ANALYSIS OF COMPLEX INTERACTIONS BETWEEN MECHANICAL ELEMENTS USING ANSYS AND SIMULINK

S. Sztendel, C. Pislaru, A. Longstaff, A. Myers, S. Fletcher, D. Ford
University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK

ABSTRACT

Modelling and computer simulation play an important role in all engineering disciplines. For the coupling of different simulation tools interface are necessary, including both aspects of physics, numerical methods, software engineering, etc. This paper describes the development of the interface between MATLAB (as a topology optimisation tool) and ANSYS (computer aided engineering software used for finite element analysis). The setting of the project application in Workbench (ANSYS toolbox) using APDL programming is described and then the ANSYS results are imported into SIMULINK.

Keywords: ANSYS, Workbench, Interface, MATLAB, FEA, Integrated modelling

1 LITERATURE REVIEW.

Yoon and Manurung (2011) address the development of an intuitive user interface for controlling a hydraulic backhoe. To build the simulator, they created a 3D model of the backhoe using SolidWorks 2007 SP0. Then 3D model was then imported into SimMechanics along with joint information, and then configured as input–output control blocks. The simulator was built using MATLAB and SIMULINK with SimMechanics and the Virtual Reality Toolbox.

Li and Tan (2007) developed a link between ANSYS and Matlab to transfer the data and the system control between these two software packages. The EM atomic fluxes are calculated in ANSYS and the atomic flux results are sent to Matlab for the computation of the AFD.

Bathelt and Jönsson (2003) presented implementation of a virtual machine is the first one using xPC Target. A common PC is used to run the MathWorks products MATLAB, SIMULINK, Real-Time Workshop, SimMechanics and the Virtual Reality Toolbox. Statespace models generated from FEMLAB can also be used in Simulink.

Dynamic Analysis and Design System (DADS (2011)) is a computer simulation tool that is used to predict the behaviour of single or multibody mechanical systems. DADS can also be interfaced with other software such as OPTIMUS, MATLAB AND Pro/ENGINEER. Virtual.Lab Motion is the next generation of the popular DADS product. The process of connecting SIMULINK with Virtual.Lab Motion is fully automated and designers have a minimum amount of work to do in using the combined solution process.

Structural Dynamics Toolbox (SDT (2011)) enhances MATLAB core capabilities in controls and signal processing through extensions linked to general 3-D finite element modeling, experimental modal analysis, test/analysis correlation. SDT FEMLINK provides interfaces with popular formats like NASTRAN, ANSYS, ABAQUS, SAMCEF, PERMAS.

SIMPACK-Interface to finite element programs (FEMBS(2011) enables the user to incorporate Finite-Element-models into SIMPACK for the representation of flexible bodies in mechanical systems. SIMAT is Linear System Interface which exports linearised *SIMPACK* model (Figure 1). The MBS model is evaluated in SIMPACK, while the controller is simulated within SIMULINK environment.

1D engine & gas dynamics simulation software package from Ricardo Software (Ricardo API(2011)) makes a link between ANSYS FLUENT and WAVE so ANSYS users are able to execute simultaneously with WAVE through the. There are occasions when it is not possible to model the entire flow domain in an engine using 3D CFD due to the excessively large mesh and subsequent run-time. CAE analysis programs with the ability to communicate during the simulation are the most effective tools to design fully integrated systems.

2 DEVELOPMENT OF ANSYS SCRIPT WHICH WILL BE USED IN MATLAB INTERFACE

To interface ANSYS with Matlab we need a script that controls the running of both programs, plus data transfer between them (APDL (2011)). So either we need to write a script in Matlab, ANSYS or in a standalone DOS script. Matlab probably has advanced programming language and is already running some of programs. We suggested writing Matlab m file that for a given input set runs APDL input file for ANSYS to run, and use Matlab to edit the parameters in the file.

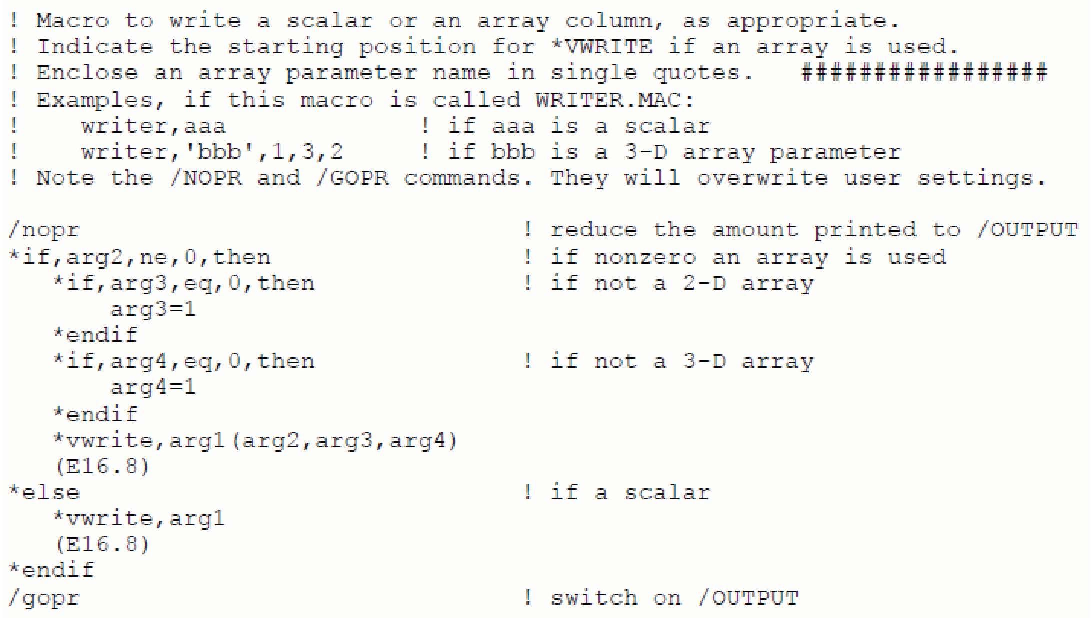
In input file (for ANSYS) we include any post processing that is required, and acquired specific results using '*get' or '*vget'. Then we write vectors to a text file using '*vwrite'. These can all be written to the same file, or to separate ones. Now, we can read the text file from your Matlab m file and transfer the parameters to Matlab.

The *VWRITE (ANSYS(2011) command can be used to output an array column, in addition to scalar parameters. The array position must be indicated when executing the *VWRITE command. *VWRITE command cannot be executed inside the GUI, it has to be executed from an input file or macro. The *VWRITE command prints the data from the starting position on down to the end of the column.

The output that results can be re-directed with the /OUTPUT or with the *CFOPEN and *CFCLOS commands. The following two macros can be used to make calling the *VWRITE command easy. The array must exist, having been created with a *DIM command.

The macro for a 3-dimensional array could be written so that it tests ARG2, ARG3, and ARG4 to see if they are zero. If they are zero, then they presumably were not entered, and the correct form of a *VWRITE command could be used to print a scalar, 1-D array, 2-D array, or 3-D array, as appropriate.

Its use would be very error prone without error checking code. A scalar need not have its name enclosed in single quotes in calling this macro, but an array would have to be enclosed in single quotes as in the above example.



```
! Macro to write a scalar or an array column, as appropriate.
! Indicate the starting position for *VWRITE if an array is used.
! Enclose an array parameter name in single quotes.   #################
! Examples, if this macro is called WRITER.MAC:
!    writer,aaa              ! if aaa is a scalar
!    writer,'bbb',1,3,2      ! if bbb is a 3-D array parameter
! Note the /NOPR and /GOPR commands. They will overwrite user settings.

/nopr                                ! reduce the amount printed to /OUTPUT
*if,arg2,ne,0,then                   ! if nonzero an array is used
   *if,arg3,eq,0,then                ! if not a 2-D array
       arg3=1
   *endif
   *if,arg4,eq,0,then                ! if not a 3-D array
       arg4=1
   *endif
   *vwrite,arg1(arg2,arg3,arg4)
   (E16.8)
*else                                ! if a scalar
   *vwrite,arg1
   (E16.8)
*endif
/gopr                                ! switch on /OUTPUT
```

The exclamation point character '(!)' sometimes called bang, is a *shell escape* and indicates that the rest of the input line is a command to the operating system. Use it to invoke utilities or call other executable programs without quitting MATLAB.

3 RUNNING EXTERNAL PROGRAMS FROM MATLAB

After the external program completes or we quit the program, the operating system returns control to MATLAB. We can add '&' to the end of the line, such as '!dir &' on Windows platforms to display the output in a separate window or to run the application in background mode. For example 'excel.exe &' opens Microsoft Excel software and returns control to the Command Window so you can continue running MATLAB language statements.

We can run created files that contain code in the MATLAB language. For a MATLAB script, we use the run function and specify the full pathname to a script.

It is necessary to ensure that the file is in the current folder or on the search path; otherwise, MATLAB returns an undefined function or variable error or a file not found error. The first thing is to make the text file using ANSYS. We can save this data in the file called /blog/vectordata.txt.

To tell MATLAB what is the name of the file to use we need to type: 'filen='W:/blog/vectordata.txt'

Now the file needs to be opened by using the ‘fopen’ command. This also assigns an integer to the file called the ‘filehandler’. This integer is automatically assigned by MATLAB and is a unique integer for any open file. The ‘r’ stands for the file being opened for reading. The other choices are ‘w’ for writing and ‘a’ for appending : ‘fh=fopen(filen,’r')‘

Now we need to get the data from the file which is identified through the file identifier variable. The ‘fgets’ command does this and every time ‘fgets’ is used, it reads the next line :’A_vector=fgets(fh)’

However, this reads the whole line of data as a single string so we need to get it as a vector A with the few entries. We do this by using the sscanf command that parses the string variable into the format used by ‘sscanf’. The format used here is ‘%f’ which is a floating point format. The other formats for reading numbers include %e for scientific format and ‘%d’ for integers. ‘A=sscanf(A_vector,’%f’)’

The data and system control exchange method can be applied to any simulation involved finite element analysis using ANSYS to enhance the data processing capability.

The MATLAB load Command

There is more than one way to read data into MATLAB from a file. The simplest, though least flexible, procedure is to use the load command to read the entire contents of the file in a single step. The load command requires that the data in the file be organized into a rectangular array. One useful form of the load command is ‘load name.ext’ where `name.ext' is the name of the file containing the data. The result of this operation is that the data in `name.ext' is stored in the MATLAB matrix variable called ‘name’. The `ext' string is any three character extension, typically `dat'. Any extension except `mat' indicates to MATLAB that the data is stored as plain ASCII text. A `mat' extension is reserved for MATLAB matrix files.

4 INTERFACING MATLAB AND ANSYS

The plan was to use finite element to produce differential equations representing a nodal structure. Then the computational and simulation environment converted them to a modal version and then to a reduced order space-state model, easy to be analyzed with the traditional control system theory and tools.

For initial Set up we start ANSYS to Set Working Directory, Preferences and Specify Initial job name. We next select the appropriate element type(s) for our problem and specify Material Properties. We make the model into finite elements to get the corresponding volume meshes (Figure 2).

All data required for the analysis should be specified in Specify Boundary Conditions & Loading-master degrees of freedom in a reduced analysis, appropriate convergence criteria for nonlinear analyses, frequency range for harmonic response analysis, and so on. Before we start the solve solution, we check our model for errors. The ANSYS is very fault prone, because of substantial actions in the graphical environment.

The SOLUTION processor is used to define the analysis type and analysis options, apply loads, specify load step options, and initiate the finite element solution. Post Processing plots the deformed and un-deformed shapes in the Graphics window (Figure 3). The structural responses, such as strains and stresses are calculate and saved in arrays (Figure 4) using the ANSYS command ‘‘*VWRITE’’, the value of the respective deformation is written into a file in a formatted sequence. ‘‘*VWRITE’’ command have to be contained in an externally prepared file and read into ANSYS. We have made a ‘namedSelection’ for the position where the displacement output of the model is required. From the Named selection the ANSYS command script have saved a file, txt.

Parametric Design Language ADPL from ANSYS is a very powerful tool, and allows making a script to manage variety of sensitivity studies in a short time. Only the essential parameters and inputs are required from the ANSYS Workbench (Figure 5). After the data is extracted from ANSYS and stored in a text file, the string/SYS, ‘C:MATLAB.exe/r filename.m’ is executed. By executing the string, the system control will be passed to Matlab environment without quitting ANSYS and the ‘filename.m’ is the Matlab m file used to extract the ANSYS data from the txt file. In the m file, the Matlab command ‘‘load’’ and a few nested ‘‘for’’ loops are used to read and sort the ANSYS data into Matlab from the txt file to calculate the flux divergence in a 3-D matrix.

To quit Matlab, the command ‘‘exit’’ is executed, and the system control will then return to ANSYS and continue with the ANSYS commands before the control was passed to Matlab. To show that the scripts are working a final SIMULINK block with imported data is presented here**-** (see Figure 6).

5 CONCLUSIONS

Whilst perfectly suited to model the elasticity ANSYS-FEA-Software is too limited to realize a full simulation of mechanism together by the process. Matlab, however, offers great freedom to develop own algorithms and routines while still providing a lot of well tested functions out of the box and is thus a good choice for simulation.
The simulation is launched in MATLAB then moving parameters and the input file to ANSYS. A detailed finite element model of the mechanism has been constructed using the FE package. In ANSYS we have set the project in the workbench with essential command lists. Setting data in ANSYS and Inputting data to Matlab has been performed automatically according to a pre-coded command flow without outside intervention. The data and system control exchange method can be applied to any simulation involving finite element analysis using ANSYS to enhance the data processing capability.

REFERENCES:

YOON J., MANURUNG A. (2010) *Development of an intuitive user interface for a hydraulic backhoe* Automation in Construction, In Press, Corrected Proof, pp. 779–790
LI EI, TAN M (2007), *Enhanced finite element modelling of Cu electro-migration using ANSYS and Matlab,* Microelectronics Reliability, Vol. 47, Issues 9-11, Pages 1497-1501
BATHELT J. AND JÖNSSON A. (2003) How to Implement the Virtual Machine Concept Using xPC Target Nordic MATLAB Conference Kopenhagen, Denmark
LMS (2011) – FEA simulation software [online] http://www.lmsintl.com/ [accessed] 20 Nov 2011
SDT (2011) - Structural Dynamics Toolbox, [online] http://www.sdtools.com/ [accessed] 12 Dec 2011
SIMPACK (2012) - Multi-body Simulation Software [online] http://www.simpack.com [accessed] 15 Jan 2012
Ricardo (2011) – Engine simulation software [online] http://www.ricardo.com/ [accessed] 18 Dec 2011
Ansys (2012) - APDL Programmer's Guide [online] http://www1.ansys.com/customer/content/documentation/100/ansys/gapdl100.pdf [accessed] 16 Jan 2012
ANSYS (2011) - Finite Element Analysis and Optimization Introduction [online] http://ansys.net/?mycat=tnt_vwrite1 [accessed] 10 Nov 2011

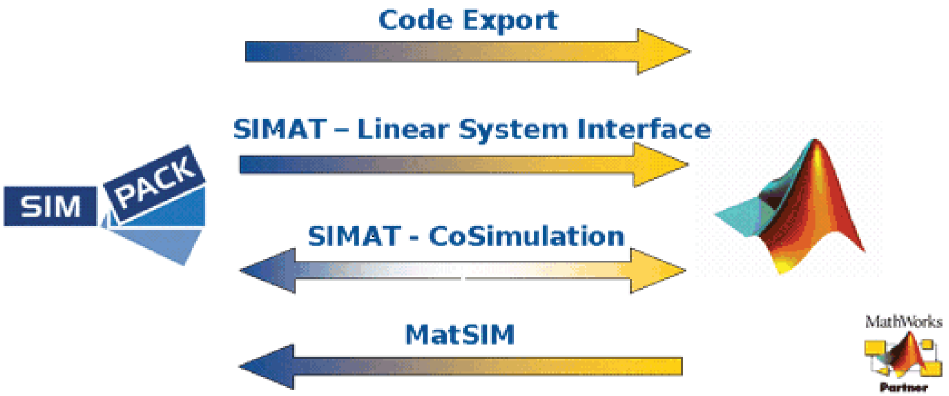
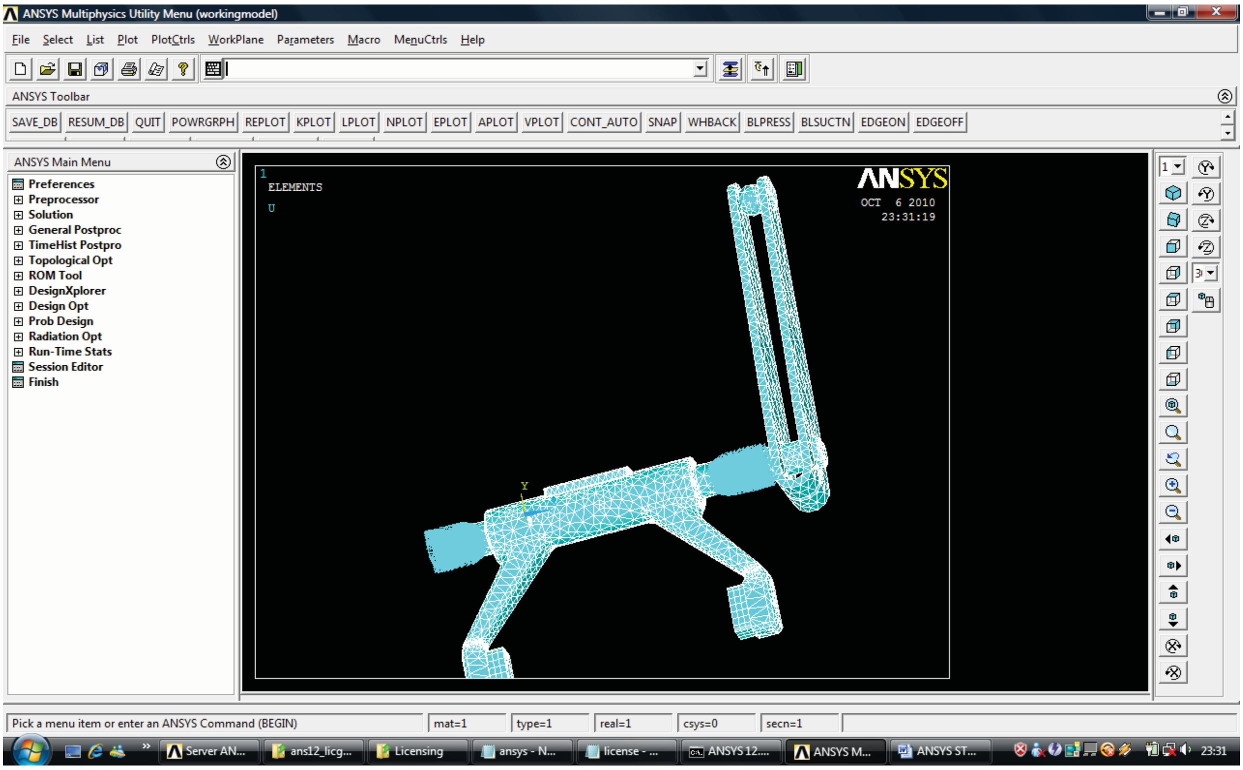


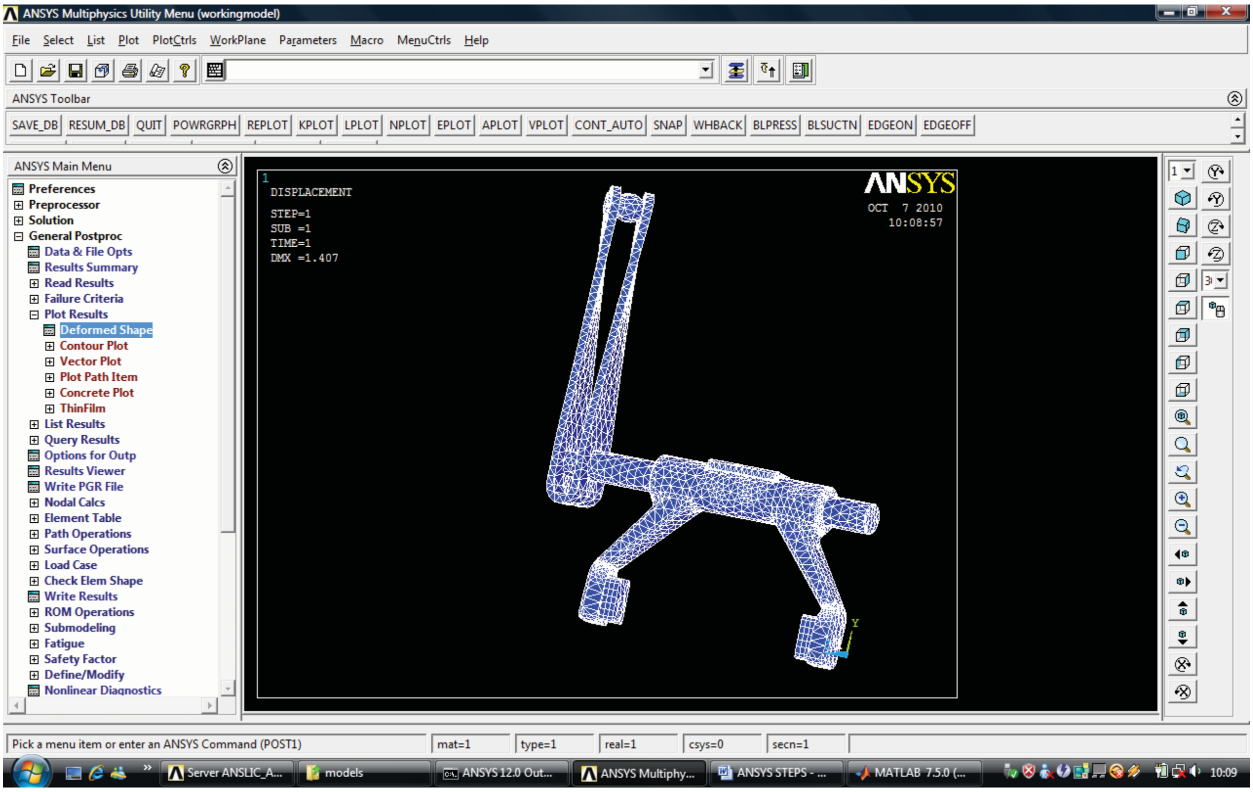
Figure 1 Co-Simulation Interface to MATLAB / SIMULINK models using SIMPACK (FEMBS, 2011)


ANSYS Multiphysics Utility Menu (workingmodel)
File Select List Plot PlotCtrls WorkPlane Parameters Macro MenuCtrls Help
ANSYS Toolbar
SAVE_DB RESUM_DB QUIT POWRGRPH REPLOT KPLOT LPLOT NPLOT EPLOT APLOT VPLOT CONT_AUTO SNAP WHBACK BLPRESS BLSUCTN EDGEON EDGEOFF
ANSYS Main Menu
Preferences
Preprocessor
Solution
General Postproc
TimeHist Postpro
Topological Opt
ROM Tool
DesignXplorer
Design Opt
Prob Design
Radiation Opt
Run-Time Stats
Session Editor
Finish
ANSYS
Y

```
1
ELEMENTS
U
OCT  6 2010
23:31:19
```

Pick a menu item or enter an ANSYS Command (BEGIN) | mat=1 | type=1 | real=1 | csys=0 | secn=1

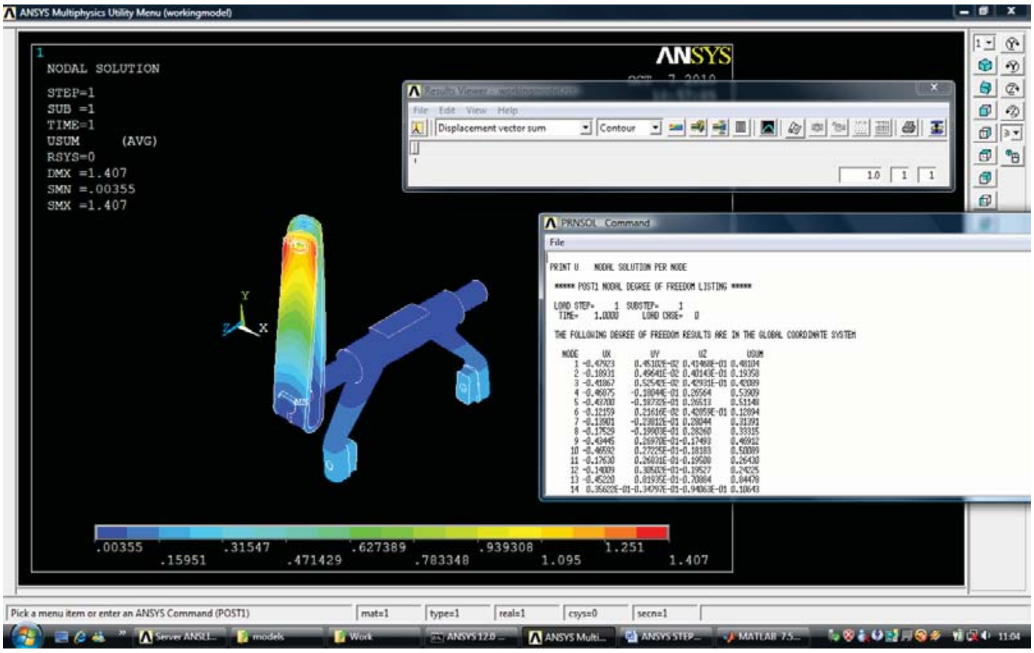
Figure 2 Modelled finite elements with corresponding volume meshes.



```
1
DISPLACEMENT
STEP=1
SUB =1
TIME=1
DMX =1.407
OCT  7 2010
10:08:57
```

Pick a menu item or enter an ANSYS Command (POST1) | mat=1 | type=1 | real=1 | csys=0 | secn=1

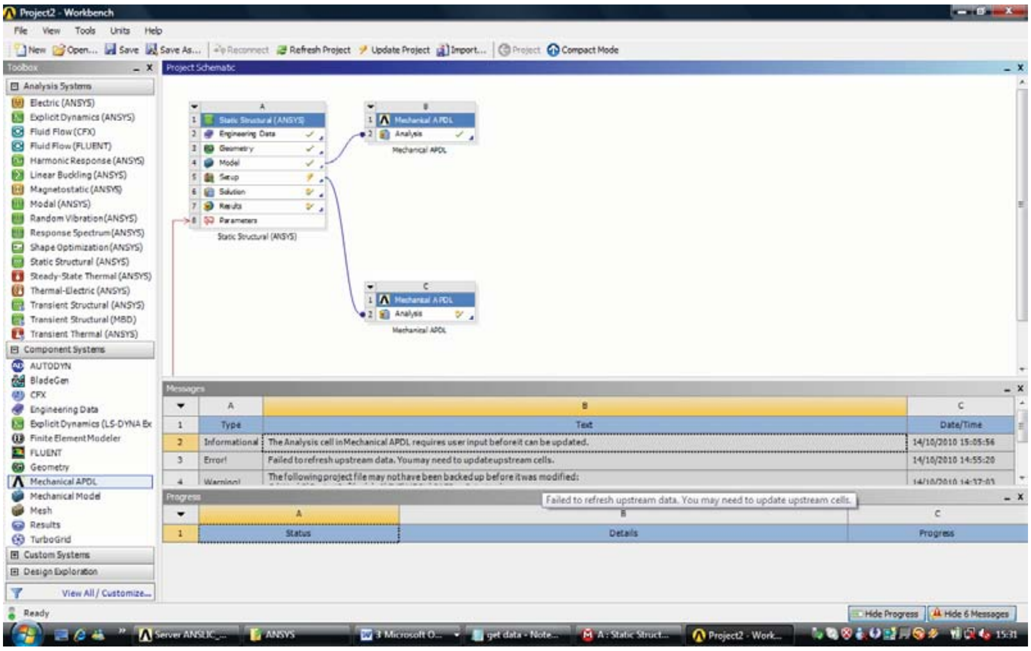
Figure 3 Post Processing plots - the deformed and un-deformed shapes



```
1
 NODAL SOLUTION

 STEP=1
 SUB =1
 TIME=1
 USUM      (AVG)
 RSYS=0
 DMX =1.407
 SMN =.00355
 SMX =1.407
```

```
PRINT U    NODAL SOLUTION PER NODE

 ***** POST1 NODAL DEGREE OF FREEDOM LISTING *****

 LOAD STEP=     1  SUBSTEP=     1
  TIME=    1.0000      LOAD CASE=   0

 THE FOLLOWING DEGREE OF FREEDOM RESULTS ARE IN THE GLOBAL COORDINATE SYSTEM

   NODE      UX          UY          UZ          USUM
      1 -0.47923      0.45102E-02 0.41468E-01 0.48104
      2 -0.18931      0.49641E-02 0.40143E-01 0.19358
      3 -0.41867      0.52542E-02 0.42931E-01 0.42089
      4 -0.46875     -0.18044E-01 0.26564     0.53909
      5 -0.43700     -0.18732E-01 0.26513     0.51148
      6 -0.12159      0.21616E-02 0.42859E-01 0.12894
      7 -0.13901     -0.23812E-01 0.28044     0.31391
      8 -0.17529     -0.19909E-01 0.28260     0.33315
      9 -0.43445      0.26970E-01-0.17493     0.46912
     10 -0.46592      0.27225E-01-0.18183     0.50089
     11 -0.17630      0.26831E-01-0.19508     0.26430
     12 -0.14009      0.30502E-01-0.19527     0.24225
     13 -0.45220      0.81935E-01-0.70884     0.84470
     14  0.35622E-01-0.34797E-01-0.94063E-01 0.10643
```

Figure 4 Results in the graphics window - can be accessed in text file.



	A	B	C
1	Type	Text	Date/Time
2	Informational	The Analysis cell in Mechanical APDL requires user input before it can be updated.	14/10/2010 15:05:56
3	Error!	Failed to refresh upstream data. You may need to update upstream cells.	14/10/2010 14:55:20
4	Warning!	The following project file may not have been backed up before it was modified:	14/10/2010 14:37:03

Failed to refresh upstream data. You may need to update upstream cells.

	A	B	C
1	Status	Details	Progress

Figure 5 Setting the project in Workbench (ANSYS toolbox)

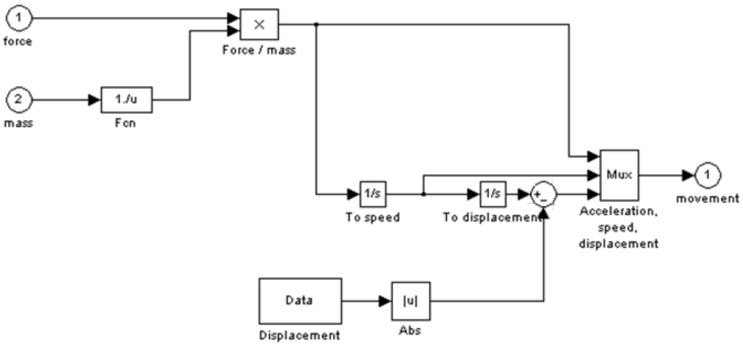


Figure 6 Importing acquired displacement data to SIMULINK model