv mode_4.asc  mode_04.asc
mv mode_5.asc  mode_05.asc
mv mode_6.asc  mode_06.asc
mv mode_7.asc  mode_07.asc
mv mode_8.asc  mode_08.asc
mv mode_9.asc  mode_09.asc
#
mv mode_1.dac  mode_01.dac
mv mode_2.dac  mode_02.dac
mv mode_3.dac  mode_03.dac
mv mode_4.dac  mode_04.dac
mv mode_5.dac  mode_05.dac
mv mode_6.dac  mode_06.dac
mv mode_7.dac  mode_07.dac
mv mode_8.dac  mode_08.dac
mv mode_9.dac  mode_09.dac
rm *.asc