MSC Nastran 2013.1

Release Guide

MSC Software

Corporate
MSC.Software Corporation
4675 MacArthur Court, Suite 900
Newport Beach, CA 92660
Telephone: (714) 540-8900
Toll Free Number: 1 855 672 7638
Email: americas.contact@mscsoftware.com

Europe, Middle East, Africa
MSC.Software GmbH
Am Moosfeld 13
81829 Munich, Germany
Telephone: (49) 89 431 98 70
Email: europe@mscsoftware.com

Japan
MSC.Software Japan Ltd.
Shinjuku First West 8F
23-7 Nishi Shinjuku
1-Chome, Shinjuku-Ku
Tokyo 160-0023, JAPAN
Telephone: (81) (3)-6911-1200
Email: MSCJ.Market@mscsoftware.com

Asia-Pacific
MSC Software (S) Pte. Ltd.
100 Beach Road
#16-05 Shaw Tower
Singapore 189702
Telephone: 65-6272-0082
Email: APAC.Contact@mscsoftware.com

Worldwide Web
www.mscsoftware.com

Disclaimer

Revision 0. November 26, 2013
NA:2013.1:Z:Z:Z:DC-REL

Contents

MSC Nastran 2013.1 Release Guide

Preface

Preface to the *MSC Nastran 2013.1 Release Guide*

This Release Guide contains descriptions for the MSC Nastran 2013.1 version, and supersedes the *MSC Nastran 2012.2 Release Guide*.

List of Books

Below is a list of some of the MSC Nastran documents. You may find any of these documents from MSC.Software at www.simcompanion.mscsoftware.com.

Installation and Release Guides
• Installation and Operations Guide • Release Guide

Reference Books
• Quick Reference Guide • DMAP Programmer's Guide • Reference Manual

User's Guides
• Getting Started • Linear Static Analysis • Dynamic Analysis • Embedded Fatigue • MSC Nastran Demonstration Problems • Thermal Analysis • Superelements • Design Sensitivity and Optimization • Implicit Nonlinear (SOL 600) • Explicit Nonlinear (SOL 700) • Aeroelastic Analysis • User Defined Services

Technical Support

For technical support phone numbers and contact information, please visit:
http://www.mscsoftware.com/Contents/Services/Technical-Support/Contact-Technical-Support.aspx

Support Center (http://simcompanion.mscsoftware.com**)**

Support Online. The Support Center provides technical articles, frequently asked questions and documentation from a single location.

Online Resources

MSC.Software (www.mscsoftware.com**)**

MSC.Software corporate site with information on the latest events, products and services for the CAD/CAE/CAM marketplace.

MSC Nastran Documentation

For quick access to the full set of MSC Nastran Documentation on Windows, one can:

1. Go to your `MSCNastran_Installation_DIR\msc20131\Doc\pdf_nastran\`
2. Click on `nastran_library.pdf` and use the Right Mouse Button to Create Shortcut
3. Move the shortcut to your Windows Desktop

1 Overview of MSC Nastran 2013.1

MSC.Software is pleased to introduce you to the exciting new technologies in MSC Nastran 2013.1, the premier and trusted CAE solution for aerospace, automotive, defense, and manufacturing industries worldwide. This release includes new features and enhancements in Contact, High Performance Computing, Acoustics, Aeroelasticity, and Explicit Nonlinear SOL 700.

Linear Analysis

- Axisymmetric Rotordynamics Enhancements (2013.1) (Ch. 2)
- Rotors in External Superelements (2013.1) (Ch. 2)
- Arbitrary Beam Cross Section (ABCS) Enhancements (2013.1) (Ch. 2)
- Enhancements to ACSRCE/RLOAD1/RLOAD2/ TOAD1/TLOAD2 Bulk Data Entries (Ch. 2)
- Support of Inter Component Force (ICF) for the FRF/FBA/TPA Capability (Ch. 2)
- Addition of New FBATOLR User Parameter for Use in the FBA Process (Ch. 2)
- Fatigue Analysis/Output Request (Ch. 2)
- Patran Support for MSC Nastran Embedded Fatigue (Ch. 2)

Acoustics

- Poroelastic Material (PEM) (Ch. 3)

Advanced Nonlinear (SOL 400)

- Contact User Interface Improvements (2013.1) (Ch. 4)
- Patran Support for MSC Nastran Contact Pair Modeling (Ch. 4)
- Linear Stress Recovery (2013.1) (Ch. 4)
- Sequential Thermal-Mechanical Analysis with Linear or Quadratic Temperature Distribution (2013.1) (Ch. 4)
- User Defined Subroutine (UDS) Improved Interface 2013.1 (Ch. 4)
- Enhancement to User Defined Subroutine Interface (Ch. 4)
- Enhancement to Enforced Relative Motion in NLTRAN for SOL 400 (Ch. 4)
- Support for Export of Adams MNF file in SOL 400 (Ch. 4)

Explicit Nonlinear (SOL 700)

- New Capabilities in Explicit Nonlinear (SOL 700) (2013.1) (Ch. 5)
- User Defined Services in SOL 700 (2013.1) (Ch. 5)
- New Material Models (Ch. 5)
- 1-D - 3-D Spherical-symmetric and 2-D - 3-D Axi-symmetric Mapping for Blast Loads (Ch. 5)
- Ignition Times for Multiple Detonations (Ch. 5)
- "LOAD_BLAST" Method for Empirical Blast Loadings (Ch. 5)
- Enhancements to FSI Algorithms to Speed Up the Simulation Time (Ch. 5)

Numerical Methods and High Performance Computing

- ACMS (Automated Component Mode Synthesis) Performance Improvements (2013.1) (Ch. 6)
- GPU Support (Ch. 6)
- New Options for MSCLDL and MSCLU Sparse Direct Solvers (Ch. 6)
- SOL 400 Parallel Performance Improvements (Ch. 6)
- New Memory Management Strategy (Ch. 6)

Optimization

- Fatigue Life Design Responses (Ch. 7)
- The Equivalent Radiated Power (ERP) Design Responses (Ch. 7)

Aeroelastic Enhancements

- Support for MONPNT2, MONPNT3 and MONSUM in Solution 146 (Ch. 8)

Miscellaneous

- Monitor Point Enhancements (2013.1) (Ch. 9)

2 Linear Analysis

Axisymmetric Rotordynamics Enhancements (2013.1)

Introduction

Until now, the rotordynamics capability in MSC Nastran has had the restriction that the rotors must be represented only as line models using bar or beam elements or as superelements reduced to a line model. This restriction has been removed in the 2013.1 version by allowing rotors to be also represented by axisymmetric configurations represented by newly developed axisymmetric harmonic elements. This facilitates the rotordynamic analysis of geometrically axisymmetric structures subjected to general non-axisymmetric loading represented by harmonics. This allows for a better capture of the geometry of the rotors and thus leads to better and more accurate predictions of the behavior of the rotordynamic models when compared to test results, so less experimental testing is needed.

In order to facilitate the above development, enhancements were made to the existing CQUADX and CTRIAX Bulk Data entries and three new Bulk Data entries (PAXSYMH, RBAX3D, and ROTORAX) were developed. The descriptions of all of these entries are given in the 2013.1 *MSC Nastran Quick Reference Guide* (QRG).

The details of the new capability are discussed in the following sections.

Elements

There are two elements. These are like solid elements since the degrees of freedom are displacements (not rotations). The QUADX element has four vertex points and up to four (optional) mid edge points. The center point of the QUADX is not allowed for harmonic analysis. The TRIAX element has three vertex points and up to three (optional) mid edge points. The connection entries list the vertex points first, and then any mid edge points. Points may be entered in either the clockwise or counterclockwise order. These become axisymmetric harmonic elements if they reference a PAXSYMH property entry. The PAXSYMH Bulk Data entry, whose description is given in the *MSC Nastran QRG*, specifies the harmonic index to be employed for the analysis as well as the element coordinate system.

Harmonic Analysis

Axisymmetric harmonic analysis is done using a cylindrical coordinate system. In the following, `x` is the radial coordinate, `y` is the axial coordinate, and `t` is azimuthal angle in an element coordinate system. Grid points lie in the (`x,y`) plane with `x` greater than or equal to zero. Harmonic analysis uses a term of a Fourier series for the azimuthal distribution of the solution variables.

For displacements:

```
Ux(x,y,t) = Uxnc(x,y) * COS(N*t) + Uxns(x,y) * SIN(N*t)
Uy(x,y,t) = Uync(x,y) * COS(N*t) + Uyns(x,y) * SIN(N*t)
Ut(x,y,t) = Utnc(x,y) * SIN(N*t) - Utnc(x,y) * COS(N*t)
```

For stresses:

```
Sxx(x,y,t) = Sxxnc(x,y) * COS(N*t) + Sxxns(x,y) * SIN(N*t)
Syy(x,y,t) = Syync(x,y) * COS(N*t) + Syyns(x,y) * SIN(N*t)
Stt(x,y,t) = Sttnc(x,y) * COS(N*t) + Sttns(x,y) * SIN(N*t)
Sxy(x,y,t) = Sxync(x,y) * COS(N*t) + Sxyns(x,y) * SIN(N*t)
```

```
Syt(x,y,t) = Sytns(x,y) * SIN(N*t) - Sytnc(x,y) * COS(N*t)
Stx(x,y,t) = Stxns(x,y) * SIN(N*t) - Stxnc(x,y) * COS(N*t)
```

N is the harmonic index and t (or theta) is the azimuthal angle. In this manner, non-axisymmetric solutions may be found for axisymmetric structures. For linear analysis, the harmonics are not coupled, and the harmonic analysis finds the solution for a single user specified N. he value for N may be 0, 1, 2, 3, …. The solution variables are not local displacements, but coefficients of either COS(N*t) or SIN(N*t). The coefficients of COS(N*t) represent solutions symmetric about the (x,y) plane while the coefficients of SIN(N*t) represent solutions anti-symmetric about the (x,y) plane. This is why Ux and Uy get multiplied by COS(N*t) while Ut gets multiplied by SIN(N*t). For N=0, Ut is uncoupled from the other two. For N=0, Ux and Uy are used for expansion and Ut is used for torsion.

Element Coordinate System

Each element refers to an element coordinate system defined on the PAXSYMH property entry. The elements must lie in the (x,y) plane of this element coordinate system. The axis of symmetry is the y axis in the element coordinate system. The radial coordinate is the x axis of the element coordinate system. The default element coordinate system is the basic system. By using CORD1R or CORD2R entries, the element coordinate system can be defined as desired. For example, if it is desired to use the basic x axis as the axis of symmetry and the basic y axis as the radial coordinate, use:

```
CORD2R, 10, 0, 0., 0., 0., 0., 0., -1., +C
+C, 0., 1., 0.
```

As another example, in order to use the basic z axis as the axis of symmetry and the basic x axis as the radial coordinate, use:

```
CORD2R, 10, 0, 0., 0., 0., 0., -1., 0., +C
+C, 1., 0., 0.
```

The element coordinate system ID (10 in this example) is specified in the PAXSYMH entry.

Material Properties

The material properties may be isotropic (MAT1) or anisotropic (MAT9). The material may be made temperature dependent by using the usual MATT1 or MATT9 records. The material temperature is assumed to be axisymmetric. For MAT9, the components 1 through 6 refer to the element coordinate system. 1 is xx, 2 is yy, 3 is tt, 4 is xy, 5 is yt, and 6 is tx. If MAT9 is used, it is required that the material be axisymmetric, so on the MAT9 record the following eight modulus components must be zero: G15, G16, G25, G26, G35, G36, G45, and G46. Material damping GE is supported.

Rotor Definition Via ROTORAX Entries

A configuration consisting of the new axisymmetric harmonic elements need not always define a rotating structure. It could very well define a stationary structure like the stator of an engine or a water tank. However, if it is part of a rotating structure, then the new ROTORAX Bulk Data entry needs to be employed to define the rotor to which the elements belong. As described in the *MSC Nastran QRG*, a ROTORAX rotor can be defined via the element IDs that comprise it, by the PAXSYMH property IDs that are referenced by such elements or by the GRID points that are on the axis of rotation or by a combination of the above.

Points on the Axis of Symmetry

A point is defined to be on the axis of symmetry if its x coordinate is less than 1.E-4 of any other point of the element. Points on the axis of symmetry require special constraints to ensure continuity of displacements. These constraints are automatically supplied by MSC Nastran, and the variables are not in the solution set. If the user does not supply SPCs, module GPSP will supply AUTOSPC constraints.

For N=0, Ux(0,y) = Ut(0,y) = 0.

For N=1, Uy(0,y) = Ut(0,y) = 0.

For N>1, Ux(0,y) = Uy(0,y) = Ut(0,y) = 0.

For N=1, an internal constraint Ut(0,y) = -Ux(0,y) has been supplied, which is why Ut(0,y) is not in the solution set.

Loads and Boundary Conditions

For loads, it is necessary to be consistent with the convention used for matrices. The stiffness, mass, and gyro matrices are for 2 pi radians.

The PLOADX1 entry is supported for pressure loads. For harmonic N greater than zero, the PA and PB fields of the PLOADX1 entry are coefficients of COS(N*t).

The GRAV load has been modified for the axisymmetric harmonic elements. All GRAV loads are in harmonic 0 or 1, higher harmonics will not be excited.

The RFORCE load on harmonic elements requires METHOD=2 which will be automatically set by the program. All RFORCE loads are in harmonic 0, 1, or 2.

The standard FORCE and SPC records can be used for loads and boundary conditions. The FORCE entries supply the coefficients of COS(N*t) or SIN(N*t). The magnitude of the FORCE record can be computed (currently by the user) using the principle of virtual work (see Appendix: Principle of Virtual Work).

For thermal expansion, the standard TEMPD and TEMP entries may be used, remembering that, for N>0, they supply the coefficients of COS(N*t).

Solution Sequences

The new axisymmetric harmonic elements are intended for linear analysis only. There is no support for differential stiffness or nonlinear forces. Design sensitivity is not supported, nor is heat transfer.

Element Stress/Strain Output

Element output is available for stresses and strains, both real and complex. These stress and strain values are the coefficients of COS(N*t) and SIN(N*t). Item codes for plotting are given in the following table:

Table 2-1 Element Stress-Strain Item Codes

	--- Real ---		---- Complex ----	
QUADX	1	(Elem ID)	1 (Elem ID)	
Harmonic	2	Harmonic	2 Harmonic	
(18)	3	CEN/grid	3 CEN/grid	
	4	Stress_xx	4 Stress_xx	RM
	5	Stress_yy	5 Stress_yy	RM
	6	Stress_tt	6 Stress_tt	RM
	7	Stress_xy	7 Stress_xy	RM
	8	Stress_yt	8 Stress_yt	RM
	9	Stress_tx	9 Stress_tx	RM
	10	VON MISES	10 Stress_xx	IP
	11-18	1st Corner	11 Stress_yy	IP
	19-26	2nd Corner	12 Stress_tt	IP
	27-34	3rd Corner	13 Stress_xy	IP
	35-42	4th Corner	14 Stress_yt	IP
			15 Stress_tx	IP
			16-28 1st Corner	
			29-41 2nd Corner	
			42-54 3rd Corner	
			55-67 4th Corner	

	--- Real ---		---- Complex ----	
TRIAX	1	(Elem ID)	1 (Elem ID)	
Harmonic	2	Harmonic	2 Harmonic	
(17)	3	CEN/grid	3 CEN/grid	
	4	Stress_xx	4 Stress_xx	RM
	5	Stress_yy	5 Stress_yy	RM
	6	Stress_tt	6 Stress_tt	RM
	7	Stress_xy	7 Stress_xy	RM
	8	Stress_yt	8 Stress_yt	RM
	9	Stress_tx	9 Stress_tx	RM
	10	VON MISES	10 Stress_xx	IP
	11-18	1st Corner	11 Stress_yy	IP
	19-26	2nd Corner	12 Stress_tt	IP
	27-34	3rd Corner	13 Stress_xy	IP
			14 Stress_yt	IP
			15 Stress_tx	IP
			16-28 1st Corner	
			29-41 2nd Corner	
			42-54 3rd Corner	

von Mises stresses and strains refer to the peak value for any value of location theta. Element strain energy (ESE) output is also available.

Vibration Modes

For harmonics, `N` greater than zero, all structural modes are doublets. The harmonic elements will only find one of the pair, the one that is symmetric with respect to the element (x,y) plane. For harmonic `N=0`, structural modes are not usually doublets. The harmonic elements will find both the expansion and torsion mode shapes. For a free body, there will be two rigid body modes for `N=0`, two rigid body modes for `N=1`, and none for `N>1`.

Gyroscopic Matrix

If N=1, a gyroscopic matrix is calculated and is used.

Connection Between Axisymmetric Harmonic Elements (QUADX and TRIAX) and Other Element Types

These elements may share grid points with other elements, however much care is needed. This connection is made between the harmonic displacement components of the axisymmetric harmonic elements and the global displacement components of other elements.

For harmonic zero, the axial degrees of freedom (in the element's y direction) may be connected to other elements. The element's y-displacement is that of the entire ring, and the element's y-force is that of the entire ring. Radial (element x) and azimuthal (element z) also represent the value for the entire ring.

For harmonic one, the radial degrees of freedom (in the element's x-direction) may be connected to other elements. The element's x-deflection represents the deflection at location theta equals zero. The element's t-deflection represents the negative deflection at location theta equals ninety degrees. One way is to combine these two to get the average component over the cross section.

```
Uaverage = (1/2) * ( Ux - Ut )
```

To see why this works, consider a rigid body motion, Ub, where b refers to the basic system. In the element's cylindrical coordinate system, this motion is represented by Ux = Ub*COS(t) and Ut = -Ub*SIN(t).

For harmonic greater than one, some complicated constraints will be needed.

Use of Constraint/Connector Element

A new RBAX3D connector has been developed to connect points of the axisymmetric harmonic elements to regular 3-D points on the axis of symmetry. This has only been implemented for harmonic one. The description of this entry is given in the *MSC Nastran QRG*.

The MPC entry may also be used with axisymmetric harmonic elements. RBE elements are generally not compatible with these elements since rotation degrees of freedom are not used. There are, however, a couple cases where RBE elements work. If there are two coincident grid points (no offset), an RBE element can be used to connect selected translation components. This can be used to simulate a spline connection for example. Another case that works is an RBE2 element with an independent point not connected to any axisymmetric harmonic elements.

Limitations

- The Beta release does not allow any axisymmetric rotors in a superelement. However, multiple line rotors can be specified as part of an external superelement.
- For line model rotors, damping specified via the GE value in a MAT1 entry shows no effects.
- For both line model rotors and axisymmetric rotors, PARAM,G is ignored. Instead, the structural damping must be specified via the GR value on an RSPINR/RSPINT entry.

- The CLAN method sometimes gives poor results, for small models the Hessian approach should be used.
- Mode tracking during Campbell diagram generation is not supported.
- The squeeze film dashpot user-defined service cannot be used with the new user subroutine method.

Sample Problem

In this sample problem, (taken from the Reference , M. Geradin, and N. Kill), critical frequencies of a realistic rotor modeled using axisymmetric elements are determined using SOL 107. As shown in Figure 2-1, the rotor is supported using bearings, where the bearing stiffness and damping are specified.

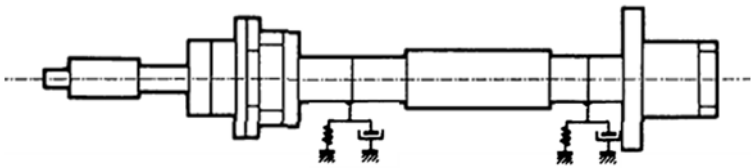


Figure 2-1 Nelson-McVaugh Rotor with Bearing Supports

The axisymmetric model for the above rotor is shown in Figure 2-2.

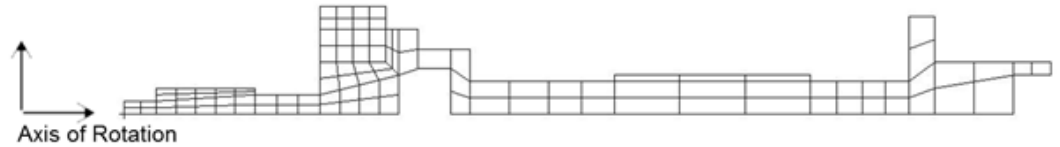


Figure 2-2 Axisymmetric Model for the Rotor

An axisymmetric rotor can be defined using the ROTORAX entries as shown here. Here the PROP option on a ROTORAX entry is used to define an axisymmetric rotor with a rotor ID of 10 comprised of all of the elements that reference PAXSYMH entries 1 or 2. In addition, the GRID option on a ROTORAX entry is used to define two points on the axis of rotation of this rotor.

```
ROTORAX 10      PROP    1       2
ROTORAX 10      GRID    1       2
```

The grid points defined above are also referenced on RSPINR entry.

```
RSPINR  10      1       2       RPM     1.
```

The operating RPM and analysis option (SYNC or ASYNC) can be defined using the RYRO entry.

```
RGYRO   100     SYNC    10      RPM             10000.
```

The CQUADX and CTRIAX elements in the axisymmetric rotor model are assigned harmonic properties by referencing PAXSYMH entries, as shown here. For most of the rotordynamics studies, harmonic analysis is performed with N=1, which is the default option. The CID field of the PAXSYMH entries defines a coordinate system 11 whose y-axis is the axis of rotation and whose x-y plane defines the two-dimensional plane of the axisymmetric harmonic element.

```
PAXSYMH 1       100     11      1
```

```
PAXSYMH 2       200     11      1
```

The coordinate system 11 referenced on the PAXSYMH entries is defined by:

```
CORD2R  11      0.      0.      0.      0.      -1.     +
+       1.
```

The axisymmetric CQUADX elements are defined in the usual manner.

```
CQUADX  1       1       19      53      56      55      0
CQUADX  2       1       53      20      57      56      0
```

In order to connect a 2-D axisymmetric grid point to a 3-D grid point, RBAX3D connector element is required. This is necessary when a bearing needs to be attached to an axisymmetric rotor model, as in this particular case. These connector elements can also be used to obtain a reduced line model for an axisymmetric rotor.

```
RBAX3D  993     9914    14
RBAX3D  994     9911    11
```

Once the 3-D grid point connected to an axisymmetric grid point is available, the CBUSH element can be defined for the 3-D grid point in the usual manner.

```
CBUSH   991     101     9911                            0
CBUSH   992     101     9914                            0
PBUSH   101     K       4.38E+074.38E+07                +
+               B       0.      0.
```

Sample results obtained for critical frequencies using the SYNC option in SOL 107 with the CLAN method are given below.

```
          C O M P L E X   E I G E N V A L U E   S U M M A R Y
0  ROOT  EXTRACTION    EIGENVALUE     FREQUENCY   DAMPING
NO.      ORDER         (REAL)  (IMAG)   (CYCLES)    COEFFICIENT
1        1    -1.293826E-13  1.615508E+03  2.571161E+02  1.601757E-16
2        2    1.399302E-13   1.758892E+03  2.799363E+02  -1.591118E-16
3        3    2.559537E-12   4.678248E+03  7.445663E+02  -1.094229E-15
```

The results obtained using axisymmetric model are compared with those obtained using the beam model, and the results available in literature (Reference 1) in Table 2-2.

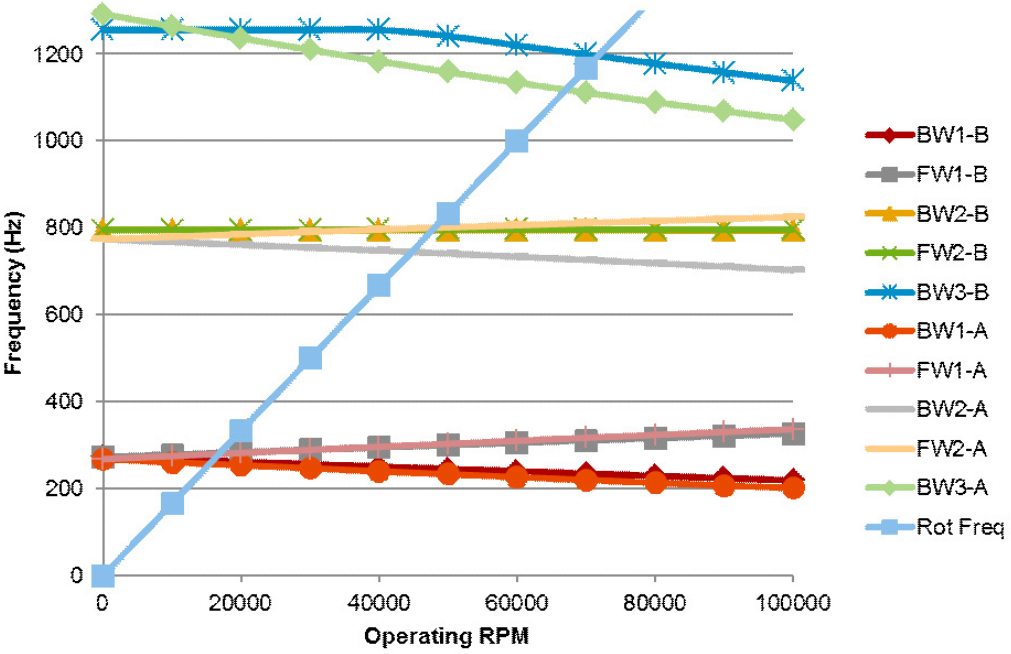
Table 2-2 Critical Frequencies

Ref. 1 (Hz)		MSC Nastran (Hz)		% Diff	
Beam	Axi	Beam	Axi	Beam	Axis
258.1	249.8	258.7	257	-0.23	-2.89
285.8	286.1	281.8	279.9	1.4	2.46
779.1	759.5	794.3	744.5	-1.96	1.97
835.4	826.6	795.5	800	4.77	3.22
1081	1040.9	1117.7	1117	-3.4	-7.31
1598	1537	1522.3	1528	4.74	0.58

The MSC Nastran data deck for this sample problem is called tpl/axiharm/`axh107a1.dat` and it is included as part of the delivery materials.

Below is the Campbell diagram showing the frequency response as a function of the rotary speed.

Campbell Diagram



Note: BW: Backward Whirl, FW: Forward Whirl, A: Axisymmetric Model, B: Beam Model.

Appendix: Principle of Virtual Work

This appendix shows how to convert loads distributed over a surface or volume to equivalent grid point loads. For a surface load `P` (unit: force/area):

```
Fi = INTEGRAL  Ni * P * dA
```

For a volume load `Q` (unit: force/volume):

```
Fi = INTEGRAL  Ni * Q * dV
```

where `Fi` is the equivalent load for component `i`, and `Ni` is the virtual displacement for grid point component `i`. The integration must be over the entire surface or volume (not just one radian). The integrals must be done in the element's cylindrical coordinate system since the shape functions refer to cylindrical coordinates. The harmonic shape functions are simply the 2-D isoparametric shapes multiplied by the appropriate `COS(N*t)` or `SIN(N*t)`.

Here is an example to show how this works. Consider a quadrilateral ring element with radius X1<x<X2 and axial y1<y<y2. Find the equivalent loads for a distributed gravity load perpendicular to the axis. First, represent the gravity in cylindrical (x,y.t) coordinates:

```
Qx = +rho*G*COS(T)
Qt = -rho*G*SIN(t)
```

Lateral gravity loads only harmonic N=1. For point 1 at x=X1 and y=Y1, the shape function for Ux is:

```
N1(x,y,t) = {(X2-x)/(X2-X1)}  * {(Y2-y)/(Y2-y1)} * COS(t)
```

This simple expression is valid only for a rectangular shape, for other shapes, a text on isoparametric analysis should be referenced. The shape function for Ut is:

```
N1(x,y,t) = {(X2-X)/(X2-X1)}  * {(Y2-y)/(Y2-y1)} * SIN(t)
dV = x * dt * dx * dy
```

Evaluating the integrals for point 1 on inner radius gives

```
Fx1 = +Ftot* (x2+2*x1) / {12*(X2+x1)}
Ft1 = -Ftot* (x2+2*x1) / {12*(X2+x1)}
```

where `Ftot = rho*G*pi*(X2^2-X1^2)*(Y2-Y1)` is the total force on the rectangular ring.

For point 2 at outer radius, x=X2 and y=Y1.

```
Fx2 = +Ftot* (2*x2+x1) / {12*(X2+x1)}
Ft2 = -Ftot* (2*x2+x1) / {12*(X2+x1)}
```

etc.

Reference

M. Geradin, and N. Kill, “A new approach to finite element modeling of flexible rotors”, Engineering Computations, 1993, Vol. 1 Issue: 1, pp.52 - 64

Rotors in External Superelements (2013.1)

Until now, the rotordynamics capability in MSC Nastran has had the restriction that the rotors must be completely contained within the residual structure of the model. Engine manufactures would find it much more convenient to deliver their rotor models to primary airframe manufactures as an external superelement rather than, as is currently done, as part of a residual structure. A limited capability to allow rotors in external superelements is provided in the 2013.1 release.

Input Requirements and Limitations

The creation of the external superelement with rotors is supported in SOL 108 using the EXTSEOUT case control command and the RGYRO case control command. The EXTSEOUT command must specify the `DMIGOP2 =` unit method of output. The RGYRO command must select a RGYRO bulk data entry. All rotor specific bulk data entries must be included the residual structure of the external superelement creation run. This includes the following entries:

ROTORG, ROTORSE, RGYRO, ROTORAX, RSPINR, and RBAX3D

Transient analysis and the RSPINT is not supported in this release. For the ROTORAX, the connections to 3-D structure using the RBAX3D must be made in the residual structure of the external superelement.

Example Problem

This example problem use a generic non-proprietary engine/strut test model developed for the IPAP project. This example uses five previously generated external superelements to assemble the engine, nacelle, and strut external superelement. This engine model includes three rotors defined with ROTORSE entries.

The MSC Nastran data decks for this sample problem are included in tpl/axiharm/* and it is part of the delivery materials. Relevant decks are:

- wing_gen.dat (Superelement Run)
- struct_gen.dat (Superelement Run)
- nac_gen.dat (Superelement Run)
- iprotor_gen.dat (Superelement Run)
- lprotor_gen.dat (Superelement Run)
- hprotor_gen.dat (Superelement Run)
- assembly_sol108.dat (Assembly Run)

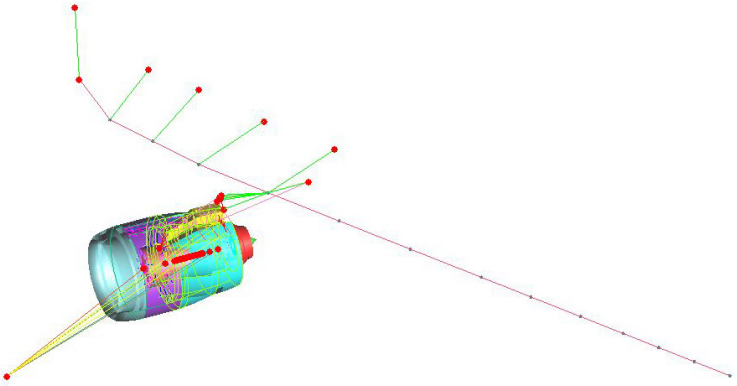


Figure 2-3 Wing/Engine Model

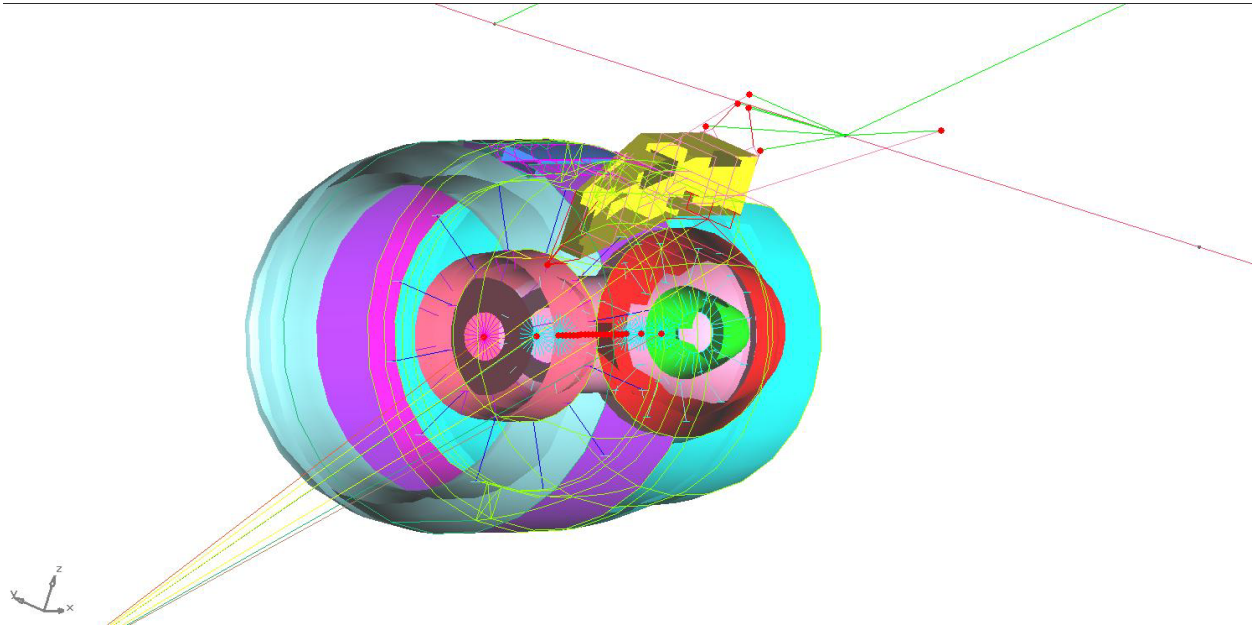


Figure 2-4 Strut and Engine External Superelement Model

Example Problem Input Data for External Superelement

```
$
$ run deck for FRF generation, external part superelements, rotor dynamics,
$ interface springs in the residual
$
NASTRAN BUFFSIZE=65537
$------------------------------------------------------------------------
$ FMS BEGIN
$------------------------------------------------------------------------
```

```
ASSIGN INPUTT2='hp_extse_op2' UNIT=35 $
ASSIGN INPUTT2='ip_extse_op2' UNIT=34 $
ASSIGN INPUTT2='lp_extse_op2' UNIT=33 $
ASSIGN INPUTT2='nac_extse_op2' UNIT=31 $
ASSIGN INPUTT2='strut_extse_op2' UNIT=32 $
ASSIGN INPUTT2='wing_extse_op2' UNIT=36 $
$------------------------------------------------------------------------
$ Executive CONTROL BEGIN
$------------------------------------------------------------------------
$
ID partSE4
DIAG 8,9,15,56 $
SOL 108
TIME 100

CEND
$
$------------------------------------------------------------------------
$ CASE CONTROL BEGIN
$------------------------------------------------------------------------
$
TITLE=GENERIC ENGINE COMPLIANCE ANALYSIS    02-FEB-2007
SUBTITLE=INTEGRATED MODEL, FIXED AT SOB, NASTRAN ROTORDYNAMICS
        ECHO=SORT
        GROUNDCHECK(DATAREC=YES,SET=(G))=YES
        SPC=1
$
        PARAM,POST,-1
$
$  superelements
SUBCASE 10
        SUPER = 10
        LABEL = WING
        METHOD=100
$       output requests
        SET 9003 = 500000
        SPCFORCE(PHASE)=9003
SUBCASE 50
        SUPER = 50
        LABEL = STRUT
        METHOD=200
$       output requests
        SET 9002 = 190101 thru 190103
        ELFORCE(PHASE)=9002
SUBCASE 60
        SUPER = 60
        LABEL = NACELLE
        METHOD=200
$       output requests
        SET 9002 = 190013 190014
        FORCE=9002
        DISP=9002
SUBCASE 66
        SUPER = 66
```

```
        LABEL = LP ROTOR
$
SUBCASE 67
        SUPER = 67
        LABEL = IP ROTOR
$
SUBCASE 68
        SUPER = 68
        LABEL = HP ROTOR
SUBCASE 100
        SUPER=0
        SPC=1
        DISP(phase)=ALL
        rsdamp=1001
        dload=1 $ fan unbalance
        freq = 1 $
        rgyro=66
BEGIN BULK
$
$  correct generic engine parameters
$
PARAM,WTMASS,0.002588
PARAM,MAXRATIO,1.0E8
PARAM,SNORM,0.0
PARAM,K6ROT,0.0
PARAM,COUPMASS,0
PARAM,AUTOSPC,YES
EIGRL   200     -1.     200.
EIGRL   100     -1.     100.
$
$
$ constraints on s-t-w rotations for roundoff-zeros
SPC1    1       456     502222  502233  502244  502266  502177  502178
SPC1    1       456     110207  110208
$
$ Fix torsional dof's
SPC1            1       6  660001    THRU  660019
$ Fix torsional dof's
SPC1            1       6  670001    THRU  670017
$ Fix torsional dof's
SPC1            1       6  680001    THRU  680019
$ Bearings
SPC1            1     456  660077  670088  670078  680081
SPC1            1    3456  660063  670071  670715  680819  660619
$
$  fix the wing SOB
SPC1,1,123456,500000
$
$   damping definition
DAMPING 1001    0.05                            0.0
$
$
$ ---------------------------------------------------------------------
$  ROTOR DYNAMICS:
```

```
$ ------------------------------------------------------------------------
$RGYRO  RID     SYNCFLG REFROTR SPDUNIT SPDLOW  SPDHIGH SPEED
RGYRO   66      SYNC    66      FREQ    0.      100.
$
$  LP rotor
$
$ROTORSE.ROTORID.SEID   SEOPT
ROTORSE 66      66      1
$$$ROTORG ROTORID GRID1   THRU    GRID2   BY      INC
$$$ROTORG  66      660001  THRU    660019
$RSPINR ROTORID GRIDA   GRIDB   SPDUNIT SPTID
$       GR      ALPHAR1 ALPHAR2 HYBRID
RSPINR  66      660003  660001  FREQ    66
        0.01
DDVAL   66      0.      100.0
$
$  IP rotor
$
$ROTORSE.ROTORID.SEID   SEOPT
ROTORSE 67      67      1
$$$ROTORG ROTORID GRID1   THRU    GRID2   BY      INC
$$$ROTORG  67      670001  THRU    670016
$RSPINR ROTORID GRIDA   GRIDB   SPDUNIT SPTID
$       GR      ALPHAR1 ALPHAR2 HYBRID
RSPINR  67      670008  670001  FREQ    67
        0.01
DDVAL   67      25.0    175.0
$
$  HP rotor
$
$ROTORSE.ROTORID.SEID   SEOPT
ROTORSE 68      68      1
$$$ROTORG ROTORID GRID1   THRU    GRID2   BY      INC
$$$ROTORG  68      680001  THRU    680019
$RSPINR ROTORID GRIDA   GRIDB   SPDUNIT SPTID
$       GR      ALPHAR1 ALPHAR2 HYBRID
RSPINR  68      680018  680001  FREQ    68
        0.01
DDVAL   68      100.0   300.0
$
$ ------------------------------------------------------------------------
$  LOADS:
$ ------------------------------------------------------------------------
$
$
$RLOAD1 SID     DAREA   DELAY   DPHASE  TAB-R   TAB-I
RLOAD1  1       1               1       100
RLOAD1  2       2               2       100
$
$DAREA  SID     GID     CID     AREA    (GID)   (CID)   (AREA)
DAREA   1       660001  1       .1618-3
DAREA   1       660001  2       .1618-3
DAREA   2       660018  1       .1618-3
DAREA   2       660018  2       .1618-3
```

```
$
$DPHASE SID     GID     CID     PHASE   (GID)   (CID)   (PHASE)
DPHASE  1       660001  2       90.
DPHASE  2       660018  2       90.
$
$TABLED4ID      X1      X2      X3      X4                              +
$+      A0      A1      A2      ...     An      ENDT
TABLED4 100     0.      1.      0.      1.0E6
+       0.      0.      39.4784 ENDT
$
FREQ1   1       1.      .25     10
$
$ ------------------------------------------------------------------
$
INCLUDE 'ModelCoordinateSystem.bdf'
INCLUDE 'EngineCoordinateSystem.bdf'
INCLUDE 'InletCoordinateSystem.bdf'
INCLUDE 'FancowlCoordinateSystem.bdf'
INCLUDE 'ThrustReverserCoordinateSystem.bdf'
INCLUDE 'StrutCoordinateSystem-v02.bdf'
INCLUDE 'NozzleCoordinateSystem.bdf'
INCLUDE 'PlugCoordinateSystem.bdf'
INCLUDE 'LHwingCoordinateSystem-v06.bdf'
$ convert to strut locations
CORD2R* 1000                1               -770.00003437    348.20009554
*           -38.90547355   -770.00001719    337.75004476     60.54700977
*          -670.00003437    348.20009554    -38.90549083

$ ------------------------------------------------------------------
$ residual interface elements
$ ------------------------------------------------------------------
$
$ strut-to-wing interface elements
INCLUDE 'Strut-to-WingInterface_EVRN.bdf'
$
$ front mount monoball springs
INCLUDE 'Monoball_EVRN.bdf'
$ front mount interface
INCLUDE 'FrontMountInterface-v04_EVRN.bdf'
SPC1,1,456,190000
$
$  rear mount links
$
CROD      190101  190101  190303  633526
PROD      190101  190002   3.000
$MAT1   ID      E       G       NU      RHO     ALPHA   T-REF   GE
MAT1      190002.300E+08        .300E+00.300E+00                0.04
CROD      190102  190101  190304  633527
CROD      190103  190101  190305  633528
$
SPC1    1       456     633526  633527  633528
$
$
$ inlet to FCSB springs
```

```
INCLUDE 'InletFCSBInterface_EVRN.bdf'
$
$ fancowl hinge springs
INCLUDE 'FancowlHingesInterface_EVRN.bdf'
$
$ thrust reverser hinges
INCLUDE 'ThrustReverserHingesInterface_EVRN.bdf'
$
$ rotor bearings
INCLUDE 'BearingNo1LInterface_EVRN-v04.bdf'
INCLUDE 'BearingNo6Interface_EVRN-v04.bdf'
INCLUDE 'BearingNo1IInterface_EVRN-v04.bdf'
INCLUDE 'BearingNo25Interface_EVRN-v04.bdf'
INCLUDE 'BearingNo34Interface_EVRN-v04.bdf'
INCLUDE 'IntershaftBearingInterface_EVRN-v04.bdf'
$
include 'HPRotor_gen.asm'
include 'IPRotor_gen.asm'
include 'LPRotor_gen.asm'
include 'nac_gen.asm'
include 'Strut_gen.asm'
include 'wing_gen.asm'
$$$
include 'HPRotor_gen.pch'
include 'IPRotor_gen.pch'
include 'LPRotor_gen.pch'
include 'nac_gen.pch'
include 'Strut_gen.pch'
include 'wing_gen.pch'
ENDDATA
```

Arbitrary Beam Cross Section (ABCS) Enhancements (2013.1)

Introduction

Arbitrary Beam Cross Section (ABCS) capability has been enhanced with the removal of following three limitations.

1. Branch Point (BRP) must not start and/or end on the first and/or last point of OUTP
2. No intersection among BRPs
3. Only one BRP is allowed to branch out from each side of a point of OUTP

Benefits

With the removal of three limitations on BRP, some previously difficult to model ABCS can now be handled with ease, see test cases.

User Interface

ABCS is supported with PBRSECT and PBMSECT bulk data entries. There is no change on PBRSECT and PBMSECT. The only exception is the usage of BRP is relaxed with the removal of above three limitations.

Parameter

ARBMSTYP

ARBMSTYP is used to select method for ABCS. `'PARAM,ARBMSTY,VKI'` is no longer available. Instead, `'PARAM,ARBMSTYP'` is applicable only for composite beam which uses Core/Layer keywords in PBMSECT.

Limitations

- PBMSECT and MAT8 will cause problem.
- The Warping stiffness does not show any influence on solution.
- A fatal error occurs when segments are defined in counter-clockwise manner.

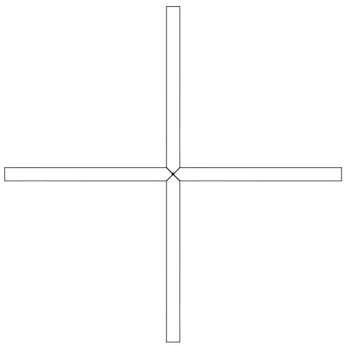
Test Cases

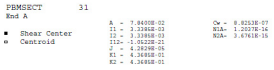
Multiple BRPs From a Single Point

For PBMSECT, 21, BRP(1), and BRP(3) are branched out from POINT,2.

```
point    1                -0.50     0.00
point    2                 0.00     0.00
point    3                 0.50     0.00
point    4                 0.00     0.50
point    5                 0.00    -0.50
```

```
$
$.......2.......3.......4.......5.......6.......7.......8.......9.......10.....
SET1    101     1       2       3
SET1    102     2       4
SET1    103     2       5
$ isotropic case using T keyword
PBMSECT 31      1       OP                                                  +
        OUTP=101,t=0.04,BRP(1)=102,BRP(3)=103
```

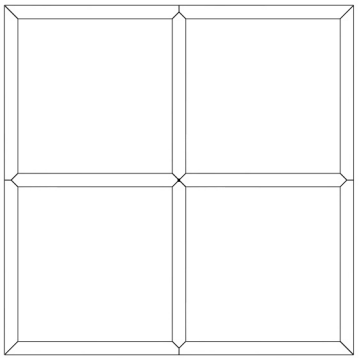




BRPs Intersect at Common Point

For PBMSECT, 33, BRP(1), and BRP(3) are intersected at POINT,29.

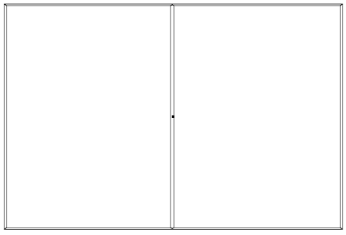
```
point   21                  -0.50    0.50
point   22                  -0.50    0.00
point   23                  -0.50   -0.50
point   24                   0.00   -0.50
point   25                   0.50   -0.50
point   26                   0.50    0.00
point   27                   0.50    0.50
point   28                   0.00    0.50
point   29                   0.00    0.00
$
$.......2.......3.......4.......5.......6.......7.......8.......9.......10.....
SET1    301     21      thru    28
SET1    302     22      29      26
SET1    303     28      29      24
$ isotropic case using T keyword
PBMSECT 33      1       CP                                                  +
        OUTP=301,t=0.04,BRP(1)=302,BRP(3)=303
```

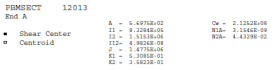


```
PBMSECT       33
End A

■  Shear Center          A  =  2.3120E-01          Cw =  2.1243E-04
□  Centroid              I1 =  2.9643E-02          Ma=  2.1734E-04
                         I2 =  2.9643E-02          Ma=  7.6239E-04
                         I12=  4.9170E-19
                         J  =  4.1067E-02
                         K1 =  4.9164E-01
                         K2 =  4.9164E-01
```

BRP Ends at the End Point of OUTP

```
For PBMSECT,12013, BRP ends at the end point 132520 of OUTP.
SET3,16,POINT,132517,THRU,132520
SET3,17,POINT,132519,132521,132522,132520
PBMSECT,12013,1,CP
        OUTP=16,
        BRP=17,
        T=1.,
        T(1)=[1.,PT=(132517, 132518)],
        T(2)=[0.75,PT=(132518, 132519)],
        T(3)=[1.5,PT=(132519, 132520)],
        T(4)=[0.75,PT=(132520, 132517)],
        T(5)=[0.75,PT=(132519, 132521)],
        T(6)=[1.,PT=(132521, 132522)],
        T(7)=[0.75,PT=(132522, 132520)]
```





Enhancements to ACSRCE/RLOAD1/RLOAD2/ TOAD1/TLOAD2 Bulk Data Entries

Several improvements have been made. These are described below.

Restriction of Unique Load Set Identification Numbers Removed

The ACSRCE/RLOAD1/RLOAD2/TOAD1/TLOAD2 entries for dynamic loads required a unique load set ID in the previous releases, forcing users to add a DLOAD Bulk Data entry to combine the dynamic loads even with unity as their scale factors. This restriction has been removed, allowing users to define multiple dynamic loads with the same load set ID.

Changes to Behavior of DLOAD Case Control Request and DLOAD Bulk Data Entry

The removal of restriction for unique Load Set IDs will change the behavior of DLOAD as used in the Case Control and Bulk data sections.

If a DLOAD Case Control request with a load set ID of `LID` points to simple dynamic load entries, then it will select ALL simple loads whose IDs match `LID`. If, instead, a DLOAD Case Control request points to a DLOAD Bulk Data entry, then any Si scale factor of this entry will apply to ALL simple loads whose IDs match the corresponding `Li` load set ID.

Real Values Allowed in Place of Table Identification Numbers

The ACSRCE/RLOAD1/RLOAD2/TLOAD1 entries reference one or more tables as part of their load definition. It is not uncommon in many cases for these tables to have the same constant value for all frequencies or throughout the time history. In such cases, there is no need to define a table since a single real value could be used to define such a table. With the MSC Nastran 2013 release, fields in these entries that reference table IDs may use real values over the range of frequencies of interest and time domain (as appropriate).

No Impact on Legacy Models

The enhancements mentioned above have no effect on legacy models which should continue to run as before.

Examples Illustrating the New Enhancements

The following examples illustrate the usage of the above enhancements.

Example 1. Using of Multiple Dynamic Load Entries With Unity Scale Factors

Old Usage

Case Control

```
DLOAD = 1000
```

Bulk Data

```
DLOAD,1000,1.0,1.0,100,1.0,200,1.0,300
,1.0,400
RLOAD1,100,101,…
RLOAD1,200,201,…
RLOAD2,300,301,…
RLOAD2,400,401,…
```

New Usage

Case Control

```
DLOAD = 1000
```

Bulk Data

```
RLOAD1,1000,101,…
RLOAD1,1000,201,…
RLOAD2,1000,301,…
RLOAD2,1000,401,…
```

(Note: There is no need for a DLOAD Bulk Data entry in this case since the DLOAD Case Control request selects ALL load entries with the same load set ID.)

Example 2. Using of Multiple Dynamic Load Entries With Non-Unity Scale Factors

Old Usage

Case Control

```
DLOAD = 1000
```

Bulk Data

```
DLOAD,1000,1.0,1.5,100,1.5,200,2.5,300
,2.5,400
RLOAD1,100,101,…
RLOAD1,200,201,…
RLOAD2,300,301,…
RLOAD2,400,401,…
```

New Usage

Case Control

```
DLOAD = 1000
```

Bulk Data

```
DLOAD,1000,1.0,1.5,100,2.5,300
RLOAD1,100,101,…
RLOAD1,100,201,…
RLOAD2,300,301,…
RLOAD2,300,401,…
```

Note: The Si scale factors in the DLOAD Bulk Data entry apply to ALL load entries with the corresponding load set ID of Li.)

Example 3. Using of Real Values in Place of Table IDs – RLOAD1

Old Usage

```
RLOAD1,100,200,,,300,400
TABLED1,300
,0.0,1.0,100.0,1.0,ENDT
TABLED1,400
,0.0,0.5,100.0,0.5,ENDT
```

New Usage

```
RLOAD1,100,200,,,1.0,0.5
```

Example 4. Using of a Real Value in Place of a Table ID – TLOAD1

Old Usage

```
TLOAD1,100,200,,,500
TABLED1,500
,0.0,1.5,100.0,1.5,ENDT
```

New Usage

```
TLOAD1,100,200,,,1.5
```

Support of Inter Component Force (ICF) for the FRF/FBA/TPA Capability

Introduction

Inter-component forces (ICFs) in an FBA process represent the forces that are acting at the connection points between and among the various components comprising the FRF assembly. These forces are helpful in understanding the load paths in the assembly and are thus useful for the design of the joints at the connection points. These ICFs were not available in earlier versions of MSC Nastran. MSC Nastran 2013 introduced the support of the ICFs in FBA. This allows for TPA (Transfer Path Analysis) to be performed using these ICFs. Details of the enhancements are discussed in the following sections.

Inter-Component Force (ICF) Processing

ICF processing in an FBA process involves two steps as follows.

Step 1:

- Specify user loads and run an FBA job to generate ICF information for specified FRF components and save the generated ICF information on an appropriate medium
- Depending upon user requests in Case Control, this step generates standard output. In addition, the user can request output of the computed ICFs using the new ICF Case Control request. This output is similar to OLOAD output.

Step 2:

- Using the saved ICF information from Step 1 and *the same loading condition as that in Step 1*, run an FBA job for a subset of the FRF components of Step 1.
- In this step, the relevant ICFs from Step 1 are treated as additional "pseudo" loads *in conjunction with the specified user loads* (if applicable) on the specified subset of FRF components.
- Depending upon user requests in Case Control, this step generates standard output. This output allows for TPA to be performed using ICFs for the specified assembly configuration.
- This step can be repeated with different subsets of the FRF components of Step 1 to study the effect of ICFs on different assembly configurations.

Combining Steps 1 and 2 in a Single Step FBA Job

The two steps discussed in previous section can be combined. This scenario of ICF processing involves two passes in a single FBA job execution as described below. However, internally, the program treats each of these two passes as a separate and distinct FBA process, each with a different assembly configuration.

Pass 1

- ICFs are generated for specified FRF components for specified user loads and are saved on an appropriate medium.

- Depending upon user requests in Case Control, this step generates standard output. In addition, the user can request output of the computed ICFs using the new ICF Case Control request. This output is similar to OLOAD output.

Pass 2

- This pass is regarded as a separate FBA process involving a subset of the FRF components of Pass 1.
- The relevant ICFs saved from Pass 1 are treated as "pseudo" loads *in conjunction with the specified user loads* (if applicable) on the specified subset of FRF components.
- Depending upon user requests in Case Control, this step generates standard output. This output allows for TPA to be performed using ICFs for the specified assembly configuration.

Check for Correctness and Validity of Step 2/Pass 2 Results

- The results for the FRF components employed in Step 2/Pass 2 must match their corresponding results from Step 1/Pass 1.
- The only difference between the Step 1/Pass 1 scenario and the Step 2/Pass 2 scenario is that the results of the former are due only to applied loads while those of the latter are due to a combination of appropriate ICFs and applicable applied loads.
- Because of the nature of the design, the validation of results will be automatically satisfied for the single step procedure. If this result validation is not satisfied for Step 2 of the two-step procedure, the user should make sure that the loading employed in Step 2 (if any) is the same as that in Step 1.

Enhancements to the FRF Case Control Command

The FRF Case Control command has been enhanced by the addition of several new keywords to facilitate ICF processing. These are described below.

New ICFGEN Keyword

This keyword specifies a list of FRF components. The FBA process generates ICFs for all FRF components specified in this list. These components are potential candidates to be employed in a subsequent FBA job or a subsequent FBA pass, with the appropriate computed ICFs acting on them, in conjunction with the original applied loads (if applicable).

ICFGEN = ALL

- Generate ICFs for all FRF components in the assembly

ICFGEN = *n* (nonzero integer)

- $n > 0$

  Generate ICFs for all FRF components in the assembly that are specified by SET ID *n*

- $n < 0$

  Generate ICFs for the single FRF component whose ID is given by $|gi|$

ICFGEN = *compname*

- Generate ICFs for the single FRF component whose name is given by *compname*

New ICFUSE Keyword

This keyword specifies a list of FRF components whose ICFs have been computed either in an earlier FBA job or an earlier FBA pass. These components are employed in an FBA process comprising only these components with the appropriate computed ICFs acting on them, in conjunction with the original applied loads (if applicable). This allows for TPA to be performed using these ICFs.

ICFUSE = *n* (nonzero integer)

- $n > 0$

  Employ only the FRF components that are specified by SET ID *n,* along with their ICFs and the original applied loads (if applicable)

- $n < 0$

  Employ the single FRF component whose ID is given by $|n|$, along with its ICFs and the original applied loads (if applicable)

ICFUSE = *compname*

- Employ the single FRF component whose name is given by *compname,* along with its ICFs and the original applied loads (if applicable)

New ICFAUTO Keyword

This keyword specifies a list of FRF components. Use of this keyword implies a single step FBA process and is equivalent to employing both ICFGEN and ICFUSE with the same specification as that of the ICFAUTO keyword in an FBA job.

The FRF components specified by ICFAUTO apply to both Pass 1 and Pass 2. In Pass 1, ICFs are computed for these FRF components for the specified user loads and saved on the appropriate medium. In Pass 2, the ICFs from Pass 1 are used in conjunction with the user loads to get the results for an assembly configuration involving only these FRF components. The results from Pass 2 allow for TPA to be performed for the ICFs.

ICFAUTO = *n* (nonzero integer)

- $n > 0$

  SET ID *n* specifies a list of FRF components. ICFs will be generated for them in Pass 1, with these ICFs being used in Pass 2 for an assembly configuration involving only these FRF components.

- $n < 0$

  $|n|$ specifies the ID of a single FRF component. ICFs will be generated for this component in Pass 1, with these ICFs being used in Pass 2 for a configuration involving just this single component.

ICFAUTO = *compname*

- *compname* specifies the name of a single FRF component. ICFs will be generated for this component in Pass 1, with these ICFs being used in Pass 2 for a configuration involving just this single component.

Using ICFGEN, ICFUSE, and ICFAUTO Keywords in the FRF

Only ICFGEN Specified

- This implies Step 1 of a two-step FBA process.

Only ICFUSE Specified

- This implies Step 2 of a two-step FBA process. In this case, the FRF components specified by ICFUSE must have their ICFs computed in an earlier FBA job.

  It is important to note that, in order for the ICFs employed in Step 2 of a two-step FBA process to be meaningful, it is absolutely essential that the loading condition of Step 1 be duplicated in Step 2. In order to satisfy this requirement and avoid inadvertent user errors, it is highly recommended that, except for the database or OUTPUT2 file specification, Case Control output requests and the FRF Case Control command, the user *employ the same data setup in Step 2 as that used in Step 1*.

Both ICFGEN and ICFUSE Specified

- This implies a single step process. In this case, the FRF components specified by ICFUSE must be among those that are specified by ICFGEN.

Only ICFAUTO Specified

- This implies a single step process and is equivalent to having both ICFGEN and ICFUSE with the same specification as ICFAUTO. In this case, the FRF components implied by ICFUSE are obviously the same as those implied by ICFGEN.

New ICFDB Keyword

- This keyword indicates that the ICF information is to be saved or is resident on the database.

New ICFOP2 Keyword

- This keyword indicates that the ICF information is to be saved or is resident on an `OUTPUT2` file. It specifies the Fortran unit number for this file.

New ICF Case Control Output Request

A new Case Control output request called ICF has been introduced. This is similar to the existing OLOAD and SPCF and requests output of the ICFs in the ICF generation phase of the two-step or single step process.

The output generated by the ICF request is similar to the OLOAD and SPCF output and is available in both the `.f06` and standard `.pch` punch files.

Summary of Inter-Component Force (ICF) Processing Using Two Steps

Step 1:

Run an FBA job as follows:

- Specify user loads and specify the ICFGEN keyword in the FRF Case Control command to indicate the FRF components whose ICFs are to be computed
- Specify the appropriate medium (database or OUTPUT2 file for saving the ICF information via the ICFDB/ICFOP2 keyword in the FRF Case Control command (ICFDB is the default)
- Optionally, request output of ICFs via the use of the new ICF Case Control command

Step 2:

Run an FBA job as follows:

- Employ the *same loading condition as in Step 1* and specify the ICFUSE keyword in the FRF Case Control command to indicate a subset of the FRF components of Step 1 that are the only ones to be included in this FBA process. ICFs must have been computed for these FRF components in Step 1. Otherwise, the program will terminate with a fatal error.
- Specify the medium (database or OUTPUT2 file) on which the ICF information is resident via the ICFDB/ICFOP2 keyword in the FRF Case Control command (ICFDB is the default)
- The program will automatically select the required data from the saved ICF information in order to meet the requirements implied by the ICFUSE keyword. No user intervention is needed.
- The output from this step will permit TPA to be performed using these ICFs
- As indicated earlier, the results for the FRF components of this step must match their results from Step 1.
- Step 2 can be repeated with different ICFUSE specifications to perform TPA using ICFs for different assembly configurations.

Summary of Inter-Component Force (ICF) Processing Using Single Step

Run a single FBA job as follows:

- Specify user loads and specify the ICFGEN/ICFUSE keywords or the single ICFAUTO keyword in the FRF Case Control command to indicate the FRF components whose ICFs are to be generated and then used
- For the ICFGEN/ICFUSE case, the FRF components in the ICFUSE specification must be part of those in the ICFGEN specification. Otherwise, the program will terminate with a fatal error.
- Specify the appropriate medium (database or OUTPUT2 file for saving the ICF information via the ICFDB/ICFOP2 keyword in the FRF Case Control command (ICFDB is the default)
- Optionally, request output of ICFs via the use of the new ICF Case Control command
- The program will automatically execute two passes, with Pass 1 generating and saving the ICF information for the ICFGEN (or ICFAUTO) FRF components, followed by Pass 2 which uses the ICF data of Pass 1 to give the results for the ICFUSE (or ICFAUTO) FRF components.

- The results for the FRF components of Pass 2 will match their corresponding results from Pass 1.
- The output from Pass 2 allows for TPA to be performed for ICFs.
- The entire procedure is automatic and completely user friendly, with no intervention called for by the user.

Summary of the Enhancements

The enhancements described above greatly enhance the FRF/FBA/TPA capability in MSC Nastran 2013, making it an excellent tool for realistic simulations and NVH studies.

Job Setup Examples

Examples of job setups that illustrate the generation and usage of ICFs described earlier are given on the following pages. For the purpose of illustration, an airplane model (consisting of five FRF components) shown on the following pages has been selected.

The examples illustrate job setups for the following cases. In addition to the following examples, an additional single shot FRF job was run using the full airplane model with a view to checking the validity and correctness of the results of the FBA jobs of Examples 2 through 7.

X-Y plots given in Figures 1(a) through 2(c) show the comparison of the displacement results for the T3 component of grid points 55 and 133 from the FBA jobs and from the single shot FRF job.

X-Y plots given in Figures 3(a) through 3(d) show the ICFs for the T3 component of connecting grid points 55 and 133 computed from the FBA job of Example 2. Since each of these points is connected to just two FRF components, the ICF plots of Figures 3(a) and 3(c) for grid point 55 are mirror images as are the ICF plots of Figures 3(b) and 3(d) for grid point 133.

- Example 1 FRF Generation Jobs -- Generate FRFs for Components 1 through 5
- Example 2 FBA Job -- Generate ICFs on Database -- Step 1 of Two-Step Process
- Example 3 FBA Job -- Use ICFs of Example 2 -- Step 2 of Two-Step Process
- Example 4 FBA Job -- Generate ICFs on OUTPUT2 File -- Step 1 of Two-Step Process
- Example 5 FBA Job -- Use ICFs of Example 4 -- Step 2 of Two-Step Process
- Example 6 FBA Job -- Generate and Use ICFs on Database -- Single Step Process
- Example 7 FBA Job -- Generate and Use ICFs on OUTPUT2 File -- Single Step Process

FRF/FBA Example

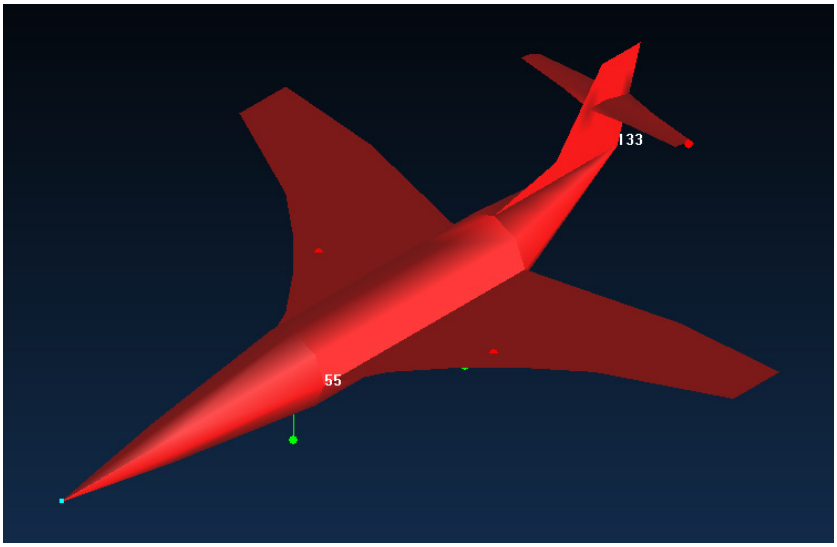


Figure 2-5 Airplane – Full Model

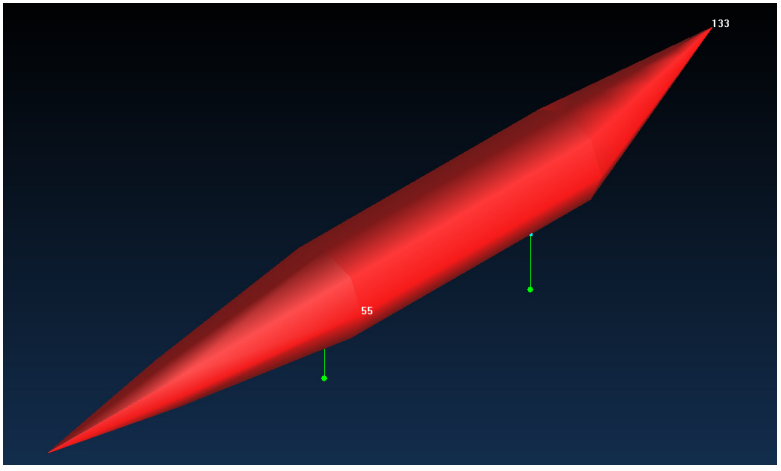


Figure 2-6 FRF Component 1 – Fuselage

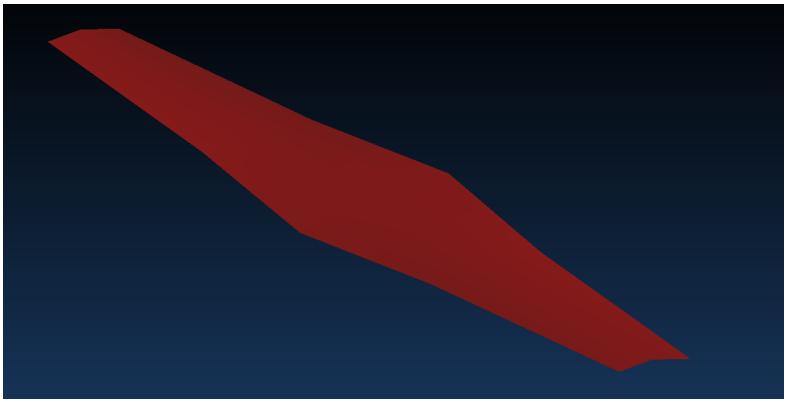


Figure 2-7 FRF Component 2 – Horizontal Tail

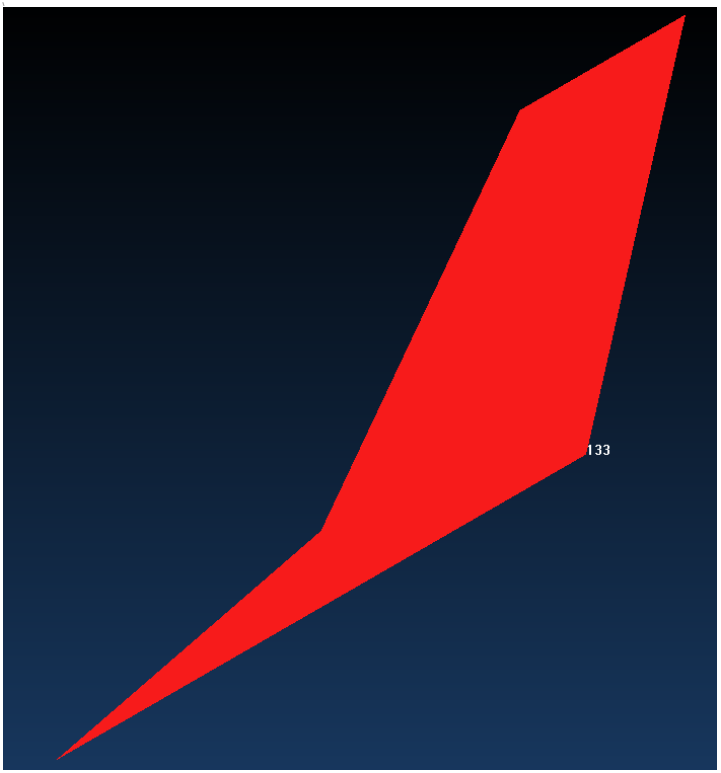


Figure 2-8 FRF Component 3 – Vertical Tail

Figure 2-9 FRF Component 4 – Inboard Wings

Figure 2-10 FRF Component 5 – Outboard Wings

Example 1

FRF Generation Jobs – Generate FRFs for Components 1 through 5

File Management Section (FMS)

FRF Component 1: ASSIGN OUTPUT2 = ‘fuselage_op2’ UNIT=25 DELETE
FRF Component 2: ASSIGN OUTPUT2 = ‘hor_tail_op2’ UNIT=26 DELETE
FRF Component 3: ASSIGN OUTPUT2 = ‘ver_tail_op2’ UNIT=27 DELETE
FRF Component 4: ASSIGN OUTPUT2 = ‘ib_wings_op2’ UNIT=28 DELETE
FRF Component 5: ASSIGN OUTPUT2 = ‘ob_wings_op2’ UNIT=29 DELETE

Case Control

FRF Component 1: FRF (COMPID = 1 COMPNAME = FUSELAGE
CONNPTS = 1000 OP2=25)
FRF Component 2: FRF (COMPID = 2 COMPNAME = HOR_TAIL
CONNPTS = 1000 OP2=26)
FRF Component 3: FRF (COMPID = 3 COMPNAME = VER_TAIL
CONNPTS = 1000 OP2=27)
FRF Component 4: FRF (COMPID = 4 COMPNAME = IB_WINGS
CONNPTS = 1000 OP2=28)
FRF Component 5: FRF (COMPID = 5 COMPNAME = OB_WINGS
CONNPTS = 1000 OP2=29)

These jobs automatically generate `.asm` files for subsequent use by the FBA process.

For these jobs, `scr = yes` may be specified on the MSC Nastran job command lines since there is no need for the databases to be saved at the end of the jobs.

Example 2

FBA Job – Generate ICFs on the Database for All Five FRF

Components – Step 1 of Two-Step Process

File Management Section (FMS)

```
ASSIGN INPUTT2 =    'fuselage_op2'  UNIT=25
ASSIGN INPUTT2 =    'hor_tail_op2'  UNIT=26
ASSIGN INPUTT2 =    'ver_tail_op2'  UNIT=27
ASSIGN INPUTT2 =    'ib_wings_op2'  UNIT=28
ASSIGN INPUTT2 =    'ob_wings_op2'  UNIT=29
```

Case Control

```
FRF (ASM  ICFGEN = ALL)
DLOAD = 1000
```

Bulk Data

```
RLOAD1,1000,2000,,,3000
FBALOAD,2000,…
FBALOAD,2000,…
TABLED1,3000,…
INCLUDE 'fuselage.asm'
INCLUDE 'hor_tail.asm'
INCLUDE 'ver_tail.asm'
INCLUDE 'ib_wings.asm'
INCLUDE 'ob_wings.asm'
```

For this job, `scr = no` should be specified on the MSC Nastran job command line since the database containing the ICF information needs to be saved for use in a subsequent FBA job.

Example 3

FBA Job – Use ICFs of Example 2 for an Assembly

Configuration Consisting of FRF Components 1, 4 and 5 -- Step 2 of Two-Step Process

File Management Section (FMS)

```
ASSIGN INPUTT2 = ‘fuselage_op2’  UNIT=25
ASSIGN INPUTT2 = ‘ib_wings_op2’  UNIT=28
ASSIGN INPUTT2 = ‘ob_wings_op2’  UNIT=29
ASSIGN ICFDATA = ‘example2.MASTER’
DBLOCATE DATABLK = (ICFDB)  LOGICAL = ICFDATA
```

Case Control

```
SET 100 = 1,4,5
FRF (ASM  ICFUSE = 100)
DLOAD = 1000
```

Bulk Data

```
RLOAD1,1000,2000,,,3000
FBALOAD,2000,…
FBALOAD,2000,…
TABLED1,3000,…

INCLUDE ‘fuselage.asm’
INCLUDE ‘ib_wings.asm’
INCLUDE ‘ob_wings.asm’
```

For this job, scr = yes may be specified on the MSC Nastran job command line since there is no need for the database to be saved at the end of the job.

Example 4

FBA Job – Generate ICFs on an OUTPUT2 File for FRF

Components 1, 2 and 3 – Step 1 of Two-Step Process

File Management Section (FMS)

```
ASSIGN INPUTT2 = ‘fuselage_op2’  UNIT=25
ASSIGN INPUTT2 = ‘hor_tail_op2’  UNIT=26
ASSIGN INPUTT2 = ‘ver_tail_op2’  UNIT=27
ASSIGN INPUTT2 = ‘ib_wings_op2’  UNIT=28
ASSIGN INPUTT2 = ‘ob_wings_op2’  UNIT=29
ASSIGN OUTPUT2 = ‘icf123_op2’  UNIT=33  DELETE
```

Case Control

```
SET 100 = 1,2,3
FRF (ASM  ICFGEN = 100  ICFOP2 = 33)
DLOAD = 1000
```

Bulk Data

```
RLOAD1,1000,2000,,,3000
FBALOAD,2000,…
FBALOAD,2000,…
TABLED1,3000,…

INCLUDE ‘fuselage.asm’
INCLUDE ‘hor_tail.asm’
INCLUDE ‘ver_tail.asm’
INCLUDE ‘ib_wings.asm’
INCLUDE ‘ob_wings.asm’
```

For this job, `scr = yes` may be specified on the MSC Nastran job command line since there is no need for the database to be saved at the end of the job.

Example 5

FBA Job – Use ICFs of Example 4 for a Configuration

Consisting of the Single FRF Component 1 -- Step 2 of Two-Step Process

File Management Section (FMS)

ASSIGN INPUTT2 = ‘fuselage_op2’ UNIT=25

ASSIGN INPUTT2 = ‘icf123_op2’ UNIT=33

Case Control

FRF (ASM ICFUSE = -1 ICFOP2 = 33)

DLOAD = 1000

Bulk Data

RLOAD1,1000,2000,,,3000

FBALOAD,2000,…

FBALOAD,2000,…

TABLED1,3000,…

INCLUDE ‘fuselage.asm’

For this job, `scr = yes` may be specified on the MSC Nastran job command line since there is no need for the database to be saved at the end of the job.

Example 6

FBA Job – Generate and Use ICFs for FRF Components 2 and 3 Using Database – Single Step Process

File Management Section (FMS)

ASSIGN INPUTT2 = ‘fuselage_op2’ UNIT=25

ASSIGN INPUTT2 = ‘hor_tail_op2’ UNIT=26

ASSIGN INPUTT2 = ‘ver_tail_op2’ UNIT=27

ASSIGN INPUTT2 = ‘ib_wings_op2’ UNIT=28

ASSIGN INPUTT2 = ‘ob_wings_op2’ UNIT=29

Case Control

```
SET 100 = 2,3
FRF (ASM  ICFAUTO = 100)
DLOAD = 1000
```

Bulk Data

```
RLOAD1,1000,2000,,,3000
FBALOAD,2000,…
FBALOAD,2000,…
TABLED1,3000,…

INCLUDE ‘fuselage.asm’
INCLUDE ‘hor_tail.asm’
INCLUDE ‘ver_tail.asm’
INCLUDE ‘ib_wings.asm’
INCLUDE ‘ob_wings.asm’
```

For this job, `scr = no` should be specified on the MSC Nastran job command line since the ICF information saved on the database in the first pass of this FBA job is needed for use in the second pass.

Example 7

FBA Job – Generate and Use ICFs for FRF Component 5 Using OUTPUT2 File – Single Step Process

File Management Section (FMS)

```
ASSIGN INPUTT2 = ‘fuselage_op2’  UNIT=25
ASSIGN INPUTT2 = ‘hor_tail_op2’  UNIT=26
ASSIGN INPUTT2 = ‘ver_tail_op2’  UNIT=27
ASSIGN INPUTT2 = ‘ib_wings_op2’  UNIT=28
ASSIGN INPUTT2 = ‘ob_wings_op2’  UNIT=29
ASSIGN OUTPUT2 = ‘icf5_op2’  UNIT=33  DELETE
```

Case Control

```
FRF (ASM  ICFAUTO = -5  ICFOP2 = 33)
DLOAD = 1000
```

Bulk Data

```
RLOAD1,1000,2000,,,3000
FBALOAD,2000,…
FBALOAD,2000,…
TABLED1,3000,…

INCLUDE ‘fuselage.asm’
INCLUDE ‘hor_tail.asm’
INCLUDE ‘ver_tail.asm’
INCLUDE ‘ib_wings.asm’
INCLUDE ‘ob_wings.asm’
```

For this job, `scr = yes` may be specified on the MSC Nastran job command line since there is no need for the database to be saved at the end of the job.

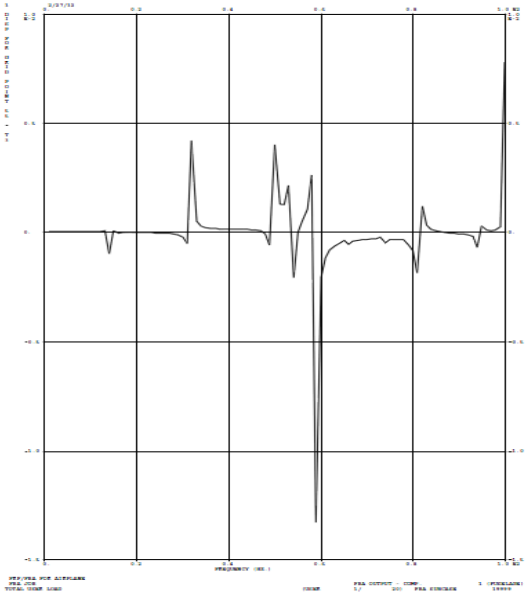


Figure 2-11 Displacement for Grid Point 55 – T3 Component (A point connecting the fuselage and an inboard wing)
FBA Job of Example 4 (ICFGEN Phase)

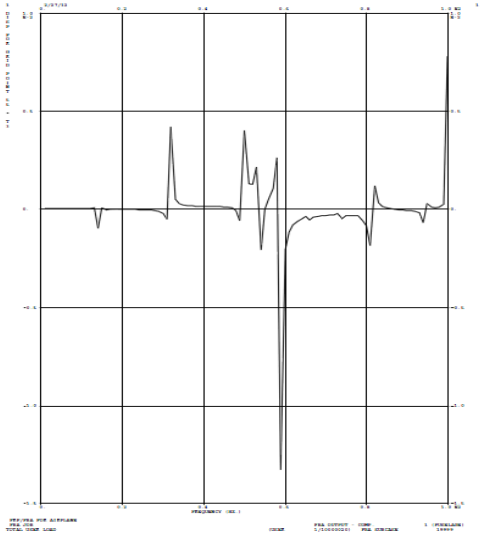


Figure 2-12 Displacement for Grid Point 55 – T3 Component (A point connecting the fuselage and an inboard wing)
FBA Job of Example 5 (ICFUSE Phase)

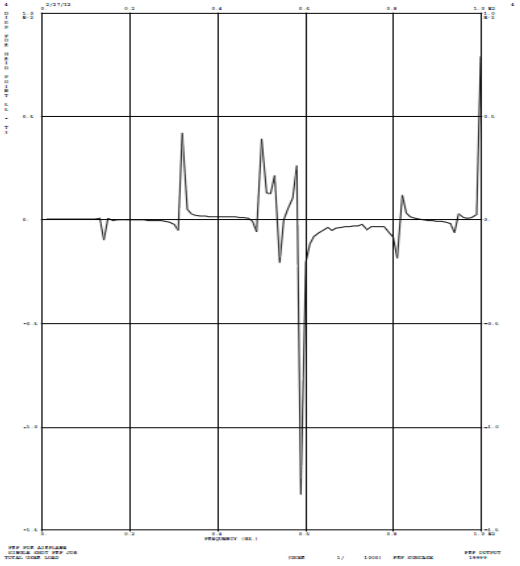


Figure 2-13 Displacement for Grid Point 55 – T3 Component (A point connecting the fuselage and an inboard wing)
Single Shot FRF Job

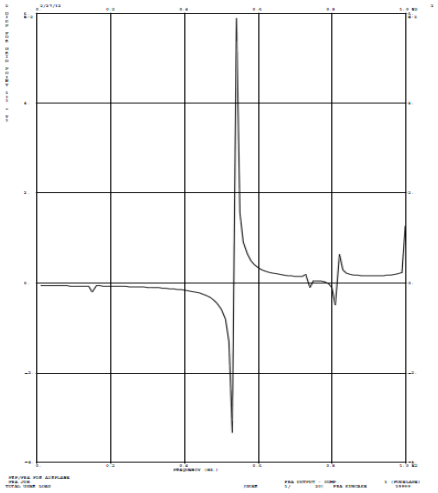


Figure 2-14 Displacement for Grid Point 133 – T3 Component (A point connecting the fuselage and the vertical tail)
FBA Job of Example 4 (ICFGEN Phase)

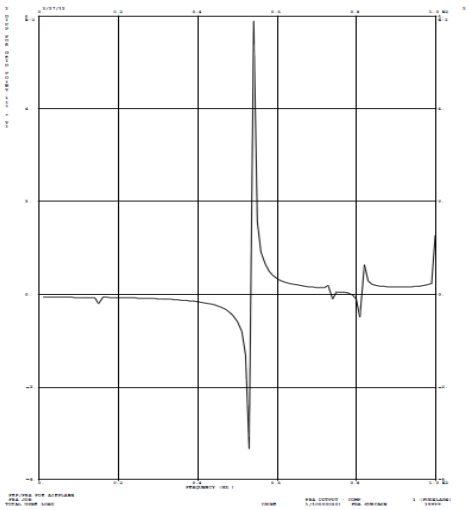


Figure 2-15 Displacement for Grid Point 133 – T3 Component (A point connecting the fuselage and the vertical tail)
FBA Job of Example 5 (ICFUSE Phase)

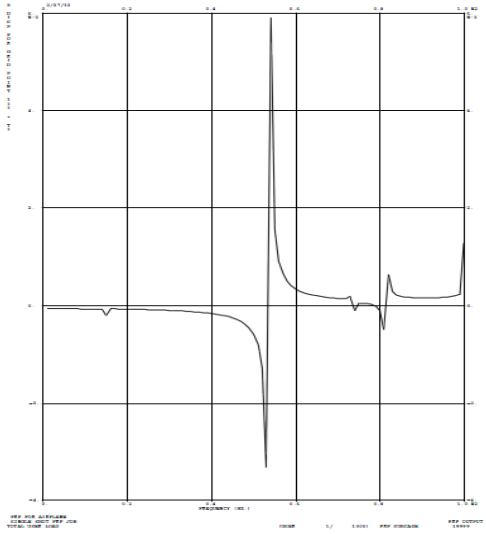


Figure 2-16 Displacement for Grid Point 133 – T3 Component (A point connecting the fuselage and the vertical tail)
Single Shot FRF Job

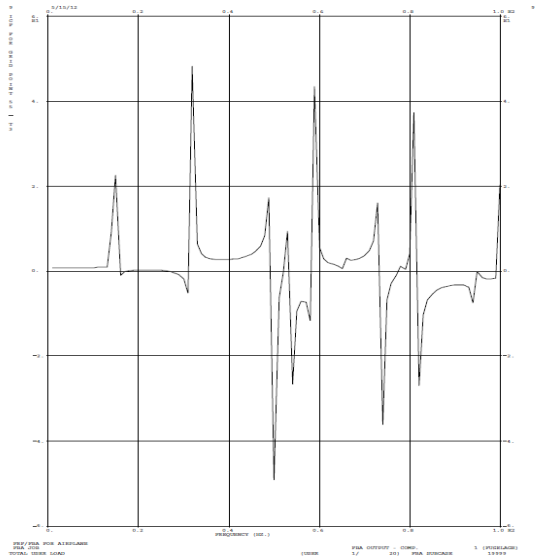


Figure 2-17 ICF for Grid Point 55 – T3 Component (As seen from the fuselage) (This is a point connecting the fuselage and an inboard wing)
FBA Job of Example 2 (ICFGEN Phase)
[Note: The ICF plots of Figures 3(a) and 3(c) are mirror images]

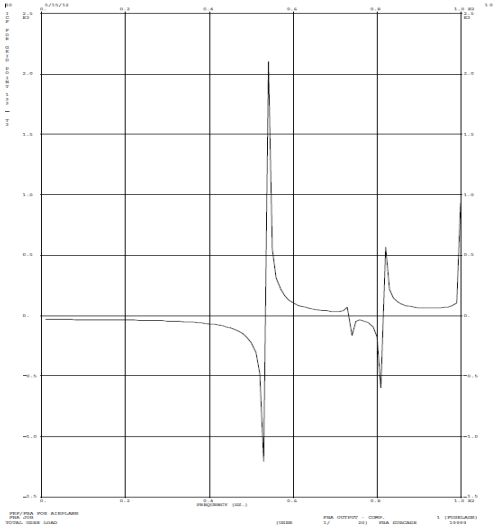


Figure 2-18 ICF for Grid Point 133 – T3 Component (As seen from the fuselage) (This is a point connecting the fuselage and the vertical tail)
FBA Job of Example 2 (ICFGEN Phase)
[Note: The ICF plots of Figures 3(b) and 3(d) are mirror images]

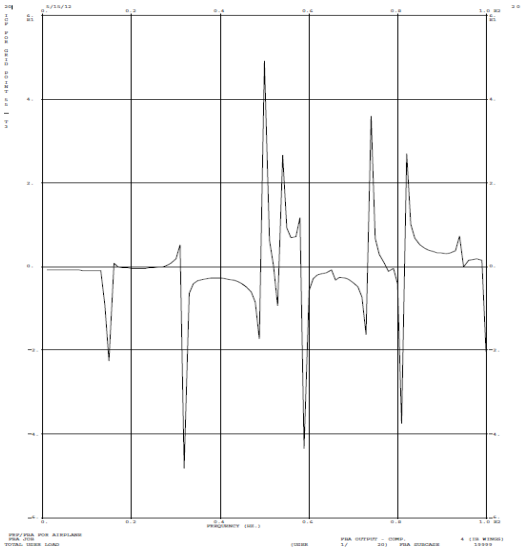


Figure 2-19 ICF for Grid Point 55 – T3 Component (As seen from the inboard wing) (This is a point connecting the fuselage and an inboard wing)
FBA Job of Example 2 (ICFGEN Phase)
[Note: The ICF plots of Figures 3(a) and 3(c) are mirror images]

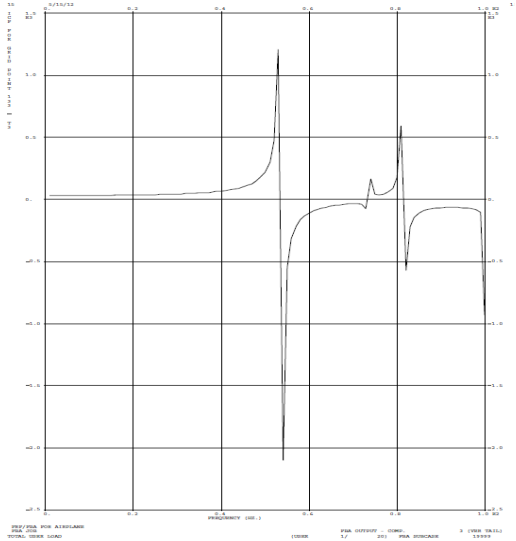


Figure 2-20 Figure 3(d). ICF for Grid Point 133 – T3 Component (As seen from the vertical tail) (This is a point connecting the fuselage and the vertical tail)
FBA Job of Example 2 (ICFGEN Phase)
[Note: The ICF plots of Figures 3(b) and 3(d) are mirror images]

Addition of New FBATOLR User Parameter for Use in the FBA Process

A new user parameter called FBATOLR has been added for use in the FBA process. This parameter is applied to grid point coordinates in order to determine connections between potential connection points of various FRF components in the FBA process.

The default value for this parameter is set 1.0E-05. This should be satisfactory for most situations. A looser tolerance may be needed in certain situations. An example is the case where the potential connection points of an FRF component are associated with the shell elements of RSSCON solid-to-shell element connectors. In this case, a looser tolerance may need to be specified in order to achieve proper connections between FRF components in the FBA process.

Fatigue Analysis/Output Request

Calculate fatigue damage and fatigue life directly within linear statics SOL 101, modal analysis SOL 103, or modal transient SOL 112 runs for materials that can be defined with MAT1 bulk data (metal fatigue analysis). This capability of embedded fatigue analysis within MSC Nastran is referred to as MSC Nastran Embedded Fatigue, or NEF for short.

Benefits

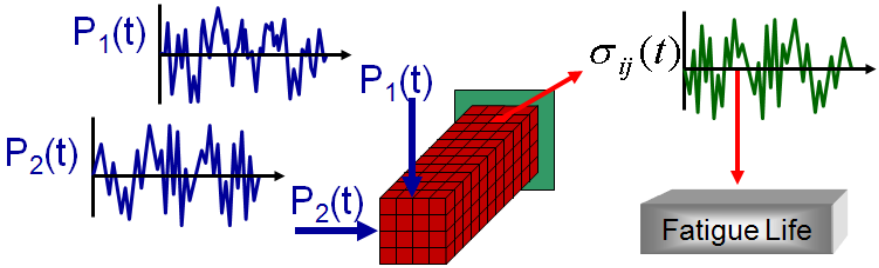
It is not about the stress! The real question is how long will it last? Fatigue life calculations used to be a tedious post-processing activity that would take place after the stresses and strains are determined, and done externally to MSC Nastran. Now users can request fatigue life and damage as an output request similar to requesting displacements, stresses, strains, and forces. An additional benefit is that the users can now run optimization in conjunction with fatigue analysis, Please see Fatigue Life Design Responses.

Feature Description

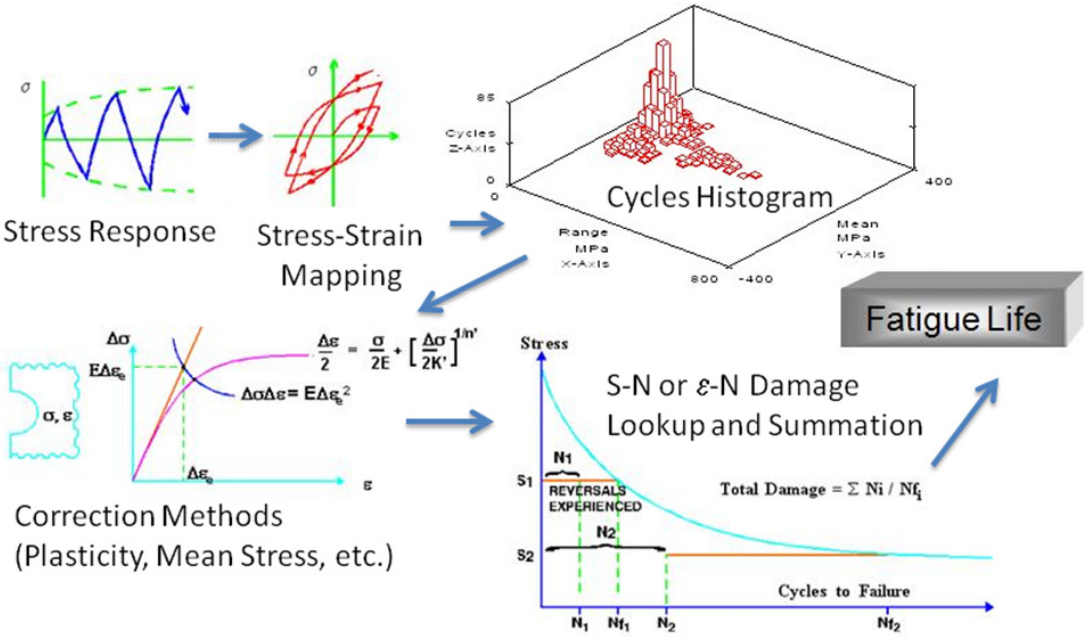
Fatigue can be described as structural failure under repeated or otherwise varying load, which never reaches a level sufficient to cause failure in a single application. Typical stress analysis is generally representative of a single application of a loading environment. Multiple or cyclic applications of the same loading environment over time can now be simulated to predict fatigue life and damage directly in MSC Nastran.

Fatigue analysis requires three main inputs: geometry, materials, and the cyclic load variations. The geometry comes from the SOL 101, 103, or 112 runs in the form of stress distributions over the entire model. Special material properties are used in the form of stress-life (S-N) or strain-life (ε -N) curves. The plasticity that occurs due to the cyclic loading is built into these curves and methods used as look up tables and corrections to equate linear stress or strain levels (range and mean) to life. The cyclic variations of the loading are defined in typical table format and are used to scale the stress distribution. Multiple, simultaneously applied loads are combined using the principle of linear superposition to produce the stress or strain time variations. These time histories are then processed through a “rainflow” cycle count algorithm to determine the range and mean of each stress/strain cycle. Damage is determined using the tried and true methods of the total life (S-N) or crack initiation (ε -N) to determine fatigue life. Damage from all cycles is summed and reported as life values. Multiple loading events can be strung together to form a sequence of events, commonly known as a duty cycle. Damage from each event is summed to give life due to the entire duty cycle.

The following illustrations give the overall fatigue calculation process. The first illustrates the process of taking the FE loads and their time variations to combine them into stress output responses time histories at various locations of the model. Ultimately this stress variation is turned into a fatigue life prediction.



The process of converting the time varying stress responses into fatigue life predictions is a two or three step process depending on the method used. Both the stress-life (S-N) and strain-life (ε-N) methods employ a well known algorithm to extract cycles of stress/strain, called rainflow cycle counting. An easy way to conceptualize this is by mapping the stress time history to the stress-strain space where each hysteresis loop represents a stress-strain cycle. Each cycle has its specific stress range and mean. Sometimes this is illustrated in the form of a histogram showing specific discrete *bins* of stress range vs. mean. In the case of the strain-life (ε-N) method, the plasticity correction is then made using techniques such as Neuber's plasticity correction method, before looking up the damage on an S-N or an ε-N curve. Both methods may employ mean stress corrections also. Damage from all such *bins* is then summed using Palgren-Miner damage summation rule and fatigue life presented as the reciprocal of damage.



Overview of Case Control and Bulk Data

One or more fatigue analyses can be called out using the new FATIGUE case control. A SET case control is used if more than one analysis is to be requested and is then referenced by the FATIGUE case control. The FATIGUE case control must appear above all subcases. This examples indicates that three separate fatigue analyses are to be performed.

Case Control

```
SET 99 = 11, 12, 13
FATIGUE(SET) = 99
```

Each ID called out by a FATIGUE case control references a set of bulk data that describe the inputs necessary for a fatigue analysis. For each fatigue analysis, a set of FTGDEF, FTGPARM, and FTGSEQ bulk data of the same ID is defined.

Fatigue Element Definitions (FTGDEF)

The FTGDEF (FaTiGue element DEFinitions) supplies the analysis with the desired locations on the model where fatigue damage is to be calculated. If no FTGDEF bulk data exists for the fatigue analysis, all the elements (solid and shells) are assumed part of the analysis, as long as there are Fatigue material properties defined. Individual elements or entire property sets of elements can be specified. Individual elements can also be excluded from the analysis. The example below simply shows that a fatigue analysis is to be performed on element 1 only and that additional properties are defined using PFTG bulk data of ID 18.

Bulk Data

```
FTGDEF, 11, , 18
       , ELSET, 1
```

Fatigue Parameters (FTGPARM)

The FTGPARM (FaTiGue PARaMeters) defines fatigue parameters. If no FTGPARM bulk data exists for the fatigue analysis, defaults are assumed. The FTGPARM specifically calls out which type of fatigue analysis is to be performed such as an S-N (total life or stress-life) or ε-N (crack initiation or strain life) analysis. S-N analysis is the default. Other parameters may also be specified on the FTGPARM to enhance the analysis, speed it up, request different correction methods, and obtain additional output. This example simply shows that a crack initiation analysis is being requested:

Bulk Data

```
FTGPARM, 11, EN, 1.0
```

Fatigue Load Sequence (FTGSEQ)

The FTGSEQ (FaTiGue SEQuence) defines the cyclic load variation. This entry is required and if not present, a fatal error is issued. The FTGSEQ is simple, yet very powerful. It can be used to define a simple oscillating time variation of -1 to +1 scaling of the load, to very complicated sequences of the loading called a duty cycle. In order to do this, additional bulk data are necessary: FTGEVNT (FaTiGue EVeNT) and FTGLOAD (FaTiGue LOADing). FTGENVT is used to define the events of the load sequence and FTGLOAD is used to define the actual time variations and associate them with a stress distribution from the analysis. This example shows the simplest request where there is only one load variation with time, thus only one event. The FTGSEQ calls out FTGEVNT 21, which in turn calls out FTGLOAD 101. FTGLOAD 101 references a TABLFTG 201 that describes the time variation and associates it to the static stresses of SUBCASE 8 (in the case of SOL 101):

Bulk Data

```
FTGSEQ, 11
      , 21
FTGEVNT, 21, 101
FTGLOAD, 101, 201, 8
TABLFTG, 201
       , 0.0, 1.0, -1.0, 0.0, ENDT
```

Fatigue Materials (MATFTG)

The MATFTG (MATerial FaTiGue) defines the fatigue material properties. It must be associated to an existing MAT1 entry with the same ID. In the example that has been used thus far, a crack initiation analysis has been called out by

the FTGPARM entry. This means that ε-N material data is necessary. This data can be derived by simply supplying the ultimate tensile strength and a material code defining the type of metal as shown or actual ε-N parameters can be entered as described in the user documentation.

Bulk Data

```
MAT1,  1, 203403.0, 78231.7, 0.3, 1.0
MATFTG,1
       ,STATIC,   , 480.0,   99
```

Fatigue Properties (PFTG)

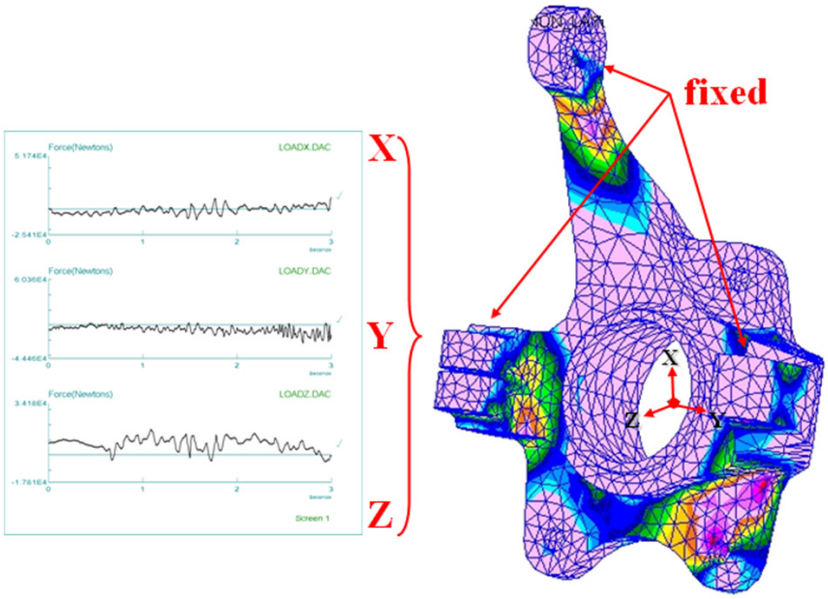
The PFTG (Property FaTiGue) defines other possible fatigue properties such as surface finish and other factors that can be applied to the requested entities from the FTGDEF entry. If this entry is not present and none is called out by FTGDEF, then default values are used. This shows PFTG of ID 18 called out from the previous example of the FTGDEF entry where a polished surface finish is specified.

Bulk Data

```
PFTG,  18, , POLISH
```

Example 1

This example shows a SOL 101 run where a single load event is defined consisting of multiple, simultaneously applied loads. The loads are combined to produce the overall stress time histories at each location of interest using the principle of linear superposition. Thus, the name pseudo-static is used. The FTGSEQ references a single FTGEVNT entry and that FTGEVNT references multiple FTGLOAD entries. The FTGLOAD entries define the time variation of each applied unit load and associates them to their subcases. No FTGDEF or FTGPARM entries are shown, thus defaults are assumed, meaning an S-N analysis will occur for every node of each element of the model for elements referencing MAT1of ID 1.



Case Control

```
SOL 101
FATIGUE = 44
SUBCASE 1
...
SUBCASE 2
...
SUBCASE 3
...
```

Bulk Data

```
MAT1    ,1, 203403.0, 78231.7, 0.3, 1.0
MATFTG ,1
               ,STATIC,        ,480.0, 99
$
FTGSEQ ,  44
       ,  21
FTGEVNT,  21, 301, 302, 303
FTGLOAD, 301, 311,   1
```

```
FTGLOAD, 302, 312,    2
FTGLOAD, 303, 313,    3
TABLFTG, 311 ...
TABLFTG, 312 ...
TABLFTG, 313 ...
```

Example 2

This example shows a SOL 103 run where the modal stress recovery method is used as opposed to the SOL 101 pseudo-static stress recovery method. Both methods are treated identically within the fatigue solver itself. This is sometimes referred to as modal superposition. The illustration below shows the differences and similarities between the two methods.

Quasi-Static Stress Recovery Method

Modal Stress Recover Method

The difference between the two methods is that the unit static loads are replaced by the mode shapes and the load time histories are replaced with the modal responses for each mode shape, sometimes know as modal participation factors or vectors or loads. Just as the load time histories must be defined by TABLFTG bulk data or external file definitions, so must the modal responses. Typically, these can come directly from an ADAMS analysis or a previously run SOL 112 analysis where `SDISPLACEMENTS(PUNCH)=ALL` has been requested to output these modal responses.

Fatigue analysis using SOL 103 only allows for a single subcase as the identifier on the FTGLOAD entries now refer to modes as opposed to static subcases. Note in this example, that 10 modes are requested via the METHOD/EIGRL entries, yet it is possible to only specify certain modes to be included in the modal superposition. Here only modes 1, 2, and 4 are used as called out by FTGLOAD entries.

Case Control

```
SOL 103
FATIGUE = 44
SUBCASE 1
METHOD = 1
...
```

Bulk Data

```
EIGRL ,1, , , 10
MAT1  ,1, 203403.0, 78231.7, 0.3, 1.0
MATFTG,1
      ,STATIC, , 480.0, 99
$
FTGSEQ, 44
      , 21
```

```
FTGEVNT, 21, 301, 302, 304
FTGLOAD, 301, 311, 1
FTGLOAD, 302, 312, 2
FTGLOAD, 304, 314, 4
TABLFTG, 311 ...
TABLFTG, 312 ...
TABLFTG, 314 ...
```

Example 3

This example shows a SOL 112 run where the modal stress recovery method is used. The difference between this example and the previous one using SOL 103 is that, since SOL 112 is used, the modal transient analysis directly provides the mode shapes and the modal responses for each subcase. Thus each subcase represents an entire load event. Because of this, no FTGEVNT or FTGLOAD entries are necessary. Multiple subcases referenced by the FTGSEQ entry can be made, the defining a duty cycle of sequential events. This example shows only a single event. Instead of referencing a FTGEVNT entry, the FTGSEQ now references SUBCASE ID. The input is much simpler for SOL 112. Internally, the fatigue solver uses the same method as both SOL 101 and 103 to determine the combined (modal superposition) time history from which fatigue live is determined.

Case Control

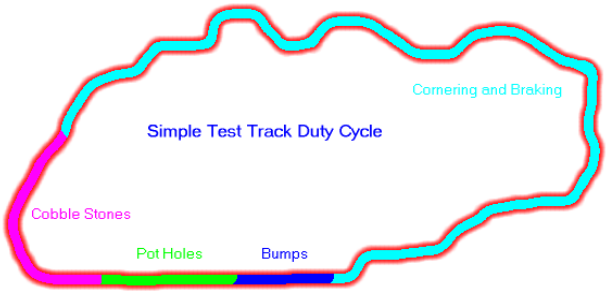
```
SOL 112
FATIGUE = 44
SUBCASE 1
METHOD = 1
...
```

Bulk Data

```
EIGRL ,1, , , 10
MAT1  ,1, 203403.0, 78231.7, 0.3, 1.0
MATFTG,1
      ,STATIC, , 480.0, 99
$
FTGSEQ, 44
      ,  1
```

Example 4

This example shows a SOL 101 run where a duty cycle has been defined. Imagine a new car being tested on the proving grounds and as it drives around the test track it is subject to various events. These events consist of a cobble stone surface, pot holes, bumps, cornering, and braking. Ten seconds of measured loading for each of these types of events has been obtained and available for the fatigue analysis. However the entire test track itself is made up of 30 seconds of cobble stones, 20 seconds of pot holes, 10 seconds of bumps, and one minute of cornering and braking. The actual loading is transferred into the car body through the four wheels making contact with the ground. Thus each event has four simultaneously acting loads. The fatigue life reported back is expressed in *Laps* around the test track.



To do this the FTGSEQ references four FTGEVNT entries and each FTGEVNT references four FTGLOAD entries. The FTGSEQ entry defines the sequence and number of repetitions of each event and the FTGEVNT entries define the simultaneously acting loads for each event. The FTGLOAD entries define the time variation of each load and associate them to the SOL 101 subcases (stress distributions due to unit loads). The FTGDEF and FTGPARM entries are shown where an S-N analysis is defined on specific elements of the model referencing MAT1 of ID 1.

Case Control

```
SOL 101
TITLE Simple Test Track Duty Cycle
$
FATIGUE = 44
$
SUBCASE 1
   SUBTITLE Unit load on front right
...
SUBCASE 2
   SUBTITLE Unit load on front left
...
SUBCASE 3
   SUBTITLE Unit load on rear right
...
SUBCASE 4
   SUBTITLE Unit load on rear left
...
```

Bulk Data

```
PSHELL, 66, 1, ...
MAT1,  1, 203403.0, 78231.7, 0.3, 1.0
$
$SN curve specifically defined
MATFTG,1
      ,STATIC, , 600.0
       61
$
$Select elements of property 66 only with polished surface finish
SET4   , 1, PROP, PSHELL, 66
FTGDEF, 44
      , ELSET, 1, 35
PFTG   , 35, 0, POLISH
```

```
$
$Specify an S-N analysis with Goodman mean stress correction
FTGPARM, 44, SN
       ,STRESS, ,GOODMAN
$
$ Duty Cycle - 3 cobble stones
$              2 pot holes
$              1 bumps
$              6 cornering and braking
FTGSEQ, 44
      , 21, 3.0, 22, 2.0, 23, 1.0, 24, 6.0
      ,UNITS, 1.0, Laps
$
$Cobble Stone event
FTGEVNT, 21, 101, 102, 103, 104
$
$Pot Hole event
FTGEVNT, 22, 201, 202, 203, 204
$
$Bumps event
FTGEVNT, 23, 301, 302, 303, 304
$
$Cornering and Braking event
FTGEVNT, 24, 401, 402, 403, 404
$
$Load association for Cobble Stone event
FTGLOAD, 101, 111, 1
FTGLOAD, 102, 112, 2
FTGLOAD, 103, 113, 3
FTGLOAD, 104, 114, 4
$
$Load association for Pot Hole event
FTGLOAD, 201, 211, 1
FTGLOAD, 202, 212, 2
FTGLOAD, 203, 213, 3
FTGLOAD, 204, 214, 4
$
$Load association for Bumps event
FTGLOAD, 301, 311, 1
FTGLOAD, 302, 312, 2
FTGLOAD, 303, 313, 3
FTGLOAD, 304, 314, 4
$
$Load association for Cornering and Braking event
FTGLOAD, 401, 411, 1
FTGLOAD, 402, 412, 2
FTGLOAD, 403, 413, 3
FTGLOAD, 404, 414, 4
$
$Tables defining load variations for each load of each event
TABLFTG, 111 ...
TABLFTG, 112 ...
TABLFTG, 113 ...
TABLFTG, 114 ...
```

```
TABLFTG, 211 ...
TABLFTG, 212 ...
TABLFTG, 213 ...
TABLFTG, 214 ...
TABLFTG, 311 ...
TABLFTG, 312 ...
TABLFTG, 313 ...
TABLFTG, 314 ...
TABLFTG, 411 ...
TABLFTG, 412 ...
TABLFTG, 413 ...
TABLFTG, 414 ...
```

Limitations

- Shell elements give incorrect life results if LAYER field on PFTG entry is top or bottom layer in the Op2 or Master/DBALL; the results are correct in the F06 output file.
- LOC=NODE on FTGPARM options shows high critical angle values from Op2 or Master/DBALL; the results are correct in the F06 output file.
- The creation of stress/strain time history and/or rainflow cycle output using the HISTOGRAM case control is not reliable.
- FATIGUE case control option SORT1/SORT2 is not supported.
- The definition of load event names using FTGEVNT may result in a fatal error.
- The flag on of the most damaged entity is not correct.
- Very large log files may be created if the maximum stress is greater than the ultimate stress.
- If the automatic S-N or e-N curve is created based upon the UTS stress on the MATFTG option, only CODE not equal to 0 is working correctly.

Other Features

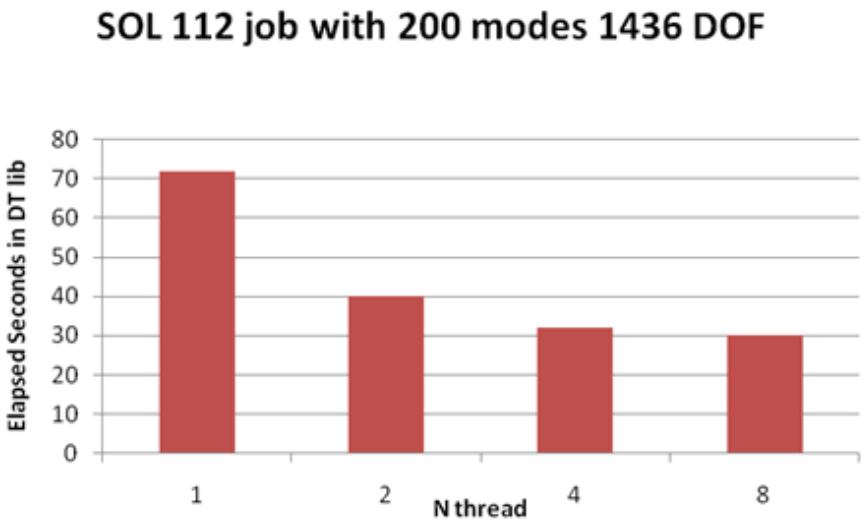
- Superelements - fatigue output available for models using both part and list superelements
- Out-of-core processing - all calculations are done in-core for performance reasons; however, if necessary out-of-core processing is automatically switched on in the case of limited memory
- Static failure flag - if static failure occurs, option may be set to stop or continue the analysis
- Restarts - quick, simple restarts are possible if only fatigue parameters, materials, and loading are changed, which do not require the recalculation of stresses

Documentation Dependencies

Please see the *MSC Nastran Fatigue Analysis User's Guide* for detailed examples of how to use these new features and the *MSC Nastran Quick Reference Guide* for details on each case control and bulk data entry to control fatigue analysis.

Multi-Threaded Fatigue Jobs

The number of threads can be specified on the FTGPARM entry for parallel processing. Below is a chart showing a SOL 112 model with 200 modes and the performance gains using multiple threads.

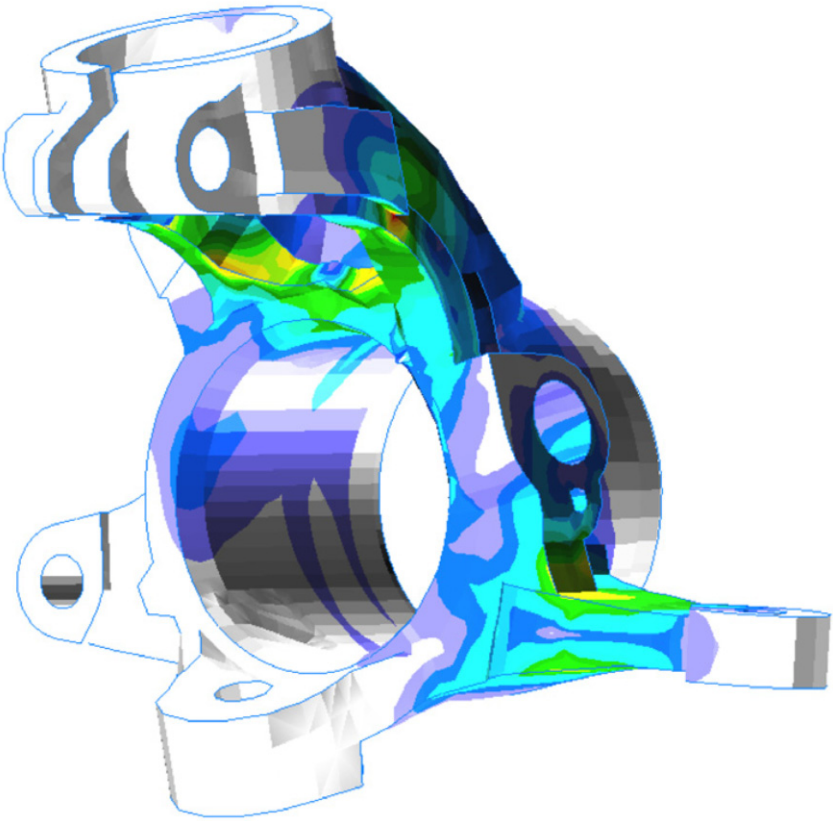


Documentation Dependencies

Please see the *MSC Nastran Fatigue Analysis User's Guide* for detailed examples of how to use these new features and the *MSC Nastran Quick Reference Guide* for details on each case control and bulk data entry to control fatigue analysis.

Patran Support for MSC Nastran Embedded Fatigue

Patran 2013 provides support for the new MSC Nastran Embedded Fatigue including materials, load and event manipulation and handling, and post-processing capabilities. It provides post support for MSC Nastran Embedded Fatigue using either the MASTER/DBALL, new OP2 file, FEF, or FER files.

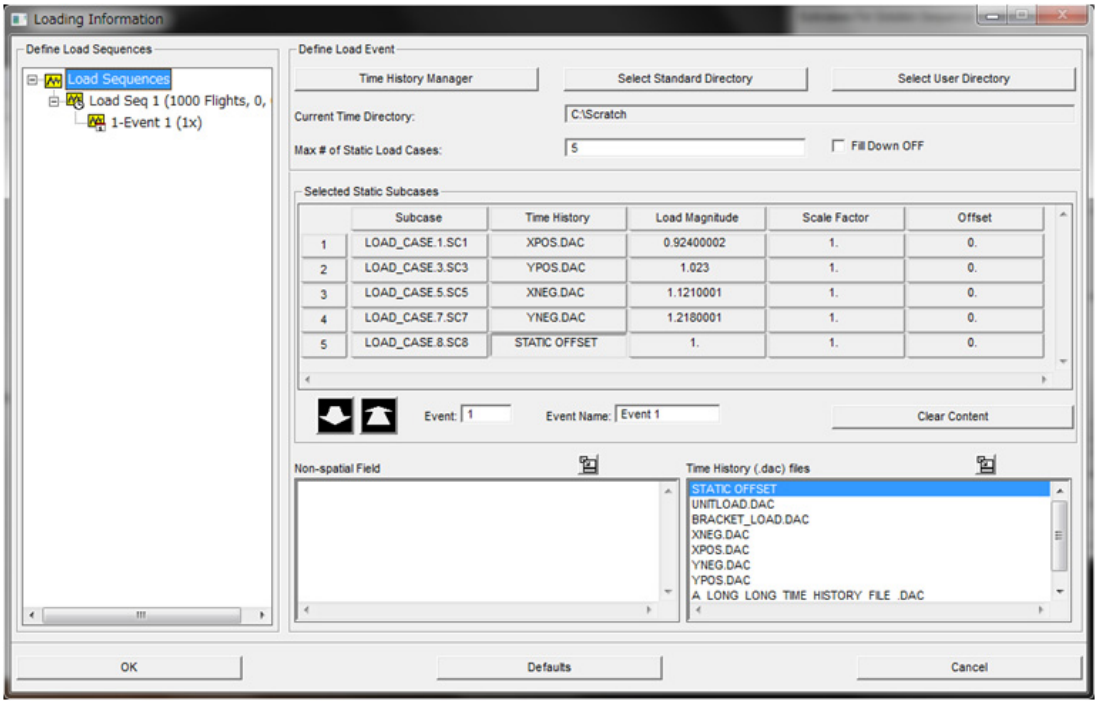


MSC Nastran Embedded Fatigue (NEF)

The fatigue calculation process is considered a post processing task and this has been an accepted convention, but now MSC Nastran Embedded Fatigue (NEF) breaks this convention by coupling the stress and fatigue calculation process into one simultaneous operation. This new immersed capability has wide ranging implications in relation to the way fatigue and reliability is handled within large mechanical engineering organizations. By combining the two separate processes into one simultaneous process the need for any kind of intermediate data is removed. Such intermediate files can sometimes be a limiting factor in the size of model that can be handled. With NEF there is no limit, theoretically, to the size of model that can be handled (practically this will be governed by normal MSC Nastran model size limitations).

Also, by embedding the process within MSC Nastran, this allows an analyst to include the materials and loading information with the model data in the MSC Nastran input file. This means that model portability becomes much easier. Another significant new capability will be created by enabling optimization procedures to be coupled with

fatigue as the constraint, via a SOL200 type analysis. And finally, because the fatigue process is far more transparent (within this solver embedded process) it will open up the opportunity for an analyst to request fatigue results output with every stress run.



Patran Support for MSC Nastran Embedded Fatigue:

- Provides Interface to New Capability
  - Material (S-N and E-N) Support
  - Defines Loading
  - Post Processing
- Provides Support in Familiar User Environment

3 Acoustics

Poroelastic Material (PEM)

Vibroacoustics with poroelastic trim components involves complex multi-physics in terms of the solid-fluid interaction at the microscopic level, as well as its unique applications in finite-element based analyses. Porous materials are widely used in automotive NVH applications for noise suppression. FFT, with its software product, Actran, has broad experiences on acoustics in general and poroelasto-vibroacoustics in particular. The integration of Actran's technologies expands the capability of Fluid-Structural Interaction analysis in MSC Nastran.

The goal of this project is to enable MSC Nastran to perform the modal frequency analysis of trimmed structure, such as a trimmed car body, for vibroacoustic simulation.

Benefits

The poroelastic materials for trimmed parts and components has the following benefits:

Poroelastic elements

In this project, poroelastic volume elements, such as CHEXA, CPENTA, and CTETRA, both in linear and quadratic orders, are created to represent the discretized poroelastic medium in a trim component.

A new user interface was created for the poroelastic elements. It includes a new MATPE1 Bulk Data entry for poroelastic materials. The existing PSOLID entry has been modified by adding a new value, `PORO`, in the field of `FCTN` to categorize the elements.

MATPE1 is a combination of, MAT1 (for the skeleton/solid-phase), MAT10 (for the fluid-phase) and additional measurable material parameters unique to the porous medium. There is also a maintenance consideration on the format. With this input format, it is easier for future expansion to include other material properties for both solid- and fluid-phases, such as MAT9.

Frequency-dependent material is also considered in this project. A new material entry, MATF1 has been created for this purpose.

Trim components

Physically, a trim component is an FE model of a sound package part. A trim component has its own characteristics, in terms of its constituents and data processing methods.

A trim component may have poroelastic elements. It could also have structural elements for modeling an elastic medium and/or fluid pressure elements for a fluid domain. It is required that the meshes at the internal interface between different domains within a trim component be congruent, while the meshes at the interface between the trim component and the residual structure/cavity are incongruent.

A new BEGIN BULK (Case) `TRMC=Trim-ID` has been created to model a trim component. The identification number of `TRMC` will serve as a qualifier for the data blocks and other computing purposes.

The reduced impedance matrix can either be projected onto the modal space or stay in the physical space. The final reduced impedance matrix has contributions from all elements, including structural and fluid elements.

Interface couplings

Two new Bulk Data entries in the *MSC Nastran Quick Reference Guide*, ACPEMCP and TRMCPL will be created for the trim component interface coupling and constraints. ACPEMCP is used to define interface coupling types and degrees of freedom, as well as constraints on a trim component. TRMCPL is used to set up search parameters for computing the interface matrices.

Solution workflow control

In automotive NVH applications, the FE model of sound package parts can be integrated into the FE model of both vehicle body and passenger compartment system. The reduced impedance matrices in physical coordinates of trim components can be reused in different design configurations of car body and/or passenger compartment as long as the interface with structural and cavity remains the same. The acoustic effects on the passengers may be investigated by the studies of various combinations of trim components.

A new Case Control command, TRIMGRP (Case), is used to define the trim components to be included for the solution.

Solution Sequences

The new trim component capability will be implemented in SOL 111, the modal frequency response. In addition, SOL 200 will perform design optimization if the design model is not associated with the trim components.

Data Recovery

Data recovery will be performed on the user-selected grid IDs located on the interface surface of trim components. The recovered data includes displacement, velocity and acceleration at the surface of trim components, which interfaces the structure and/or cavity.

Background Theory

The theories of Actran's technology have been documented in [1]. In this section, they are briefly reviewed for the integrity of this document.

Biot Theory (M. A. Biot, 1956)

The Biot theory of poroelasticity was developed by M. A. Biot [2, 3]. The governing dynamic equations of motion in the time domain are shown as follows.

$$\nabla \cdot \boldsymbol{\sigma}^s = \rho_{11}\ddot{\mathbf{u}}^s + \rho_{12}\ddot{\mathbf{u}}^f + b(\dot{\mathbf{u}}^s - \dot{\mathbf{u}}^f)$$

$$\nabla \cdot \boldsymbol{\sigma}^f = \rho_{12}\ddot{\mathbf{u}}^s + \rho_{22}\ddot{\mathbf{u}}^f - b(\dot{\mathbf{u}}^s - \dot{\mathbf{u}}^f)$$

Figure 3-1 Poroelastic Dynamic Equations of Motion in Time Domain

In Figure 3-1, ∇ is the gradient operator; σ the stress tensor and u the displacement vector; the superscripts, s and f, stand for the solid- and fluid-phase of a porous medium, respectively; both (i,j=1,2), and b are material constants.

Their counterparts in the frequency domain are

$$\nabla \cdot \boldsymbol{\sigma}^s = -\omega^2\left(\rho_{11}\mathbf{u}^s + \rho_{12}\mathbf{u}^f\right) + jb\omega(\mathbf{u}^s - \mathbf{u}^f)$$
$$\nabla \cdot \boldsymbol{\sigma}^f = -\omega^2\left(\rho_{12}\mathbf{u}^s + \rho_{22}\mathbf{u}^f\right) - jb\omega(\mathbf{u}^s - \mathbf{u}^f)$$

Figure 3-2 Poroelastic differential equations in frequency domain

where ω is the angular velocity and $j = \sqrt{-1}$

"U-p Formulation: Differential Equations

Since there exists a relation between the pressure of fluid-phase and the displacements, Atalla, et al [4, 5] developed a simpler form of differential equations in the frequency domain for vibro-acoustics with poroelastic materials.

$$\nabla \cdot \hat{\boldsymbol{\sigma}}^s(\mathbf{u}) + \widetilde{\rho}\omega^2\mathbf{u} + \widetilde{\gamma}\nabla p = \mathbf{0}$$
$$\Delta p + \frac{\widetilde{\rho}_{22}}{\widetilde{R}}\omega^2 p - \frac{\widetilde{\rho}_{22}}{\phi^2}\widetilde{\gamma}\nabla \cdot \mathbf{u} = 0$$

Figure 3-3 Poroelastic Differential Equations in U-p Formulation

In Equations 3, p is the pressure of fluid-phase; the Δ aplace operator; the porosity of porous material (Ω as in [1]); and the others are material related constants. The tilde symbol above a constant indicates that constant is complex and frequency-dependent. It must be pointed out that the superscript, *s*, is dropped in the equations whenever it does not cause any ambiguity. Detailed information on background theory is available in Ref. [1].

The stresses in the solid-phase are given in Figure 3-4, respectively.

$$\hat{\boldsymbol{\sigma}}^s(\mathbf{u}) = \left(K_b - \frac{2}{3}N\right)\nabla \cdot \mathbf{u}\mathbf{I} + 2N\boldsymbol{\varepsilon}^s$$

$$\boldsymbol{\sigma}^s(\mathbf{u}^s, \mathbf{u}^f) = \hat{\boldsymbol{\sigma}}^s - \phi\frac{\widetilde{Q}}{\widetilde{R}}p\mathbf{I}$$

Figure 3-4 Stresses of solid-phase

where I is the identity tensor; the strain tensor of solid-phase; and the rest of notations the material related constants.

User Interface

To support trim components with PEM, several new case control and bulk data entries, e.g. TRIMGRP and ACPEMCP respectively, were implemented. In addition, some existing entries, e.g. DISP and PSOLID, received updates to expand its functionalities. An abbreviated documentation is provided here. Complete documentation for all entries is available in *MSC Nastran Quick Reference Guide*.

Case Control Command Summary

Trim Component Definition and Selection	
BEGIN BULK (Case) TRMC	Creates the finite element sub-model of a trim component.
TRIMGRP (Case)	Selects a group of trim components for analysis.

Trim Component Physical Set Output Requests	
DISPLACEMENT (Case)	Requests the form and type of displacement output
VELOCITY (Case)	Requests the form and type of velocity output
ACCELERATION (Case)	Requests the form and type of acceleration output

Parameter and Bulk Data Entries

TRMBIM	Defines formulation for boundary reduced impedance matrix

Bulk Data Entry Summary

Poroelastic Element Property and Materials	
PSOLID	Defines the properties of poroelastic volume elements (CHEXA, CPENTA, and CTETRA entries).
MATPE1	Defines an isotropic poroelastic material which is frequency-independent.
MATF1	Defines a frequency-dependent and isotropic material of the skeleton (solid-phase) of poroelastic medium.

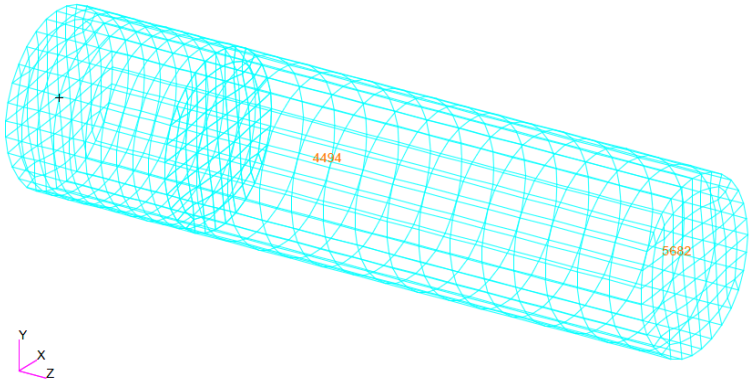
Trim Component Interface Coupling Conditions and Constraints	
ACPEMCP	Defines the interface coupling conditions and constraints of a trim component.
TRMCPL	Defines parameters for computing the interface coupling matrices of a trim component.

Test Cases

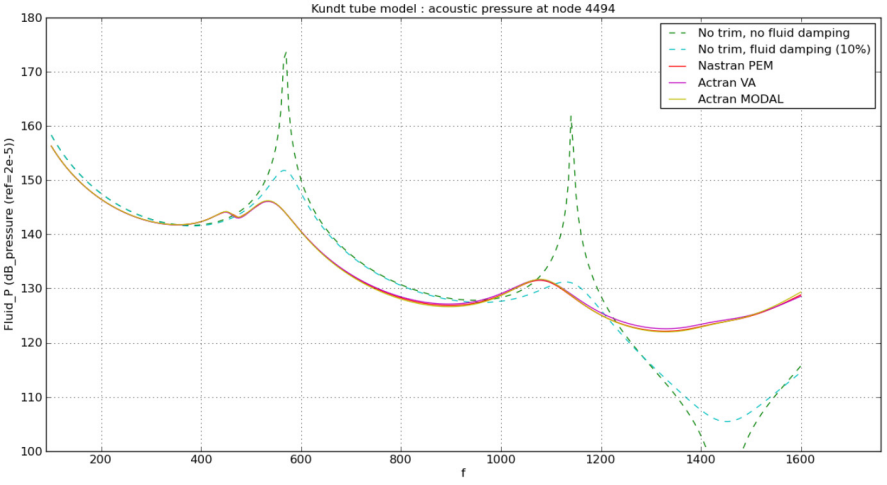
Kundt's Tube

The Kundt's Tube is 400mm in length and 50mm in radius. Glass Wool of 100mm in length and 50mm in radius fills one end of Kundt's Tube. The rest of the Kundt's Tube is filled with air. An unit acoustic source is placed at 280mm (Grid ID) away from the face of Glass Wool. The back face of Glass Wool is clamped and sides are constrained in lateral direction. The PEM properties of Glass Wool is shown in following table.

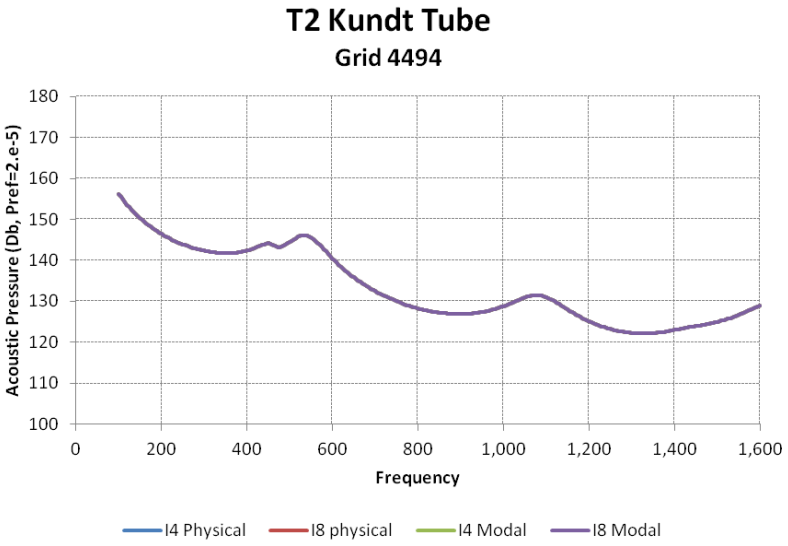
Properties	Value
Porosity (POR)	0.94
Tortuosity (TOR)	1.06
Resistivity (AFR)	40000
Viscous Length (VLE)	5.6E-5
Thermal Length (TLE)	1.1E-4



The acoustic pressure at Grid 4494 as 'MSC Nastran PEM' is compared with results from 'Actran VA' and other results. The comparison is shown in following plot.



In addition, MSC Nastran PEM results from I4 Physical, I8 Physical, I4 Modal and I8 Modal are compared in following plot.

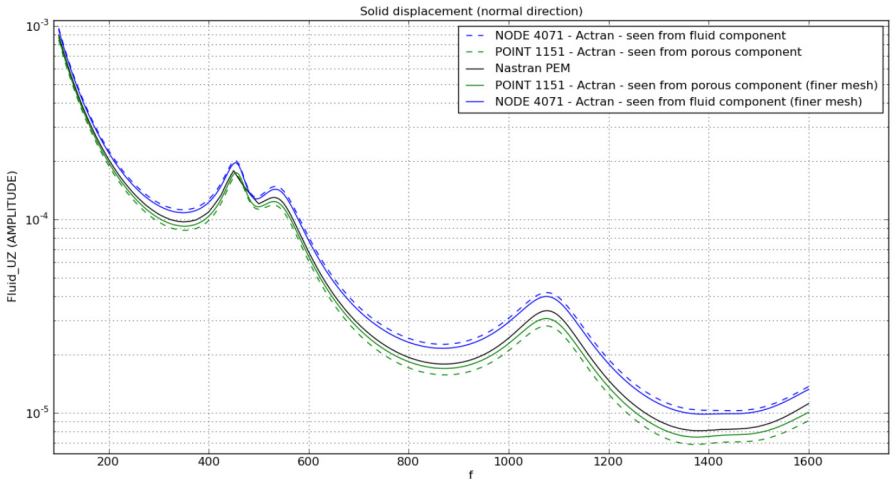


Data recovery for grid 1151 of trim component has been performed. The following figure shows the amplitude of normal fluid displacement at the PEM/cavity interface. The results obtained by the MSC Nastran PEM implementation are compared to results obtained by Actran. The correlation between the results is good, taking into account the fact that the simulation in Actran is performed in physical coordinates whereas the MSC Nastran PEM implementation relies on a projection into the modal space.

The evaluation of the fluid displacement in Actran seems to be more dependent on the mesh resolution than the MSC Nastran PEM implementation. In fact, for a converged mesh, the fluid displacement seen from the porous component should match the fluid displacement seen from the fluid component. This discrepancy is related to the fact that fluid displacements in Actran are secondary variables. As such, they are evaluated from the gradient of the fluid pressure through the Euler's force equation:

$$u_f = \frac{grad(p)}{-\rho \cdot i\omega}$$

The numerical evaluation of the gradient operator may require a longitudinal mesh refinement for better accuracy. In the MSC Nastran PEM implementation, the fluid displacement is directly evaluated from the reduced impedance matrices and the finite element solution, which seems to confer better convergence properties.



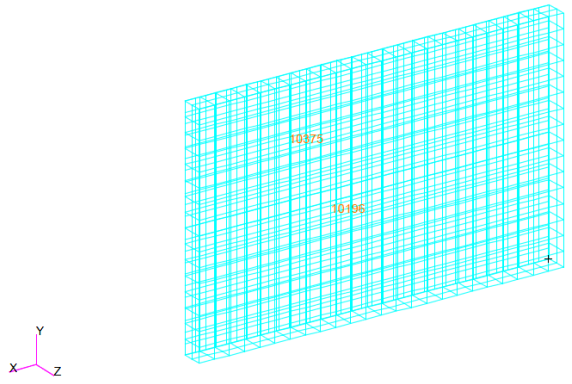
Test decks: tpl/pem/`pemt2.dat`, `pemt2m.dat`

Results: `pemt2.n`, `pemt2m.n`

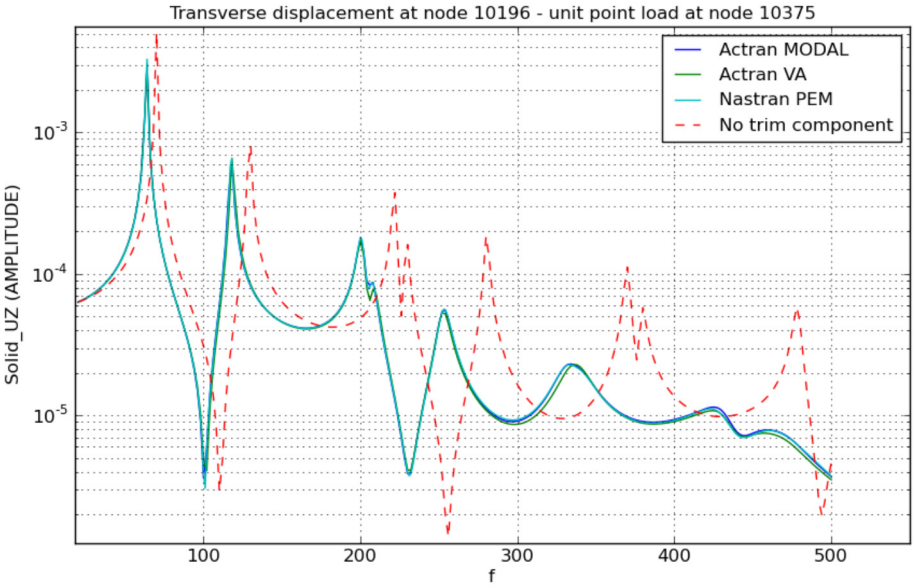
Plate Coated with Foam Layer

An aluminum plate of 350mmx220mm with 1mm in thickness is coated with a layer of foam with a thickness of 20mm. The PEM properties for the foam layer is as follows:

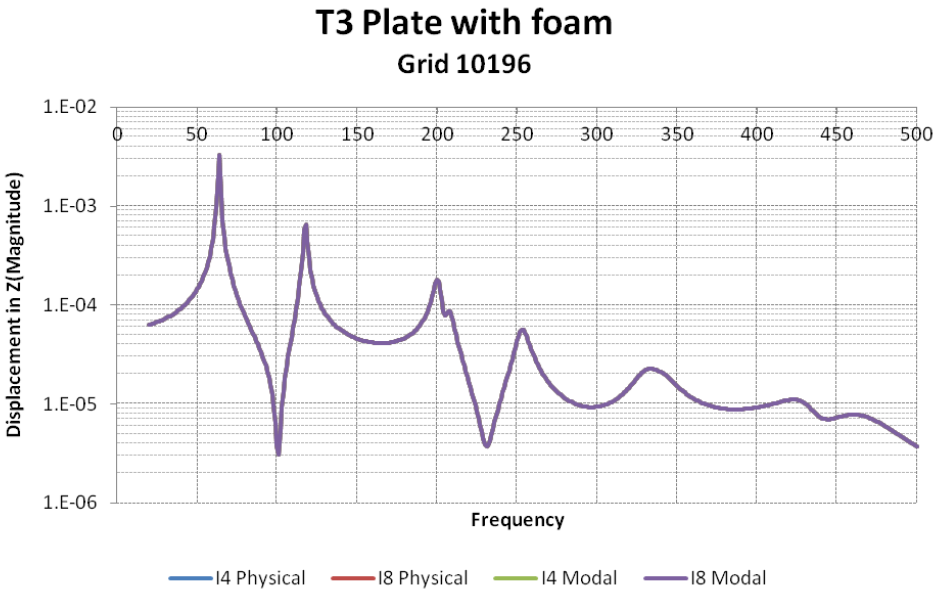
Properties	Value
Porosity (POR)	0.95
Tortuosity (TOR)	1.40
Resistivity (AFR)	2.5E-5
Viscous Length (VLE)	9.32E-2
Thermal Length (TLE)	9.32E-



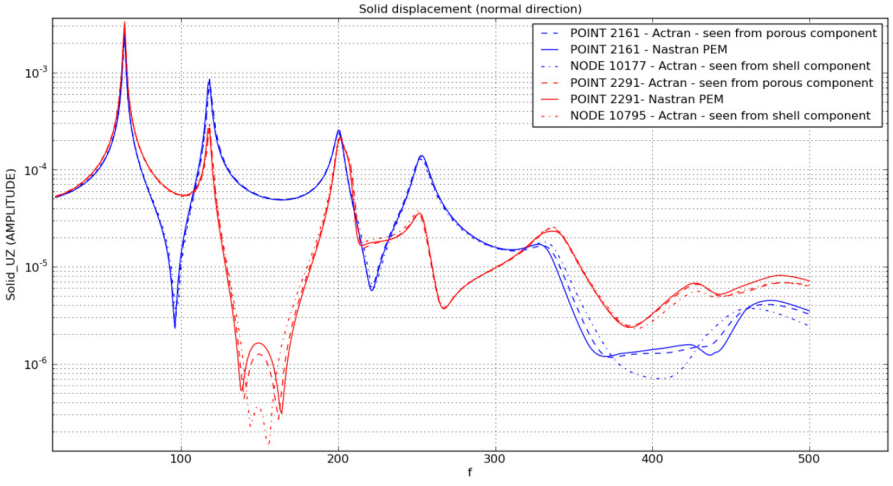
A unit load is applied at Grid 10375 of the plate. The transverse displacement at Grid 10196 of ‘MSC Nastran PEM’ is compared with results from ‘Actran VA’ and other results. The comparison is shown in following plot.



In addition, MSC Nastran PEM results from I4 Physical, I8 Physical, I4 Modal and I8 Modal are compared in following plot.



Data recovery for grid 2161 and 2291 of trim component are performed. The figure below shows the amplitude of the normal skeleton displacement at the porous/plate interface. The match between results obtained by MSC Nastran PEM and Actran is very good. Small discrepancies come from the fact that the Actran results are obtained from a simulation in physical coordinates while the values obtained by MSC Nastran result from a projection into the modal space. Moreover, minor differences between shell formulations in MSC Nastran and Actran are present. Finally, there are small differences between the Actran results seen from the plate component and seen from the porous component. These differences are explained by the fact that non-congruent meshes were used in the Actran analysis.



A brief section of F06 for PEM data recovery is shown as follows. Please note the highlighted area where the interface and TRMC ID are identified. This information is only available in F06, not in PCH.

```
1    PLATE (ALU 1MM)  COATED WITH FOAM (20MM)                              AUGUST  11, 2012  MSC.NASTRAN  8/11/12   PAGE   113
                                                                                                 ST/PEM  TRIMC ID       1
0                                                                                                   SUBCASE 1
      POINT-ID =      2161
                                    C O M P L E X   D I S P L A C E M E N T   V E C T O R
                                                (MAGNITUDE/PHASE)

      FREQUENCY   TYPE          T1             T2             T3             R1             R2             R3
0  2.000000E+01     G      3.907143E-11   2.270205E-11   5.174727E-05   0.0            0.0            0.0
                            10.6483        9.8379         359.2686         0.0            0.0            0.0
0  2.100000E+01     G      4.016117E-11   2.331277E-11   5.228292E-05   0.0            0.0            0.0
                            10.5803        9.8542         359.2577         0.0            0.0            0.0
0  2.200000E+01     G      4.117159E-11   2.387472E-11   5.286079E-05   0.0            0.0            0.0
                            10.5094        9.8716         359.2463         0.0            0.0            0.0
0  2.300000E+01     G      4.210259E-11   2.438796E-11   5.348335E-05   0.0            0.0            0.0
                            10.4355        9.8903         359.2345         0.0            0.0            0.0
0  2.400000E+01     G      4.295398E-11   2.485248E-11   5.415329E-05   0.0            0.0            0.0
                            10.3581        9.9101         359.2220         0.0            0.0            0.0
0  2.500000E+01     G      4.372549E-11   2.526830E-11   5.487369E-05   0.0            0.0            0.0
                            10.2771        9.9314         359.2090         0.0            0.0            0.0
```

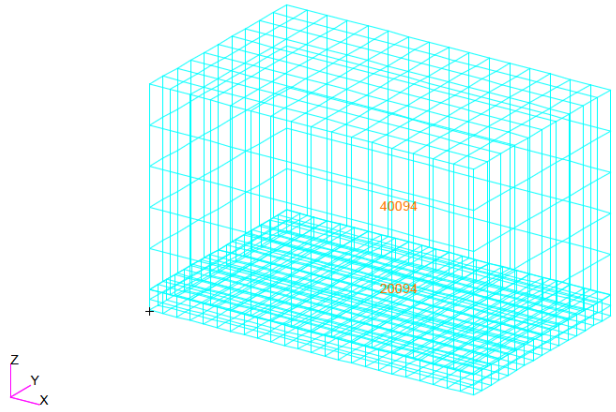
Test decks: tpl/pem/`pemt3.dat, pemt3m.dat`

Results: `pemt3.n, pemt3m.n`

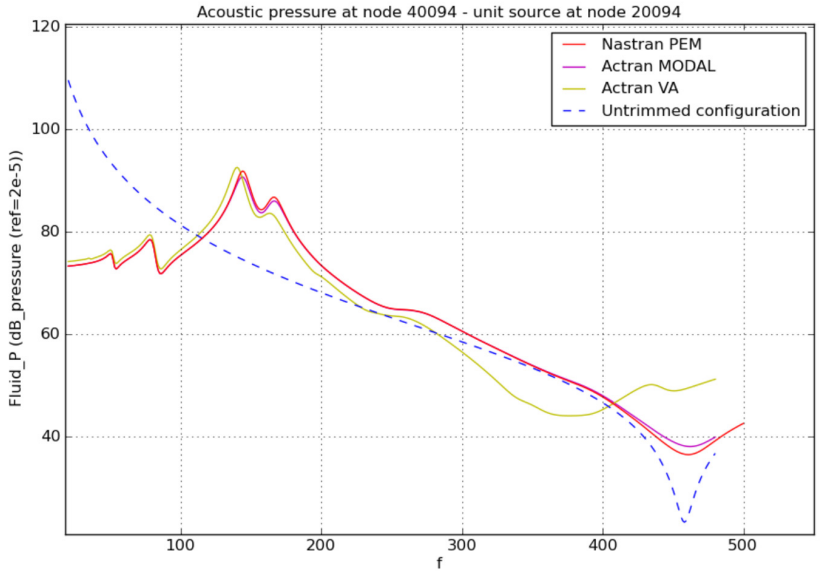
Cavity Coupled with a Foam Layer

A cavity of 350mmx220mm with 200mm in length is coupled with a layer of foam.of 20mm in thickness. The PEM properties for the foam layer is as follows

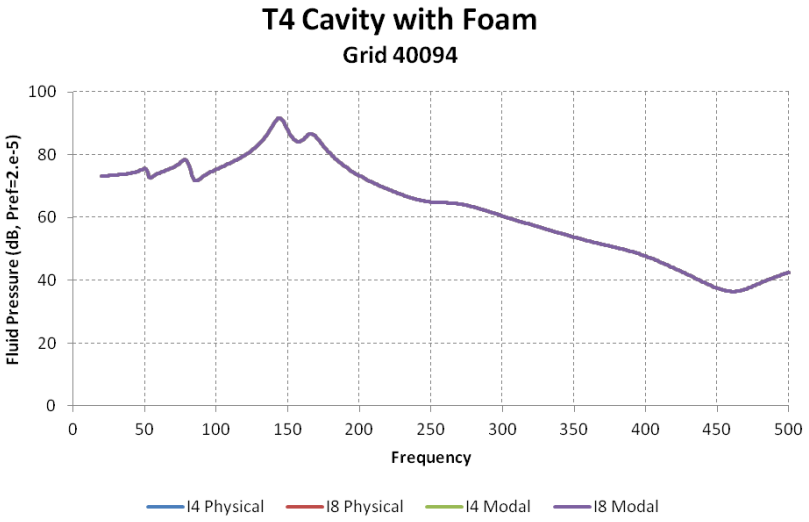
Properties	Value
Porosity (POR)	0.95
Tortuosity (TOR)	1.40
Resistivity (AFR)	2.5E-5
Viscous Length (VLE)	9.32E-2
Thermal Length (TLE)	9.32E-2



A unit load is applied at Grid 20094 of the plate. The acoustic pressure at Grid 40094 of 'MSC Nastran PEM' is compared with results from 'Actran VA' and other results. The plot of comparison is shown in following plot.



In addition, MSC Nastran PEM results from I4 Physical, I8 Physical, I4 Modal and I8 Modal are compared in following plot.

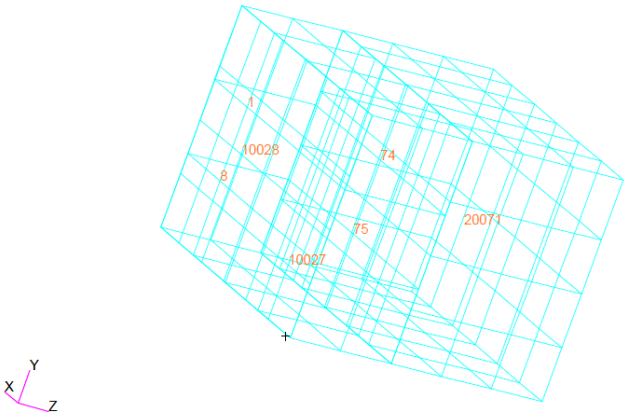


Test decks: tpl/pem/`pemt4.dat`, `pemt4m.dat`

Results: `pemt4.n`, `pemt4m.n`

Frequency Dependent Material in Trim Component

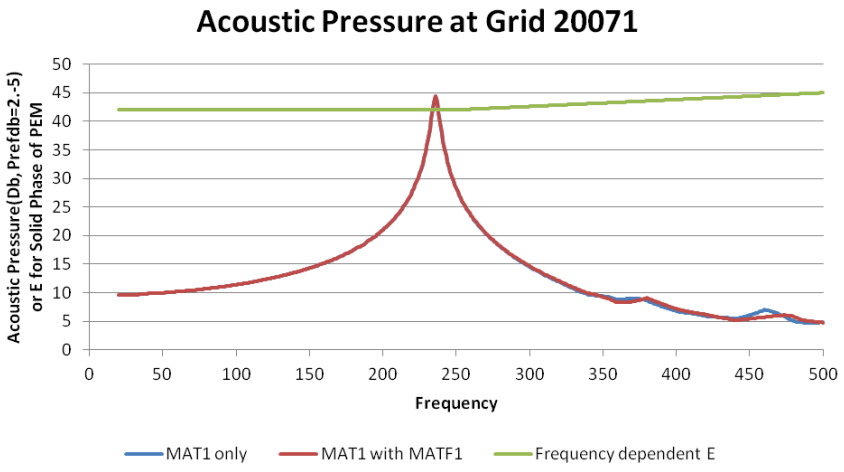
A small test deck is utilized to test the frequency dependent material, MATF1, for a trim component. The plate is 58.333x41.25x1mm. The trim component resides in between structure plate and acoustic cavity.



The material entries are as follows

```
$
$  PSOLID Data
$
$ Poroelastic material: Foam
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
PSOLID          4       1                                PORO
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
$MATPE1 MID     MAT1    MAT10   BIOT
$       VISC    GAMMA   PRANDTL POR     TOR     AFR     VLE     TLE
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
MATPE1  1       2       3                                                +
+       1.84-8  1.40    7.13-1  9.5-1   1.4     2.5-5   9.32-2  9.32-2
$
$  MAT1 Data
$
$ FOAM: Solid phase
$
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
MAT1     2      42.0            0.0     6.00E-07                0.05
$
$ Frequency-dependent isotropic material
$
MATF1   2       11
TABLEM1 11                                                               +
+       20.0    42.0    250.    42.0    500.    45.0    ENDT
```

Following grids are selected for data recovery: 10027, 10028 from structure, 20071 from fluid/cavity, 1,8 on the structure/PEM interface and 74, 75 on PEM/cavity. The acoustic pressure at grid 20071 is shown in following chart.

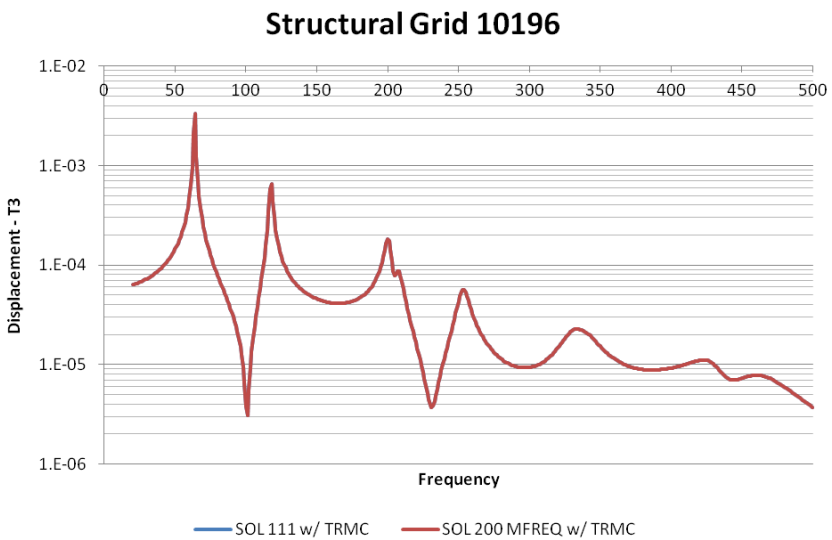


Test decks: tpl/pem/`pemsmlx.dat`(no MATF1), `pemsmlx_matf1.dat`(MATF1 with constant E)

Results: `pemsmlx.n`, `pemsmlx_matf1.n`

SOL 200 support for Trim Component

Test deck for SOL 200 support is converted from `pemt3.dat`, plate coated with foam. First deck is tpl/pem/`pemopt.dat` which has a simple change from SOL 111 to SOL 200 and no design model. The results of Grid 10196 is shown as follows



The second deck is `pemopt2.dat` which is `pemopt.dat` with following design model added.

```
$
$-------------------------------------------------------------------------
$ DESIGN MODEL
```

```
$------------------------------------------------------------------------
$
$...DESIGN VARIABLE DEFINITION
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV(OPTIONAL)
DESVAR, 1,      A1,     1.0,    0.1,     3.0
$
$...IMPOSE X3=X1 (LEADS TO A3=A1)
$DLINK, ID,     DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,     +
$+,     IDV3,   C3,     ...
$
$...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER RELATIONS
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,10,     PSHELL,  3,     T,      ,       ,       ,       ,       +DP1
+DP1,   1,      1.0
$
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1, 20,     W ,     WEIGHT
DRESP1, 21,   g196,     FRDISP, ,       ,       3,   MAX,   10196
DRESP1, 22,   g375,     FRDISP, ,       ,       3,   MAX,   10375
DRESP1, 23,   g815,     FRDISP, ,       ,       3,   MAX,   10815
$
$...CONSTRAINTS
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,21,     21,     -5.-4  ,5.-4
DCONSTR,21,     22,     -5.-4  ,5.-4
DCONSTR,21,     23,     -5.-5  ,5.-4
```

The optimization results is shown as follows,

```
                    ************************************************************
                    S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                    ************************************************************

                                    (HARD CONVERGENCE ACHIEVED)

                                    (SOFT CONVERGENCE ACHIEVED)

                       NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           5
                       NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     4

                                  OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-------------------------------------------------------------------------------------------------------------
                  OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
  CYCLE             APPROXIMATE               EXACT                    OF                       OF
  NUMBER          OPTIMIZATION              ANALYSIS              APPROXIMATION             CONSTRAINT
-------------------------------------------------------------------------------------------------------------

  INITIAL                                 2.079000E-01                                    1.271315E+00

        1          2.102719E-01           2.102715E-01           1.842523E-06             1.456190E+00

        2          2.130226E-01           2.130240E-01          -6.575358E-06             1.147145E+00

        3          2.122993E-01           2.122996E-01          -1.824921E-06             8.520881E-01

        4          2.122996E-01           2.122996E-01           0.000000E+00             8.520881E-01
-------------------------------------------------------------------------------------------------------------
```

Test decks: tpl/pem/`pemopt.dat`, `pemopt2.dat`

Results: `pemopt.n`, `pemopt2.n`

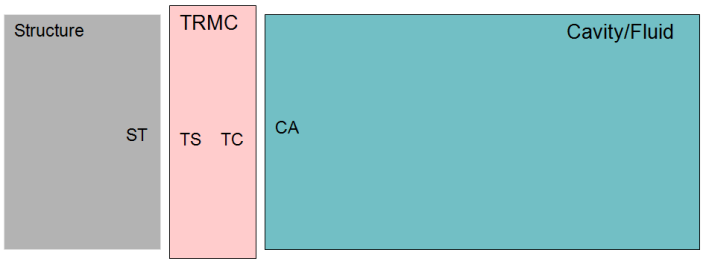
Guidelines

Modeling Technique

a. Since the trim component can be modeled separately from the structure and fluid cavity, it is important to know how to prepare the finite element model for trim component. There are two basic techniques, namely taped-over and wedged-in. Each technique is demonstrated in following graph with simple models.

Taped-over: trim component overlap portion of structure and fluid models

Wedged-in: trim component model placed in-between structure and fluid models



The following table shows the characteristics and guidelines on each technique.

	Characteristic	Guidelines
Taped-over	TRMC model may encroach on both structure and cavity space. Coupling can be handled with tolerances of the algorithm.	• Good for adding TRMCs for existing FSI decks • It is also good for TRMC thickness is much smaller than the dimension of cavity
Wedged-in	TRMC model fits in the space between structure and cavity	• This is a recommended method if the thickness of TRMC is in the same order of magnitude as the dimension of cavity • If separation of structure and fluid is significant, tolerances on ACMODL may need to increase if TRMC is removed.

b. If data recovery for PEM will not be attempted, it is recommended to use `'PARAM,TRMBIM,MODAL'` to obtain faster turnaround.

c. SOL200 support for PEM is limited to `analysis=MFREQ`. In addition, design variables and design constraints must stay on the properties and responses of structure and fluid models. Any design variables or design constraints placed on TRMC will be ignored. As a performance measure, the RIM will be computed only at the analysis stage of 1st design cycle. Those RIM will be assumed to remain constant throughout the rest the SOL 200 job.

d. For SOL 200, the change in design variables will alter the eigenvalues eigenvectors. If `'PARAM,TRMBIM,MODAL'` is selected by the user, the RIM for TRMC will be required to re-compute due to the change of eigenvalues and eigenvectors.

e. FREQ, FREQ1, and/or FREQ2 may be used to specify master frequencies for each TRMC. Under `'BEGIN BULK TRMC'`, the ID of FREQx entries must be the same as TRMC ID to get used. In addition, it is FATAL if no FREQx entry with the ID of TRMC ID can be found. FREQ3, FREQ4, and FREQ5 entries are not supported to specify master frequencies of a TRMC.

4 Advanced Nonlinear (SOL 400)

Contact User Interface Improvements (2013.1)

Introduction

The powerful contact capabilities in MSC Nastran have been enhanced by providing an improved user interface. The input requires less data for large models, is better organized, and is more readable.

A set of new Bulk Data entries which splits the contact data into several entries with fewer continuation lines is introduced to replace the existing BCTABLE and BCBODY entries. The contact parameters can be shared by all contact bodies so that duplicate specifications are not required. The existing BCTABLE and BCBODY entries are still supported to ensure upper compatibility, but within one input file, the old format should not be mixed with the new format. Utilizing the new input, it is easier to define the contact interaction between pairs of contact bodies.

Benefits

This new input interface provides a user friendly interface to facilitate the specifications of contact data. It also allows better support of MSC Nastran from Patran and SimX.

Note: While this material is in the SOL 400 section, the same input options are available in the SOL 100.

Feature Description

The following fourteen new Bulk Data entries are implemented.

- "BCTABL1: Defines a Contact Table (New Form).
- "BCONECT: Defines the Touching and Touched Contact Bodies.
- "BCONPRG: Geometric Contact Parameters of Touching Bodies.
- "BCONPRP: Physical Contact Parameters of Touching Bodies.
- "BCBODY1: Flexible or Rigid Contact Body in 2-D and 3-D (New Form).
- "BCBDPRP: Contact Body Parameters.
- "BCRIGID: Defines a Rigid Contact Body.
- "BCRGSRF: Rigid Contact Body Surface List.
- "BCPATCH: Defines a Rigid Contact Body Made up of Quadrilateral Patches.
- "BCBZIER: Defines a Rigid Contact Body Made up of Bezier Surfaces.
- "BCNURB2: Defines a 2-D Rigid Contact Body Made up of NURBS.
- "BCNURBS: Defines a Rigid Contact Body Made up of NURBS.
- "BCTRIM: Defines the Geometry of Trimming Curves.
- "BCGM700: Defines a Geometrical Contact Body in SOL 700.

User Inputs

Replacement of BCTABLE

The BCTABLE entry is replaced by a new BCTABL1 entry, which references BCONECT entries that define the pairs of contact bodies that interact with one another identification numbers of new BCONECT entries. Each BCONECT entry specifies a list of touching bodies (SLAVE) and another list of bodies (MASTERS) touched by touching bodies. Both bodies are defined by the identification numbers of BCBODY1 Bulk Data entries. In the case of segment-to-segment contact, the terminology of MASTER and SLAVE is not significant.

The geometric properties such as the contact distance are defined on the BCONPRG enter. The physical properties such as the friction coefficient are defined on the BCONPRP entry. As multiple pairs can reference these geometric and physical properties, redundant data can be eliminated. Furthermore, the data blocks can be placed in ay order. The data is entered as a keyword value to make the input and output more readable.

The following schematic flowchart illustrates the relationships of these four new Bulk Data entries.

Case Control Command:

BCONTACT = BCID

Bulk Data entries:

BCTABL1	BCID	ID1	ID2	ID3	ID4	ID5	ID6	ID7	
	ID8	ID9	-etc.-						

BCONECT	ID	BCGPID	BCPPID	IDSLAVE	IDMASTER				
	"SLAVES"	IDSL1	IDSL2	IDSL3	IDSL4	IDSL5	IDSL6	IDSL7	
		IDSL8	IDSL9	-etc.-					
	"MASTERS"	IDMA1	IDMA2	IDMA3	IDMA4	IDMA5	IDMA6	IDMA7	
		IDMA8	IDMA9	-etc.-					

BCONPRG	BCGPID		PARAM1	VAL1	PARAM2	VAL2	PARAM3	VAL3	
	PARAM4	VAL4	PARAM5	VAL5	-etc.-				

BCONPRP	BCPPID		PARAM1	VAL1	PARAM2	VAL2	PARAM3	VAL3	
	PARAM4	VAL4	PARAM5	VAL5	-etc.-				

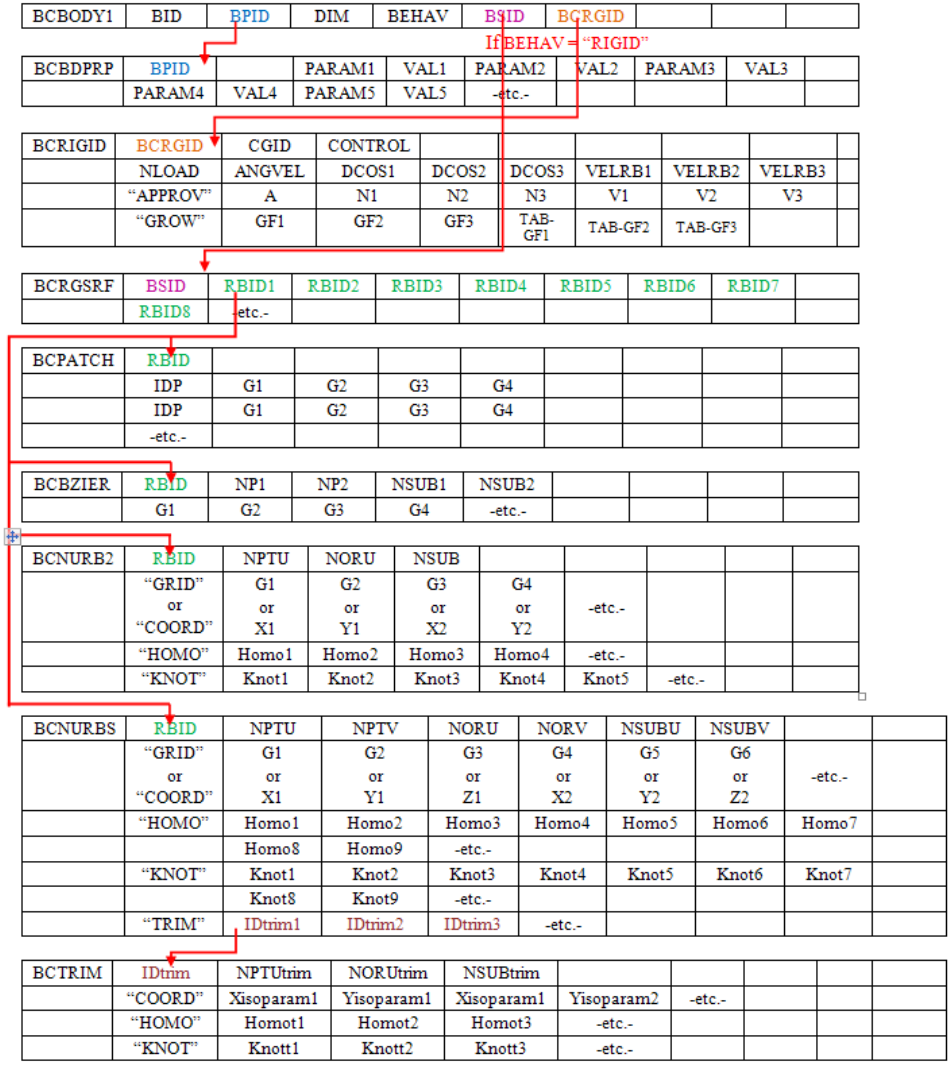
Replacement of BCBODY (SOLs 101 and 400)

The BCBODY entry is replaced by a new BCBODY1 entry, which refers the new BCBDPRP entry to define parameters of contact bodies in random order. For rigid contact, a new Bulk Data entry BCRIGID referred by the BCRGID of BCBODY1 entry defines the geometry, approaching velocity and growth of the rigid body.

An optional new BCRGSRF entry referred by the BSID of BCBODY1 entry is used to group a list of rigid contact forms by RBID. If BCRGSRF does not exist, the rigid contact forms will be referred directly from the BSID of BCBODY1 entry.

The original BCBODY forms (PATCH3D, BEZIER, NURBS2-D, and NURBS) describing the properties of a rigid body are separated into individual new Bulk Data entries BCPATCH, BCBEZIER, BCNURB2, and BCNURBS. These entries are referred by the RBID of BCRGSRF entry or BSID of BCBODY1 entry if BCRGSRF entry does not exist. In addition, a new BCTRIM entry defines the geometry of trimming curves, which is referred by BCNURBS.

The relationships of these new Bulk Data entries are illustrated below.

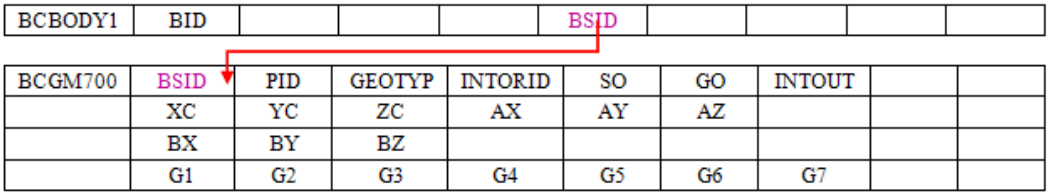


Replacement of BCBODY (SOL 700)

For SOL 700, the BCBODY entry is replaced by a new BCBODY1 entry, which refers to BSURF, BCBOX, BCPROP, BCMATL, BCSEG, BCGRID, or BCGM700 Bulk Data entry through BSID field. The new Bulk Data entry BCGM700 defines the rigid geometry of a contact body.

The relationships of these two new Bulk Data entries are illustrated below.

Bulk Data entries:



Limitation

In a contact model, the new Bulk Data entries cannot coexist with BCTABLE and BCBODY entries.

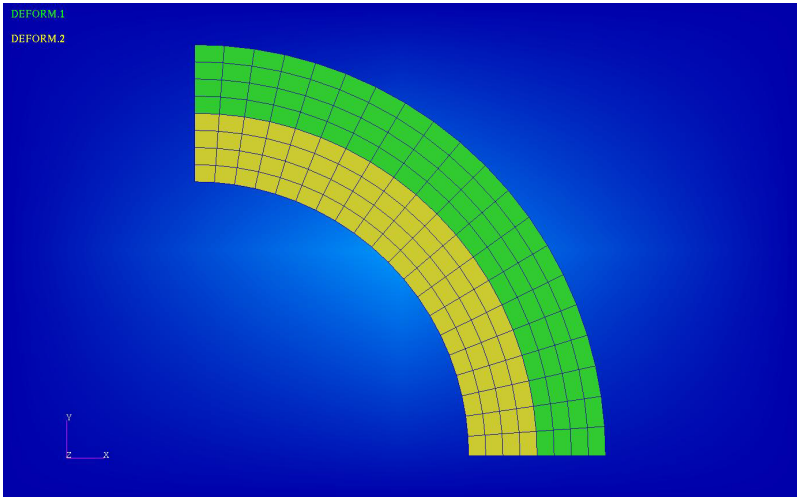
Test Cases

The MSC Nastran TPL has a subdirectory `ncntinp` that contains a series of simple examples, which apply the new contact entries across a variety of solution sequences. The names of the test cases and the original models are summarized in the following table.

Test Case	Original ModelTPL or ETL Subdirectory/Name	Description
nwct02a	tpl/cntglue/cglumlt7	SOL 101 permanent glued contact.
nwct02b	tpl/cntglue/cgluwlwl	SOL 103 permanent glued contact.
nwct02c	tpl/cntglue/cglub105	SOL 105 permanent glued contact.
nwct02e	tpl/cntglue/cglub108	SOL 108 permanent glued contact.
nwct02f	tpl/cntglue/cglub109	SOL 109 permanent glued contact.
nwct02h	tpl/cntglue/cglub111	SOL 111 permanent glued contact.
nwct02i	tpl/cntglue/cglub112	SOL 112 permanent glued contact.
nwct04	tpl/md400ug/nug_20	SOL 400 node-to-segment contact.
nwct05a	tpl/nsg_400/nsg001e	SOL 400 segment-to-segment contact with small sliding.
nwct05b	etl/err20121/nas11109	SOL 400 segment-to-segment contact with finite sliding.
nwct06	tpl/htmph_400/htmph13	SOL 400 thermal contact.
nwct08a	tpl/nsg_400/nsg020c	SOL 400 ALLBODY contact.
nwct09	tpl/3dcnt400/nlc002f	Contact in transient analysis of SOL 400.
nwct12	tpl/mdr3s400f/nug_49a	SOL 400 contact analysis with friction stress limit.
nwct13	tpl/mdr4s400f/dqnear_g1	SOL 400 thermal contact.
nwct14	tpl/mdr3s400e/nanac04	SOL 400 rigid body contact.
nwct15	tpl/mdr4s400a/hsh_flm_ns1a	SOL 400 thermal contact.
nwct16	tpl/nlc_400a/nlc034b	SOL 400 rigid body contact.

Test Case	Original ModelTPL or ETL Subdirectory/Name	Description
nwct18	tpl/nlc_400a/nlc031b	SOL 400 rigid body contact.
nwct20	tpl/mdr4s400c/nnlst05b	SOL 400 rigid body contact.
nwct21	tpl/mdr4s400c/nnlst05g	SOL 400 rigid body contact.

A typical example is `nwct14` as shown below.



This is a 2-D contact model with automatic detection of discontinuities. The original BCTABLE entries are listed as follows.

```
BCTABLE  0                              5
         SLAVE    1        0.        0.       0.       0.       0
                  0        0         0
                  FBSH   1.+20      .9        0.
         MASTERS  2
         SLAVE    1        0.        0.       0.       0.       0
                  0        0         0
                  FBSH   1.+20      .9        0.
         MASTERS  3
         SLAVE    1        0.        0.       0.       0.       0
                  0        0         0
                  FBSH   1.+20      .9        0.
         MASTERS  4
         SLAVE    2        0.        0.       0.       0.       0
                  0        0         0
                  FBSH   1.+20      .9        0.
         MASTERS  3
         SLAVE    2        0.        0.       0.       0.       0
                  0        0         0
                  FBSH   1.+20      .9        0.
```

```
          MASTERS 4
BCTABLE   1                        5
          SLAVE   1        0.      0.       0.       0.       0
                  0        0       0
                  FBSH   1.+20    .9        0.
          MASTERS 2
          SLAVE   1        0.      0.       0.       0.       0
                  0        0       0
                  FBSH   1.+20    .9        0.
          MASTERS 3
          SLAVE   1        0.      0.       0.       0.       0
                  0        0       0
                  FBSH   1.+20    .9        0.
          MASTERS 4
          SLAVE   2        0.      0.       0.       0.       0
                  0        0       0
                  FBSH   1.+20    .9        0.
          MASTERS 3
          SLAVE   2        0.      0.       0.       0.       0
                  0        0       0
                  FBSH   1.+20    .9        0.
          MASTERS 4
```

The converted new entries are listed below.

```
BCTABL1  0       11      12      13      14      15
BCTABL1  1       11      12      13      14      15
BCONECT  11      2001    3001    1       2
BCONECT  12      2001    3001    1       3
BCONECT  13      2001    3001    1       4
BCONECT  14      2001    3001    2       3
BCONECT  15      2001    3001    2       4
BCONPRG  2001            BIAS    .9
BCONPRP  3001            FRLIM   1.+20
```

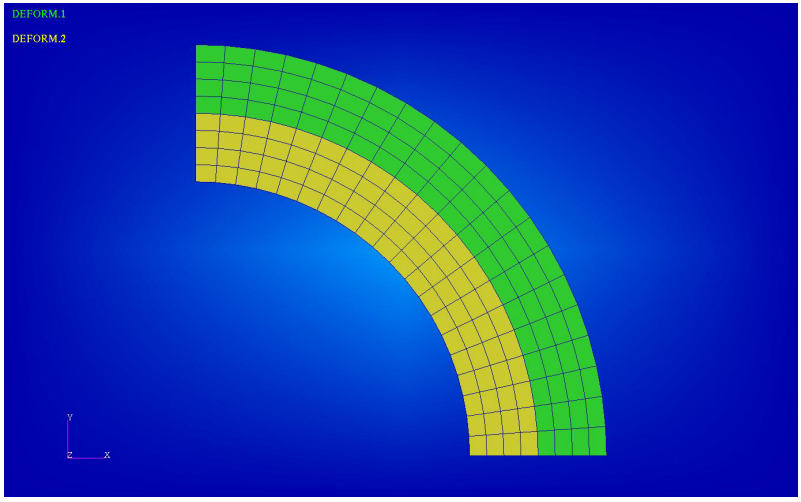
The original BCBODY entries are shown below.

```
BCBODY   1       2D      DEFORM  1       0               -1
BCBODY   2       2D      DEFORM  2       0               -1
BCBODY   3       2D      SYMM            1                1       0
         0       0.      0.      0.      0.      0.       0.      0.
        RIGID   0       1       BODY_3
         NURBS2D -2      2       50
                 0.     75.              0.     180.
                1.      1.
                 0.      0.     1.      1.
BCBODY   4       2D      SYMM            1                1       0
         0       0.      0.      0.      0.      0.       0.      0.
        RIGID   0       1       BODY_4
         NURBS2D -2      2       50
                180.     0.              75.      0.
                1.      1.
                 0.      0.     1.      1.
```

The converted new entries are listed as follows:

```
BCBODY1  1        4001     2D       DEFORM   1
BCBODY1  2        4001     2D       DEFORM   2
BCBODY1  3        4002     2D       SYMM     6001     5001
BCBODY1  4        4002     2D       SYMM     6002     5001
BCBDPRP  4001              IDSPL    -1
BCBDPRP  4002              IDSPL    1
BCRIGID  5001     0        0
         0        0.       0.       0.       0.       0.       0.       0.
BCNURB2  6001     -2       2        50
         COORD    0.       75.      0.       180.
         HOMO     1.       1.
         KNOT     0.       0.       1.       1.
BCNURB2  6002     -2       2        50
         COORD    180.     0.       75.      0.
         HOMO     1.       1.
         KNOT     0.       0.       1.       1.
```

The above example demonstrates that it is easier to specify contact data using the new Bulk Data entries. The following figure shows the contact status that indicates the contact pairs have been defined correctly.



Interface Digimat to Nastran SOL400/SOL700 (2013.1)

Introduction

Digimat is the state of the art linear and nonlinear multi-scale material modeling software suite from *e-Xstream engineering*. The Digimat software enables the prediction of the constitutive behavior of heterogeneous and/or anisotropic materials such as Polymer Matrix Composites (PMC), Rubber Matrix Composites (RMC) and Metal Matrix Composites (MMC). One typical and commonly used application of Digimat is in the engineering plastics area where Digimat is used to model the linear and nonlinear material behavior of glass-reinforced thermo-plastic injection molded parts. In such a case, Digimat will typically take into account the fiber orientation predicted by injection molding software and act as the micro-mechanical material model within the structural finite element analysis software.

Note: The interface to Digimat is not supported on the Windows 32 platform.

Please contact support@e-xstream.com for a demonstration or for more detailed information concerning the Digimat software suite or consult http://www.e-xstream.com/.

Benefits

Digimat to MSC Nastran SOL400/SOL700 is powerful software which enables you to bridge the gap between processing simulation, e.g. injection molding and drape molding, and predictions carried out on the structural side, e.g. implicit and explicit structural FEA and life time prediction. The interfaces handle data per element. In other words, they account for all kind of effects resulting from e.g. fiber orientation, residual stresses or even temperature fields to further perform most accurate predictions on the structural part. In short, Digimat-CAE enables an integrative, accurate and efficient approach to multi-scale material and structure modeling by taking into account the process-induced material microstructure in the FEA of the final part structure. Moreover, failure indicator can be defined not only at composite level, but also at phase level. They can also be coupled together.

Digimat helps material suppliers and end users from across the industries to:

- Accurately predict the nonlinear micromechanical behavior of complex multiphase composites materials and structures (PMC, RMC, MMC, nano, ...)
- Leverage the exceptional properties of composite materials such as stiffness to weight ratio and design freedom
- Faster to market, innovative and high quality products
- Produce lighter weight parts to reduce the induced gas emissions and raw material consumption
- Reduce the number of material testing and prototypes to lower development time and costs

Feature Description

MATDIGI entry

Defines material data for the advanced composites with Digimat.

MATDIGI **Material Digimat**

Defines material data for the advanced composites with Digimat from e-Xstream engineering (SOL 400 and SOL 70) only). For more information about e-Xstream engineering and Digimat, please contact support@e-xstream.com or consult http://www.e-xstream.com/

Format:

1	2	3	4	5	6	7	8	9	10
MATDIGI	MID	UDID	K	NU	E	G	RHO		

Example:

MATDIGI	5	10	6789	0.29	1.577e+03	10000.	3000.		

Field	Contents
MID	Material identification number. (Integer > 0)
UDID	References UDNAME entry (Required, Integer>0)
K	Bulk Modulus (Real?0.0 or blank) Used SOL700 only.

To view the complete entry, please see MATDIGI (p. 2562) in the *MSC Nastran Quick Reference Guide*.

UDNAME entry

Provides the name of a file that can be referenced for MATDIGI.

Format:

1	2	3	4	5	6	7	8	9	10
UDNAME	UDID								
	NAME								

Field	Contents
UDID	Unique UDID (Integer>0). See Remark 1.
NAME	Name of Digimat material file (Character).

Remarks:

1. The UDID is referenced by a MATDIGI entry.
2. The NAME can be of any length with a maximum of 64 words which corresponds to four lines of data in fields 2 thru 9.
3. If only a NAME with no path (e.g., *DigimatMaterial.mat*) is supplied, the file is assumed to be located in the same directory as the Nastran input file. If an absolute or relative path is supplied (e.g, */local/user/digimat/DigimatMaterial1.mat*), it will be used.
4. The filenames must be all lowercase.

UDSESV entry

Define the number and names of user state variables for material user subroutines (SOL 400 and SOL 700 only). This entry is required when MATDIGI is defined to define the number of state variables.

NLOUT entry

Selects additional nonlinear output quantities as referenced by NLSTRESS Case Control Command in SOL 400. This command is requested for post processing Digimat External State Variables (ESV).

Remarks:

1. The post processing of ESV is only possible by using the new OP2 format (set POST 1).

DYPARAM,LSDYNA,DBEXTENT entry

Specify output files to be written in SOL700. This entry can be used to post process Digimat External State Variables (ESV). For solid elements, user must use the NEIPH option followed by the number of state variables to be written in d3plot result file. For shell elements, you must use the NEIPS option followed by the number of state variables to be written in d3plot result file.

Commands usage related to Digimat material MATDIGI

The new MATDIGI command is not yet available in Patran and SimXpert. Moreover, the number of state variables changes from a Digimat material, depending on the Digimat material model that has been setup inside Digimat platform (not provided by MSC. For more information or a demonstration about Digimat graphical user interface, please contact support@e-xstream.com).

To overcome this issue, Digimat-CAE graphical user interface can automatically generate all bulk data entry that are needed to make Digimat to Nastran SOL400/SOL700 computation. The content of these files have to be copy and pasted in the bulk data section. Examples of such interface files are presented below for SOL400 and SOL700.

```
$ Digimat-MSC Nastran SOL400 5.0.1 - 24/10/2013  11:29:44
$.... The following lines give information on the MSC Nastran SOL400 commands
$.... that are needed to build-up a coupled Digimat-MSC Nastran SOL400
$.... material model.
$
$ To access to the Digimat state variables, insert in SUBCASE section:
   NLSTRESS(PLOT,NLOUT=1)=ALL
$
$ USER MATERIAL CARD
$ To setup the definition of the Digimat user material card in the MSC Nastran
$ input deck, copy/paste the following command lines in the BULK DATA section.
$   1  ><   2  ><   3  ><   4  ><   5  ><   6  ><   7  ><   8  ><   9  ><  10  >
MATDIGI 1       1                                       1.4E-009
$   1  ><   2  ><   3  ><   4  ><   5  ><   6  ><   7  ><   8  ><   9  ><  10  >
UDNAME  1
        DigimatMaterialt1
$ STATE DEPENDENT VARIABLES
$.... A given number of state dependent variable must be defined to initiate
$.... the coupling between Digimat and MSC Nastran SOL400.
$.... This number varies depending upon the material model the user works with.
$   1  ><   2  ><   3  ><   4  ><   5  ><   6  ><   7  ><   8  ><   9  ><  10  >
UDSESV          37
$
        SV2     HV2     SV3     HV3     SV4     HV4     SV5     HV5
        SV6     HV6     SV7     HV7     SV8     HV8     SV9     HV9
        SV10    HV10    SV11    HV11    SV12    HV12    SV13    HV13
        SV14    HV14    SV15    HV15    SV16    HV16    SV17    HV17
        SV18    HV18    SV19    HV19    SV20    HV20    SV21    HV21
        SV22    HV22    SV23    HV23    SV24    HV24    SV25    HV25
        SV26    HV26    SV27    HV27    SV28    HV28    SV29    HV29
        SV30    HV30    SV31    HV31    SV32    HV32    SV33    HV33
        SV34    HV34    SV35    HV35    SV36    HV36    SV37    HV37
$   1  ><   2  ><   3  ><   4  ><   5  ><   6  ><   7  ><   8  ><   9  ><  10  >
NLOUT   1       TOTTEMP
        ESV     SV2     SV3     SV4     SV5     SV6     SV7
                SV8     SV9     SV10    SV11    SV12    SV13
                SV14    SV15    SV16    SV17    SV18    SV19
                SV20    SV21    SV22    SV23    SV24    SV25
                SV26    SV27    SV28    SV29    SV30    SV31
                SV32    SV33    SV34    SV35    SV36    SV37
```

Figure 4-1 Example of Digimat to Nastran SOL400 interface file generated by Digimat graphical user interface.

```
$ Digimat-MSC Nastran SOL700 5.0.1 - 15/08/2013  16:29:58
$.... The following lines give information on the MSC Nastran SOL700 commands
$.... that are needed to build-up a coupled Digimat-MSC Nastran SOL700
$.... material model.
$
$ USER MATERIAL CARD
$ To setup the definition of the Digimat user material card in the MSC Nastran
$ input deck, copy/paste the following command lines in the BULK DATA section.
MATDIGI 1       1       4.0E+004                3.0E+0041.4E-009
UDNAME  1
        digimatmaterialt21
$ STATE DEPENDENT VARIABLES
$.... A given number of state dependent variable must be defined to initiate
$.... the coupling between Digimat and MSC Nastran SOL700.
$.... This number varies depending upon the material model the user works with.
UDSESV          51
$
        SV1     HV1     SV2     HV2     SV3     HV3     SV4     HV4
        SV5     HV5     SV6     HV6     SV7     HV7     SV8     HV8
        SV9     HV9     SV10    HV10    SV11    HV11    SV12    HV12
        SV13    HV13    SV14    HV14    SV15    HV15    SV16    HV16
        SV17    HV17    SV18    HV18    SV19    HV19    SV20    HV20
        SV21    HV21    SV22    HV22    SV23    HV23    SV24    HV24
        SV25    HV25    SV26    HV26    SV27    HV27    SV28    HV28
        SV29    HV29    SV30    HV30    SV31    HV31    SV32    HV32
        SV33    HV33    SV34    HV34    SV35    HV35    SV36    HV36
        SV37    HV37    SV38    HV38    SV39    HV39    SV40    HV40
        SV41    HV41    SV42    HV42    SV43    HV43    SV44    HV44
        SV45    HV45    SV46    HV46    SV47    HV47    SV48    HV48
        SV49    HV49    SV50    HV50    SV51    HV51
```

Figure 4-2 Example of Digimat to Nastran SOL700 interface file generated by Digimat graphical user interface.

Example: Tensile test on a fiber reinforced plastic using Digimat to SOL400 interface (Doc/mdug/ch089)

Test files:
ch089solids400.dat (Nastran input deck),
digimatsolidsol400.mat (Digimat material file)
OT_aligned_dumbbell.dof (Fiber orientation file)

This exercise presents the use of Digimat and Nastran SOL400 within the framework of short fiber reinforced plastics (SFRP) modeling. Practicing the proposed exercise will help understand the concepts lying behind the Digimat multi-scale modeling approach.

The application is to compute the tensile response of a dumbbell through a Digimat coupled SOL400 analysis. This dumbbell is cut of a plate made of polyamide, glass fiber reinforced. Typical linear stress-strain curves at two loading angles are presented at Figure 4-3. Mesh and boundary condition are presented at Figure 4-4 Left part of the dumbbell is clamped while an imposed displacement of 2.5mm is imposed to right part. The fiber orientations along x-direction are predicted by Moldflow and are presented at Figure 4-5. It can be interpreted as follows: at the skin of the dumbbell, the fibers are nearly fully aligned along X-direction, while at the core of dumbbell, the fibers are oriented along transverse direction. Typical result are presented at Figure 4-6. This simple example illustrates the interest of taking into account different fiber orientation at each element.

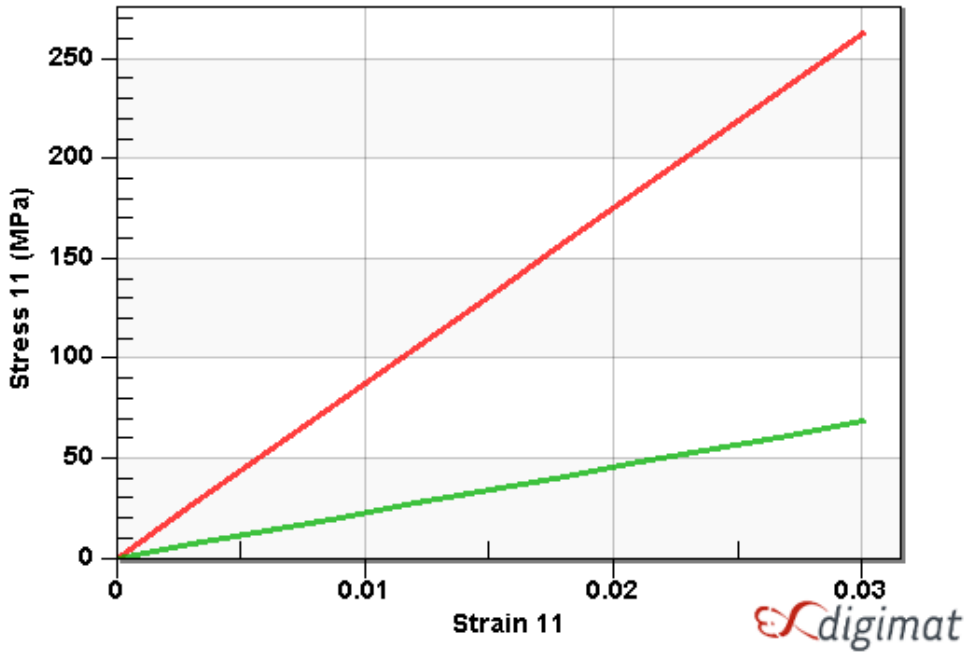


Figure 4-3 Typical Stress-Strain curve for Polyamid glass fibers reinforced. In red: Traction along fiber direction. In green: Traction along transverse direction.

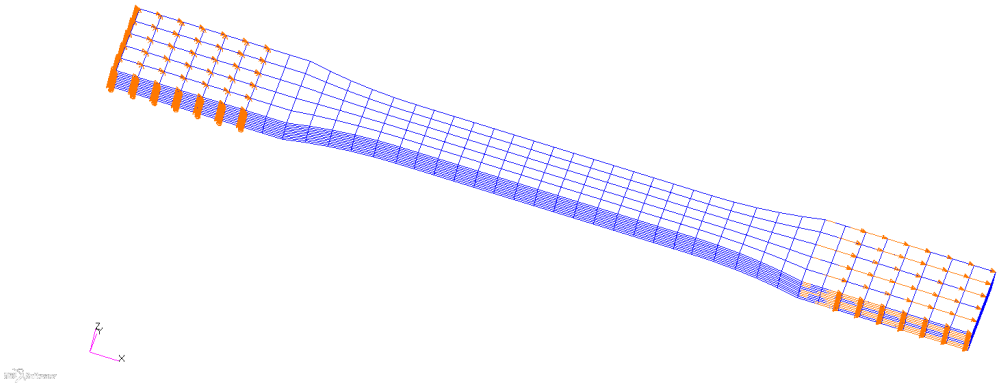


Figure 4-4 Mesh and boundary conditions.

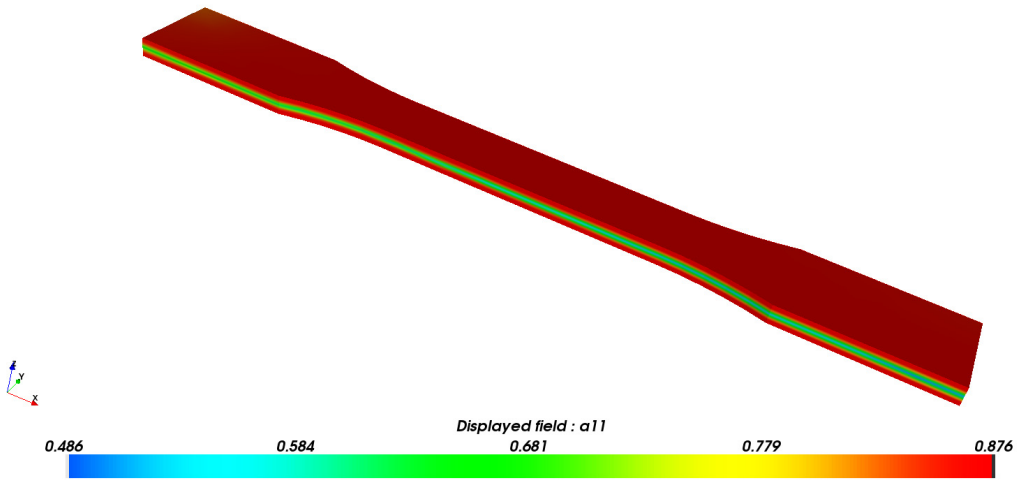


Figure 4-5 Fiber orientation along x-direction.

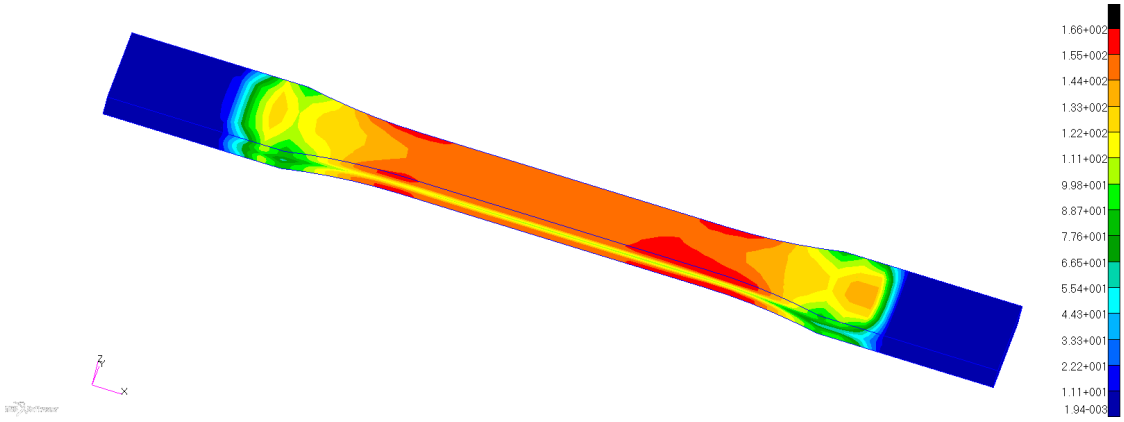
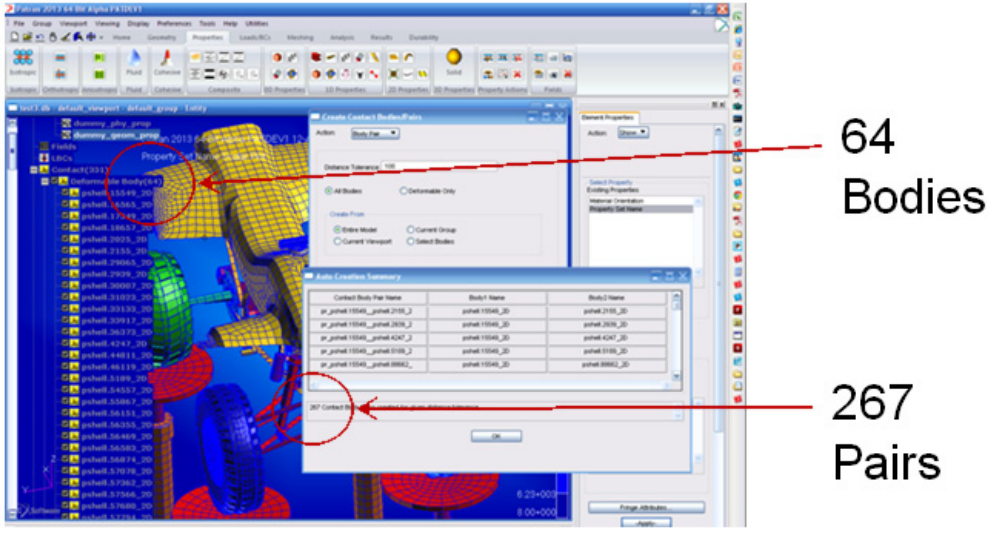


Figure 4-6 Equivalent Von Mises stress distribution.

Patran Support for MSC Nastran Contact Pair Modeling

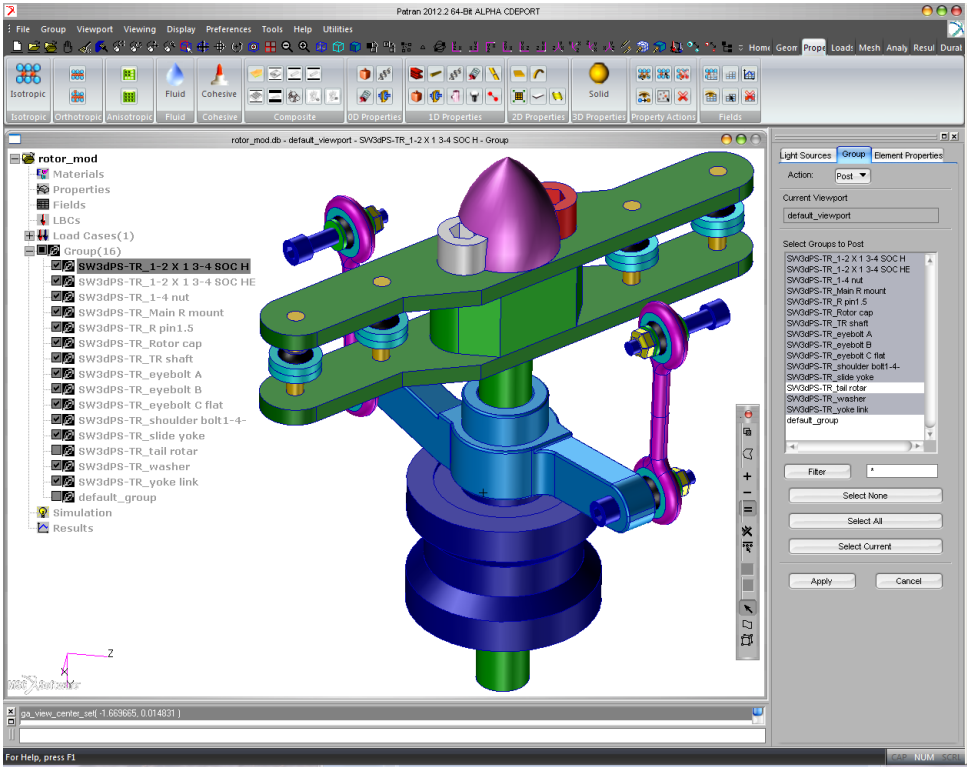
MSC Patran 2013 supports the new MSC Nastran 2013.1 contact pair user interface by providing a new Contact Pair Load and Boundary Condition (LBC) graphical user interface. The interface allows users to create contact pairs using individual contact property sets, or re-use existing interface definitions by referencing existing property sets. This capability, along with Patran's global property editing capability, provide a powerful, scalable UI that is suited equally well to creating both individual contact pairs, and large numbers of complex contact pairs, using common properties. Along with the new contact pair user interface is a set of tools that allow users to automatically create contact bodies and contact pairs based on user-defined criteria. Here is a summary of the Patran capabilities:

- Write New BCTABL1/BCONECT MSC Nastran Bulk Data entry
- Contact Pair Graphical User Interface
- Automatic Contact Body Creation Tool
  - Based on Property Sets
  - Based on Material Sets
  - Based on Group Membership
  - Based on Element Topology
  - Based on Connectivity
  - Based on Geometry
- Automatic Contact Pair Creation Tool Based on Proximity
- Model Browser Tree Interaction
- Powerful Global Property Editing
- Scalability for Large Number of Contact Bodies



Contact Pair Modeling Capability

Since the introduction of surface to surface contact in MSC Nastran almost 10 years ago, the general 3-D contact capabilities in MSC Nastran have been based on contact tables and multi-body contact. As the size and complexity of nonlinear models, along with the use of glued contact to connect dis-similar meshes, has grown, so has the complexity of the contact table input. To minimize the complexity of the contact input MSC has introduced an alternate form of contact body interaction control in the form of **contact pairs**. The idea of using contact pairs is to simplify contact modeling by simplifying the input. This is done by providing a scalable, flexible user interface that can be very simple and straight forward for a simple contact problem, but can be extended to support the most complex contact simulations that may include dozens or even hundreds of contact bodies. The key to this flexibility is using property sets that can be attached to the contact pairs. One pair can be used by an individual contact pair, or by many contact pairs. This provides the capability to attach the properties to the contact pair LBC set, but still edit a large number of contact pairs at one time (through a shared property set) using the global property editor.



Linear Stress Recovery (2013.1)

In analyses where there is a nonlinear step followed by linear perturbation steps, the element stiffness and mass are currently formed in the nonlinear static (NLSTAT) or nonlinear dynamic (NLTRAN) step. The global assembled matrices can consist of both classical MSC Nastran elements and Advanced elements which are then used in linear perturbation steps like modal analysis (MODES), direct frequency (DFREQ), modal frequency (MFREQ), modal transient (MTRAN), direct complex-eigenvalue analysis (DCEIG) and modal complex-eigenvalue analysis (MCEIG). However, the Advanced elements are not part of the conventions SOL 101 solution sequence. This has the following repercussions:

- Incorrect results – For Marc elements that share the same topology as MSC Nastran elements (e.g., CHEXA, CQUAD4), stress recovery is carried out based on existing flow for MSC Nastran elements. This would be incorrect for non-linear materials, composites, etc. since the classical MSC Nastran element's formulation is completely different from the Advanced element's formulation.
- Missing results - For Advanced elements that are unique (e.g., CAXISYM), recovered results are missing.

Note when performing perturbation analysis in SOL 400, NLIC is not supported so, one can only perform a single perturbation step in the simulation.

Note that the buckling perturbation is available with SOL 400, but stress recovery is with the advanced elements is not available.

Background

The nonlinear prestressing steps solve the system of equations:

$K_T\,\delta u = F - I$ for NSLSTAT and

$\left(\frac{aM}{\Delta t^2} + \frac{bB}{\Delta t} + cK^T\right)\delta u = F(t) - I - X$ for NLTRAN

where M is the mass matrix, B is the damping matrix, K_T is the tangent stiffness matrix, δu is the iterative displacement, F is the nodal load vector, I is the internal force vector, X is additional right-hand-side terms for dynamics based on the previous displacement state, a, b, and c are dynamic coefficients based on the type of dynamic operator used.

For the Marc elements are supported in SOL400, all the matrices above M, B and K^T are supported.

The linear steps that follow the nonlinear step use these matrices in various ways:

MODES: $(K^T - \lambda M)\,\phi = 0$

DFREQ: $(-\omega^2 M + K^T)\,\upsilon = P$ (various forcing frequencies are provided through FREQx cards)

MFREQ: 2 steps

a. same as MODES
b. DFREQ analysis but with generalized modal coordinates instead of physical degrees of freedom

MTRAN: 2 steps

a. Same as MODES

b. Direct transient analysis but with generalized modal coordinates instead of physical degrees of freedom

DCEIG: $(-\omega^2 M + i \omega B + K^T) \upsilon = P$

This is similar to DFREQ but with complex damping. This causes the displacement vector to have 2 parts : real vector and imaginary vector

MCEIG: $(-\omega^2 M + i \omega B + K^T) \phi = 0$: complex eigenvalue analysis (similar to MODES)

In all cases, after these linear steps, a displacement vector is obtained. This displacement vector can be a eigenvector (MODES, MCEIG) or function of time (MTRAN) or function of frequency (DFREQ, MFREQ, DCEIG). Also, the displacement vector is real-valued (MODES, MTRAN, DFREQ, MFREQ) or complex value (real and imaginary parts) (MCEIG, DCEIG)

The displacement vector has to be passed into the stress recovery routines for advanced elements and the stresses and strains have to be computed.

In a linear step following the nonlinear step, the strain increment is linearized about the nonlinear deformed state. This linearization depends on the analysis options used:

- For Total Lagrange, the linearized Green-Lagrange strain is formed,
- For Updated Lagrange (Additive), the linearized logarithmic strain is formed,
- For Updated Lagrange (Multiplicative), the deformation gradient is formed.

The linearized stress increment is computed from the linearized strain increment by using the material tangent. At the end of the nonlinear step, the material tangent could be a highly nonlinear one (accounting for plasticity, creep, failure, etc.). The kind of material tangent to be used for the subsequent linear steps could be a function of the type of linear analysis and needs to be determined:

- For linear perturbation steps (modes, buckling) where the actual displacement shape magnitude has no physical meaning, the convention is to use the linear elastic tangent.
- For other linear steps where the actual displacement shape has a physical meaning the elastic tangent is used.

Limitations

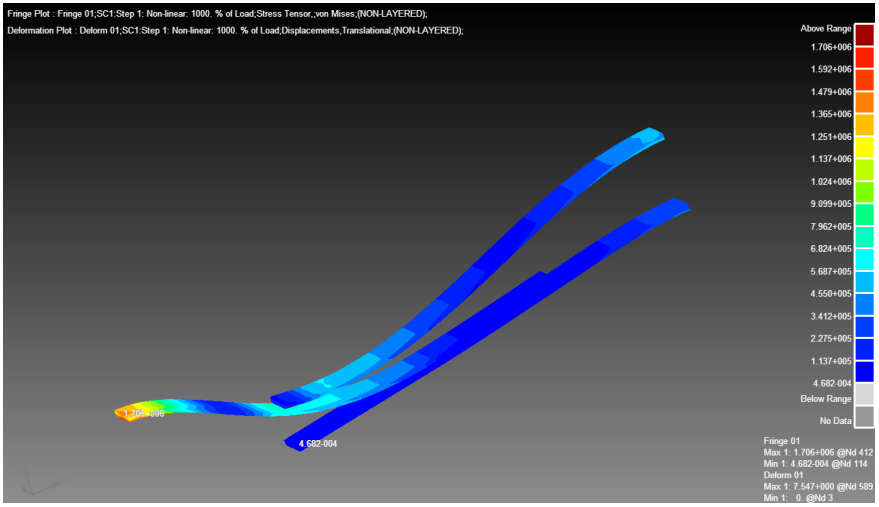
- Does not with interface elements and cohesive material model (MCOHE).
- Cannot perform a nonlinear analysis after a perturbation step.
- Perturbation analysis with composite materials has assorted issues.
- Contact condition not satisfied in perturbation analysis when segment-to-segment contact.
- The element ID is missing in output (`f06`) file in using `MFREQ`, `MTRAN`, and `MODES` perturbation analysis.

- Temperature dependent material behavior is not considered in perturbation analysis.
- `RESVEC` is not available in perturbation analysis.
- The iterative solvers should not be used in perturbation analysis.

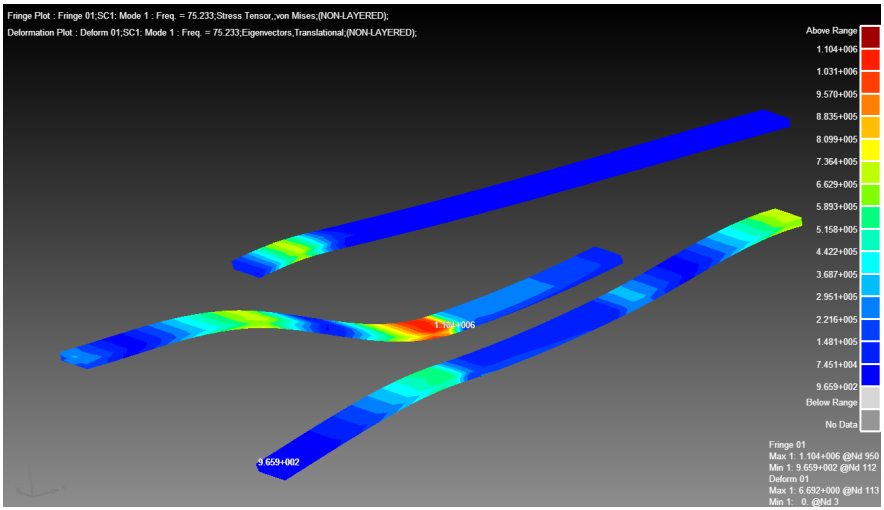
Example

In a simple example, three beams are used as shown below where a rigid tool is use to deflect the top beam such that it contacts the middle beam which proceeds to contact the bottom beam:

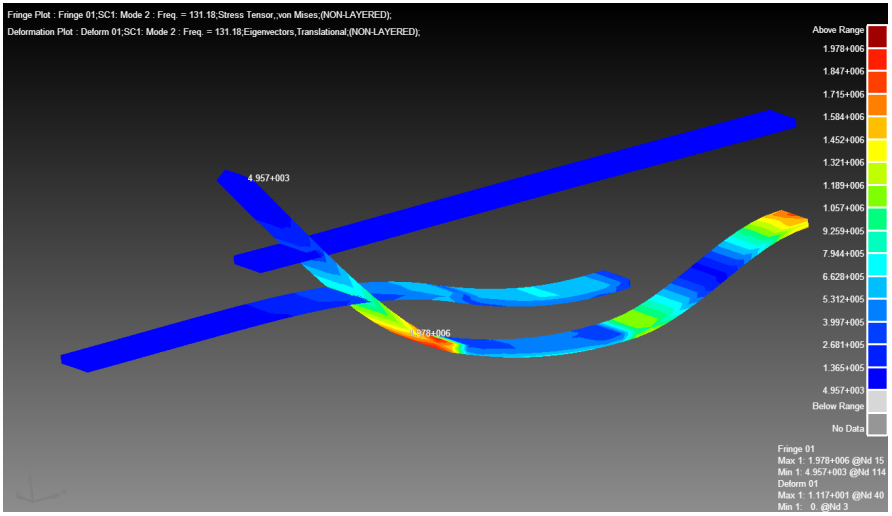
Initial Geometry



After Contact



Mode 1 – 75.2 Hz



Mode 2 – 131.18 Hz

Example of Input – MODES

```
$! NASTRAN Control Section
$! File Management Section
$! Executive Control Section
SOL 400
CEND
ECHO = SORT
```

```
$! Case Control Section
SUBCASE 1
STEP 1
   SPC = 28
   BCONTACT = 1
   ANALYSIS = NLSTAT
   NLSTEP = 2
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CORNER)=ALL
STEP 2
   ANALYSIS = MODES
   SPC = 28
   METHOD = 3
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CENTER)=ALL
BEGIN BULK
$! Bulk Data Pre Section
PARAM   POST          -1
```

Sequential Thermal-Mechanical Analysis with Linear or Quadratic Temperature Distribution (2013.1)

In MD Nastran 2010, a shell element type with multiple degrees of freedom through the element thickness was introduced to simulate constant, linear or quadratic temperature distribution across the thickness for thermal analysis. This capability provides a more accurate simulation through the shell thickness. It also offers ease of modeling to avoid using 3-D brick elements on thin shell structure.

Besides the above advanced thermal element type, a bi-directional coupled thermal-mechanical analysis was implemented in the same version using an increment-level staggered approach. This feature allows a realistic modeling of both thermal and mechanical physics for a structure. Although the coupled thermal-mechanical analysis supports the temperature mapping of multi-degrees of freedom heat shell elements, this staggered coupling scheme must be solved at the same time step for both physics.

Prior to the 2013.1 release, mechanical analysis in single physics did not have the data structure to interpret the temperatures of the grids generated internally for 3-D thermal shell elements. It can only process the temperatures defined on the element grids (top temperatures). The bottom and middle temperatures associated with the virtual nodes are unrecognized. As a result, incorrect mechanical analysis results are encountered.

Since thermal and structural analyses may be performed by different engineer groups with huge models, the coupled thermal-mechanical analysis is often impractical. Therefore, a new capability is added to allow temperature mapping of multi-degrees of freedom heat shell elements in sequential thermal-mechanical analysis.

Benefits

This feature offers a better representation of the temperature distribution for mechanical analysis within single physics.

Feature Description

The thermal results with 3-D shell elements can be saved in a punch file. In the subsequent structural run, the user will be able to access the results by specifying a NLMOPTS,TEMPP Bulk Data entry. The bottom and middle temperatures of the 3-D shell elements will be processed to determine the effect of thermal loading.

User Interface

To apply sequential thermal-mechanical analysis with linear or quadratic temperature distribution, both thermal and mechanical models must specify the Bulk Data entry NLMOPTS,TEMPP as follows.

- NLMOPTS,TEMPP,LINE - linear distribution
- NLMOPTS,TEMPP,QUAD - quadratic distribution

Inputs

Thermal Run

In the Case Control section, request the punch outputs of temperatures by the following command.

```
THERMAL(SORT1,PRINT,PUNCH) = ALL
```

The multi-degrees of freedom thermal shell elements are specified by NLMOPTS, TEMPP, LINE, or NLMOPTS, TEMPP, QUAD Bulk Data entry for linear or quadratic temperature distribution across the shell thickness.

Structural Run

Define temperature load using the following Case Control command.

```
TEMPERATURE(LOAD) = n
```

where `n` is the set ID of TEMP entries that user wants to apply thermal load analysis.

In the Bulk Data section, include the punch file by

```
Include thermal_job_name.pch
```

or specify TEMP Bulk Data entries directly to input the temperatures of virtual grids. The initial temperatures of the virtual grids can be defined by TEMPN1 or TEMPD entry.

To activate the thermal effects of virtual grids, the identical NLMOPTS,TEMPP entry defined in the previous thermal run must be specified in the mechanical model.

Outputs

MSC Nastran will save HRMSP data block in OP2 and DBALL. This data block contains relationships between the top grid IDs and the internally generated grid IDs for bottom and middle points. The user may also output the mapping of user specified grids (top grids) to bottom or middle internal grids by the Bulk Data entry

```
NLMOPTS,TEMGO,YES
```

- Thermal Run

  The program will create a punch file with name `thermal_job_name.pch` which consists of temperatures of user specified grids as well as the virtual grids created internally.

- Structural Run

  Besides HRMSP data block, outputs are same as the existing mechanical results.

Guidelines and Limitations

- Both thermal and mechanical models must have consistent type of temperature distribution. Either quadratic or linear temperature distribution should be specified for both physics.
- In mechanical models, the temperatures of dummy thermal grids or points must be removed from the temperature load set. These TEMP entries are used to define temperature boundary conditions in thermal analysis. The dummy thermal grids or points do not exist in mechanical models.
- Only one set of TEMP(LOAD) is allowed for each mechanical model.
- The element types supported include CQUAD4, CTRIA3, and CQUAD8, which are associated with PSHELL property entries. The program switches these shell elements to advanced elements automatically by generating PSHLN1 entries internally if the original model does not have any PSHLN1 or PSHLN2 entries.
- This capability is not supported in a chained analysis.
- This functionality is available for SOL 400 only.

Example (tpl/uncoupl400/rg_h3ht.dat and rg_h3st.dat)

This pair of thermal and mechanical models demonstrates how to transfer three values of quadratic temperatures of CQUAD4 shell elements from steady state thermal analysis into a separate mechanical analysis.

The thermal model consists of two free convection boundary conditions; one is applied on the top surface, while the other is applied on the bottom face using the feature of quadratic temperature distribution through the shell thickness. Two dummy SPOINTs with IDs 212 and 213 are used to define ambient temperatures for convection boundary conditions. A fixed time stepping scheme with five increments is specified.

To output temperatures of 3-D thermal shell and the mapping of top, bottom and middle grids, define THERMAL(PUNCH) Case Control Command and NLMOPT Bulk Data entry as follows.

```
SOL 400
CEND
ECHO = SORT
$
TEMPERATURE(INITIAL) = 102
SUBCASE 1
$
   THERMAL(SORT1,PRINT,PUNCH)=ALL
   FLUX(PRINT)=ALL
   ANALYSIS = HSTAT
   SPC = 104
   NLSTEP = 1
BEGIN BULK
$
NLMOPTS TEMPP   QUAD
        TEMGO   YES
PARAM   POST            1
$
MAT4            1      1.       1.  0.2804
MATT4           1       1        2
PSHELL          1       1     0.4       1               1
PSHELL          2       1     0.4       1               1
GRID            1         9.00057 9.00014 0.09999
GRID            2        8.000554 9.00018 0.09999
```

```
:
CQUAD4          1       1     201      83       81     202
CQUAD4          2       1      83      85       80      81
:
PCONV          12       2       0     0.0
MAT4            2                        0.000393
SPOINT        212
SPC             1     212       1232.2462
TEMP          102     212232.2462
CONV          540      12               0     212
CHBDYG        540           AREA4
              201      83      81     202
:
PCONV          13       3       0     0.0
MAT4            3                        0.00142
SPOINT        213
SPC             2     213       1 730.391
TEMP          102     213 730.391
CONV          640      13               0     213
CHBDYE        640       1       6
:
TEMPD         102     70.
SPCADD        104       1       2
NLSTEP          1                                                    +
+       FIXED   5
ENDDATA
```

The temperature results of the above model are written to a punch file `rg_h3ht.pch`. Since there are five increments during analysis and the results are saved for each increment (default), this punch file has five sets of temperature results. In a modified file `rg_h3ht_inc5.pch` shown below, the results of the last increment are saved with the temperatures of dummy points 212 and 213 removed.

```
$TITLE    =
$SUBTITLE=
$LABEL    =
$TEMPERATURE
$REAL OUTPUT
$SUBCASE ID =           1
$LOAD FACTOR =   1.0000000E+00
TEMP*                   5               1    5.084365E+02
TEMP*                   5               2    5.084365E+02
:
TEMP*                   5             211    5.084365E+02
$TEMP*                  5             212     2.322462E+02
$TEMP*                  5             213     7.303910E+02
TEMP*                   5       101000001    6.539524E+02
:
TEMP*                   5       101000240    5.820754E+02
TEMP*                   5       101000241    6.539524E+02
TEMP*                   5       101000242    5.820754E+02
```

Another output file `rg_h3ht.f06` (part of zipped `rg_h3ht.n`) lists the mapping of user specified grids (top grids) to bottom and middle internal grids as follows.

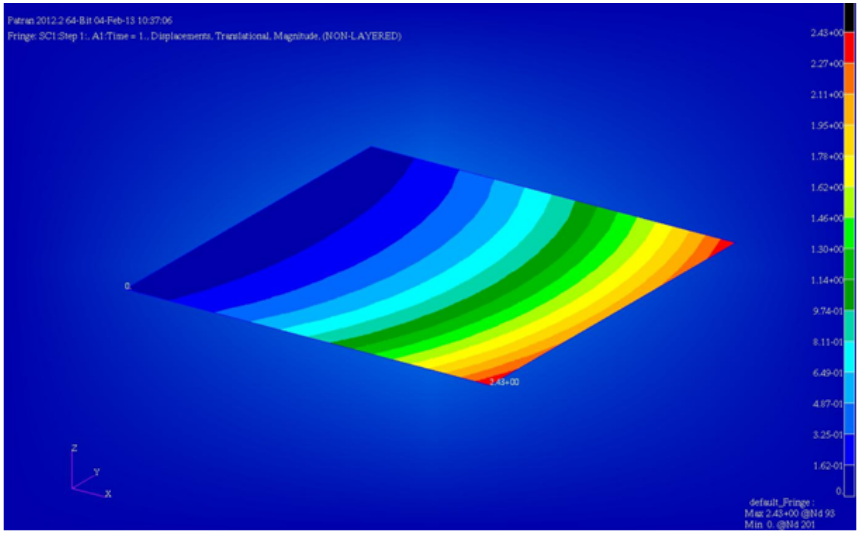
```
          U S E R    G R I D    P O I N T (T O P)    M A P P I N G    T O    B O T/M I D    I N T E R N A L    G R I D

   TOP          1          2          3          4          5          6          7          8          9
10
         BOT    101000001  101000003  101000005  101000007  101000009  101000011  101000013  101000015  101000017  101000019
         MID    101000002  101000004  101000006  101000008  101000010  101000012  101000014  101000016  101000018  101000020
```

```
   TOP                  11              12              13              14              15              16              17
18              19              20
       BOT    101000021  101000023  101000025  101000027  101000029  101000031  101000033  101000035  101000037   101000039
       MID    101000022  101000024  101000026  101000028  101000030  101000032  101000034  101000036  101000038   101000040
   TOP                  21              22              23              24              25              26              27
28              29              30
       BOT     101000041  101000043  101000045  101000047  101000049  10100005 1 101000053  101000055  101000057  101000059
       MID     101000042  101000044  101000046  101000048  101000050  101000052  101000054  101000056  101000058  101000060
   :
```

In structural run, a fixed time stepping scheme with ten increments is specified. The thermal load is defined through TEMPERATURE(LOAD) Case Control command with ID=5. This thermal load is the only load applied to the model, with all six degrees of freedom along one edge of the patch (grid IDs 201 to 211) constrained. The temperatures are imported by including the modified punch file created from previous thermal model as follows.

```
SOL 400
CEND
ECHO = SORT
$
TEMPERATURE(INITIAL) = 70
SUBCASE 1
   SPC = 1
   TEMPERATURE(LOAD) = 5
   ANALYSIS = NLSTAT
   NLSTEP = 1
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CORNER)=ALL
BEGIN BULK
$
include rg_h3ht_inc5.pch
NLMOPTS TEMPP   QUAD
PARAM   POST            1
$
MAT1            1   3.E+7             0.31  0.2804  0.0001
PSHELL          1       1      0.4       1               1
PSHELL          2       1      0.4       1               1
GRID            1         9.00057 9.00014 0.09999
GRID            2        8.000554 9.00018 0.09999
:
CQUAD4          1       1      201      83       81      202
CQUAD4          2       1       83      85       80       81
:
SPC1            1  123456     201    THRU     211
$
NLSTEP          1
+       FIXED   10
TEMPD         70     70.
ENDDATA
```

The displacement contours resulted from the thermal loading effect are shown below.



User Defined Subroutine (UDS) Improved Interface 2013.1

MSC Nastran supports user defined subroutines (UDS) through SCA services. The definition and implementation of SCA services originally require the user to create interface related files, build SCA component, and set up the environment. The work involved is too much for a simple user case. The UDS ease of use project aims to simplify the user's tasks. MSC Nastran will now build SCA service behind the scene based on user model or source files; there will be no direct SCA development work required for user.

Benefits

This release will relieve user from developing SCA component for user defined subroutines. Users will not need to write any SCA related files; only user subroutine source files are required. This will allow the user to focus on engineering application not the overhead work associated with SCA interfaces and services.

The Patran 2013 Beta supports the identification of user defined subroutines and the job initiation using these user subroutines.

Feature Description

Two MSC Nastran command line keywords, `uds` and `udssave`, are added in this release. These command line keywords are used to specify user source file and SCA component build location. In user source file, there may be one or more predefined user subroutines. These predefined user subroutines will be used to define interfaces in user SCA service. The build location is where the SCA component will be built.

uds

This command line keyword is used to specify user source file with predefined user subroutines. It is specified at MSC Nastran job submittal time.

```
nast20131 myjob.dat uds=mysource.F
```

The specified source file will be compiled and implemented. Since only one user source file is allowed, the user must put all the predefined user subroutines and other related subroutines in one source file. The user source file can be in FORTRAN or C++.

An alternative method is provided to the user to handle source code by means of the model input file with `BEGIN UDS` section. The `BEGIN UDS` section is a newly added section starting with `BEGIN UDS` and ending with `ENDDATA` or another `BEGIN` statement. The user can put source code in this section and set *uds* command line keyword to `model`. MSC Nastran will extract source code in this section by the following command:

```
nast20131 myjob.dat uds=model
```

The user service name in `CONNECT SERVICE` statement in input file is needed. Only one `CONNECT SERVICE` statement is allowed in this release.

An example of this method is shown as follows:

```
$ Connect service SCA.MDSolver.Obj.Uds.Materials,  name it material.
$ The material service has implemented UMAT interface.
connect service material 'SCA.MDSolver.Obj.Uds.Materials'
SOL 400
CEND
...
BEGIN BULK
...
PSOLID,1,1,,,1,
CHEXA   1    1    17    20    53    31    19    22
        54   32
...
$ Specify UMAT type UDS from material service for the element
MATUSR,1,,,3,
MATUDS  1     MATUSR  material UMAT
,REAL,5.e10,2.5e10,
...
GRID   17        -1.   .1    -.1
GRID   19        -1.   .1    .1
...
BEGIN [BULK] UDS= material
  Subroutine ext_umat
  ...
  end
ENDDATA
```

udssave

This command line keyword is used to specify where the SCA component will be built and whether the built component will be saved for later use.

```
nast20131 myjob.dat uds=mysource.F udssave=/home/temp
```

If `udssave` is not given, the MSC Nastran output directory will be used as build location, and the built component will be deleted after MSC Nastran run. If `udssave` is specified, the user component will be saved in specified location and could be reused later.

To reuse a built user component without building it again, only specify the `udssave` in command line.

```
nast20131 myjob.dat udssave=/home/temp
```

Limitation and Potential Enhancement

There is only one `BEGIN UDS` section and `CONNECT SERVICE` statement supported. If there is more than one in the input file, only the first one is used.

MSC SDK 2013 version is required and the `<SDK_INSTALL_DIR>/Tools` directory must be added in the `PATH` environment variable.

Test Case

Model file `genuds_numat.dat` and source file `ext_umatnotify.F` are used for testing. The source file has two predefined user subroutines: `EXT_UMAT` and `EXT_NOTIFY`. The model file has `CONNECT SERVICE` statement with user service name `'UDSTest.Myapp'`.

Run MSC Nastran as the following:

```
nast20131 genuds_numat.dat uds=ext_umatnotify.F
```

The user SCA component is built in current working directory and loaded successfully in MSC Nastran. Utility function `print06` is called in `EXT_NOTIFY` to print out messages.

To save the built component and reuse it in later MSC Nastran run, specify the `udssave` in command line.

```
nast20131 genuds_numat.dat uds=ext_umatnotify.F udssave=/home/temp
nast20131 genuds_numat.dat udssave=/home/temp
```

Enhancement to User Defined Subroutine Interface

Introduction

In this release, the enhanced User Defined Subroutine (UDS) features include:

2. A new entry UDSESV to allow user to define the number and name of state variables.
3. A new user subroutine UMAT for general material. User-defined state variables, element Gaussian point volume, procedure phase and convergence flag are passed to this subroutine. The MATUDS entry is modified to support the UMAT type subroutine.
4. A new user subroutine UCOHES for material used for cohesive element. User-defined state variables, procedure phase and convergence flag are passed to this subroutine. The MATUDS entry is modified to support UCOHES type subroutine.
5. Utility functions GET_ELEM_PARAM, GET_NODE_PARAM, and GET_GLOBAL_PARAM can be called to obtain the information of element and node as well general data in material user subroutines.
6. A new SCA interface SCAIMDSolverRuntimeInfo and its method *notify*. The *notify* method in a service will be called at the beginning of load case, the end of load case, the beginning of increment and the end of increment.
7. A new entry GENUDS to allow user to define input data for the *notify* method in SCAMDSolverRuntimeInfo interface. User can define integer, real and character data in this entry, these data will be passed to the *notify* method as arguments when it gets called.
8. The C++/FORTRAN template implementation files for UMAT and UCOHES subroutines and the SCAIMDSolverRuntimeInfo interface.
9. Output of user defined state variables. Up to 100 state variables can be selected to output to F06, OP2 and DBALL files. DRA access to state variables output is available for GUI modelers.

Benefits

This release enhances the capabilities of user defined materials. The user could define state variables for user material subroutines. More arguments are added in the UMAT and UCOHES subroutines for passing internal data, such as element Gaussian point volume, procedure phase and convergence flag. Within material user subroutines, the information of element and node as well analysis data can be obtained using utility functions. A new SCA interface is available for user to get notification from MSC Nastran when the analyzing procedure runs into the beginning and end of load case and increments. The output support of user state variables makes post-processing of state variables possible.

Technical Discussion

UDSESV Entry

The UDSESV defines user state variables. It is a global entry used for all material UDS with state variables. Each state variable has its default name as SVi with the *i* being the index number of the state variable. For example, the 3rd state

variable has its default name SV3. User can define another name for state variable using the UDSESV entry. These state variables will be passed to material UDS as arguments when the UDS gets called.

One special note is for the first state variable, it is reserved for temperature. The temperature is passed as the first element of the state variable array in UDS. User defined state variables are available from the second position in the state variable array.

Format:

UDSESV		NSTATS							
	SV2	SV2_NAME	SV3	SV3_NAME	SV4	SV4_NAME	SV5	SV5_NAME	
	SV6	SV6_NAME	.etc.						

Example:

UDSESV		3							
	SV2	VAR2	SV3	VAR3					

Field	Contents
NSTATS	The number of user defined state variables. (Integer >= 1)
SVi	The default nominal name of state variable (CHARACTER, *i* >= 1, where *i* is the index number of the state variable)
SVi_NAME	The state variable name defined by user (CHARACTER, Default = SVi, where *i* is the index number of the state variable)

Remarks:

1. This is a global entry that defines user state variables for material user subroutines. The temperature will always be passed to material use subroutine as the first state variable; its name should not be redefined in this entry.
2. If a state variable is not given a name, SVi will be used as its name. The number
3. *i* is the index number of the state variable.
4. For output, either state variable names given in UDSESV or default SVi names can be used in NLOUT entry in case control. The state variables names will be used as keywords for output selection.
5. The 1st state variable is always temperature. The remaining user defined state variables are defined and used only by user, MSC Nastran will not use them.

UMAT User Subroutine and SCAIMDSolverUmat Interface

The SCAIMDSolverUmat interface and two methods, usrUmat_32 and usrUmat_64, are defined for UMAT type material UDS. The number of state variables can be defined in the UDSESV entry. Arguments to pass internal data, such as phase number, convergence flag and integration point volume, are added in the methods.

Corresponding to interface methods, FORTRAN subroutine is provided to user to implement UMAT in FORTRAN.

UCOHES User Subroutine and SCAIMDSolverUcohesive Interface

The UCOHES type user subroutine is defined in the SCAIMDSolverUcohesive interface for cohesive materials. Like the UMAT subroutine, user state variables and internal data arguments are passed in this subroutine.

SCAIMDSolverRuntimeInfo interface and *notify* method

The interface SCAIMDSolverRuntimeInfo and its method notify is defined. The notify method will be called at some specific analyzing points, including the beginning of a load case, the beginning of an increment, the end of increment and the end of the load case. The current subcase number, step number, increment number, current time, incremental time and a `RESTART` flag are passed in the call. In addition, user can supply input data to the notify method. The user supplied data is defined in GENUDS entry.

GENUDS entry

The GENUDS is to specify SCA service that implements the SCAIMDSolverRuntimeInfo interface. The user supplied input data for the *notify* method is also defined in this entry. When the *notify* method is called, the input data will be passed as arguments to this method.

Format:

GENUDS	SRV_ID								
	“INT”	IDATA1	IDATA2	IDATA3	IDATA4	IDATA5	IDATA6	IDATA7	
		IDATA8	IDATA9	…	IDATAn				
	“REAL”	RDATA1	RDATA2	RDATA3	RDATA4	RDATA5	RDATA6	RDATA7	
		RDATA8	RDATA9	…	RDATAn				
	“CHAR”	CDATA1	CDATA2	…	CDATAn				

Example:

GENUDS	MY_SRV								
	INT	1	2	100					

Field	Contents
SRV_ID	The service identifier used in the CONNECT SERVICE statement. (Character, no default)
“INT”	Keyword indicating that the following data is integer. (Character)
IDATAi	User supplied integer data. (Integer, no default)
“REAL”	Keyword indicating the following data is real. (Character)
RDATAi	User supplied real data. (Real, no default)
“CHAR”	Keyword indicating the following data is character. (Character)
CDATAi	User supplied character data. (Character, no default)

Remarks:

1. The SER_ID is the service identifier of SCA service in the CONNECT SERVICE statement. The SCA service should have implemented the RuntimeInfo interface.
2. A CDATAi entry cannot be the character “INT”, “REAL” or “CHAR”.

Utility Functions to Access MSC Nastran Data in UDS

Three utility functions GET_ELEM_PARAM, GET_NODE_PARAM, and GET_GLOBAL_PARAM are provided in this release. In material UDS, these utility functions can be called to get model data. A keyword and related input arguments are used to indicate what kind of data to retrieve, the utility functions will returned data in output arguments. Both FORTRAN and C++ callable functions are provided.

General Parameters

The functions and keywords are used to get model, machine and analysis procedure information, the available keywords in this category are:

- SUBCASE_NUMBER
- STEP_NUMBER
- INCREMENT_NUMBER
- SUB_INCREMENT_NUMBER (if applicable)
- ITERATION_NUMBER
- CURRENT_TIME
- INCREMENTAL_TIME
- TIME_OF_PREVIOUS_STEP
- TIME_OF_PREVIOUS_INCREMENT
- FRACTIN_OF_STEP_COMPLETED
- LARGE_DISP_FLAG
- JOB_NAME
- JOB_DIRECTORY
- WORKING_DIRECTORY
- SCRATCH_DIRECTORY
- NUM_PROCS
- NUM_CPUS

Element Parameters

The functions and keywords are used to get element related data, the available keywords in this category are:

- ELEMENT_TYPE

- DIRECT_STRESS_QUANTITIES
- SHEAR_STRESS_QUANTITIES
- NODES_OF_THE_ELEMENT
- INTEGRATION_POINTS_OF_THE_ELEMENT
- MATERIAL_ID_FOR_THE_ELEMENT
- ELEMENT_CLASS
- MAJOR_ENGINEERING_STRAIN
- MINOR_ENGINEERING_STRAIN
- CURRENT_VOLUME
- ORIGINAL_VOLUME
- TOTAL_TEMPERATURE
- INCREMENTAL_TEMPERATURE
- EQUIVALENT_VON_MISES_STRESS
- EQUIVALENT_STRESS_YIELD_STRESS_RATIO
- EQUIVALENT_ELASTIC_STRAIN
- EQUIVALENT_CREEP_STRAIN
- TOTAL_STRAIN_ENERGY_DENSITY
- ELASTIC_STRAIN_ENERGY_DENSITY
- PLASTIC_STRAIN_ENERGY_DENSITY
- GASKET_PRESSURE
- GASKET_CLOSURE
- PLASTIC_GASKET_CLOSURE
- FAILURE_INDEX
- TOTAL_VALUE_OF_FIRST_STATE_VARIABLE
- TOTAL_VALUE_OF_SECOND_STATE_VARIABLE
- TOTAL_VALUE_OF_THRID_STATE_VARIABLE
- VOLUME_FRACTION_OF_MARTENSITE
- EQUIVALENT_PHASE_TRANSFORMATION_STRAIN
- EQUIVALENT_TWIN_STRAIN
- EQUIVALENT_TRIP_STRAIN
- COMPONENTS_OF_CAUCHY_STRESS
- COMPONENTS_OF_TOTAL_STRAIN
- COMPONENTS_OF_ELASTIC_STRAIN
- COMPONENTS_OF_PLASTIC_STRAIN

- COMPONENTS_OF_CREEP_STRAIN
- COMPONENTS_OF_THERMAL_STRAIN
- COMPONENTS_OF_STRESS_PREFERRED_SYSTEM
- PHASE_TRANSFORMATION_STRAIN_TENSOR
- INTERLAMINAR_SHEAR_THICK_ELEMENTS_TXZ
- INTERLAMINAR_SHEAR_THICK_ELEMENTS_TYZ
- INTERLAMINAR_NORMAL_STRESS
- INTERLAMINAR_SHEAR_STRESS

Nodal Parameters

The functions and keywords are used to get nodal data, the available keywords in this category are:

- DISPLACEMENT
- ROTATION
- VELOCITY
- ROTATIONAL_VELOCITY
- ACCELERATION
- ROTATIONAL_ACCELERATION
- COORDINATE

UMAT and UCOHES Type User Subroutines in MATUDS Entry

The UMAT and UCOHES type are added in MATUDS entry for material user subroutines.

MTYPE	UNAME	SOL400
MATHE	uelastomer	X
MATUSR	hypela2	X
MAT1	Crplaw	X
MATF	Ufail	X
MATF	uprogfail	X
MATORT	Orient	X
MATUSR	umat	X
MCOHE	ucohes	X

Output of State Variables

The user-defined state variables can be output in MSC Nastran F06, OP2 and DBALL files. The NLOUT is used for state variables output selection. In this release, up to 100 state variables are allowed to be selected for output. The DRA support to access state variable output is available for modelers post-processing.

Test Case

To use UMAT and UCOHES material subroutines, user should implement the material SCA interface. For users convenience, template files have been provided and distributed with MSC Nastran installation. These template files can be found at:

```
[$InstallationDir]/[MSC Version]/nast/services/Implementations/Materials/src/
```

Users can write user code in the `ext_umat.F` or `ext_ucohes.F` subroutines under the *umant* and *ucohesive* directories for special material behavior. The material user services can be built using MSC SDK tools. The SDK is a separate installer that provides SCA build tools and environment. For information about SCA services and build, please see MSC Nastran SCA and User Defined Services documents.

For demonstration, model and template files using UMAT are provided and can be found at:

```
[$InstallationDir]/[MSC Version]/nast/services/Implementations/Materials/src/umat/
```

Enhancement to Enforced Relative Motion in NLTRAN for SOL 400

Introduction

In this release, the enhanced features of enforced relative motion include:

1. Apply enhanced relative displacement with SPCR in dynamic analysis.
2. Apply enhanced relative velocity with SPCR in dynamic analysis.
3. Apply enhanced relative acceleration with SPCR in dynamic analysis.
4. Using the results of displacement, velocity, and acceleration in the previous load case/step as the initial condition for the variables in SPCR.
5. Allow SPCR for nonlinear static analysis, nonlinear dynamic analysis, as well nonlinear static and dynamic chain analysis.

Benefits

This release enhances the capabilities of enforced relative motion in dynamic analysis of SOL 400. The capability of applying enforced relative motion is important and helpful for users to apply SOL 400 in nonlinear dynamic analysis. For example, when a GRID has deformation due to an applied load or motion associated with it in the previous STEP and the user wishes to pick up the resulting displacement as an enforced displacement, or applying specific relative displacement in the current STEP, then capability of applying relative motion will provide an efficient way. However, implementation of the enforced relative motion including displacement, velocity, and acceleration in nonlinear dynamic analysis is much more complicated than one used in static analysis due to the much more complicated loading conditions in dynamic analysis

Feature Description

With enhancement of this capability in dynamic analysis, enforced relative motion may be used in both static analysis and dynamic analysis, as well the static and dynamic chain analysis. User may apply the relative motion control at any load step in any type of structure analysis. The limitation for SPCR may be removed now.

1. For static analysis, SPCR may be used in the any steps including the first load step, which is kept unchanged as before;
2. For dynamic analysis, SPCR may be used in the any steps including the first load step. The type of SPCR, i.e., displacement, velocity, or acceleration, is determined by parameter of `TYPE` in TLOADi card as follows:

TLOAD1	SID	EXCITEID	DELAYI/DELAYR	TYPE	TID/F	US0	VS0		

3. It should be noted that the SPCR is applying the relative motion based on the deformed configuration of previous load step, therefore, the initial condition of SPCR is the status of the previous load step. The input values in TLOADi card `US0` and `VS0` will be ignored even they are given in the TLOADi cards.

4. The enforced relative motion, SPCR, may be time function, by TABLED. As mentioned above, the initial condition of SPCR is based on the previous load step, therefore, the initial value of TABLED should be zero to avoid the incompatibility of the deformation.

Limitation and Potential Enhancement

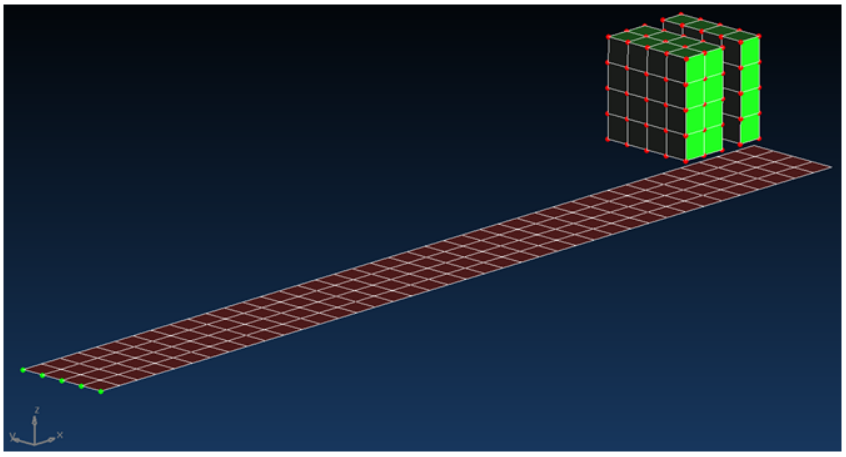
In this release, both SPCD and SPCR are supported in static and dynamic analysis for SOL 400. With SPCR, the enforced relative motion is applied based on the deformed configuration of the previous load step. With SPCD, it always determines the final position. In the current SOL 400, its initial condition for velocity and acceleration are given in TLOADi cards. Their default values are zero. For chain analysis, users should determine the initial values for them and should keep the compatibility with the results of previous load step. The potential enhancement is that for chain analysis, the initial value of enforced motion is always taken as the results of last load step for displacement, velocity, and acceleration.

Test Case

Test cases may be found in `tpl/spcdr2012`. Here `spcdrd010.dat` is taken as example as shown in Figure 4-7. This is static and dynamic chain analysis. The first step is static analysis, and second step is dynamic analysis. This FE model consists of 3-D solid element and four-node shell element. The thickness of the shell is 0.1 mm. Large displacement is considered in the analysis. Three contact bodies are deformable, which are plate, left block, and right block, respectively. Two block contact bodies may contact with plate contact body.

In the first step, the left end of the plate is constrained fully. X- and Y- direction displacements of two blocks are constrained, and the Z-direction displacement is given -0.6 and -0.25 to the left block and right block, respectively. Figure 4-8 shows the deformed configuration.

In the second step, ANALYSIS=NLTRAN is used. The displacement is applied during 0.015 seconds. Based on the deformed configuration after the first load step, the enforced relative motions are applied to two blocks again with SPCR. The relative motions applied to left and right blocks are -0.6 and -0.25, respectively. Therefore, the total displacement of the final position of the blocks are -1.2 and -0.5, respectively. Figure 4-9 shows the final deformation of the configuration.



(A)

(B)

Figure 4-7 Schematic of Finite Element Model

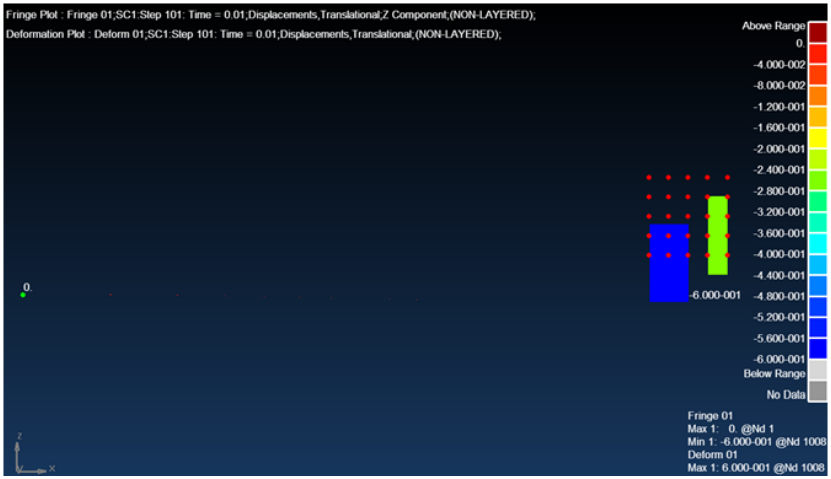


Figure 4-8 Deformed Configuration after the First Load Step

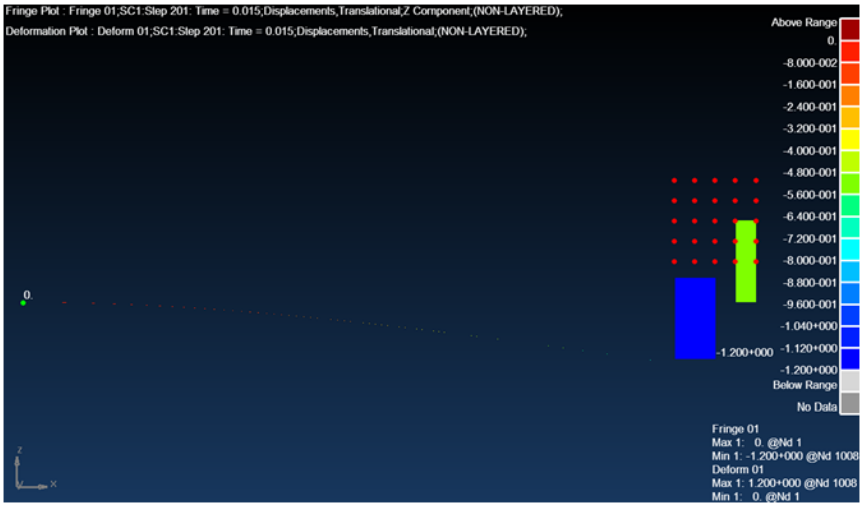


Figure 4-9 Deformed Configuration after the Second Load Step

Support for Export of Adams MNF file in SOL 400

Introduction

Support for the export of Adams MNF file is now available in SOL 400. An MNF file can be exported in SOL 400 (nonlinear) static analysis at a deformed or undeformed configuration. MNF files are required by Adams to represent deformation in Adams flexible bodies (flexbodies) by a set of Craig-Bampton (CB) modes. The capability of exporting the MNF file at deformed configuration would be useful for modelling components whose operating configuration are significantly different from undeformed configuration such that their eigenmodes/eigenfrequencies are considerably different from those calculated in undeformed configuration. However, it should be noted that the vibrations around this deformed operating configuration are still assumed to be small in the dynamic simulation conducted in Adams.

Benefits

This functionality will be of benefit for applications in Aerospace and Auto industries where some components of multibody systems are significantly deformed during operation and/or they are internally loaded during operation such that the eigenmodes/eigenfrequency of these components are significantly different from the unloaded state. MNF export in SOL400 will also benefit from SOL400's advanced modelling capabilities, e.g., advanced elements capable of representing material nonlinearity and large deformation, nonlinear offsets and contact. SOL400 also provides the convenience of exporting a nonlinearly preloaded MNF in one analysis as opposed to a two step procedure which has to be employed in SOL103 to export a nonlinearly preloaded MNF, i.e., SOL106 preloading followed by a SOL103 restart to export the MNF.

Theory

A more comprehensive description of the basic theory and methods used in the design is available in *Adams/FLEX User Guide* and *MSC Nastran Quick Reference Manual* Section 13.12. MNF files are required by Adams to represent deformation in flexbodies by a set of Craig-Bampton (CB) modes.

The idea behind using Craig-Bampton modal coordinates to represent the deformation of a flexible body is the same as using a truncated set of modal coordinates to reduce the size of the flexible bodies degrees of freedom for the sake of computational efficiency. Where, CB modes differ from physical eigenmodes of the system is that unlike a set of typical truncated eigenmodes the CB modes exactly capture the motion of the attachment point because they are explicitly constructed to contain them. This is very important as they capture the boundary interactions (for example joint constraint between two bodies) of the flexible body better if the attachment points are correctly specified.

It should be noted that currently, we are interested in extracting the CB modes in the reference deformed configuration of the body around which the small vibrations are to be modeled. The main difference in such a situation (from the linear case) is that here, instead of the usual linear stiffness matrix, we consider the tangent stiffness matrix (which is composed of the linear stiffness, material nonlinear part of stiffness, differential stiffness, and the follower force part of the stiffness) for generating the CB modes.

Input

A new Bulk Data section labeled BEGIN FLXBDY = id must be included with the SOL 400 run to export a MNF file (see ADAMSMNF* (Case), 221 case control statement Remark 21 in the *MSC Nastran Quick Reference Guide*). This new bulk section must contain the attachment a-set for identifying attachment points and must also contain the q-set for specifying the desired number of modal amplitudes for orthonormalization as shown below:

```
$ FLEXBODY Bulk section
BEGIN BULK FLXBDY = 10
$ Attachment point and component mode (A-SET) selection
  ASET1,123456,1,11,111,121
  QSET1,0,100001,THRU,100020
```

Typical Input

Typically SOL 400 is used to produce a preload for an Adams flexbody MNF run. In the preload run, the structure should be statically supported and follower loading must be applied as a self-equilibrating load set (not with SPC relationships!). In the ANALYSIS=MODES step the structure must be a free-free structure as the resulting orthonormalization requires that six rigid body modes be present. In order to produce modal amplitudes and mode shapes and to ensure residual vector calculations, SPOINTs and Q-sets are required. The SPOINTs must be included in the MAIN Bulk Data as they are included in the overall matrix size.

A new bulk data section labeled BEGIN FLXBDY =id must be included with the run. This new bulk section must contain the Q-set associate with the SPOINTs (in main bulk!) for modal amplitudes and the A-set required for attachment point designation. The example below is a typical SOL400 problem setup:

```
SOL 400
CEND
$ Case Control Section
$ Output AdamsMNF REQUIRED ABOVE SUBCASE
AdamsMNF flexbody=yes, psetid=all, outgstrs=yes, outgstrn=yes
SUBCASE 1
  $ Preload
  STEP 10
    $ Static load and support for preload
    SUBTITLE = PRELOAD
    ANALYSIS = NLSTATICS
    NLSTEP = 110
    LOAD = 120
    SPC  = 130
    BCONTACT = 140
    SPCF = ALL
    $ Generate stress and strain grid shapes
    STRESS(PLOT) = ALL
    STRAIN(PLOT) = ALL
    GPSTRESS(PLOT) = ALL
    GPSTRAIN(PLOT) = ALL
  $ Modal Step for Producing MNF
  $ Default: Select the end of previous load step to output AdamsMNF
  STEP 20
    ANALYSIS = MODES
    $ Select real Eigen Value Parameters
```

```
  METHOD = 210
  $ Turn residual vectors on
  RESVEC = COMPONENT
  STRESS(PLOT) = ALL
  STRAIN(PLOT) = ALL
  GPSTRESS(PLOT) = ALL
  GPSTRAIN(PLOT) = ALL

$ FLEXBODY Bulk section
BEGIN BULK FLXBDY = 10
$ Attachment point and component mode (A-SET) selection
  ASET1,123456,1,11,111,121
  QSET1,0,100001,THRU,100020
```

In the above example, the SPC set in the ANALYSIS=NLSTAT must be a static (non-redundant) constraint condition. Note that in the ANALYSIS=MODES STEP, the SPC constraint set has been removed. In SOL 400, the definition of the attachment a-set for identifying attachment points and for q-set for specifying the desired number of modal amplitudes for orthonormalization is done in a separate new FLXBDY Bulk Data Section shown above.

Guidelines

1. In SOL400, the ASET/ASET1 and QSET/QSET1, must only, appear in the FLXBDY bulk data section.
2. If contact is required as part of the preloading for the FLEXBODY=YES run it is highly recommended that the friction option be turned on by using an appropriate BCPARA bulk data entry setting, e.g.,

   ```
   $ Select bilinear Coulomb friction for all subcases
   BCPARA, 0, FTYPE, 6
   ```

   If contact friction is not turned on, the tangential motion between the two parts coming into contact will most likely not be constrained and incorrect or fatal results will occur.
3. If RIGID elements (RBE1/RBE2/RBE3/RBAR/RROD/RJOINT) are in the model, then the Case Control RIGID= LAGRANGE (default for SOL400) should be used to avoid possible wrong results. If an attachment point happens to touch a rigid element, the point should be associated with the independent degree of freedom of the rigid element. Though not recommended, if for some modeling requirement, a dependent rigid element grid is required to be in the attachment set, the user must include at least one independent/reference grid for that specific rigid element in the ASET.

Limitations

1. Current release does not support stress strain output in MNF for advanced nonlinear elements.
2. Only one FLXBDY Bulk Data section (with a positive Flexbody ID) is supported in SOL 400.
3. Currently, for a SOL400 analysis in which an ADAMS MNF is requested, only one mode step would be supported (this of course would be the step in which ADAMS MNF would be exported); i.e., multiple mode steps in one/multiple SUBCASE/SUBCASES would not be supported.
4. Transient Dynamics not supported.
5. AUTOQSET option is not supported.

Example Problem

The plate example problem provided here for SOL 400 has been previously discussed in *MSC Nastran Quick Reference Manual* Sec.13.12 for SOL 103 restarted from SOL 106.

The square plate is divided into a 10 x 10 mesh (see Figure 4-10). The four corner points, grid point 1 (.0,.0,.0), grid point 11 (0.,1.,0.), grid point 111 (1.,0.,0.), and grid point 121 (1.,1.,0.) are considered the attachment points. The geometric and material definitions are provided by the file included in the end.

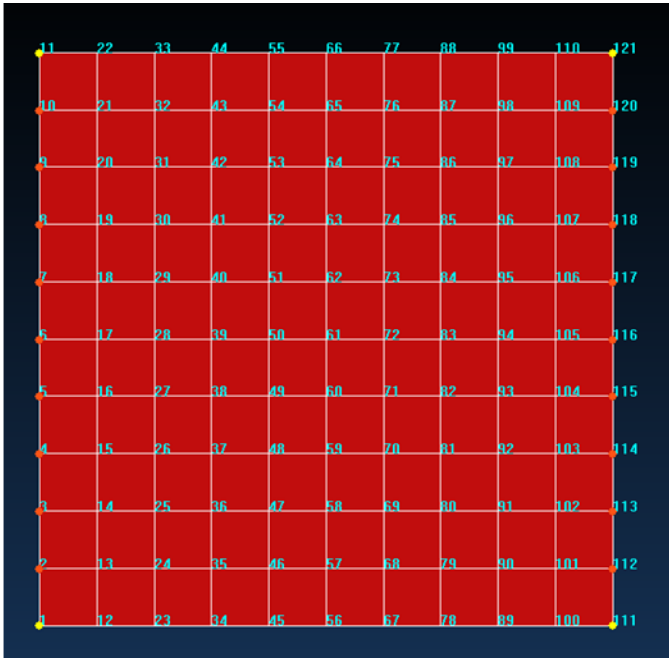


Figure 4-10 Plate Model

The plate is modeled with geometric nonlinearities. Step 1 is nonlinear static step that preloads the plate under simple tension and Step 2 is modal step in which MNF is exported. For validation, the MNF from SOL400 is compared with the MNF from SOL103 restarted from SOL 106. These MNF's are imported in ADAMS as flex/bodies and the eigenvalues/modes of plates fixed at all four ends calculated. The results (lower 7 modes) are shown below:

SOL 103 Restarted from SOL 106	SOL 400
29.2195	29.2195
47.848	47.8296
62.8388	62.8474
86.5278	86.5278
124.855	124.855

SOL 103 Restarted from SOL 106	SOL 400
141.29	141.29
150.832	150.832

The results are in agreement with each other.

Plate Input

```
SOL 400
CEND
TITLE= SIMPLE PLATE MODEL 10 X 10 ELEMENTS SOL 400 NL PRELOAD
ECHO= NONE
$
$
$ Initiate an MSC.Nastran/ADAMS interface run
$  FLEXBODY=YES is REQUIRED
$  ADMOUT=YES also output op2 file
$  OUTGSTRS=YES output element stress shapes
$  OUTGSTRN=YES output element stress shapes
$
ADAMSMNF FLEXBODY=YES, ADMOUT=YES, OUTGSTRS=YES,
         OUTGSTRN=YES, FLEXONLY=NO
$
SUBCASE 100
 STEP 100
  ANALYSIS=NLSTATICS
  NLPARM = 1
  $ Generate constraint forces
  SPCF = ALL
  SPC = 100 $
  LOAD=100
  STRESS(PLOT) = ALL
  STRAIN(PLOT) = ALL
  GPSTRESS(PLOT) = ALL $
  GPSTRAIN(PLOT) = ALL $
  NLSTRESS(PLOT) = NONE $
$
 STEP 200
   ANALYSIS = modes
   $
   $ Select real eigen value parameters
   $
   METHOD=300
   $
   $ Turn on residual vectors
   $
   RESVEC = COMPONENT
   DISP(PLOT)=ALL
   STRESS(PLOT) = ALL
   STRAIN(PLOT) = ALL
   GPSTRESS(PLOT) = ALL $
   GPSTRAIN(PLOT) = ALL $
$
$ Define surface for stress and strain grid shapes
$
OUTPUT(POST)
SET 9998 = ALL
SURFACE 9998 SET 9998 FIBRE Z2 NORMAL X3
$
BEGIN BULK
$
$ Turn on large displacements
```

```
$
PARAM,LGDISP,1
$
$ Nonlinear parameters
$
NLPARM,1,4,,,,,UPW,YES
$
$ If wanted, turn on gridpoint weight generator
PARAM,GRDPNT,0
$
$ Default value - ADAMS must use the DTI,UNITS
$
PARAM,WTMASS,1.0

$ Select number of modes:
$
$ ================================================================
EIGR    300     LAN                                10
$ ================================================================
$
$ SCALAR Point to define DOFs to use for component modes
SPOINT,80001,THRU,80019
$
$ ADAMS REQUIRES following DTI
$
DTI,UNITS,1,KG,N,M,SEC
$
$ Add in plate tensioning follower load
$
$ Preload left side
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
FORCE1  100     111     30000.    1       111
FORCE1  100     112     60000.    2       112
FORCE1  100     113     60000.    3       113
FORCE1  100     114     60000.    4       114
FORCE1  100     115     60000.    5       115
FORCE1  100     116     60000.    6       116
FORCE1  100     117     60000.    7       117
FORCE1  100     118     60000.    8       118
FORCE1  100     119     60000.    9       119
FORCE1  100     120     60000.    10      120
FORCE1  100     121     30000.    11      121
$ Preload right side
FORCE1  100     1       30000.    111     1
FORCE1  100     2       60000.    112     2
FORCE1  100     3       60000.    113     3
FORCE1  100     4       60000.    114     4
FORCE1  100     5       60000.    115     5
FORCE1  100     6       60000.    116     6
FORCE1  100     7       60000.    117     7
FORCE1  100     8       60000.    118     8
FORCE1  100     9       60000.    119     9
FORCE1  100     10      60000.    120     10
FORCE1  100     11      30000.    121     11
$
$
$ static support set for preload
$
SPC1    100     123     1
SPC1    100     13      11
SPC1    100     3       111
$
$ Get model data
$
include 'plate_mesh.bdf'
$
$ New Bulk Flxbdy bulk data section
$
```

```
BEGIN BULK FLXBDY = 77
$
$ The corner grids 1, 11, 111, 121 are the exterior
$ or attachment point grids
ASET1,123456,1,11,111,121
$ QSET1 to define DOFs to use for component modes
QSET1,0,80001,THRU,80018
ENDDATA
```

Helicopter Rotor Blade System

The Adams model of the helicopter rotor blade system is shown in Figure 4-11. The three blades are attached to the rotor head through fixed joints as shown. Two Adams models are generated one in which the rotor blades are modelled as SOL400 flexbodies and one in which they are modelled as SOL103 flexbodies.

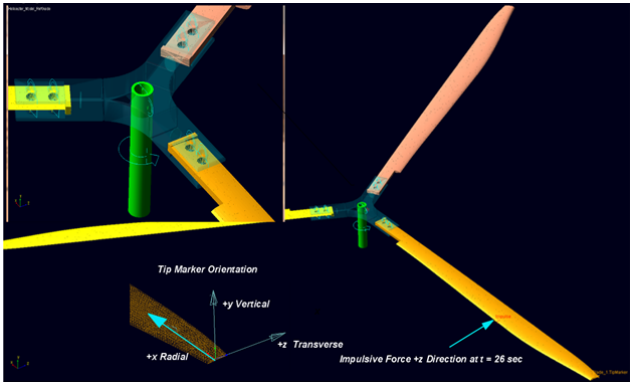


Figure 4-11 Helicopter Rotor Blade System

The SOL400 analysis to generate the MNF is done in two steps. Step 1, the blade is loaded under (1) self-weight, (2) axial tensile loading of 4.224e+5 N which represents centrifugal loading at 2000°/second (333.33 rpm). Step 2, is the modal step in which the MNF is exported. The MNF generated, therefore models the stiffening effects at the operating speed of 2000°/second. Due to preloading the blade is stiffened; eg., calculating the cantilevered modes of the blade reveals the first (bending) mode of the beam in the vertical and horizontal direction is 2.658 Hz and 8.454 Hz for the unloaded blade and 3.108 Hz and 8.769 Hz for the loaded blade. Rayleigh damping (Parameters 1 = 2508 and 2 = -1.276e-5) is used to model the damping behavior of the blades.

The rotor blade system is now simulated in Adams. A motion driver is applied to the rotor shaft to linearly ramp up the rotational speed from 0 to 2000°/second at t = 25 seconds. After 25 seconds, the shaft rotational speed is held constant. At t = 26 seconds, an impulsive load (see Figure 4-11) of 1.0e+5 N in z-direction (transverse) is applied for t = 0.01 sec.

The results of the simulation are summarized through three Figures. Figure 4-12, shows the vertical displacement of the blade tip during the simulation. The translational deformation in transverse direction at the blade tip after the impulsive load is applied is shown in Figure 4-13 for both SOL400 (includes preload) and SOL103 (no preload). One can easily identify the transients associated with the impact in this figure. It should be noted that the preload applied in SOL400 causes a 5mm of initial deformation in transverse direction. The stiffening of the blade due to preload in SOL400 is also apparent in Figure 4-12 due to the reduction in time period of transverse oscillations. Figure 4-14 shows the von Mises stress distribution in the rotor shortly after the impact (t = 26.029 sec).

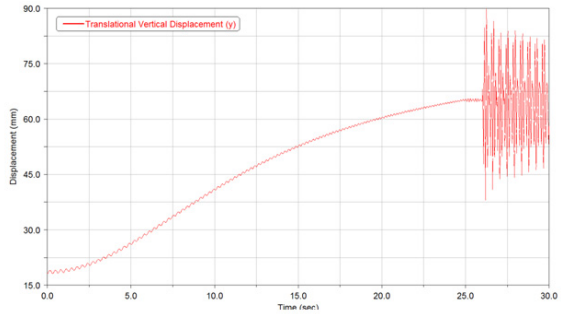


Figure 4-12 Vertical Displacement at Blade Tip

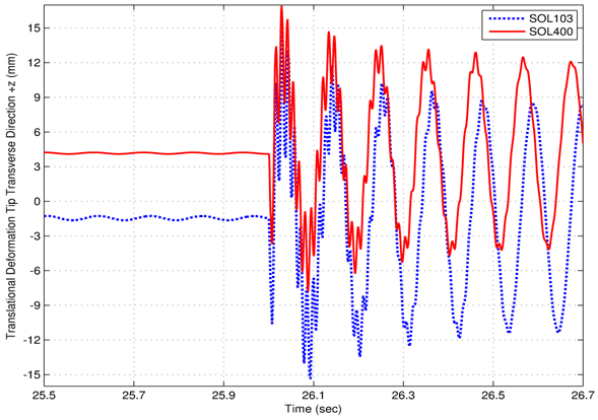


Figure 4-13 Translational Deformation at Blade Tip

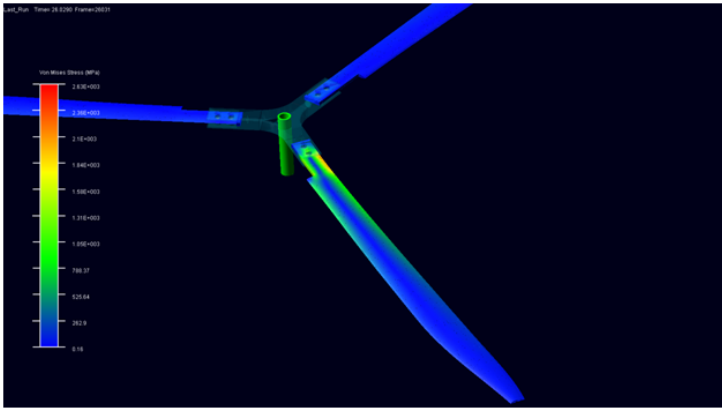


Figure 4-14 Von Mises Stress at t = 26.029 sec (SOL400 MNF with damping)

Helicopter Rotor Blade System Input Deck

```
SOL 400
CEND
$
$
$
$ Initiate an MSC.Nastran/Adams interface run
$  FLEXBODY=YES is REQUIRED
$  ADMOUT=YES also output op2 file
$  OUTGSTRS=YES output element stress shapes
$  OUTGSTRN=YES output element stress shapes
$
ADAMSMNF FLEXBODY=YES, psetid=all, OUTGSTRS=YES, OUTGSTRN=YES
SUBCASE 1
STEP 1
   ANALYSIS = NLSTAT
   NLSTEP = 2
   SPC = 3
   LOAD = 100
   $ Generate constraint forces
   SPCF(PLOT) = ALL
   MPCF(PLOT) = ALL
   DISPLACEMENT(PLOT)=ALL
   NLSTRESS(PLOT) = ALL
   STRESS(PLOT)=ALL
   STRAIN(PLOT) = ALL
   GPSTRESS(PLOT) = ALL
   GPSTRAIN(PLOT) = ALL
STEP 2
   ANALYSIS = MODES
   $ Select real eigen value parameters
   METHOD = 4
   $ Turn on residual vectors
   RESVEC = COMPONENT
   DISP(PLOT)=ALL
   STRESS(PLOT) = ALL
   STRAIN(PLOT) = ALL
   SET 1 = 1
   GPSTRESS(PRINT) = 1
   SET 2 = 2
   GPSTRAIN(PRINT) = 2
OUTPUT(POST)
   SET 3 = ALL
   SURFACE 1 SET 3 FIBRE ALL AXIS X1 NORMAL R TOPOLOGICAL
    BRANCH BREAK
   VOLUME 1 SET 3
   SET 4 = ALL
   SURFACE 2 SET 4 FIBRE ALL AXIS X1 NORMAL R TOPOLOGICAL
   BRANCH BREAK
   VOLUME 2 SET 4
BEGIN BULK
$ Rayleigh Damping
PARAM,ALPHA1,.2508,0.0
PARAM,ALPHA2,-1.276-5,0.0
$ Other Parameters
PARAM,LGDISP,1
PARAM,POST,1
PARAM,SNORM,20.
PARAM,K6ROT,100.
PARAM,WTMASS,1.0E-03
$ Adams REQUIRES following DTI
DTI,UNITS,1,KG,N,MM,SEC
$ Nonlinear parameters
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
NLSTEP  2       1.0
        GENERAL                 5
```

```
        FIXED   100
        MECH    UPW     0.01    0.01    0.01    PFNT
$ Select number of modes:
EIGRL          4                15                              MASS
$ Get model data
include 'blade_mesh.bdf'
$ RBEs for the Model
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
RBE2     42496   80060   123456  7979    7980    7981    7982    7983
         7984    7985    7986    7987    8061    8289    8290    8291
         8292    8293    8294    8295    8296    8297    8298    8299
         8354    8355    8356    8357    8358
RBE2     42497   80061   123456  7988    7989    7990    7991    7992
         7993    7994    7995    7996    7997    8300    8301    8302
         8303    8304    8305    8306    8307    8308    8309    8348
         8349    8350    8351    8352
RBE3     42498           8500    123456 1.       123     72      74
         8502    205     87      8501    207     209     103     102
         211     213     101     215     244     217     221
RBE3     42499           8453    123456 1.       123     8458    8512
         8456    8455    8454    8452    8451    8450    8449    8438
         8435    8448    8447    8446    8445    8444    8443    8442
         8441    8439    8440    8409    8407    8406    8405    8404
         8403    8402    8401    8400    8399    8398    8432    8468
         8467    8466    8465    8464    8463    8462    8461    8459
         8460    8431    8429    8428    8427    8426    8425    8424
         8423    8422    8420    8421
$ Loading
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
GRAV           2           9810.     0.0     -1.     0.0
FORCE1  10      8500    4.224+5 8453    8500
FORCE1  10      8453    -4.224+58453    8500
LOAD    100     1.0     1.0     2       1.0     10
$ Static support set for preload
SPC1           3  123456   80060   80061
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
$ SCALAR Point to define DOFs to use for component modes
SPOINT,80001,THRU,80059
$ New Bulk Flxbdy bulk data section
BEGIN BULK FLXBDY = 77
$-------2-------3-------4-------5-------6-------7-------8-------9-------0-------
ASET1   123456  80060   80061
QSET1   0       80001   THRU    80059
ENDDATA
```

5 Explicit Nonlinear (SOL 700)

The following new capabilities have been added to the Explicit Nonlinear Solution - SOL 700 in MSC Nastran 2013.1

New Capabilities in Explicit Nonlinear (SOL 700) (2013.1)

The following new capabilities are added in this release:

1. Support for User Defined Services (UDS)
2. New Material Models

In addition several software defects are corrected.

User Defined Services in SOL 700 (2013.1)

The MSC Nastran 2013.1 release now supports the use of user subroutines for both the fluid (Eularian) and structural (Lagrange) region through User Defined Services.

This capability allows the users to take customize functionalities such as user-defined material models, equation of states, flow boundary conditions, friction models, etc. in their simulation. The capability allows users to transition from the Dytran product to MSC Nastran.

The following Dytran services and user-defined subroutines are supported in this release:

Service name	User Subroutine Name (C++/Fortran)	Description
EulFlow	usrFlow/EXT_FLOW	Simple flow definition on the boundary of Euler elements. FLOWUDS with option UNAME=EXFLOW must be used.
	usrFlow3/EXT_FLOW3	Advanced flow definition on the boundary of Euler elements. FLOWUDS with option UNAME=EXFLOW3 must be used.
	usrPor/EXT_POR	Flow definition on a coupling surface. PORUDS must be used and be referenced from a LEAKAGE entry.
EulInOut	usrOut/EXT_OUT	User defined output request. NLOUTUD must be used to activate this user subroutine.
	usrInit/EXT_INIT	User defined initialization of Euler elements at the start of the simulation. TICEUDS must be used to activate this user subroutine.
EulMat	usrMatyld/EXT_MATYLD	User defined yield material model. YLDUDS must be used and referenced from MATDEUL to activate this user subroutine.
	usrMateos/EXT_MATEOS	User defined equation of state material model. EOSUDS must be used and referenced from MATDEUL to activate this user subroutine.
	usrMatfail/EXT_MATFAIL	Simplified user defined failure material model. FAILUDS with option USER=EXFAIL must be used and referenced from MATDEUL to activate this user subroutine.
	usrMatfail2/EXT_MATFAIL2	Advanced user defined failure material model. FAILUDS with option USER=EXFAIL2 must be used and referenced from MATDEUL to activate this user subroutine.
	usrMatshr/EXT_MATSHR	User defined shear material model. SHRUDS must be used and referenced from MATDEUL to activate this user subroutine.

The following LS-Dyna services and subroutines are also supported in MSC Nastran 2013.1.

Service Name	User Subroutine Name (C++/Fortran)	Description
LagMat	usrVumat_32/EXT_VUMAT_32 usrVumat_64/EXT_VUMAT_64	User-defined material for Lagrangian elements. MATUDS must be used to activate the user subroutine.
LagStruct	usrVfric_32/EXT_VFRIC_32 usrVfric_64/EXT_VFRIC_64	User-defined friction for contact interfaces. BCONUDS must be used to activate the user subroutine.
	usrCtrl1_32/EXT_CTRL1_32 usrCtrl1_64/EXT_CTRL1_64	Simulation control by the user. CNTRUDS must be used to activate the user subroutine.

Each subroutine has 32- and 64-bit types. The single-precision of dytran-lsdyna will use the 32-bit type of user subroutines and the double precision one will use the 64-bit type of user subroutines. The 64-bit is required if DBL=YES or DBL700=YES

For a detailed description of the user routines, please refer to the *MSC User Defined Services Manual*.

The user-defined subroutines are supported through Simulation Component Architecture (SCA) services in MSC Nastran. The definition and implementation of SCA services, the mechanism of loading services and calling subroutines are all based on SCA component and architecture. Under this scenario, there is no need for creating new executables to use a user routines. The user routines can be in Fortran or C++ language.

UDS requires the SDK module that is available for download from the Software Download Center (SDC) in MSC Software's website. User Defined Services is extended to support SOL 700 in MSC Nastran 2013.1. This is an important capability that enables the users to utilize the user-defined subroutines from LS-Dyna or Dytran in SOL 700.

Limitations

This implementation allows only the UDS to be in either MSC Nastran SOL 700 or other MSC Nastran solutions. This capability does not support a combined UDS for MSC Nastran SOL 700 and other MSC Nastran solutions at the same time.

No third party library is allowed in user subroutines, since the building procedure would not recognize the names and their locations.

Examples

Two examples are prepared to demonstrate how to use a user routines with SOL 700. The first example will use a Dytran user routine EulFlow to define flow boundary condition and the second example uses an LS-Dyna friction model. Both examples are included in the *MSC Nastran Demonstration Problems Manual*. Please refer to SOL700 User Defined EulFlow Subroutine (Ch. 84) and SOL700 User Defined Friction Subroutine (Ch. 85) in the *MSC Nastran Demonstration Problems Manual* for more details.

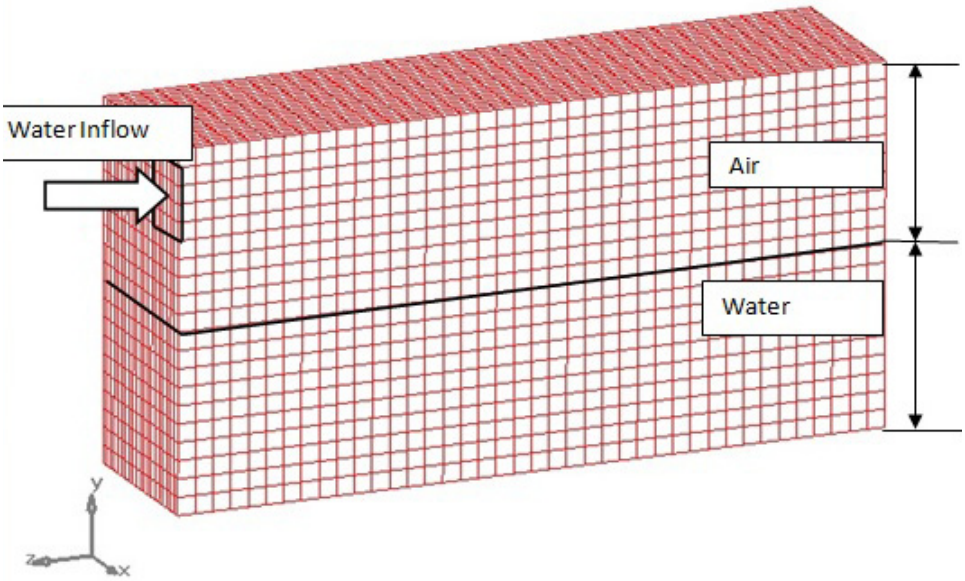
We will summarize the Dytran EulFlow examples here to show how the users routines are used in SOL 700.

SOL700 User Defined EulFlow Subroutine

This Demo Example demonstrates how to utilize the User Defined Services (UDS) to define a user flow boundary model in the fluid part of SOL700.

This example uses the EulFlow service. This service enables users to write their own Fortran or C++ routines to prescribe inflow and outflow. In addition, the fluid part of Sol700 provides the EulMat and EulInOut services. The EulMat service enables user defined materials and EulInOut provides user defined Euler mesh initialization and Euler mesh output.

The problem simulates the inflow of water into a container that is half filled with water and half filled with air. The inflow is prescribed by a user subroutine.



Input File

MSC Nastran input model

Define the connect service at the top.

```
CONNECT SERVICE exfl 'SCA.MDSolver.Obj.Uds.Dytran.EulFlow'
```

`exfl` is the group name and will be used in the input deck to assign the user defined service.

`'SCA.MDSolver.Obj.Uds.Dytran.EulFlow'` is the service name which is defined for sol700 user EulFlow service.

Sol700 is an executive control that activates an explicit nonlinear transient analysis:

```
SOL 700,129 STOP=1
CEND
```

Case control cards for default setting, problem time, loads and initial conditions

```
PARAM,DYDEFAUL,DYNA
$
DLOAD = 1
IC = 1
ENDTIME =    0.1
```

Bulk data section:

BEGIN BULK

Define Output results request for every 0.002 second

```
DYPARAM   LSDYNA BINARY D3PLOT    0.002
```

The tank is modeled by a half symmetrical 3-D Euler mesh. The number of Euler elements in the X-, Y-, and Z-direction is 20, 20 and 40.

```
MESH,1,BOX,,,,,,,+
+,0.,0.,-1.,.5,1.,2.,,,+
+,20,20,40,,,,EULER,1
```

The multi-material Euler solver will be used:

```
PEULER1,1,,MMHYDRO,19
```

Two materials will be defined. These are air and water:

```
MATDEUL 3       1.3     3
MATDEUL 4       1.e3    1
EOSGAM,3,1.517,226.45
EOSPOL  1       2.2e9
```

The Euler mesh is initialized with these materials as:

```
TICEUL1,19,19
TICREG,2,19,BOX,1,3,5,2.0
TICREG,1,19,SPHERE,2,4,6,1.0
BCBOX,1,COORD,,,,,,,+
+,,,-100,  0.5,-100, 100,  0.5,-100,+
+,,, 100,200.5,-100,-100,200.5,-100,+
+,,,-100,  0.5, 100, 100,  0.5, 100,+
+,,, 100,200.5, 100,-100,200.5, 100

SPHERE,2,,0.0,0.0,0.0,500.0
TICVAL,5,,SIE,400000.,DENSITY,1.3
TICVAL,6,,SIE,400000.,DENSITY,1000.0
```

Two flow boundaries are defined.

The first one prescribes inflow of water by employing a user subroutine:

```
TLOAD1,1,2,,4
FLOWUDS,2,exfl,,,,,,,+
+,0.4,0.6,0.7,0.9,1.0
```

This boundary condition prescribes inflow for all eulerian boundary faces that are inside the square (x = 0.4 to 0.6, y = 0.7 to 0.9, and z= 1.0)

The second flow boundary condition prescribes a transmitting flow condition

```
FLOW,1,,,POSZ,,,,,+
+,,,,,,,,,+
+,ZVEL,0
```

In addition a gravity loading will be prescribed

```
TLOAD1,  2,  444, , 0
GRAV,  444 ,  -0.98E+1 ,   0.00,    1.00,    0.00
DLOAD          1      1.       1.       1       1.       2
ENDDATA
```

User EulFlow Subroutines

SOL700 supports C++ and Fortran user subroutine for each capability.

The routines are located in `EulFlow/src/exflow` (fortran) and `EulFlow/src` (C++). Since the fortran routine is called by the C++ routine, both routines have the same arguments. Users can choose one of them.

Fortran Routine

The fortran routine name is `ext_flow.F`.

In this example a constant velocity of 20 m/s in the negative Z-direction is defined:

```
DO 100 NF = ISTART,IEND
IMATNO(NF) = 4
PFAC(NF)  = PZON(NF) + QZON(NF)
UXFAC(NF) = 0.0d0
UYFAC(NF) = 0.0d0
UZFAC(NF) = -20d0
RHOFAC(NF) = 1000d0
SIEFAC(NF) = 0.0d0
100  CONTINUE
```

C++ Routine

In EulFlow.cpp

```
  SCAInt64 nf;
      printf("hallo\n");
  for(nf=istart-1; nf < iend; nf++)
       {
         imatno[nf] = 4;
         pfac[nf]   = pzon[nf] + qzon[nf];
         uxfac[nf]  = 0.0e0;
         uyfac[nf]  = 0.0e0;
         uzfac[nf]  = -20.e0;
         rhofac[nf] = 1000.e0;
```

```
    siefac[nf] = 0.0e0;
  }
```

Compile User Subroutine

To compile user subroutines, the SDK module is required, which can be downloaded from the Software Download Center.

On windows platform, it is installed by default in `C:\MSC.Software\SDK`. Users can install the SDK module in any directory.

If the user-defined subroutine Eulflow and all other folders are located in `c:\test`, then the EulFlow folder must also reside under `c:\test`. All folders and files must have the same directory structures as Input File example and they must be located under `c:\test`.

1. Set the environment variable in order to set the location where you want to put the user subroutine library.

   Type in the command:

   ```
   set SCABOBJ=”location” For example, set SCABOBJ=c:\users\xxxx\obj
   set SCA_SERVICE_CATALOG=%SCABOBJ%\res\SCAServiceCatalog.xml
   set SCA_LIBRARY_PATH=%SCABOBJ%\WIN8664\lib
   ```

2. Go to `c:\test` folder where the user subroutine folders are located.
3. Compile the library using the commands listed below. EulFlow will be changed depending on the example. If SDK is installed in `C:\MSC.Software\SDK\20131`, you can compile using the following commands:

   ```
   C:\MSC.Software\SDK\2013\tools\scons EulFlow APPS_LOCAL=%SCABOBJ%
   SCA_OBJECT=%SCABOBJ%
   PPS2_SYSTEM=C:\MSC.Software\MSC_Nastran\20131\msc20131\dyna\win64\sdk
   ```

4. You can confirm if the compilation has been successful. If it is compiled correctly, it will give the following messages

   ```
   Creating library
   C:\Users\json\obj\WIN8664_EULFLOW_OPT\EulFlow\src\EulFlow.lib and object
   C:\Users\json\obj\WIN8664_EULFLOW_OPT\EulFlow\src\EulFlow.exp
   copy C:\Users\json\obj\WIN8664_EULFLOW_OPT\EulFlow\src\EulFlow.dll
   C:\Users\json\obj\WIN8664\lib\SCA\MDSolver\Obj\Uds\Dytran\EulFlow.dll
   Adding component from EulFlow\src\EulFlow.cdl to catalog
   c:\users\json\obj/res\SCAServiceCatalog.xml
   copy C:\Users\json\obj\WIN8664_ EULFLOW_OPT\EulFlow\src\EulFlow.lib
   C:\Users\json\obj\WIN8664\lib\SCA\MDSolver\Obj\Uds\Dytran\EulFlow.lib
   copy C:\Users\json\obj\WIN8664_ EULFLOW_OPT\EulFlow\src\EulFlow.exp
   C:\Users\json\obj\WIN8664\lib\SCA\MDSolver\Obj\Uds\Dytran\EulFlow.exp
   Processing SCA manifest
   C:\Users\json\obj\manifest\WIN8664\SCA\MDSolver\Obj\Uds\Dytran\EulFlow
   Building Visual Studio project file
   C:\Users\json\obj\WIN8664_EULFLOW_OPT\EulFlow\src\EulFlow.vcxproj
   scons: done building targets.
   ```

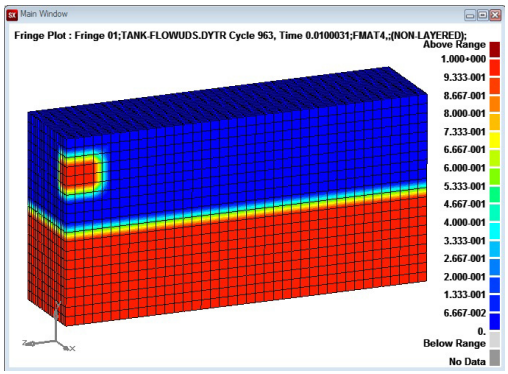
5. Go to input deck location and run a job without any option. If the input deck name is bndtnkf.dat, just type:

   ```
   C:\MSC.Software\MSC_Nastran\20131\bin\nast20131 bndtnkf.dat
   ```

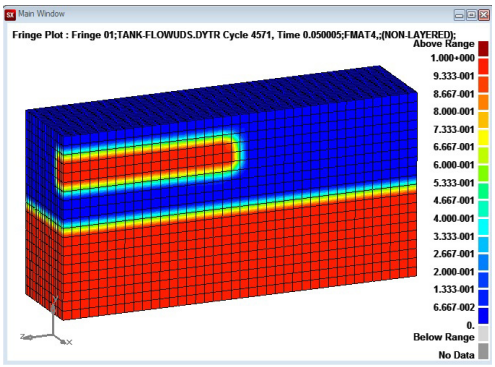
Results

FMAT4 (Water) Plots

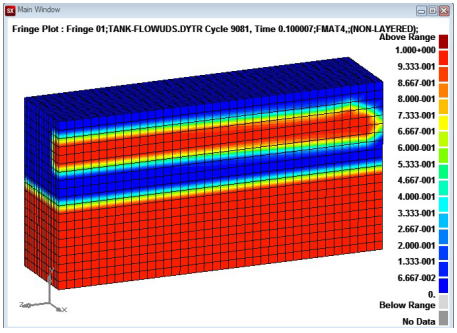
The following figures show the material fraction of water. The inflow of water is clearly visible.



Time = 0.01 s

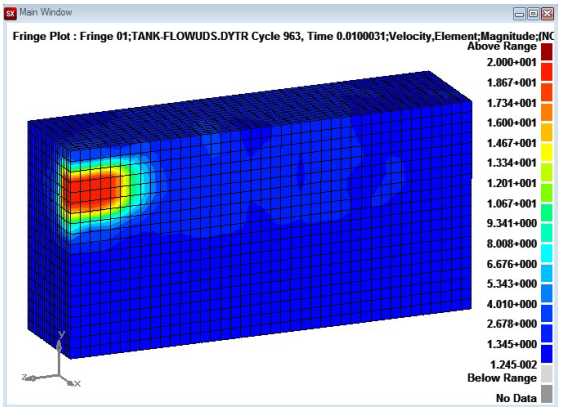


Time = 0.05 s

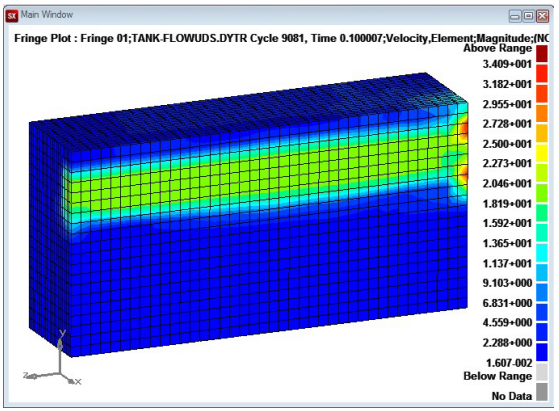


Time = 0.1 s

Water Velocity Plots



Time = 0.01 s



Time = 0.1 s

New Material Models

Three new material models have been added to SOL 700

MATD107

This material is the same as MATDO15 (Johnson-Cook) but is has more complicated stress-strain relation and camage criteria.

MATD120

The Gurson model is used to describe the plastic flow. This model is only available for shell and solid elements. This is similar to similar to the Gurson Model defined in MATEP for SOL 400.

MATD12J

This is the same as Gurson flow model but it includes an additional Johnson-Cook failure criterion. This model is only available for shell and solid elements.

MATD12R

This is the same as Gurson flow model but it includes additional the Wilkins Rc-Dc [Wilkins, et al., 1977] fracture model. This model is only available for shell and solid elements.

MATD224

This material is the same as MATD015 (Johnson-Cook) but is has tabulated variables for Johnson-Cook failure criterion.

1-D - 3-D Spherical-symmetric and 2-D - 3-D Axi-symmetric Mapping for Blast Loads

Two techniques are introduced in this release to compute the blast loads in 1-D and 2-D meshes followed by a re-map in a full 3-D mesh.

Blast wave simulations require fine mesh within and around the explosive to capture the details of the pressure wave propagation. As a result a large three dimensional fine Eulerian mesh is often constructed that result in an increase in simulation time. During most of this time, the pressure wave is just expanding in the medium without hitting the structure. This is particularly true for far-field explosions where the distance of the detonation point with respect to the structure is rather large.

An efficient method is introduced in this release to compute the blast wave pressure by using a spherical-symmetric 1-D or axi-symmetric 2-D mesh prior to impact to the structure in a 3-D model. This will require a two steps simulation process where the blast loads will be written into an archive file with a 2-D or 1-D results and then will be read as a “remap” file in a subsequent 3-D simulation with the structure.

How it works: 2-D - 3-D Axi-symmetry Mapping

To import a 2-D-axial symmetric run into a 3-D run, the 2-D axi-symmetric element variables have to be mapped onto 3-D elements. Each element in the 2-D axi-symmetric Euler archive defines a cylinder. For each cylinder a number of element variables like density and specific energy and velocity are specified. These cylinders are used to initialize the Euler domain in the 3-D mesh using an approach called micro-zoning. A cylindrical shape in general covers only a fraction of an Euler element. To compute this fraction the 3-D element is divided into a number of smaller elements. These smaller elements are called micro zones. Each micro zone is then examined to determine whether it is inside the cylinder. The micro zones approach will be used to do the 2-D-axial symmetric to 3-D remap. The number of micro zones is by default 1000, but can be modified by using PARAM, MICRO to increase accuracy. In most simulations using 1000 micro zones is sufficiently accurate.

Similar approach will be followed to remap 1-D spherical elements to 3-D elements using the micro-zoning methodology where the Euler archive of the 1-D spherical run defines spheres.

The 2-D mesh can be put in the 3-D mesh under an arbitrary angle. The direction of the axial axes as viewed in the 3-D mesh is specified by the direction vector given on the DYPARAM,AXREMAP entry. This direction will be called the 3-D axial axis. In intersecting the 2-D elements with 3-D elements only the distance to the 3-D axial axis and the height along this axial axis is of importance. This height will be called the axial height. To speed-up the computation, the 2-D axi-symmetric elements are sorted. This makes it easy to determine what 2-D elements are in the vicinity of a 3-D element.

The following steps are performed for remapping the blast loads in a 3-D model:

- Create a model with 2-D wedge.
- Run the model and see when the blast wave approaches the structure. Since 2-D axi-symmetric meshes do not have that many elements, Euler archives can be requested several times.
- Select a time at which the blast wave has come in the vicinity of the structure.
- Create a 3-D model and use the option EULINIT/EID to point to the archive of the 2-D model. Also enter the desired cycle number.

Two new parameters are created, one for 2-D axi-symmetric run and another to read the loads and remap in the 3-D model as follows:

1. For Axial Symmetric run DYPARAM AXIALSYM is required
2. The Archive result files from this run can be remapped in the follow-up run (c) by using DYPARAM, AXREMAP and `sol700.pth` file where the regarding Archive file is defined by means of `EID` option.

The following example demonstrates the application of this method. For details please refer to the Axial – Spherical Symmetry to 3D Euler Remap (Ch. 82) in the *MSC Nastran Demonstration Problems Manual*.

Blast Against a Structure using 2-D Axial Symmetric – 3-D Remap

For the detailed examples please refer to the 2D Axial Symmetric to 3D Remap (Ch. 82) in the *MSC Nastran Demonstration Manual*.

The 3-D model consists of rectangular box with dimensions of 12m x 6m x 6m, with its origin at [0,0,0] and occupying the positive quadrant (Figure 4-1). A cylindrical charge is assumed to be placed with its axis coinciding with the Z-axis. The box is meshed with a uniform Cartesian mesh, 120x60x60. The symmetry walls are rigid walls by default. The positive x, y and z directions, are defined as non-reflecting boundaries to prevent unwanted reflections.

Figure 4-1 3-D Model of Cylindrical Shape Charge Against a Structure

The axially symmetric run was calculated in the XZ-plane, for a 5m x 6m domain (see Figure 4-2). This domain was meshed by 200x240 elements, so that the charge radius included 14 elements. The 250kg charge is represented by an “energetic” air, to allow a single material calculation. The charge is a cylinder with L/D=2, elevated 1m above ground.

For Axial Symmetric Analysis the following parameter is used:

```
DYPARAM, AXIALSYM, AXIAL, Z, ZX, 2.5
```

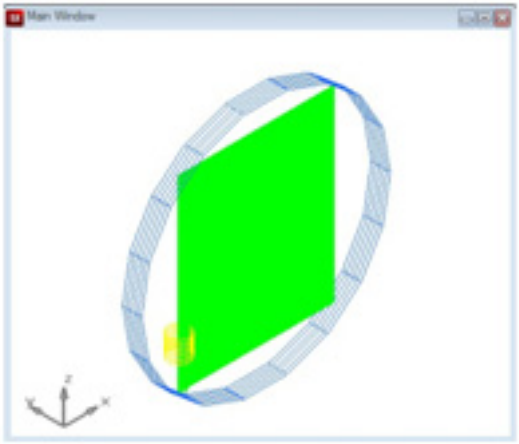


Figure 4-2 Model for 2-D - Axial Symmetry

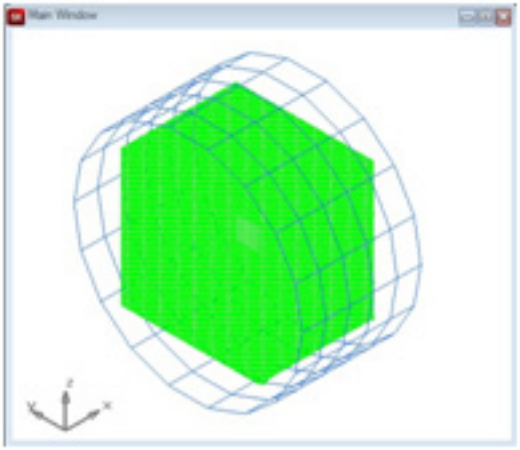


Figure 4-3 3-D Model for Remap

2-D - Axial Symmetry model

```
$! Executive Control Section
SOL 700,129 STOP=1
CEND
ECHO = SORT
$! Case Control Section
PARAM, DYDEFAUL, DYNA
$! Subcase name : DefaultLoadCase
   TSTEPNL = 1
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CORNER)=ALL
BEGIN BULK
$! Bulk Data Pre Section
PARAM   DYDTOUT     0.04
PARAM   POST           0
$! Bulk Data Model Section
PARAM    INISTEP  0.0001
PARAM    STEPFCT     0.6
PARAM      MICRO      20
DYPARAM AXIALSYM   AXIAL       Z      ZX     2.5
DYPARAM   LSDYNA  BINARY  D3PLOT     0.1
EOSGAM          1     1.4                                          EOSGAM_1
MATDEUL         1   1.225       1
PEULER1         1           HYDRO       1                          PEULER_1
TICEUL1         1       1
TICREG          1       1CYLINDER       1       1       1      1.
TICREG          2       1CYLINDER       2       1       2      2.
MESH            1     BOX
+             0.0   0.001     0.0      5.    0.02      6.
+             200       1     240                           EULER       1
CYLINDR         1             2.5    -0.3      3.     2.5     0.3      3.
               4.
CYLINDR         2             0.0     0.0      1.     0.0     0.0  1.6828
          0.3414
TICVAL          1         DENSITY   1.225     SIE  0.2068
TICVAL          2         DENSITY   1000.     SIE    3.52
$!  SX Names for Materials
$!-1---|---2---|---3---|---4---|---5---|---6---|---7---|---8---|---9---|---0---|
$! Bulk Data Post Section
TSTEPNL         1     100    0.02
ENDDATA c62a81e6
```

Please note the following:

1. For Axial Symmetric run DYPARAM AXIALSYM is required
2. The Archive result files from this run can be remapped in the follow-up run (c) by using DYPARAM, AXREMAP and `sol700.pth` file where the regarding Archive file is defined by means of `EID` option
3. The input deck of the 3-D remap run has the following entries for remap:
   - Addition of PATH=3 in SOL 700 card to activate `sol700.pth` file
   - Addition of DYPARAM, AXREMAP
   - Removal of some initialization entry for the blast wave region that will be replaced by the data from the axial symmetric run
   - Additions of `sol700.pth` file to define the remapped Archive file.

3-D Model for Remap

Figure 4-3 shows the 3-D model that is used for remap the blast load. The following input file summarizes the entries that are required for remapping. These are highlighted with a red box.

```
$! Executive Control Section
SOL 700,129 STOP=1 PATH=3
CEND
ECHO = SORT
$! Case Control Section
PARAM, DYDEFAUL, DYNA
$! Subcase name : DefaultLoadCase
   TSTEPNL = 1
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CORNER)=ALL
BEGIN BULK
$! Bulk Data Pre Section
PARAM   DYDTOUT     0.12
PARAM   POST           0
$! Bulk Data Model Section
PARAM    INISTEP    0.001
PARAM    STEPFCT      0.6
PARAM      MICRO       16
PARAM   EULERCUB     2000
DYPARAM  AXREMAP       0.      0.0      0.0       0.      0.0      1.0
DYPARAM   LSDYNA  BINARY  D3PLOT      0.5
DYTIMHS    BINARY   0.001
        CMARKOUT
$!
EOSGAM          1     1.4                                              EOSGAM_1
MATDEUL         1   1.225        1
PEULER1         1            HYDRO        1                           PEULER_1
PMARKER         2   FIXED                                             PMARKER_
PMARKER         3   FIXED                                             PMARKER_
GRID          991             4.999   0.001   1.341
GRID          992             4.999   0.001   2.999
GRID          993             4.999   0.001   3.999
GRID          994             4.999   0.001   0.001
$!
$!
CMARKN1         1       3      991
CMARKN1         2       3      992
CMARKN1         3       3      993
CMARKN1         4       3      994
TICEUL1         1       1
TICREG          1       1CYLINDER       1       1       1       1.
$TICREG         2       1CYLINDER       2       1       2       2.
MESH            1     BOX
+             0.0     0.0      0.0      5.      6.      6.
+              50      60       60                              EULER       1
CYLINDR         1              -0.1      3.      3.      5.1      3.      3.
              5.
$CYLINDR        2               0.0      0.0      1.      0.0      0.0  1.6828
$          0.3414
TICVAL          1          DENSITY   1.225      SIE   0.2068
$TICVAL         2          DENSITY   1000.      SIE     3.52
$!  SX Names for Materials
$!-1---|---2---|---3---|---4---|---5---|---6---|---7---|---8---|---9---|---0---|
$! Bulk Data Post Section
TSTEPNL         1     100    0.06
ENDDATA e285ff9c
```

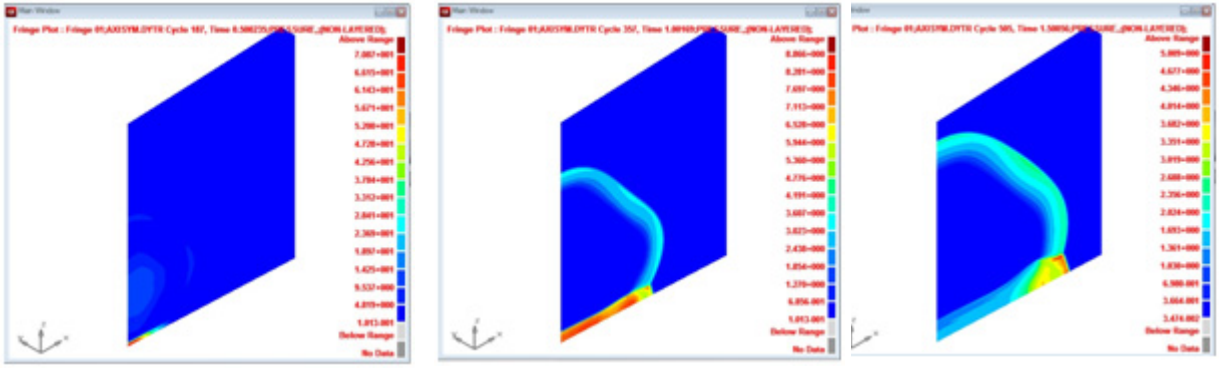
Sol700.pth (Sol700_A.pth)

```
C:\MSC.Software\MSC_Nastran\20122\msc20122\dyna\win64\run_dytran
exe=D:\MSC_NASTRAN_2013\dytran-lsdyna
memory=100m
delete=yes
debug=yes
gid=NUG82_A2.DYTR_EULER_187.ARC
```

Result Comparisons for the Variation of the Remap Time

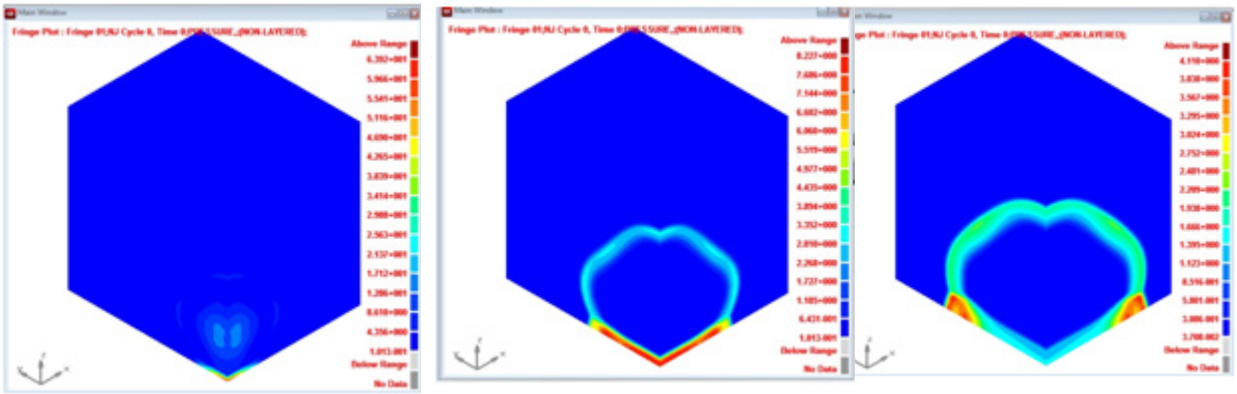
Axial Symmetric 2-D Run

Remap Time = 0.5 ms **Remap Time = 1.0ms** **Remap Time = 1.5 ms**

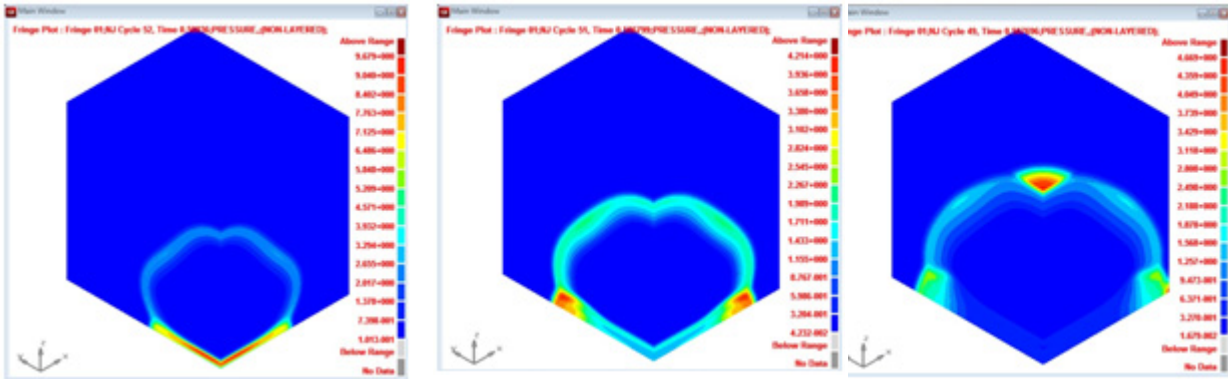


3-D- Remap Run

(c) Time = 0 ms **(d) Time = 0 ms** **(e) Time = 0 ms**



(c) Time = 0.5 ms **(d) Time = 0.5 ms** **(e) Time = 0.5 ms**



How it works: 1-D - 3-D Spherical-symmetry Mapping

Another method to remap the load blasts is by using a spherical 1-D wedge model. Again this is suitable for those applications with a spherical charge at a relatively far distance from the target.

The simulation is carried out in two stages. First, a spherical (1-D) calculation is carried out with a fine mesh to a desired time that would bring the blast wave in close proximity of the structure. Then the spherical blast load profile is mapped into the 3-D mesh which includes the structure. The remapping of the fine mesh solution into the 3-D relatively coarse mesh is accomplished with mass, energy and momentum conservation.

For detailed examples please refer to the Axial – Spherical Symmetry to 3D Euler Remap (Ch. 82) in the *MSC Nastran Demonstration Problems Manual*.

As an example, consider a model that consists of a cube with dimensions of 1m x 1m x 1m, with its origin at [0,0,0] and occupying the positive quadrant (Figure 4-4). A spherical charge is assumed to be placed with its center coinciding with the origin. Thus only one eighth of the charge will be included in the cube. The cube is meshed with a uniform Cartesian mesh, 80x80x80. The boundary walls are assumed rigid (by default). Some gauge points are placed to trace the blast response.

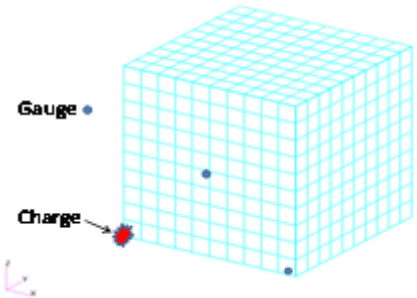


Figure 4-4 Model for Spherical Symmetry Remap

Figure 4-5 shows a full 3-D model that was run for comparative purposes with the 1-D-3-D example. Please refer to the MSC *Nastran Demonstration Problems Manual* to see the results comparisons for the full 3-D model. Figure 4-6 and Figure 4-7 show the model for the 1-D spherical wedge to compute the blast wave. Figure 4-8 shows the 3-D remap model where the results for the 1-D runs are used to map the load.

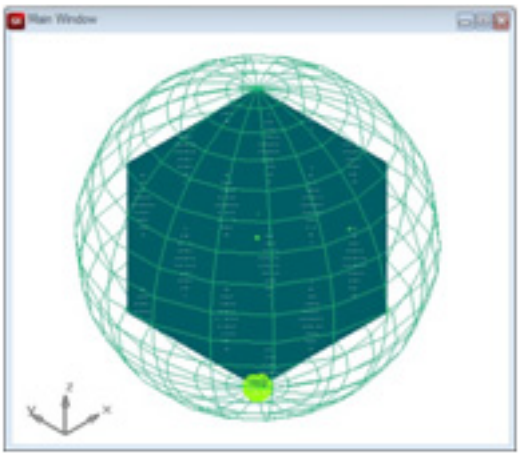


Figure 4-5 Model (a) for Full Run

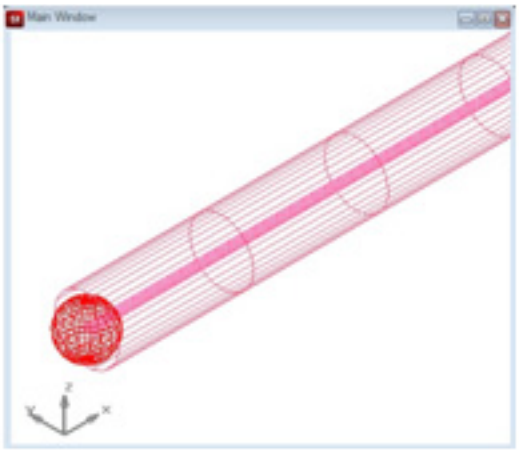


Figure 4-6 Geometry input for 1-D wedge

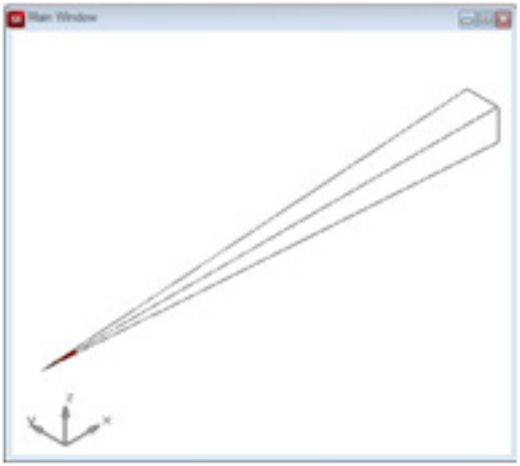


Figure 4-7 Model (b) Geometry Result

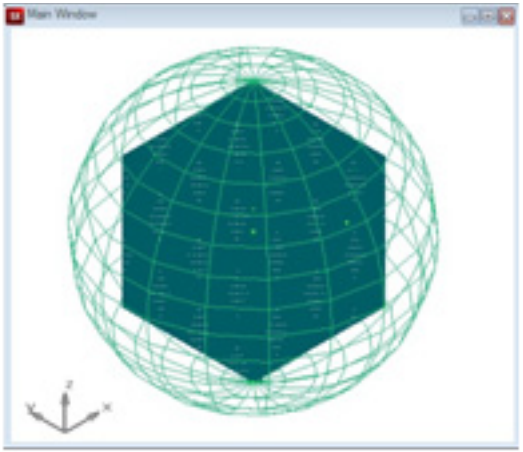


Figure 4-8 Model for Remap

Please note that the following new parameters are used for spherical symmetry:

1. For 1-D Spherical Symmetric run DYPARAM SPHERSYM is required
2. For Remap run DYPARAM SPREMAP is required

1-D - Spherical Symmetry Model

```
$! Executive Control Section
SOL 700,129 STOP=1
CEND
ECHO = SORT
$! Case Control Section
PARAM, DYDEFAUL, DYNA
$! Subcase name : DefaultLoadCase
   TSTEPNL = 1
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CORNER)=ALL
BEGIN BULK
$! Bulk Data Pre Section
PARAM   DYDTOUT     0.02
PARAM   POST           0
$! Bulk Data Model Section
PARAM    INISTEP  0.0001
PARAM    STEPFCT     0.6
PARAM      MICRO      20
DYPARAM SPHERSYM    RECT        X        2
DYPARAM   LSDYNA  BINARY  D3PLOT     0.08
EOSGAM          1     1.4                                          EOSGAM_1
MATDEUL         1   1.225        1
PEULER1         1            HYDRO        1                       PEULER_1
TICEUL1         1        1
TICREG          1        1CYLINDER        1       1       1      1.
TICREG          2        1  SPHERE        2       1       2      2.
MESH            1      BOX
+             0.0    -0.01    -0.01       2.    0.02    0.02
+             400        1        1                          EULER       1
CYLINDR         1             0.0     0.0     0.0     2.1     0.0     0.0
              0.1
SPHERE          2             0.0     0.0     0.0 0.07816
TICVAL          1          DENSITY   1.225     SIE  0.2068
TICVAL          2          DENSITY   1000.     SIE    3.52
$!  SX Names for Materials
$!-1---|---2---|---3---|---4---|---5---|---6---|---7---|---8---|---9---|---0---|
$! Bulk Data Post Section
TSTEPNL         1      100    0.01
ENDDATA d887e3b2
~
```

1-D - Spherical Symmetry - 3-D Remap Model

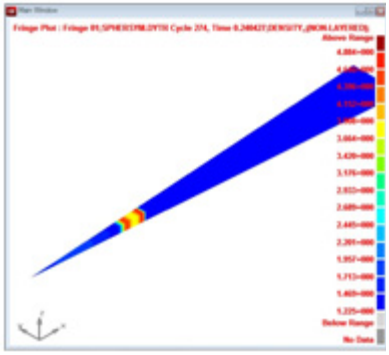
Sol700.pth (Sol700_S.pth)

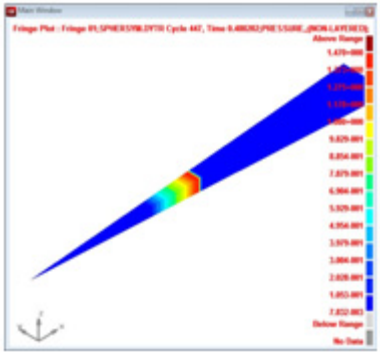
```
C:\MSC.Software\MSC_Nastran\20122\msc20122\dyna\win64\run_dytran
exe=D:\MSC_NASTRAN_2013\dytran-lsdyna
memory=100m
delete=yes
debug=yes
gid=NUG82_S2.DYTR_EULER_274.ARC
```

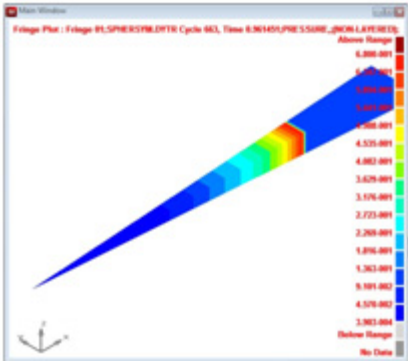
```
$! Executive Control Section
SOL 700,129 STOP=1 PATH=3
CEND
ECHO = SORT
$! Case Control Section
PARAM, DYDEFAUL, DYNA
$! Subcase name : DefaultLoadCase
   TSTEPNL = 1
   DISPLACEMENT(SORT1,PRINT,REAL)=ALL
   STRESS(SORT1,PRINT,REAL,VONMISES,CORNER)=ALL
BEGIN BULK
$! Bulk Data Pre Section
PARAM   DYDTOUT    0.024
PARAM   POST           0
$! Bulk Data Model Section
PARAM    INISTEP  0.0001
PARAM    STEPFCT     0.6
PARAM      MICRO      16
PARAM   EULERCUB    2000
DYPARAM  SPREMAP       0       0    0
DYPARAM   LSDYNA  BINARY  D3PLOT    0.25
DYTIMHS    BINARY   0.001
        CMARKOUT
$!
EOSGAM         1     1.4                                          EOSGAM_1
MATDEUL        1   1.225       1
PEULER1        1           HYDRO       1                         PEULER_1
PMARKER        2   FIXED                                         PMARKER_
PMARKER        3   FIXED                                         PMARKER_
GRID         991           0.999   0.001   0.001
GRID         992           0.707   0.001   0.707
GRID         993           0.577   0.577   0.577
GRID         994           0.001   0.001   0.999
GRID         995           0.001   0.001   0.001
$!
$!
CMARKN1        1       3     991
CMARKN1        2       3     992
CMARKN1        3       3     993
CMARKN1        4       3     994
CMARKN1        5       3     995
TICEUL1        1       1
TICREG         1       1  SPHERE       1       1       1      1.
$TICREG        2       1  SPHERE       2       1       2      2.
MESH           1     BOX
+            0.0     0.0     0.0      1.      1.      1.
+             80      80      80                      EULER       1
SPHERE         1             0.5     0.5     0.5      1.
$SPHERE        2             0.0     0.0     0.0 0.07816
TICVAL         1         DENSITY   1.225     SIE  0.2068
$TICVAL        2         DENSITY   1000.     SIE    3.52
$!  SX Names for Materials
$!-1---|---2---|---3---|---4---|---5---|---6---|---7---|---8---|---9---|---0---|
$! Bulk Data Post Section
TSTEPNL        1     100   0.012
ENDDATA d3a5741c
```

Result for Spherical Symmetric 1-D Run

Remap Time = 0.24 **Remap Time = 0.48** **Remap Time = 0.98**

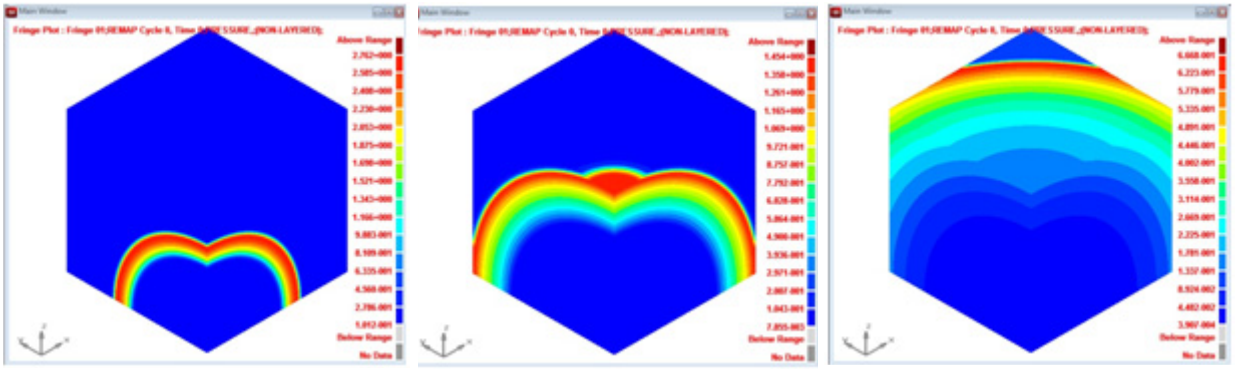




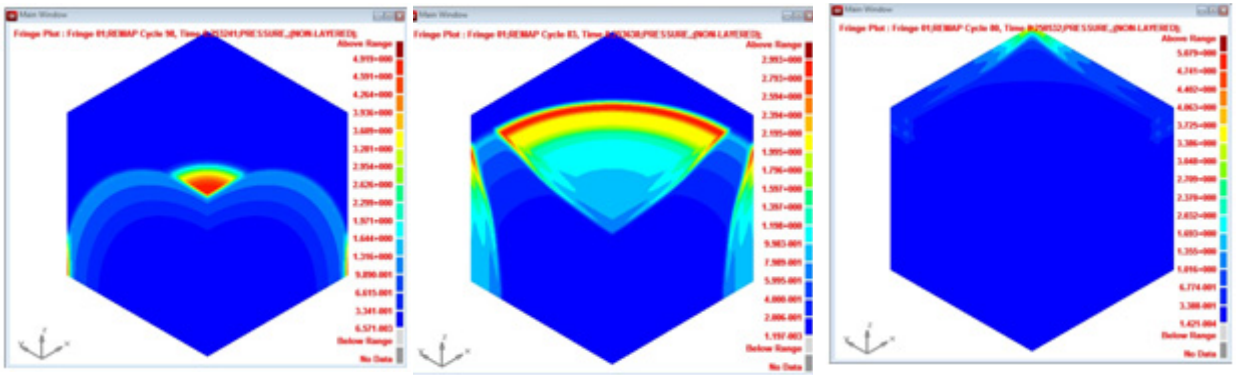


Results for 1-D - 3-D Remap

(c) Time = 0 ms **(d) Time= 0 ms** **(e) Time = 0 ms**



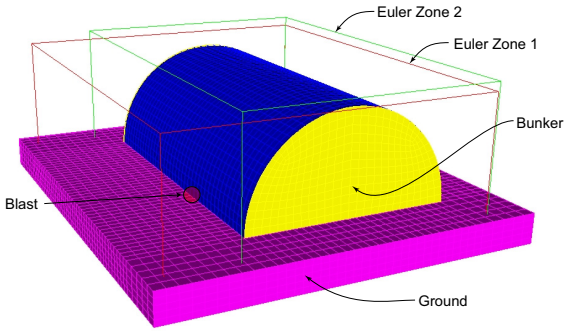
(c) Time = 0.25 ms **(d) Time= 0.25 ms** **(e) Time = 0.25 ms**

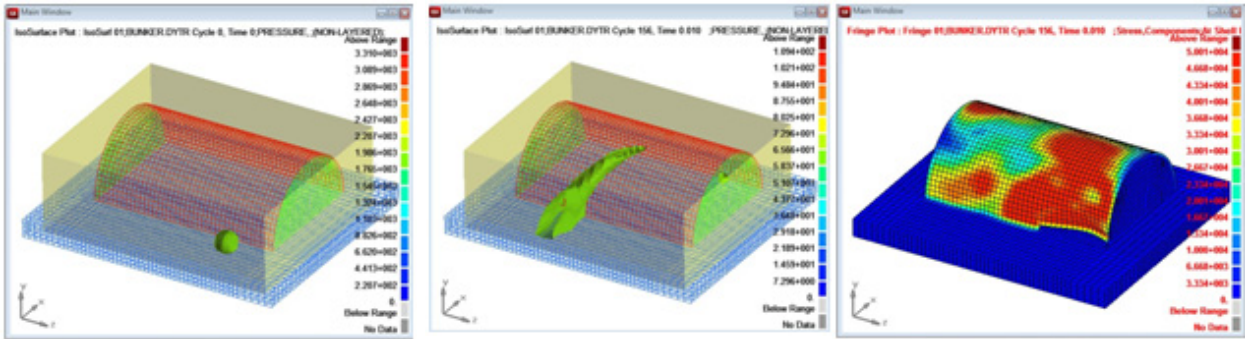


In addition to the simple examples shown above, two real life applications are demonstrated in the *MSC Nastran Demonstration Problems Manual* (Ch. 82). These are the blast against a bunker structure using the 2-D - 3-D mapping method and an UNDEX (Under Water Shock Explosion) using the 1-D - 3-D spherical mapping.

Blast Against a Bunker using 2-D Axial Symmetric - 3-D Remap

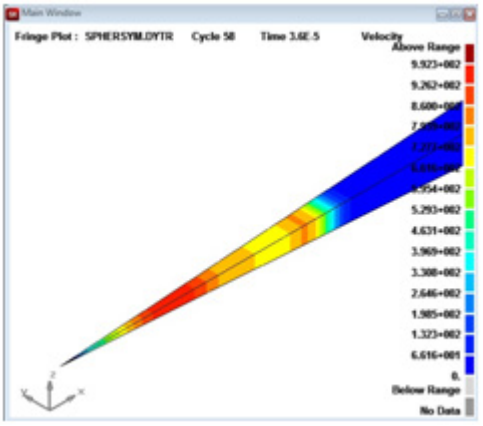
This is the same problem as in Blastwave Hitting a Bunker (Ch. 43) in the *MSC Nastran Demonstration Problems Manual*. An explosive is placed next to a bunker structure as shown below. The 2-D - 3-D remap approach is used on this problem as documented in Axial – Spherical Symmetry to 3D Euler Remap (Ch. 82) in the *MSC Nastran Demonstration Problems Manual*.



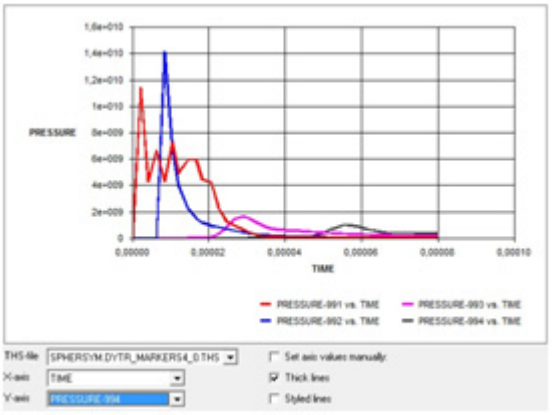


UNDEX Example using 1-D Axial Symmetric - 3-D Remap

The Underwater Shock Analysis (UNDEX) example was re-run using 1-D - 3-D remapping. The detailed entries are documented in Axial – Spherical Symmetry to 3D Euler Remap (Ch. 82) in the *MSC Nastran Demonstration Problems Manual*.



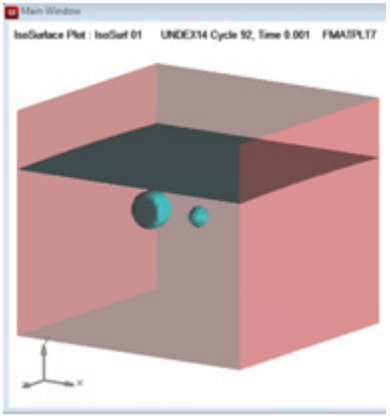
Time = 3.8E-4 S (Cycle 58, to be remapped)



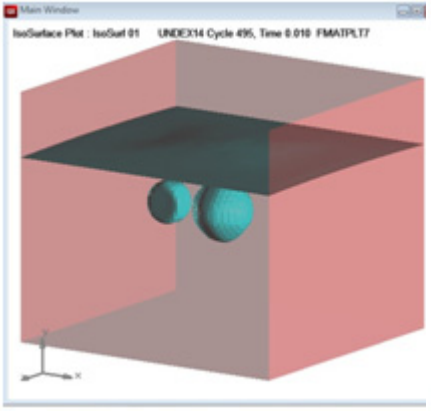
Pressure History on Markers

Result

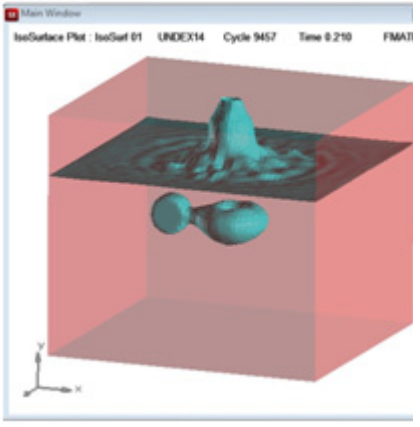
Time = 0.001 S



Time = 0.01 S

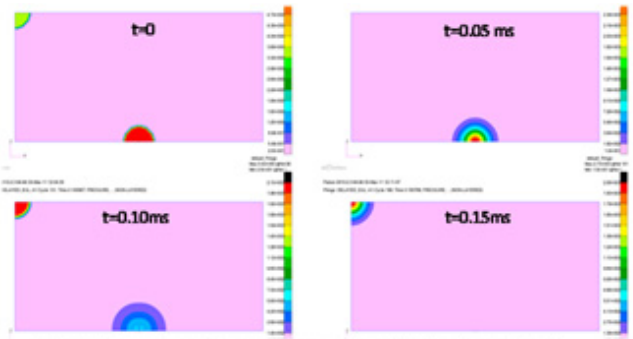


Time = 0.21 S

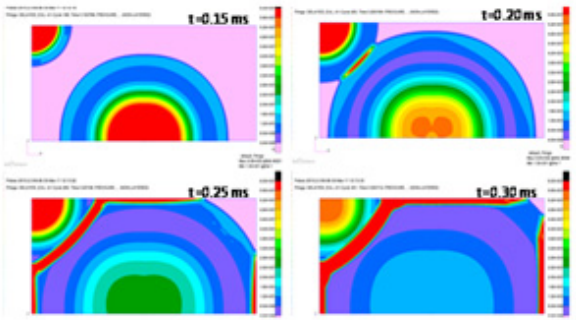


Ignition Times for Multiple Detonations

The blast wave created by multiple explosions is an important safety consideration for ammunition magazines. In certain cases, it is important to take into account the delay time of the ignition of the various charges present in the magazine. To get a delayed ignition, the EOSJWL material model has to be used for each explosive. In addition, the detonation process needs to be specified by a DETSPH entry. In principle, for each EOSJWL material, a distinct DETSPH entry can be used. This enables the user to define the ignition time for each explosive separately. By default, the DETSPH definition of one explosive also applies to the other explosive. Therefore, the blast wave of one explosive can ignite the other explosive. When this happens, the explosive is triggered by a DETSPH definition of a different material. This is not always suitable. With MSC Nastran 2013 an option has been added that switches off this ignition of explosive by another explosive. This option is activated by DYPARAM,JWLDET. When using DYPARAM,JWLDET,NOLINK, the explosive can no longer ignite the other material. Then, the ignition is simulated exactly as specified by the specific DETSPH entries.



Internal energy and pressure maps for a double charge blast wave. The scale of this map was chosen automatically by Patran



Pressure maps for the double charge blast wave. The scale of these maps was chosen manually to have a maximum of 60 MPa

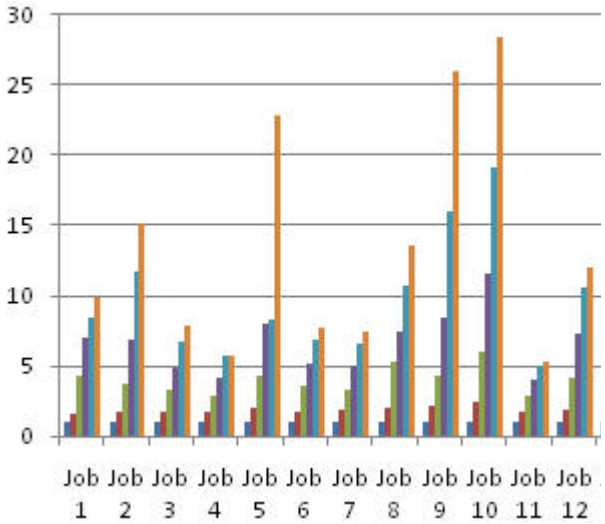
Use of Markers for Time History Data

Another useful feature that is introduced in this release is the use of markers to obtain the time histories of the blast response at arbitrary locations. These are obtained by either requesting a time history for an Euler element or a time history for a marker. When requesting element time histories, the user has to specify an element so he has first to look what element is present at a specific position. With markers, the user only needs to give the coordinates of the specific position. This makes markers more user-friendly. Markers can be used in both orthogonal and non-orthogonal meshes.

"LOAD_BLAST" Method for Empirical Blast Loadings

The LOAD BLAST boundary condition in MSC Nastran SOL700 is based on the work by Randers-Pearson and Bannister (1997) that was implemented in the CONWEP code (Conventional Weapons Proliferation) to simulate the empirical blast loading. The blast loading can be utilized in two cases:

1. Surface detonation of a hemispherical charge
2. Free air detonation of a spherical charge



This method is widely used in the defense industry due to the abundance of empirical data and relatively simple models. However, the LOAD BLAST method is not adequate for buried explosive devices such as land mines.

Load Blast Simulation (Ch. 83) in the *MSC Nastran Demonstration Problems Manual* has two different examples to compare the results of the LOAD BLAST to empirical data as well conventional FSI methods using the Eulerian approach.

In the first example a plate model was subjected to a blast loading and the results of LOAD BLAST were compared to those from the paper “Dynamic Stress Analysis of the effect of an Air Blast Wave on a Stainless Steel Plate” (see Part 1: Plate Model (Ch. 83) in the *MSC Nastran Demonstration Problems Manual*). The plate was modeled with both shells and solids to make sure that the results were consistent across different element types.

In the second example shown as follows, an explosive was detonated underneath an armored vehicle. The blast load was modeled with LOAD BLAST method and then the results were compared with general coupling method.

Armored Vehicle Model

In this model a blast wave loading of 7 kg TNT is applied under an armored vehicle at a distance of 0.3 m from the lowest vehicle floor.

Two types of blast loading will be analyzed and the results will be compared with those from the Sol700 FSI calculations.

1. Hemispherical surface burst - charge is located on or very near the ground surface.
2. Spherical free-air burst - no amplification of the initial shock wave due to interaction with the ground surface

Blast Load of 7 kg TNT Under Vehicle

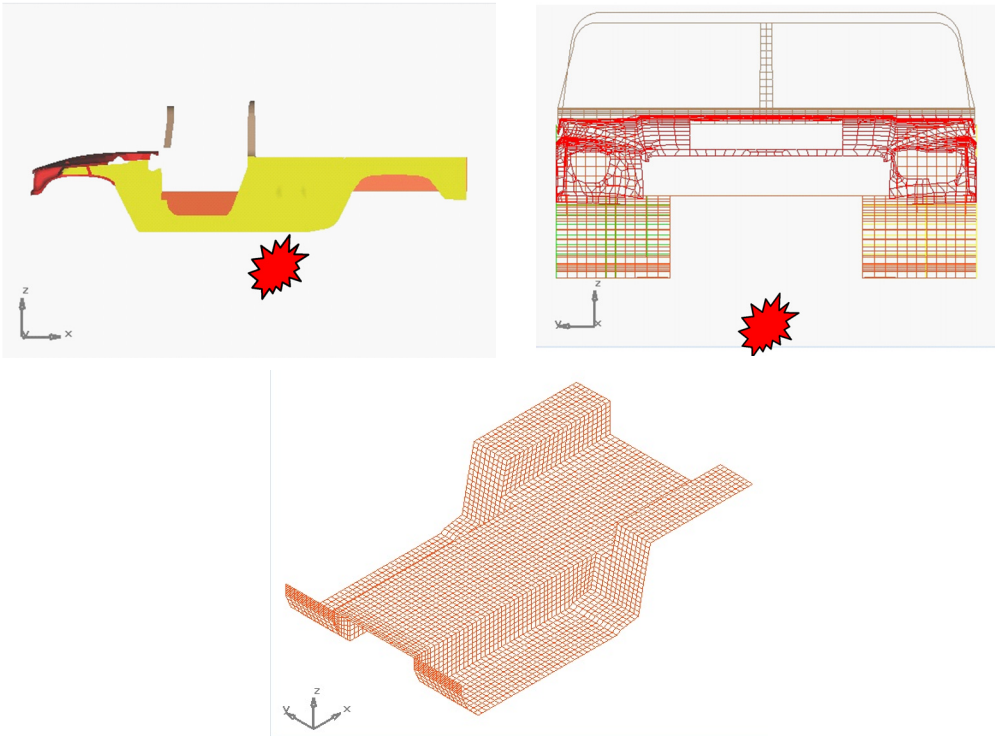


Figure 4-9 PLBLAST loaded Segments

MSC Nastran SOL700 Input for Load Blast Model

The LOAD BLAST method is defined by using the EXPLSV and PLBLAST entries. The PLBLAST defines a surface subjected to air blast pressure. An air blast source must be defined in EXPLSV as a function of pressure loads caused by the detonation of conventional charge. This feature includes enhancements for treating reflected waves, moving warheads and multiple blast sources. The shape of explosive such as partially buried hemispherical or spherical in free air can also be defined using the EXPLSV entry.

DYPARAM,LSDYNA,BLSTFOR generates blast pressure history database (`*.blastfor`). PLBLAST is used for applying blast pressure generate by this entry.

The following tables demonstrate the use of these entries to define a hemispherical and free air burst for the armored vehicle:

Type 1: Hemispherical Surface Burst

```
BEGIN BULK
$------- Parameter Section ------
$
PARAM,DYNINT,15000000
PARAM,DYNREAL,15000000
DYPARAM,STEPFCTL,0.9
PARAM*,DYINISTEP,.5E-6
PARAM*,DYMINSTEP,1.E-7
$
DYPARAM,LSDYNA,BINARY,D3PLOT,.00005
PARAM,LSDYNA,BINARY,BLSTFOR,0.00005
$
$
EXPLSV,7,7.,4.600, 0.000,  2.0,0.0,2,1,+
+,,,,,,1.E20,,,+
+,,,,,,,,,+
+
$
INCLUDE HV_SI.bdf
INCLUDE plblast7.bdf
ENDDATA
```

Type 2: Free air Burst

```
BEGIN BULK
$------- Parameter Section ------
$
PARAM,DYNINT,15000000
PARAM,DYNREAL,15000000
DYPARAM,STEPFCTL,0.9
PARAM*,DYINISTEP,.5E-6
PARAM*,DYMINSTEP,1.E-7
$
DYPARAM,LSDYNA,BINARY,D3PLOT,.00005
PARAM,LSDYNA,BINARY,BLSTFOR,0.00005
$
EXPLSV,7,7.,4.600, 0.000,  2.0,0.0,2,2,+
+,,,,,,1.E20,,,+
+,,,,,,,,,+
+
$
INCLUDE HV_SI.bdf
INCLUDE plblast7.bdf
ENDDATA
```

A full 3-D FSI model (see below) was constructed to compare the results of the LOAD BLAST method to conventional FSI methods using the general coupling (For complete description of the Fluid-Structure-Interaction model please refer to Mine Blast Under a Vehicle (Ch. 42) in the *MSC Nastran Demonstration Problems Manual*.

Deformation results for Solid model

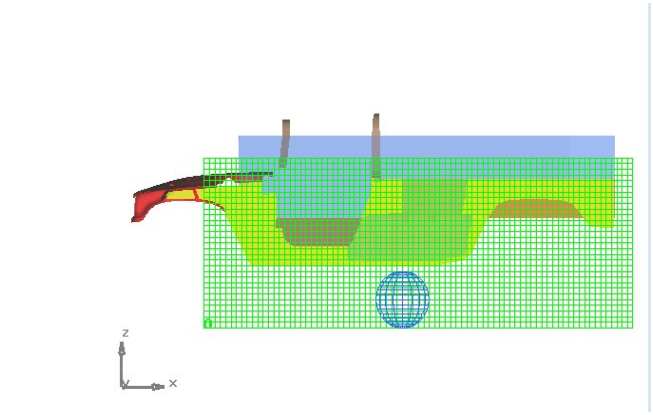


Figure 4-10 Explosive Under the Vehicle Floor

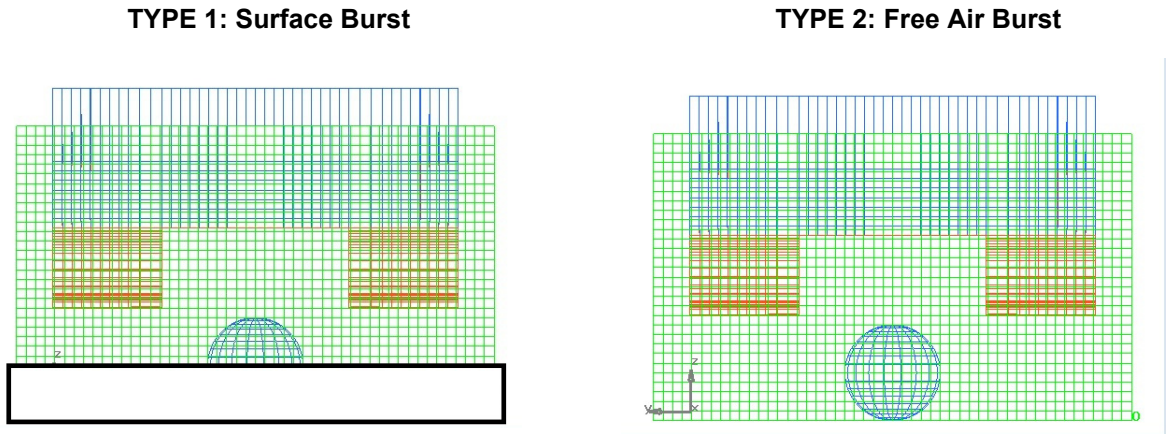


Figure 4-11 Surface Burst vs. Free Air Burst

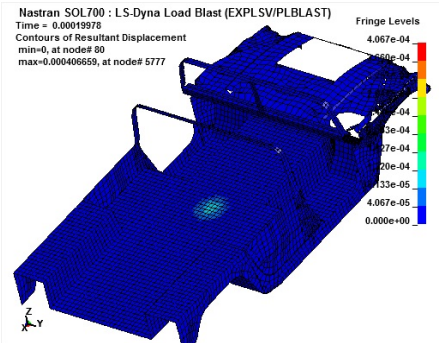
Result Comparison for Hemispherical Surface Burst (Type 1)

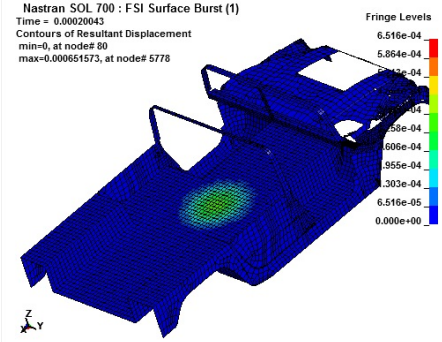
Results were compared between the LOAD BLAST method using the empirical approach and conventional FSI methods for both hemispherical surface burst (Type 1) and spherical free air (Type 2) explosive as shown below. Here we only show the result comparisons for the Type 1. Please refer to the Load Blast Simulation (Ch. 83) in the *MSC Nastran Demonstration Problems Manual* for detailed example.

The time history of the displacements, pressure and energies at select locations on the vehicle are reasonably close between the two approaches as shown in the following plots.

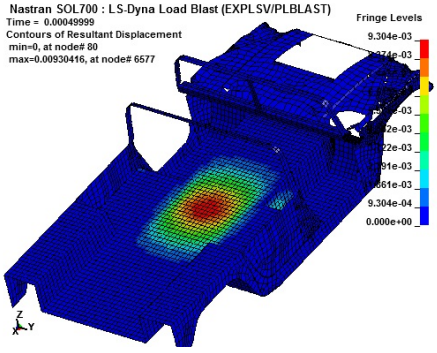
Load Blast using EXPLSV-PLBLAST **FSI-Eulerian**

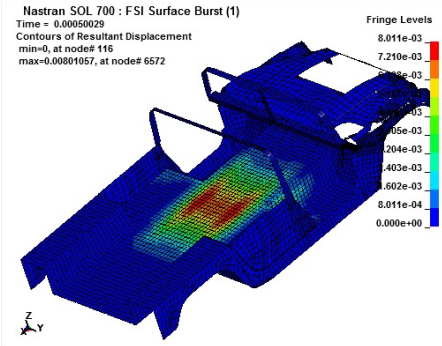
Displacement at Time = 0.2 ms



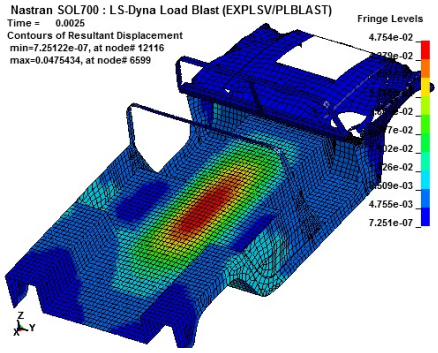


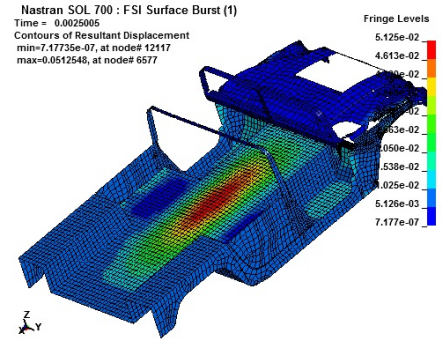
Displacement at Time = 0.5 ms





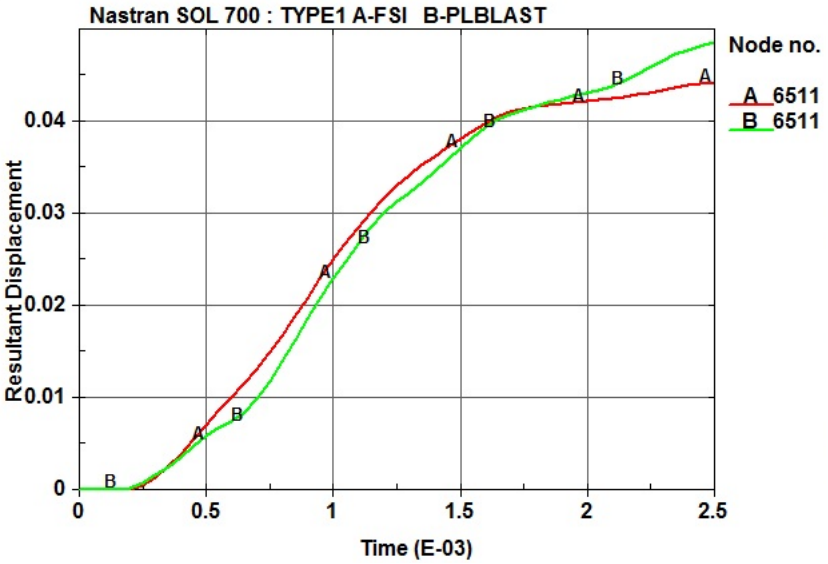
Displacement at Time = 2.5 ms



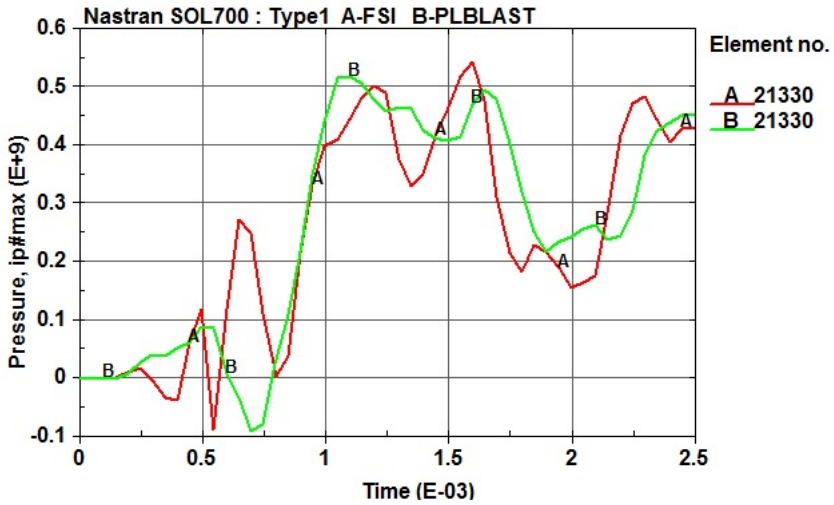


Results Comparison between Load Blast and FSI-Blast Wave Calculation Type 1

Displacement: Red FSI - Green PLBLAST

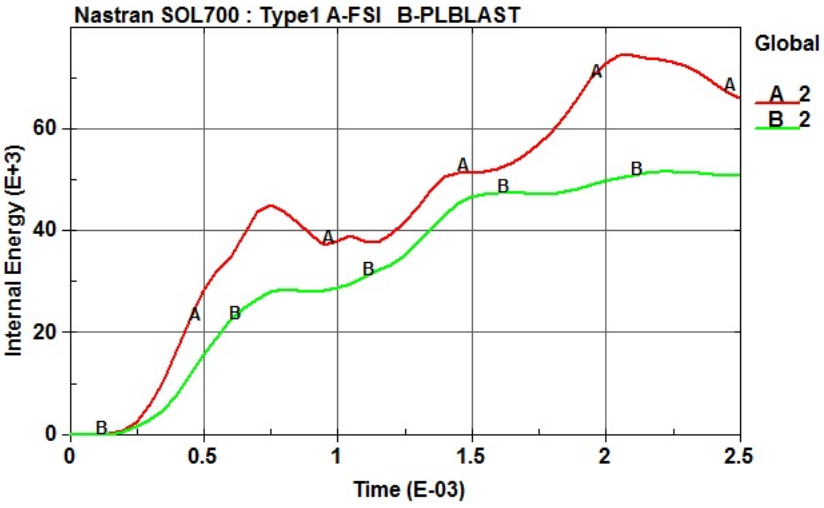


Pressure: Red FSI - Green PLBLAST



Internal energy of the entire structure:

Red FSI - Green PLBLAST



Enhancements to FSI Algorithms to Speed Up the Simulation Time

Several enhancements were done to Distributed Memory Parallel (DMP) algorithms to speed up the performance of the FSI simulations. In particular, one area of the focus has been improving the MPI calls in a multiple node with multiple core cluster environment.

A total of 23 benchmark problems (see Table 4-1 for model sizes) were run to study the performance on a single Linux 64 node with 32 cores called "EM64TE" (see Table 4-2) and on a cluster called "Janus" that was provided by the HPC Advisory Council to see the DMP performance on a multiple node, distributed environment. The platform configurations are as follow:

32 Core Linux Server

Red Hat Enterprise Linux Server release 5.4 (Tikanga)

Intel(R) Xeon(R) CPU E7- 8837

Processor composition:

Processors(CPUs) : 32

Packages(sockets) : 4

Cores per package : 8

Threads per core : 1

64 Core Linux Cluster

- Dell™ PowerEdge™ M610 38-node cluster
- Dual-socket six-Core Intel® Xeon® processor X5670 @ 2.93 GHz
- Intel Cluster Ready certified cluster
- Mellanox ConnectX®-2 40Gb/s InfiniBand mezzanine card
- Mellanox M3601Q 36-Port 40Gb/s InfiniBand Switch
- Memory: 24GB memory per node

The cluster consists of 38 nodes. Each node has two CPU's and each CPU has 6 cores. Access was limited to 16 of the nodes with a total of 64 cores due to licenses available at the time of testing.

The benchmarks were run with MSC Nastran 2013 and the previous version 2012.2 on single processor as well as 64 core (8 Nodes x 8 cores each) to show the performance improvements with respect to the previous release. These

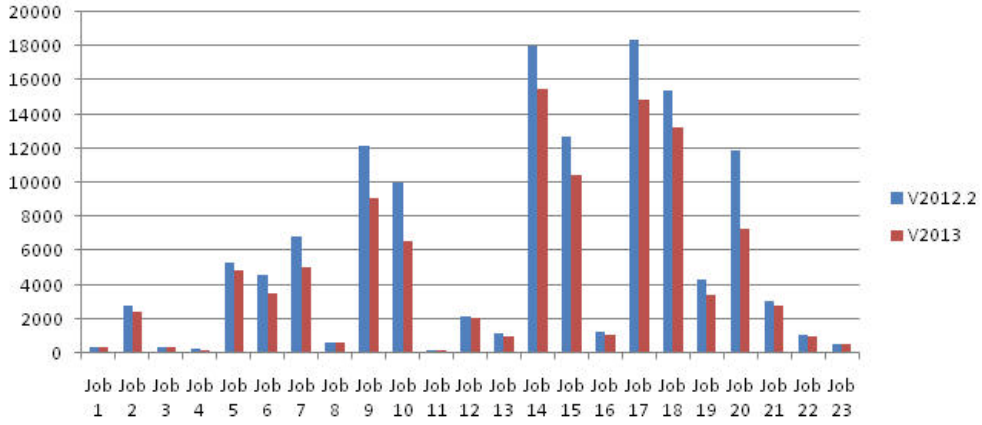
results are shown in Table 4-1 and the following charts.

Several runs were made with 8 MPIs, 16 MPIs, 32 MPIs and 64 MPI tasks using different number of nodes with different cores to study the influence of the network configuration and load overhead. For example for 8 MPI task, jobs were run on 1 node and 8 cores (1Nx8C), 2 nodes with 4 cores each (2Nx4C), 4 nodes with 2 cores each (4Nx2C) and 8 nodes with 1 core each (8Nx1C). These results were very encouraging and will be published separately in a paper.

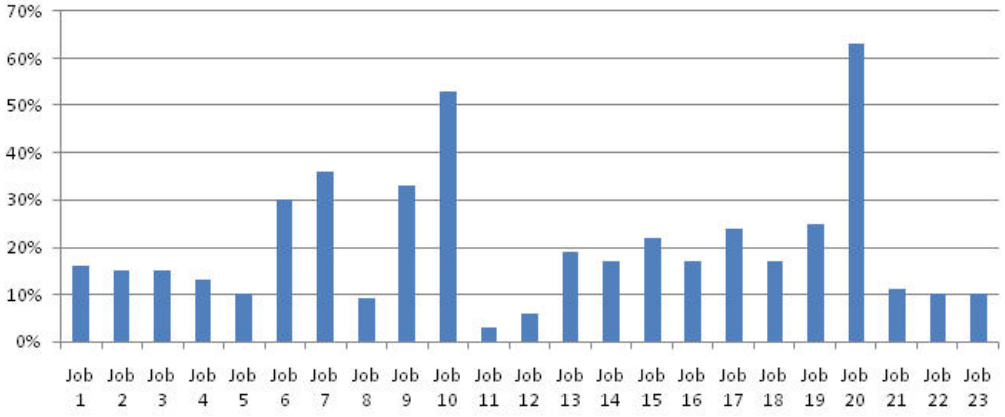
Table 4-1 The DMP Benchmark Problems on Single Core and 64 Cores (8Nx8C)

Job	Lagrangian Elements	Number of Euler Elements	Total no. of Elements	V2012.2	V2013	Performance on Single core	V2012.2	V2013	Performance on 64 core 8Nx8C
Job 1	3394	110592	113986	370	320	16%			
Job 2	312	360000	360312	2827	2459	15%	256	231	11%
Job 3	6302	612720	619022	402	349	15%	90	83	8%
Job 4	3280	451200	454480	245	216	13%	75	71	6%
Job 5	161	240000	240161	5323	4847	10%	242	191	27%
Job 6	9660	82944	92604	4617	3549	30%	1386	594	133%
Job 7	9961	410435	420396	6820	5029	36%			
Job 8	29	168345	168374	659	605	9%			
Job 9	2000	1572864	1574864	12133	9106	33%	379	302	25%
Job 10	181	90000	90181	10013	6551	[illegible]			
Job 11	26	294912	294938	154	150	3%	68	64	6%
Job 12	0	524288	524288	2164	2049	6%	191	226	-15%
Job 13	9961	378594	388555	1191	1005	19%	220	167	32%
Job 14	178908	1304576	1483484	18025	15452	17%	5481	2378	130%
Job 15	56900	2195200	2252100	12716	10409	22%	724	621	17%
Job 16	300	192000	192300	1296	1108	17%			
Job 17	147327	884736	1032063	18372	14852	24%	3663	2399	53%
Job 18	32	1000000	1000032	15445	13252	17%	808	644	25%
Job 19	2000	983040	985040	4326	3464	25%	270	205	32%
Job 20	0	884736	884736	11906	7291	63%	278	209	33%
Job 21	0	1162851	1162851	3064	2765	11%	208	196	6%
Job 22	59738	108000	167738	1067	974	10%			
Job 23	6	68921	68927	552	501	10%			

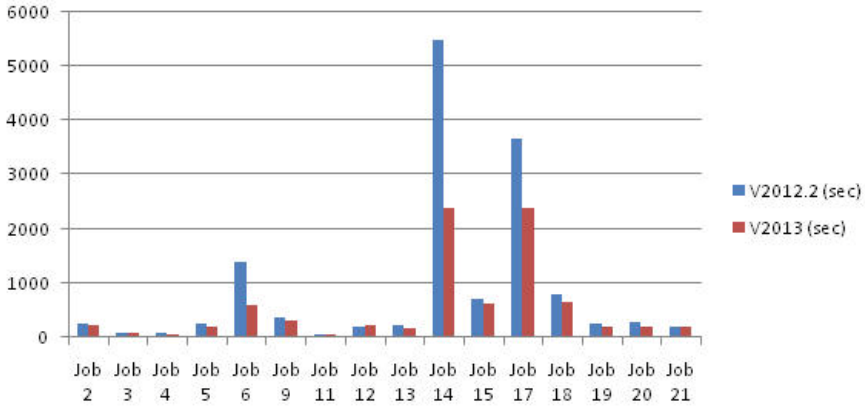
Elapsed Time (sec) on Single Core Between MSC Nastran 2013 vs 2012.2



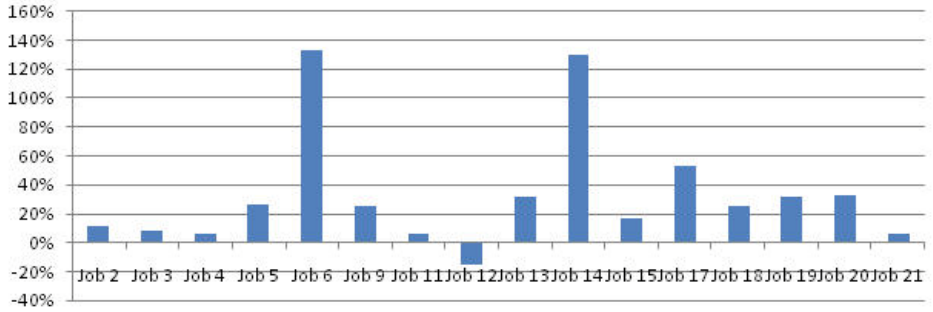
Performance Improvements on Single Core Between MSC Nastran 2013 and 2012.2

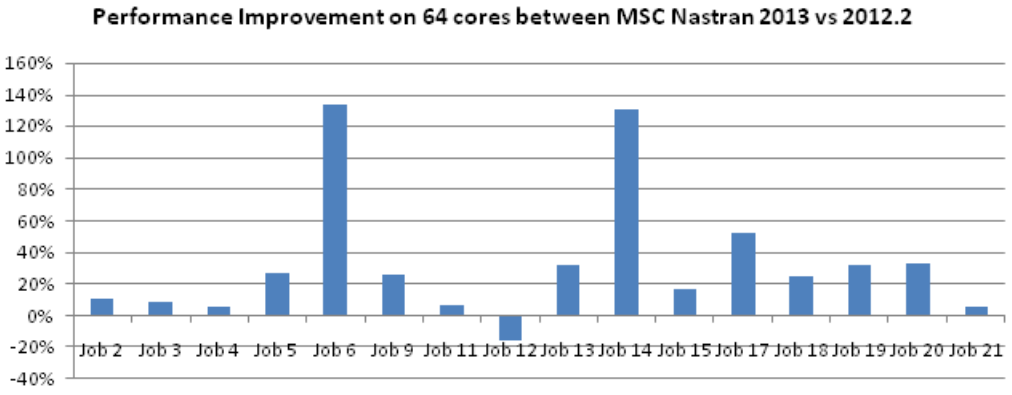


Elapsed Time (sec) on 64 Cores Between MSC Nastran 2013 vs 2012.2



Performance Improvement on 64 Cores Between MSC Nastran 2013 vs 2012.2





Elapsed Time (sec) on Single Core vs 64 Cores with MSC Nastran 2013

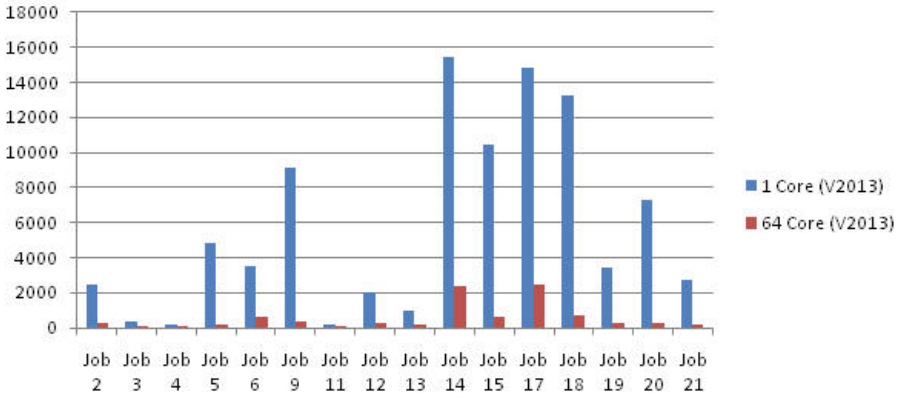
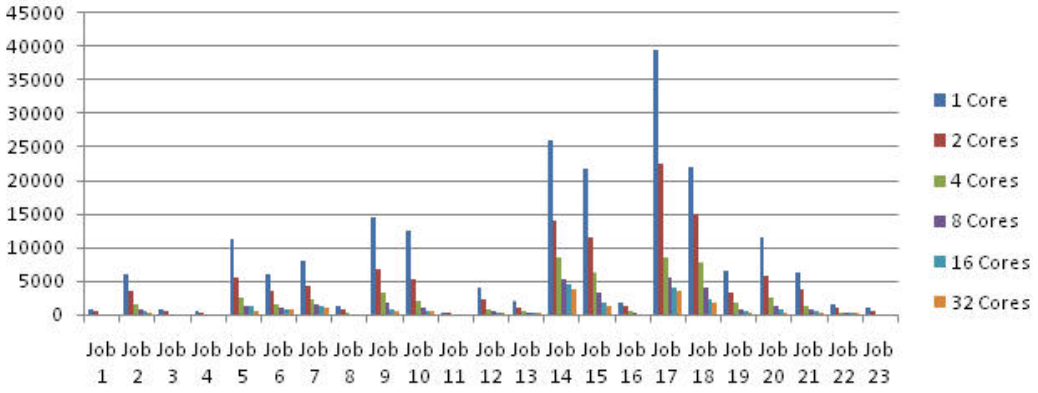


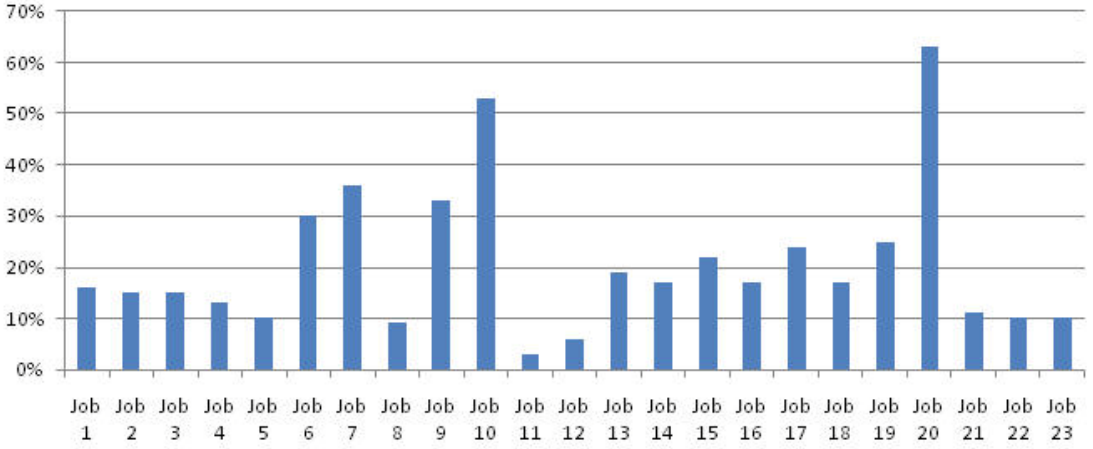
Table 4-2 Performance Results on Single Node with 32 Cores

Elapsed Time (sec)							Speedup Factors						
All	1 Core	2 Cores	4 Cores	8 Cores	16 Cores	32 Cores	All	1 Core	2 Cores	4 Cores	8 Cores	16 Cores	32 Cores
Job 1	643	410	149	91	76	65	Job 1	1	1.57	4.32	7.07	8.46	9.89
Job 2	5899	3361	1592	866	505	388	Job 2	1	1.76	3.71	6.81	11.68	15.2
Job 3	719	408	222	147	107	92	Job 3	1	1.76	3.24	4.89	6.72	7.82
Job 4	420	239	147	103	73	73	Job 4	1	1.76	2.86	4.08	5.75	5.75
Job 5	11273	5508	2599	1411	1354	493	Job 5	1	2.05	4.34	7.99	8.33	22.87
Job 6	5909	3476	1660	1143	861	770	Job 6	1	1.7	3.56	5.17	6.86	7.67
Job 7	7887	4272	2448	1587	1191	1056	Job 7	1	1.85	3.22	4.97	6.62	7.47
Job 8	1336	671	251	181	124	98	Job 8	1	1.99	5.32	7.38	10.77	13.63
Job 9	14541	6790	3409	1712	906	560	Job 9	1	2.14	4.27	8.49	16.05	25.97
Job 10	12449	5264	2065	1070	649	438	Job 10	1	2.36	6.03	11.63	19.18	28.42
Job 11	321	187	113	81	64	61	Job 11	1	1.72	2.84	3.96	5.02	5.26
Job 12	4080	2151	983	555	384	340	Job 12	1	1.9	4.15	7.35	10.63	12
Job 13	1954	994	661	379	244	223	Job 13	1	1.97	2.96	5.16	8.01	8.76
Job 14	25887	13870	8474	5424	4529	3879	Job 14	1	1.87	3.05	4.77	5.72	6.67
Job 15	21805	11400	6371	3332	1874	1280	Job 15	1	1.91	3.42	6.54	11.64	17.04
Job 16	1840	1197	557	311	185	141	Job 16	1	1.54	3.3	5.92	9.95	13.05
Job 17	39380	22360	8530	5496	3975	3669	Job 17	1	1.76	4.62	7.17	9.91	10.73
Job 18	22065	15050	7783	3974	2311	1747	Job 18	1	1.47	2.84	5.55	9.55	12.63
Job 19	6524	3314	1874	886	565	388	Job 19	1	1.97	3.48	7.36	11.55	16.81
Job 20	11527	5659	2562	1416	737	417	Job 20	1	2.04	4.5	8.14	15.64	27.64
Job 21	6265	3625	1440	784	482	365	Job 21	1	1.73	4.35	7.99	13	17.16
Job 22	1501	861	453	271	212	191	Job 22	1	1.74	3.31	5.54	7.08	7.86
Job 23	930	398	159	96	70	60	Job 23	1	2.34	5.85	9.69	13.29	15.5

Elapsed Time (sec) on 1, 2, 4, 8, 16 and 32 Cores



Speed up Factors on 1, 2, 4, 8, 16 and 32 Cores



Results Discussion

The following conclusions can be made based on the previous charts:

Performance of MSC Nastran 2013 vs 2012.2 on a 64 Core Linux Server

1. MSC Nastran 2013 has dramatic performance improvements compared to previous version 2012.2. The performance improvements on a single core can be as high as 63% based on the 23 problems that were run. When the same problems were run on 64 cores with 8Nx8C configurations, the performance improvements in v2013 can be as high as 133% compared to the previous release. There are jobs that had poor performance on 64 cores due to the relatively small size of the models. These jobs were excluded from the 64 core runs. There was an outflier (job 12) that had no lagrangian elements and had a poor performance on 64 core.

2. When comparing the speed ups between the serial (single core) and 64 core on Janus with MSC Nastran 2013, the performance improvements ranges from approximately 100% (1X) for job 11 to an impressive 3300% (33X) for job 20. Again it has to be noted that some of the jobs were not tested on 64 cores due to the small sizes of the models. The model sizes, domain decomposition of the eulerian cubes, the type of analysis and platform configuration play important roles in performance.

Performance of MSC Nastran 2013 on a 32 core Linux server

Same benchmarks were also run with MSC Nastran 2013 on another Linux machine with 32 cores on 1, 2, 4, 8, 16 and 32 cores. All cores located on the same node.

Table 4-2 and the previous charts summarize the results. In all cases there were performance improvements and reasonable scalabilities. The performance improvements on 32 core Linux server ranges from 5.26X for job 11 to an impressive 28.42X for job 10 on 32 cores compared to a serial run.

6 Numerical Methods and High Performance Computing

ACMS (Automated Component Mode Synthesis) Performance Improvements (2013.1)

The MSC Nastran modal-based analysis includes normal modes, modal frequency, modal transient analysis (SOLs 103, 111, and 112, respectively), and their appropriate subsets within design optimization (SOL 200). ACMS is provided as an alternative and more efficient version of the Lanczos method for those cases when model sizes become prohibitively large (size > 500,000 degrees of freedom) or for moderate models where the desired number of modes is high (# modes > 1000). In the MSC Nastran 2013.1 version, particular attention has been directed towards reducing I/O usage by ACMS as a means of delivering more efficient modal analysis. Additionally, default parameters have been changed to provide improved partitions in the ACMS method and sparse calculations have replaced dense calculations in places for even further performance improvements.

Benefits

The MSC Nastran 2013.1 version has a 10% to 30% reduction in the ACMS part of a simulation time for many large-scale models. This will lead to a reduction in overall runtime for analyses for which ACMS or modal extraction consumes most of the computation time. In particular, those models where I/O is a dominant feature of the simulation will be most impacted. A reduction in simulation time for ACMS allows customers to perform more modal analysis simulations over a given time period and therefore answer more engineering questions of relevance to their organization.

Technical Discussion

ACMS invokes an automatic partition to divide the global matrix structure into a given number of sub-domains, or components. A partition that generates sub-domains with large boundary sizes can yield inferior performance. In order to minimize boundary sizes, a multi-level “reduction” tree is constructed. Current implementation restricts the shape of the tree to a binary tree, thus limiting the number of “tip” components to a power of two. Another current implementation issue is related to processing the reduction tree one level at a time. This is referred to in the literature as “level order” tree traversal. An example is shown in Figure 6-1.

For this case, the global matrix structure is divided into 16 components. Note that the tree diagram in Figure 6-1 has an inverted shape in contrast to what is often seen in literature. After the division of the global matrix, we define the reduction tree leading to component “0” where system modes are computed. This tree consists of five levels: level 1 consists of domains 1 through 16; level 2 consists of domains 17 through 24, and so on. In Figure 6-1, components are color coded to indicate a static assignment of domains to DMP processes (in this case, dmp=4).

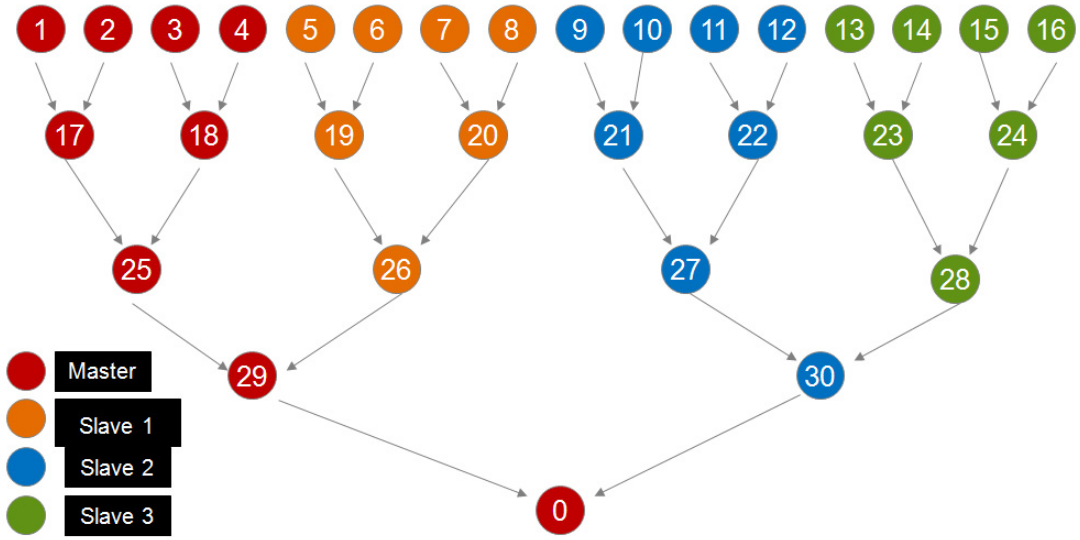


Figure 6-1 Level Order Tree Traversal

Note the order of processing: all tip components are processed (1 through 16) before the first "collector" component is processed (domain 17). This means that all contributions from level 1 of the tree must be maintained (on the data base or in the Buffer Pool) before they are used to process level 2. This fact puts a burden on the Buffer Pool, which may be exhausted before the end of level 1 is reached, necessitating flushing of buffers to the data base; this increases I/O traffic and database size. One way to alleviate this condition is to change the processing order. Figure 6-2 shows the same example using "post order" tree traversal.

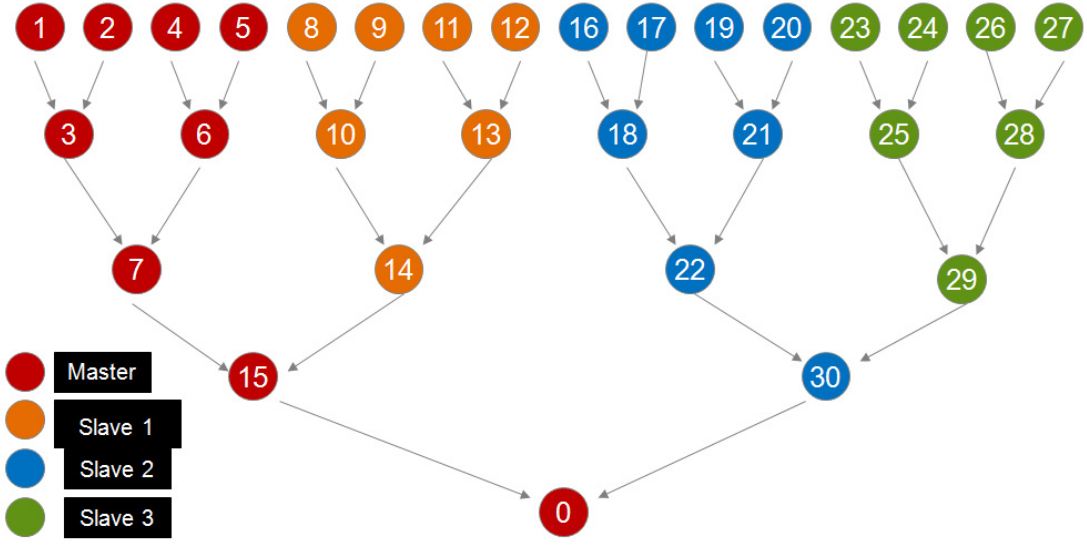


Figure 6-2 Post Order Tree Traversal

Traversing the tree with a post order strategy, the first collector component is processed as soon as its upstream matrices are available. For example, when computations for components 1 and 2 are complete, component 3 is then processed. When component 3 is complete, intermediate matrices belonging to components 1 and 2 may be deleted. This process is repeated for all components. The result is that fewer data blocks (less data) must remain active on the data base, which in turn places a lesser burden on the Buffer Pool; this, in turn, reduces I/O and database size. The net result is a performance improvement due to the following factors:

- Less buffer pool flushing (CPU time spent in buffer pooling)
- Less CPU time doing I/O
- Less elapsed time waiting for I/O
- Better memory utilization, especially when multiple DMP processes are contending for the same scratch file system (i.e. DMP running on a single host)

The end result of these changes is the 25% to 50% improvements for many models where the ACMS reduction is a significant factor in the overall analysis time.

Guidelines and Limitations

Optimal performance is achieved by setting *mem=max* on the command line. Note that *max* is the default value for *mem=* during installation; if this setting is not changed during the installation process, you do not need to set it on the command line. Additionally, *mem=max* can be set in a Resource Configuration File (RCF). Use of *mem=max* allows MSC Nastran to allocate memory and Buffer Pool automatically. For more information, see New Memory Management Strategy in this chapter.

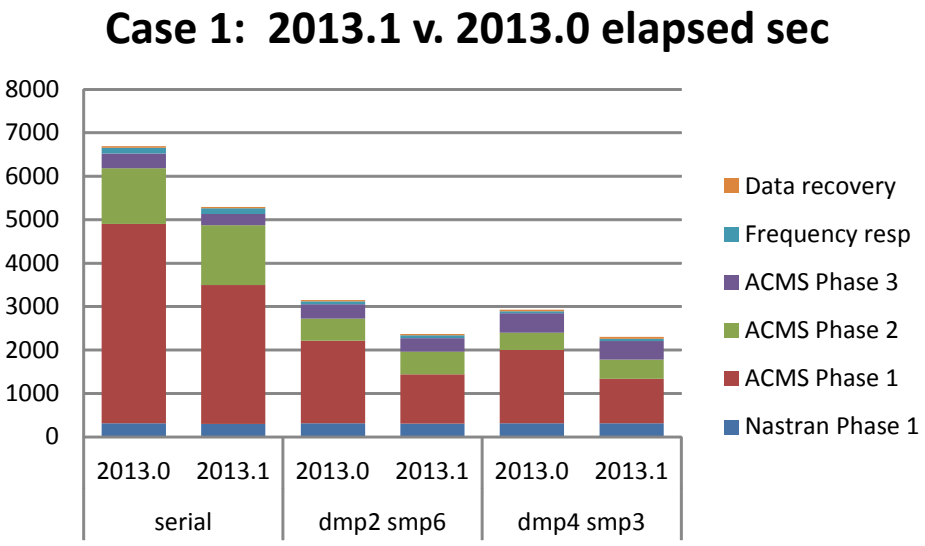
To improve performance, use SCR=YES if you have not plans on using restart.

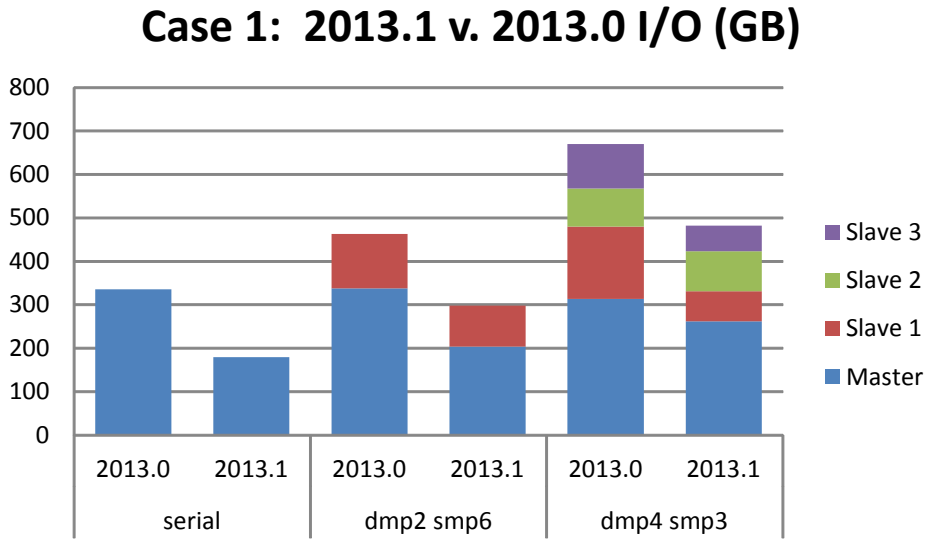
Test Cases

Test cases were run on a Linux machine with two 6-core Intel© Xeon© X5670 processors (2930 MHz). The machine has 96GB memory.

CASE 1: Interior Acoustics

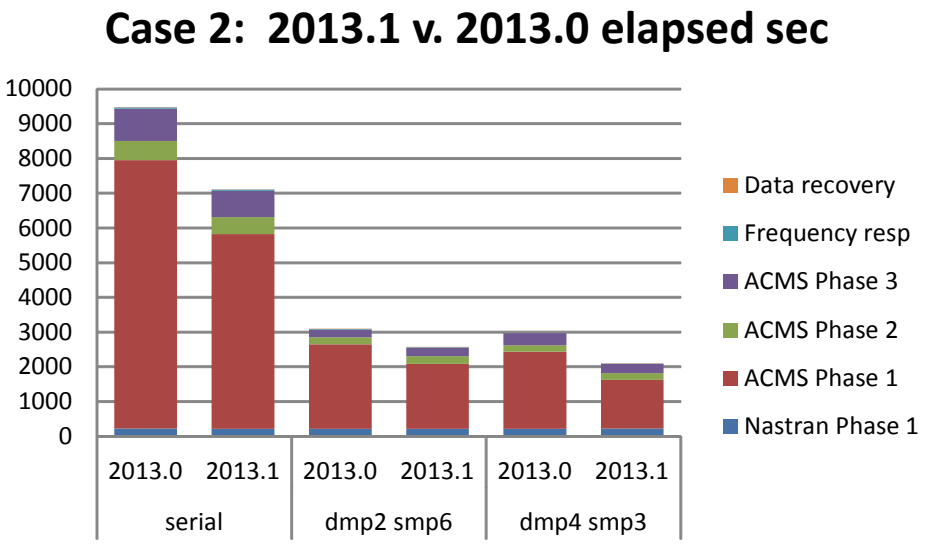
SOL	111
Number of grid points	1.9 Million
Number of global DOF	11.6 Million
Number of load cases	66
Number of eigenvectors required	5900
Number of forcing frequencies	450

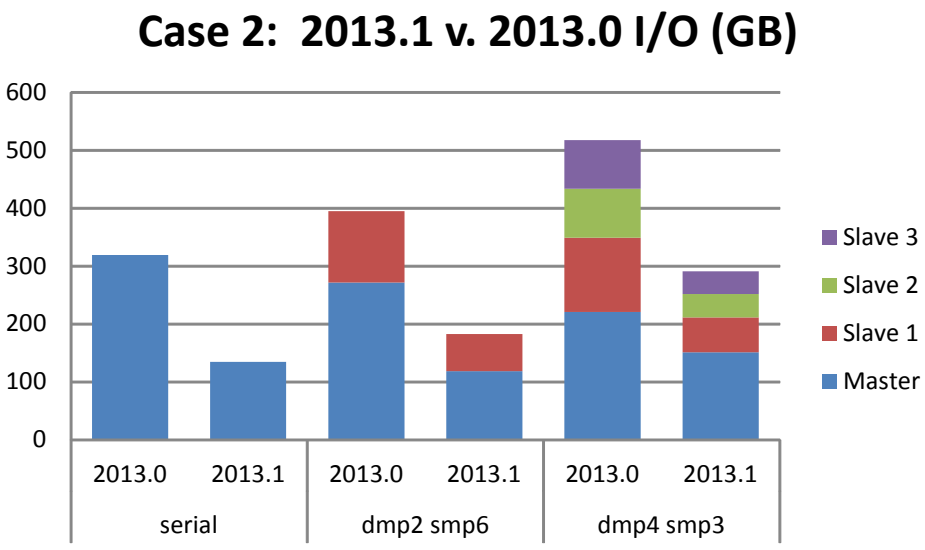




CASE 2: Structural NVH

SOL	111
Number of grid points	1.2 Million
Number of global DOF	7.3 Million
Number of load cases	6
Number of eigenvectors required	3620
Number of forcing frequencies	320





GPU Support

General Purpose computation on Graphics Processing Units (GPGPU) is a new trend in high performance computing and is having an increased presence in the FEA simulation market. For floating point intensive applications, the new computing paradigm can offer unrivalled performance and cost effectiveness. See http://gpgpu.org, for more information on GPGPU technology.

The GPU related feature in MSC Nastran was first released in 2012.1. The MSC Nastran GPU acceleration is delivered by a set of compute kernels for the symmetric (MSCLDL) and the nonsymmetric (MSCLU) sparse direct solvers, for NVIDIA CUDA-capable GPU cards.

Benefits

The performance advantages of GPU computing are most prominent in large sparse direct solvers in intensive SOL 101 and SOL 108 analyses. Other solution sequences with high sparse direct solver contents also may show significant improvements on GPU.

System Requirements

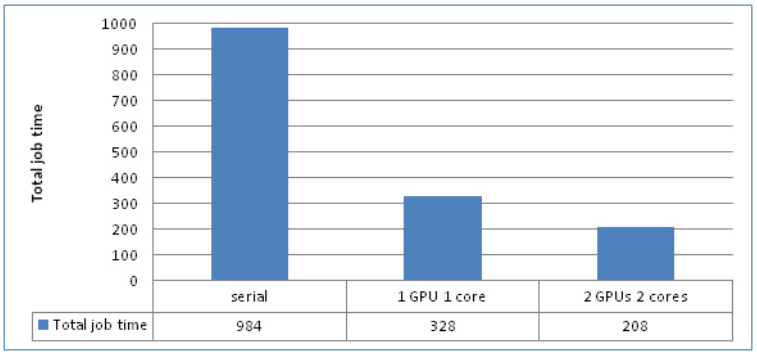
- Nvidia CUDA-capable GPGPU card(s) with at least 1.5GB on-board memory
- CUDA 5.0 drivers which consists of:
  - Nvidia Developer Drivers 304.54 (or later) for Linux.
  - Nvidia Developer Drivers 306.94 (or later) for Windows
- MSC Nastran GPGPU license

Technical Discussion

An interface has been developed for the default MSC Nastran sparse direct factorizations (MSCLDL and MSCLU) for NVIDIA GPU devices. There is no new functionality introduced. Utilization of GPU devices is for reduced run times only.

A good GPU kernel implementation overlaps compute on GPU, data transfer in the PCI-E bus, and compute on CPU, with multiple CUDA streams. To have enough floating point computations, such that these overlaps can occur, the

front size, i.e. NFRONT, has to be sufficiently large. In addition, to make the task more compute bound instead of PCI-E communication bound, the rank update size; i.e., NRANK also needs to be sufficiently large.



Therefore, improved performance only occurs for relatively large models. In particular, only matrices whose front sizes larger than a certain threshold benefit. To get good performance, the GPU capability works in conjunction with the new pivoting options, so that rank update sizes greater than 320 can be used effectively with MSCLDL and MSCLU. For details on the new pivoting options, please refer to the New Options for MSCLDL and MSCLU Sparse Direct Solvers section of this release guide.

Input

There is one run time parameter that controls GPU execution. The `gputhresh` keyword in the previous releases is deprecated.

`gpiud gpuid=id,id` or `gpuid=id:id Default: none`

id The ID of a licensed GPU device to be used in the analysis. Multiple IDs may be assigned to MSC Nastran distributed memory processor (DMP) runs. Separate a list of IDs with a comma or a colon. Each DMP process will be assigned a GPU ID in round robin fashion.

Note that for best performance, NONUPIV=3, or NONUPIV=1 for positive definite or diagonally dominant models, should be set, and the rank update size should be set to 320 or higher but no greater than 400. Again, see the New Options for MSCLDL and MSCLU Sparse Direct Solvers section for more details.

Examples

1. Select GPU ID number 1 for computation of your MSC Nastran analysis:
   `nastran myinput gpuid=1…`
2. Select GPU IDs 0 and 1 for your Distributed Memory Processing SOL108 analysis:
   `nastran myinput gpuid=0:1…`
3. Select GPU ID number 0 for linear static analysis of a positive definite model:
   `nastran myinput gpuid=0…`

Output

If a GPU device is successfully employed to run a MSC Nastran analysis job, a User Information Message 7840 will be printed in the F06 file:

```
*** USER INFORMATION MESSAGE 7840 (DFMRRD)
     MSCLDL SPARSE FACTORIZATION WILL BE RUN ON GPU DEVICE           1 IN
ADDITION TO CPU.
```

If a GPU ID is specified on the command line when there is no GPU device, System Fatal Message 7840 will be printed:

```
*** SYSTEM FATAL MESSAGE 7840 (DFMRRD)
     NASTRAN CANNOT FIND ANY CUDA-CAPABLE DEVICE ON THE SYSTEM.
     CONTACT THE APPROPRIATE HARDWARE SUPPORT/SYSTEM ADMIN.
```

The above error message also can be due to a mis configured software environment, for example if the Nvidia libraries cannot be found. Please see known issues, below.

Guidelines and Limitations

The GPU capability is limited to MSCLDL and MSCLU sparse factorizations.

Speedup is limited to medium to large sized models where the `ESTIMATED MAXIMUM FRONT SIZE` is greater than 10000 and 5000, for real and complex data types respectively.

`NONUPIV` should set to 3 or 1 if the model is positive definite or diagonally dominant.

For optimal performance, the system cells 205, 219, 220, 221 should be increased to 320 from the default value of 64.

The upper bound of the front size that can be processed by the MSC Nastran GPU kernel is 87250.

Known Issues

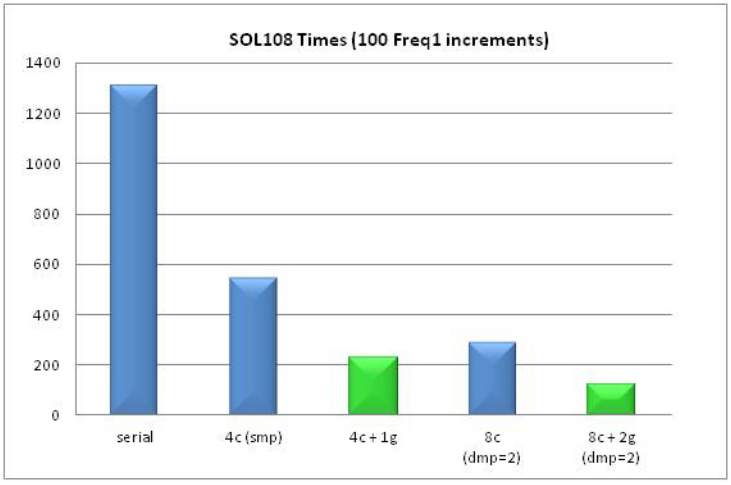
If a GPU card cannot be detected by MSC Nastran, change to the TCC (Tesla Compute Cluster) mode with the NVIDIA-SMI utility. Refer to the pertinent Nvidia documents for details.

Test Cases

In the following charts, `c` indicates CPU and `g` indicates GPU.

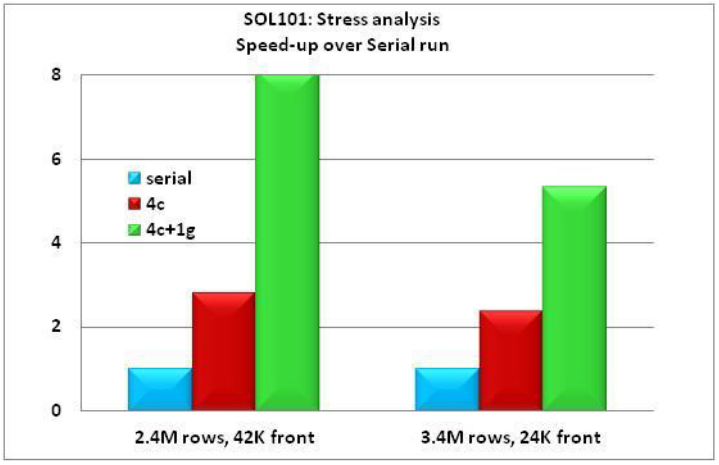
Case 1: Car Body NVH Analysis (sys653=3)

SOL:	108
Number of DOF:	710K
Max Front Size:	8939



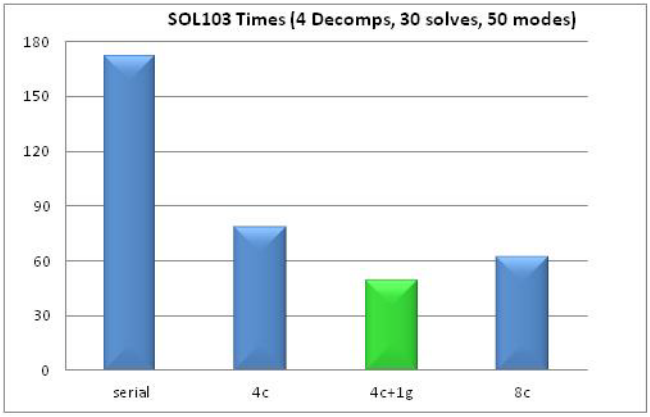
Case 2: Stress Analysis (sys653=3)

SOL:	101
Number of DOF:	2.4M and 3.4M
Max Front Size:	42K and 24K



Case 3: Normal Modes Analysis (sys653=1)

SOL:	103
Number of DOF:	2.6M
Max Front Size:	18K



New Options for MSCLDL and MSCLU Sparse Direct Solvers

The compute-intensive kernels in MSCLDL and MSCLU are now organized in three separate branches selectable by the MDLPRM,NONUPIV parameter, which is paired with system cell 653.

The three options - NONUPIV=0 (existing), 1(new), and 3(new) - offer different levels of computational efficiency, numeric accuracy, and hardware resource requirement. In general, NONUPIV=0 provides the most numerically stable solution and the least memory consumption, but also the lowest performance. For slightly more memory consumption, a positive definite, or diagonally dominant, model can be solved by NONUPIV=1; since this particular option does not do numeric pivoting, the performance is typically the best among these three options. The user is advised to select the one option that is most appropriate for the particular MSC Nastran analysis task.

Benefits

The user can select the most appropriate sparse direct solver method to analyze the particular model at hand to achieve maximum possible performance.

Technical Discussion

It is well known that, on cache-based micro-processors, only BLAS-3 like compute kernels can reach the theoretical floating point performance of the arithmetic units. The new options - NONUPIV=1 and NONUPIV=3, for MSCLDL and MSCLU utilize this concept, which leads to better computational efficiency and faster sparse solver turn-around times. However, it also has been challenging to maintain numeric stability and gain performance at the same time. Therefore, the NONUPIV=1 and NONUPIV=3 options assume only limited roles in this MSC Nastran release until further improvement.

A quick summary of the three options are as follows:

- NONUPIV=0 (default): segmented rank1 factorization and solve, segmented s/d/c/zgemm rankN update, thresholding 1x1 and 2x2 pivoting, small segment/rankN update size, least memory usage, most accurate, GPU enabled.
- NONUPIV=1(new): segmented rank-1 factorization, segmented s/d/c/ztrsm solve, segmented s/d/c/zgemm rankN update, no pivoting, large segment/rankN update size, slightly more memory usage. GPU enabled.
- NONUPIV=3 (new): s/d/c/zsytrf and s/d/c/zgetrf factorization, s/d/c/ztrsm solve, segmented s/d/c/zgemm rankN update, supernodal 1x1 and 2x2 pivoting, large segment/rankN update size, largest memory usage, GPU enabled.

The increase in memory usage by NONUPIV=1 over NONUPIV=0 would be (rank update size)*{2*(rank update size) + (max front size)}*(floating point storage unit size). That would result in a less than 5% increase in the sparse direct solver numeric phase memory usage, for typical jobs. The sparse direct solver memory usage increase from NONUPIV=0 to NONUPIV=3 can be as high as 30%. Set sys166=2 on the MSC Nastran submission command line to get sparse direct solver statistics in the F06 file. If insufficient open core is given, the job would abort with a fatal message requesting a memory increase.

Note that the total memory consumption in the numeric phase of sparse factorization is typically the sum of the MEMORY REQR'D TO AVOID SPILL from USER INFORMATION MESSAGE 4157 in F04 and the

ADDITIONAL CORE STRUCTURE IA in F06. The ADDITIONAL CORE STRUCTURE information is printed out only if sys166=2 is set. If a job doesn't do any row/column interchange with NONUPIV=0 or NONUPIV=3, then it is a candidate for NONUPIV=1.

There is one new run time parameter that controls the method selection. The selection also can be done with a NONUPIV keyword switch on the MSC Nastran submission command line or a MSC Nastran card in the input file.

NONUPIV	Parameter to select the numeric compute kernel and pivoting methods in MSCLDL and MSCLU sparse direct solvers.	
	0	Use the native Bunch-Kauffman threshold pivoting in MSCLDL, and the native threshold partial pivoting in MSCLU. (Default)
	1	Use no numeric pivoting in MSCLDL and MSCLU. BLAS3 TRSMs are called to compute the pivot column update to improve performance. Ill-conditioned models may terminate with "singular matrix" during sparse factorization.
	2	Not documented.
	3	LAPACK SYTRFs with Bunch-Kaufman pivoting and GETRFs with partial pivoting are called to perform factorizations, and BLAS3 TRSMs are called to compute pivot column update to improve performance. (Default)

Note that to maximize the performance potentials of NONUPIV=1 and NONUPIV=3, the appropriate rank update size; i.e., sys205 for real symmetric, sys219 for complex symmetric, sys220 for real unsymmetric, and sys221 for complex unsymmetric needs to be set to 320 or higher but no greater than 400. The default for sys205/219/220/221 is 64, which is adequate for NONUPIV=0 but not the other two options.

Examples

1. Select with NONUPIV switch on the submission command line:

   ```
   Nastran  myinput  NONUPIV=3 sys219=320 …
   ```

2. Select with MSC Nastran card in the input file:

   ```
   Nastran  system(653)=3,  system(219)=320
   ```

3. Select with parameter card: (sys219 sets separately)

   ```
   MDLPRM,NONUPIV,3
   ```

Output

User Fatal Message 7843 will be printed in the F06 if the rank update size for the sparse direct solver exceeds 400:

```
*** USER FATAL MESSAGE 7843 (DFMSA)
     MSCLDL SPARSE DIRECT SOLVER RANK UPDATE SIZE         420 EXCEEDS 400.
     USER ACTION: REDUCE SYS205/SYS219, AND RERUN.
```

Since NONUPIV=1 doesn't perform numeric pivoting, a job may terminate with the following User Fatal Message if an exact zero is encountered on the diagonal:

```
*** USER FATAL MESSAGE 6133 (DFMN)
    SINGULAR MATRIX IN SPARSE DECOMPOSITION AT ROW =            xxx
    USER ACTION: CHECK MODEL
```

Guidelines and Limitations

For the best performance, a sparse direct solver intensive SOL101 or SOL108 job should set `NONUPIV=3`, or `NONUPIV=1` if the model is positive definite or diagonally dominant.

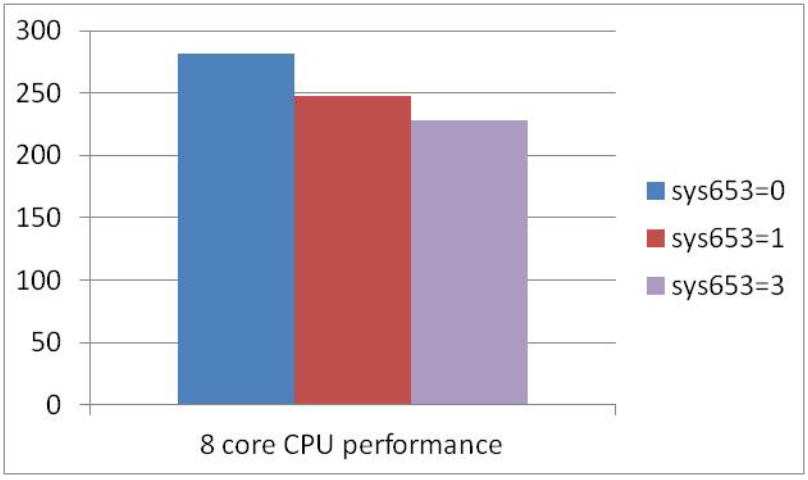
Known Issues

The pivoting method in `NONUPIV=3` is not as robust as that in `NONUPIV=1`. Therefore, for models that have large numbers of Lagrange multipliers, such as those encountered in SOL400 and SOL200, `NONUPIV=3` should be avoided. Generally speaking, only SOL101 and SOL108 jobs should use `NONUPIV=3` until future improvement.

Test Cases

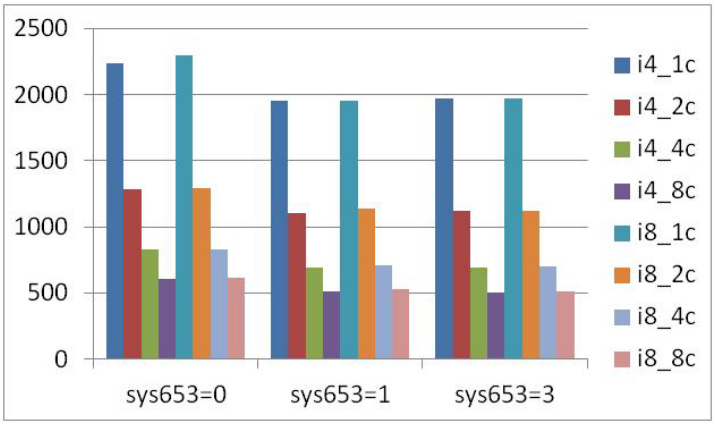
Case 1: Linear Static Analysis

SOL:	101
Number of DOF:	15263326



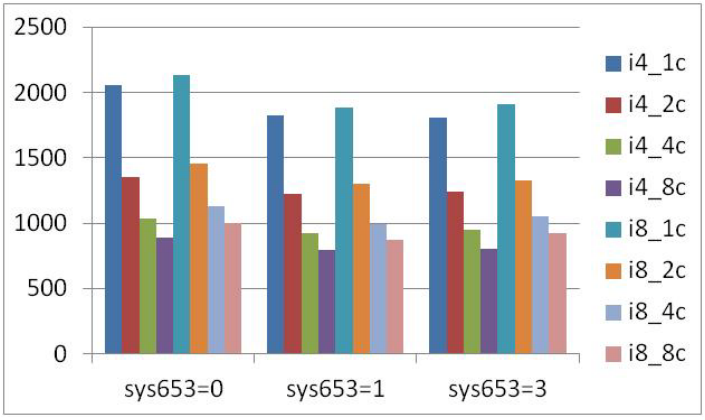
Case 2: Exterior Acoustics Analysis

SOL:	108 (4 frequencies)
Number of DOF:	164650



Case 3: Car Body NVH Analysis

SOL:	108 (2 frequencies)
Number of DOF:	710542



SOL 400 Parallel Performance Improvements

The MSC Nastran nonlinear analysis solution (SOL 400) is capable of running jobs in both distributed and shared memory parallel modes (DMP and SMP, respectively). Originally, the parallel portion of the analysis in DMP mode was limited to the solution of equations [A]{x}={b} to compute displacements during a nonlinear cycle. In version 2012.2, DMP parallel computation was extended to calculation of nonlinear element stiffness, stress, and forces. In the current 2013 version, these DMP capabilities have been improved by making them more efficient, and by removing significant limitations.

Benefits

Advanced nonlinear capabilities require significantly more processing time to compute nonlinear element stiffness, stress, and force quantities compared to the conventional nonlinear calculations available in MSC Nastran. The time required to perform these calculations may often be longer than the time of the numerical solution itself. Efficiency enhancements to MSC Nastran enable parallel scaling beyond the previous version. In addition, removal of key limitations such as support for stress-based separation allows broader application of DMP parallelism in MSC Nastran.

Technical Discussion

For conventional nonlinear analysis, large models typically spend a large majority of time computing a linear solution of equations. For these cases, achieving parallel speedup may be accomplished by simply parallelizing the linear equation solver.

However, more realistic modeling and analysis requirements have brought about new analysis capabilities that in turn require more computation in element stress, force, and stiffness calculations. Often, the advanced nonlinear calculations require more processing time than the linear solution of equations. These calculations are carried out by a sub-process of the main nonlinear solution module (NLSOLV) known as NLEMG. The NLEMG process is responsible for computing nonlinear forces, recovering intermediate element stresses, calculating element stiffness, and maintaining element summary table information.

Advanced nonlinear element specification is often specified by additional element property input data present in the bulk data, such as PSHLN1, PSHLN2, etc. Alternately, the `SPROPMAP` keyword specified on the NLMOPTS bulk data entry is used to convert all nonlinear elements to advanced nonlinear elements. For more information about advanced nonlinear capabilities available in SOL 400, please see the *MSC Nastran User's Guide*.

The NLEMG process was adapted for distributed parallel processing in MSC Nastran version 2012.2 (July 2012). Parallel scalability was effective for `DMP=2` (two distributed processes) and in some cases for four distributed processes (`DMP=4`). One significant limitation was the fact that MSC Nastran DMP was confined to "multiple master" (MULTIMST) mode. In this parallel run mode, all DMP processes execute the entire solution sequence; parallel processing and communication takes place where possible. Multiple master mode is most appropriate for execution on a distributed cluster with a high speed interconnect. For execution on a single host, multiple master mode places a heavy memory and I/O burden on the single host system.

For version 2013, the parallel NLEMG process is available in "master-slave" (MSTSLV) mode. In master-slave mode, only a single "master" DMP process executes the entire solution sequence. "Slave" DMP processes enter a wait state until signaled by the Master process. Parallel communication then takes place and parallel computations follow.

Master-slave mode places a smaller I/O burden on a single host system and thus is more suitable for single host execution. MSC Nastran automatically selects master-slave mode when running DMP on a single host.

Input

Changes to user input are confined to the activation of the master-slave run mode (see the Technical Discussion above). This option may be specified on the DOMAINSOLVER Executive command:

```
DOMAINSOLVER  NLSOLV ( RUNOPT=MSTSLV )
```

Manual specification of the parallel run mode is optional. By default, MSC Nastran detects the number of hosts used for the DMP job. If the number of hosts is one (i.e., single host execution), `RUNOPT` is set to `MSTSLV`. Otherwise, multiple-master mode is used (`RUNOPT=MULTIMST`). Previously, specification of master-slave mode would result in User Warning Message 530, and the DMP job would continue in multiple-master mode.

If no `DOMAINSOLVER` entry is specified, then a SOL 400 job submitted with `DMP>1` will automatically execute both the numerical solution of equations, and the element calculations, in DMP parallel mode (`MDSTAT` and `NLEMG`, respectively). These two computational tasks may be run in serial or DMP mode in any combination. See Table 6-1 for details.

Table 6-1 DOMAINSOLVER Options in SOL 400

DOMAINSOLVER Entry	[A]{x}={b}	NLEMG
DOMAINSOLVER STAT, NLSOLV	DMP	DMP
DOMAINSOLVER STAT	DMP	Serial
DOMAINSOLVER NLSOLV	Serial	DMP
No DOMAINSOLVER entry	DMP	DMP

If parallel NLEMG is active, but there are no advanced nonlinear elements in the model, the parallel NLEMG processing is automatically de-activated.

The DOMAINSOLVER entry is documented in Section 3, Executive Control Statements, of the *MSC Nastran Quick Reference Manual*.

Output

There are no new outputs.

Guidelines and Limitations

For MSC Nastran in general, and parallel SOL 400 in particular, the new default value for `MEM` (i.e., `mem=val` on the ‘nastran’ submit line) is `max` (i.e., `mem=max`), except on Win32 where the default *`mem=estimate`* is used. MSC Nastran will use a predefined amount of memory, and automatically allocates this memory for both numerical

calculations and for I/O caching. The I/O caching capability is done via enhanced “buffer pooling” that was introduced in MSC Nastran 2012.2.

For more information on `mem=max` and automatic memory allocation, see the section entitled New Memory Management Strategy elsewhere in this document.

Limitations

The VCCT capability in SOL 400 is not supported for DMP parallel `NLEMG`. If attempted, a Fatal Error message will be generated, and the MSC Nastran job will terminate.

Test Cases

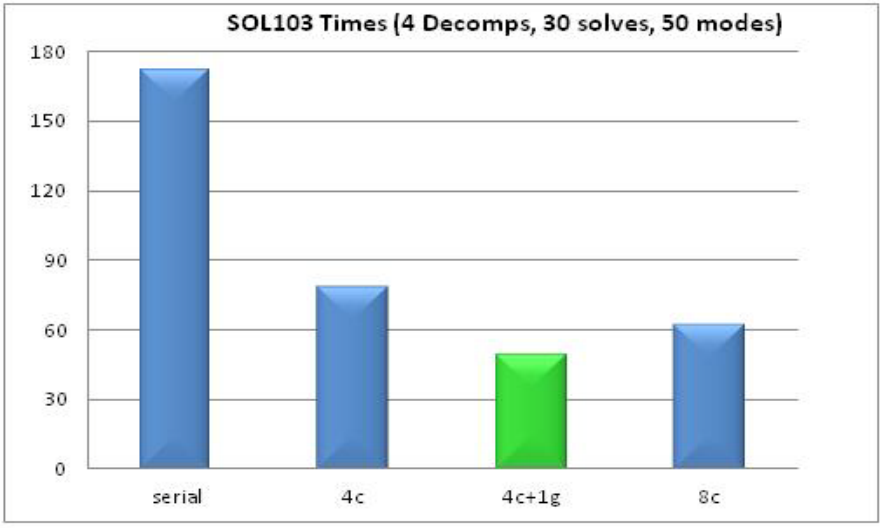
Test cases come from customers. The model data may not be transmitted outside MSC.

Hardware used in the examples below is 3470MHz Intel Westmere CPUs, 48GB main memory, Red Hat Enterprise Linux Server release 5.4.

Charts show elapsed time performance comparing MSC Nastran version 2012.2 to version 2013. For version 2013, note that no memory or buffer pool parameters were explicitly set. Memory allocation was determined automatically via `MEM=MAX`. Memory and buffer pool parameters were explicitly set for version 2012.2.

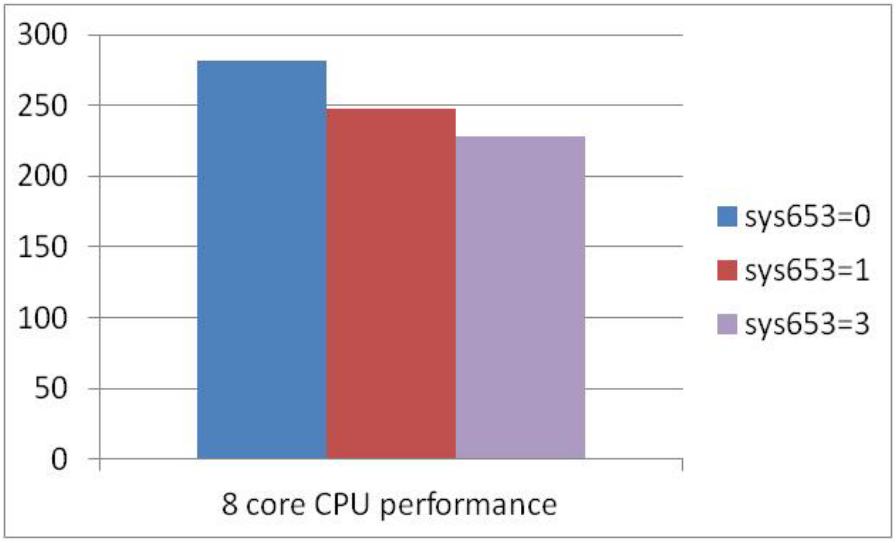
Case 1

SOL	400
Number of grid points:	118,145
G-size DOF:	824,463
Number of load steps:	3
Total number of nonlinear cycles:	210



Case 2

SOL	400
Number of grid points:	101,053
G-size DOF:	644,256
Number of load steps:	1
Total number of nonlinear cycles:	57



New Memory Management Strategy

Introduction

MSC Nastran memory management is geared toward minimizing memory use in order to conserve resources in a multi-user environment. Numerical methods employed in MSC Nastran allow for out-of-memory algorithms (or "spill logic") enabling MSC Nastran to solve problems of virtually unlimited size.

As large memory systems become common, the MSC Nastran memory management strategy has evolved in an effort to improve overall throughput by automatically allocating a significant portion of memory to the MSC Nastran buffer pool (see "Improved I/O Performance via Buffer Pooling" in the *MSC Nastran 2012.2 Release Guide*, Chapter Six). Thus, in addition to fulfilling the original goals of problem size scalability, MSC Nastran version 2013 will intelligently allocate sufficient memory to minimize disk I/O traffic.

Benefits

Users no longer need to set memory and buffer pool parameters. MSC Nastran automatically calculates memory lengths and allocates the memory resources accordingly. For most applications, this will result in less disk I/O, more efficient CPU utilization, and faster elapsed times for MSC Nastran jobs.

Performance penalties for under-estimating memory requirements for MSC Nastran can sometimes prove prohibitive for large analysis jobs. The new memory management strategy is based on machine resources and may be customized to meet any general user scenario.

Technical Discussion

By default, MSC Nastran will occupy memory up to a size specified as a percentage of the real (physical) memory on the system. This is accomplished in one of two ways:

1. Set `mem=max` on the 'nastran' command line.
2. Do not set `mem=` on the 'nastran' command line at all. The default MEM specification (set during installation) is `mem=max`.

The amount of memory specified by `mem=max` depends on the `memorymaximum` keyword. For version 2013, the default `memorymaximum` is "0.5xphysical" which means that 50% of the physical memory on the machine will be used for a MSC Nastran job.

Buffer pooling is achieved by dedicating a portion of `MEM` to buffer pool activity (`BPOOL`). The default amount of memory allocated to `BPOOL` is 25% of `MEM`. This is equivalent to setting `bpool=25x`, where `x` signifies a percentage of memory. An exception to this process occurs when running SOL 101 or SOL 400. In these cases, `mem=max` invokes the *estimate* program, which determines the `BPOOL` size.

For Distributed Memory Parallel (DMP) jobs, the memory length set by `memorymaximum` is divided by the number of DMP processes, and `MEM` is set accordingly. This guarantees that the `memorymaximum` is never exceeded.

Note that for i4 mode, MEM is limited to 8GB, so that a single MSC Nastran process is limited to using 8GB of memory. (On Windows, the limit of MEM is slightly less than 8GB, due to operating system restrictions.)

For example, on an 8GB system, consider the following 'nastran' command line:

```
nastran myjob mem=max memorymax=0.5xphysical bpool=25x
```

Memory allocation for this job would be as follows:

- MEM will be set to 4GB; total memory used will not exceed 4GB
- BPOOL will be set to 1GB

On a 32GB system, consider the following:

```
nastran myjob mem=max memorymax=0.5xphysical bpool=25x mode=i8
```

Memory will be allocated as follows:

- MEM will be set to 16GB (this is possible via mode=i8); total memory used will not exceed 16GB
- BPOOL will be set to 4GB

Defaults may be set in MSC Nastran RC files. Program defaults may be duplicated in the system-wide RC file by including these commands:

```
memorymax=0.5xphysical
bpool=25x
```

These may be changed via RC files at various levels (system-wide, user-specific) and/or on the 'nastran' submit line.

More examples are shown below.

Note: On Win32 the default is "*mem=estimate*" and that is the best choice for this operating system. The rest of this section is not applicable to Win32.

Input

There are essentially three changes to MSC Nastran input parameters that encompass the new memory management strategy.

1. The default memory value is changed from mem=estimate to mem=max. Note that the functionality of mem=estimate is not changed. Note also that for SOL 101 and SOL 400, mem=max invokes estimate in order to set BPOOL length.
2. Specifying mem=max no longer uses estimate to set SMEM; it uses estimate to set BPOOL instead.
3. Memory for buffer pooling (BPOOL) may now be set as a percentage value. The percentage is taken from MEM. Specify BPOOL in one of three ways:
   a. bpool=N specifies number of GINO blocks for buffer pooling
   b. bpool=size specifies a memory size for buffer pooling
   c. bpool=Nx specifies a percentage of MEM to be used for buffer pooling (new for version 2013)

Reverting to the previous MSC Nastran memory management strategies is possible by setting appropriate values on the ‘nastran’ command line or in RC files.

Guidelines and Limitations

Allocating 100% of a system’s physical memory for MSC Nastran is not generally recommended.

When running in i4 mode, the maximum memory possible is 8GB per MSC Nastran process. See the Examples section below for illustration.

When running on a single user system, and running one MSC Nastran job at a time, you can use more physical memory by setting *memorymax=0.8xphysical*. This is generally a safe maximum.

Multiple-user environments (i.e., systems where multiple MSC Nastran jobs may be run simultaneously) require more caution. Set the `memorymaximum` parameter so that memory is not over-subscribed. For example, if there is a maximum of two simultaneous MSC Nastran jobs, set `memorymaximum` to “0.4xphysical” so that an 80% general threshold is maintained.

Buffer Pool performance and I/O savings are marginally reduced when *scr=no* is specified for data base restart.

Examples

Below are examples of memory specified with new defaults. Defaults can be easily changed depending on user and resource requirements.

Default memory settings used in the examples are:

```
memorymax=0.5xphysical
bpool=25x
```

Memory settings are shown assuming analysis other than SOL 101 and SOL 400. For SOL 101 and SOL 400, BPOOL is set via the `estimate` program.

Example 1: Running MSC Nastran on an 8GB system

MSC Nastran Command	MEM (mb)	BPOOL (mb)
nastran myjob	4096	1024
nastran myjob mode=i8	4096	1024
nastran myjob dmp=2	2048	512
nastran myjob dmp=2 mode=i8	2048	512
nastran myjob dmp=4	1024	256
nastran myjob dmp=4 mode=i8	1024	256

Example 2: Running MSC Nastran on an 16GB system

```
memorymax=0.5xphysical
```

```
bpool=25x
```

MSC Nastran Command	MEM (mb)	BPOOL (mb)
nastran myjob	8192	2048
nastran myjob mode=i8	8192	2048
nastran myjob dmp=2	4096	1024
nastran myjob dmp=2 mode=i8	4096	1024
nastran myjob dmp=4	2048	512
nastran myjob dmp=4 mode=i8	2048	512

Example 3: Running MSC Nastran on an 32GB system

```
memorymax=0.5xphysical
bpool=25x
```

MSC Nastran Command	MEM (mb)	BPOOL (mb)
nastran myjob	8192	2048
nastran myjob mode=i8	16384	4096
nastran myjob dmp=2	8192	2048
nastran myjob dmp=2 mode=i8	8192	2048
nastran myjob dmp=4	4096	1024
nastran myjob dmp=4 mode=i8	4096	1024

Example 4: Running MSC Nastran on an 48GB system

```
memorymax=0.5xphysical
bpool=25x
```

MSC Nastran Command	MEM (mb)	BPOOL (mb)
nastran myjob	8192	2048
nastran myjob mode=i8	24576	6144
nastran myjob dmp=2	8192	2048
nastran myjob dmp=2 mode=i8	12288	3072
nastran myjob dmp=4	6144	1536
nastran myjob dmp=4 mode=i8	6144	1536

Example 5: Running MSC Nastran on an 64GB system

```
memorymax=0.5xphysical
bpool=25x
```

MSC Nastran Command	MEM (mb)	BPOOL (mb)
nastran myjob	8192	2048
nastran myjob mode=i8	32768	8192

nastran myjob dmp=2	8192	2048
nastran myjob dmp=2 mode=i8	16384	4096
nastran myjob dmp=4	8192	2048
nastran myjob dmp=4 mode=i8	8192	2048

Example 6: Running MSC Nastran on an 96GB system

```
memorymax=0.5xphysical
bpool=25x
```

MSC Nastran Command	MEM (mb)	BPOOL (mb)
nastran myjob	8192	2048
nastran myjob mode=i8	49152	12288
nastran myjob dmp=2	8192	2048
nastran myjob dmp=2 mode=i8	24576	6144
nastran myjob dmp=4	8192	2048
nastran myjob dmp=4 mode=i8	12288	3072

7 Optimization

Fatigue Life Design Responses

Introduction

Optimize based on fatigue life, damage, or safety factor responses.

Benefits

It is not about the stress! The real question is how long will it last? With the addition of fatigue life calculations in standard SOL 101 analyses, it is now possible to design and optimize structures and components for ANALYSIS=STATICS based on fatigue life. See Chapter 2 Linear Analysis for a discussion of the new FATIGUE output request case control and associated bulk data.

Feature Description

This is a simple but significant enhancement to SOL 200. With the ability to define fatigue life or damage responses, design objectives and design constraints can be defined with respect to fatigue life or damage. It is possible to set the design objective to maximize fatigue life (or minimize damage), or more commonly, to set a fatigue life constraint with an objective of weight minimization.

Overview of Bulk Data

A new response type can be defined in the RTYPE field of the DRESP1 bulk data entry called FATIGUE. The PTYPE field specifies whether elements or property sets are specified in the ATTi fields. If PTYPE is left blank, all the elements specified by the fatigue analysis called out in the ATTB field are used (not recommended). The ATTA field is used to specify the fatigue response item code of interest. These can be life, damage or scale factor from a factor of safety analysis. The life value can be defined in either repeats of the loading sequence or by specifying life in the user defined fatigue equivalent units as specified on the FTGSEQ bulk data entry. The following shows a fatigue life design response in fatigue equivalent units (fatigue item code 6 in ATTA field) on element IDs 1 through 5 (ATTi fields) only of FATGUE ID 44 (ATTB field).

```
Bulk Data
DRESP1, ID  , LABEL  , RTYPE  , PTYPE, REGION, ATTA, ATTB, ATT1
      , ATT2, etc.

DRESP1, 11  , Flights, FATIGUE, ELEM ,       , 6,  , 44  , 1
      , 2   , 3      , 4      , 5
```

Example 1:

This example shows a SOL 200 optimization run of the duty cycle example in Chapter 2. Everything is the same for defining the actual fatigue analysis/output request. The design objective is set to minimize weight, and to ensure that a specific fatigue life is obtained in a particular element of the structure.

Case Control

```
SOL 200
TITLE Optimization - Simple Test Track Duty Cycle
$
DESOBJ(MIN) = 15
DESGLB      = 16
$
FATIGUE = 44
$
ANALYSIS = STATICS
$
SUBCASE 1
   SUBTITLE Unit load on front right
...
SUBCASE 2
   SUBTITLE Unit load on front left
...
SUBCASE 3
   SUBTITLE Unit load on rear right
...
SUBCASE 4
   SUBTITLE Unit load on rear left
...
```

Bulk Data

```
$Design Response - used as objective
DRESP1,15, W, WEIGHT
$
$Design Response on element 1 only - used as constraint
DRESP1, 99, Laps, FATIGUE, ELEM, , 6, 44, 1
$
$Design Constraint (500 Laps) - applied to fatigue response 99
DCONSTR, 16, 99, 500
$
PSHELL, 66, 1, ...
MAT1,  1, 203403.0, 78231.7, 0.3, 1.0
$
$SN curve specifically defined
MATFTG,1
      ,STATIC, , 600.0
       61
$
$Select elements of property 66 only with polished surface finish
SET4   , 1, PROP, PSHELL, 66
FTGDEF, 44
      , ELSET, 1, 35
PFTG   , 35, 0, POLISH
$
$Specify an S-N analysis with Goodman mean stress correction
FTGPARM, 44, SN
       ,STRESS, ,GOODMAN
$
$ Duty Cycle - 3 cobble stones
$              2 pot holes
```

```
$              1 bumps
$              6 cornering and braking
FTGSEQ, 44
      , 21, 3.0, 22, 2.0, 23, 1.0, 24, 6.0
      ,UNITS, 1.0, Laps
$
$Cobble Stone event
FTGEVNT, 21, 101, 102, 103, 104
$
$Pot Hole event
FTGEVNT, 22, 201, 202, 203, 204
$
$Bumps event
FTGEVNT, 23, 301, 302, 303, 304
$
$Cornering and Braking event
FTGEVNT, 24, 401, 402, 403, 404
$
$Load association for Cobble Stone event
FTGLOAD, 101, 111, 1
FTGLOAD, 102, 112, 2
FTGLOAD, 103, 113, 3
FTGLOAD, 104, 114, 4
$
$Load association for Pot Hole event
FTGLOAD, 201, 211, 1
FTGLOAD, 202, 212, 2
FTGLOAD, 203, 213, 3
FTGLOAD, 204, 214, 4
$
$Load association for Bumps event
FTGLOAD, 301, 311, 1
FTGLOAD, 302, 312, 2
FTGLOAD, 303, 313, 3
FTGLOAD, 304, 314, 4
$
$Load association for Cornering and Braking event
FTGLOAD, 401, 411, 1
FTGLOAD, 402, 412, 2
FTGLOAD, 403, 413, 3
FTGLOAD, 404, 414, 4
$
$Tables defining load variations for each load of each event
TABLFTG, 111 ...
TABLFTG, 112 ...
TABLFTG, 113 ...
TABLFTG, 114 ...
TABLFTG, 211 ...
TABLFTG, 212 ...
TABLFTG, 213 ...
TABLFTG, 214 ...
TABLFTG, 311 ...
TABLFTG, 312 ...
TABLFTG, 313 ...
```

```
TABLFTG, 314 ...
TABLFTG, 411 ...
TABLFTG, 412 ...
TABLFTG, 413 ...
TABLFTG, 414 ...
```

Documentation Dependencies

Please see the *MSC Nastran Fatigue Analysis User's Guide* for detailed examples of how to use these new features and the *MSC Nastran Quick Reference Guide* for details on each case control and bulk data entry to control fatigue analysis.

The Equivalent Radiated Power (ERP) Design Responses

In automotive applications, the noise inside the passenger compartment can be caused by many sources including vibrating body panels. The Equivalent Radiated Power (ERP) calculation focuses on the vibration of body panels, which radiate acoustic power to the passenger cabin. Understanding which panels are responsible for the radiated power is important in understanding the structural behavior and acoustic consequences. ERP analysis was initially introduced in MD Nastran R3.1 (MD Nastran 2010 Release Guide or MD Nastran R3.1 Release Guide). MSC Nastran 2013 adds the capability to access ERP results in SOL 200 as a design response that can be applied as an objective or design constraint in an optimization task.

Benefit

The ERP sensitivity calculation can be used to understand which parameters are the primary contributors to the radiated power. Engineers can use the **ERP** sensitivities and optimization to improve design.

Theory

As a brief overview, ERP squares the normal velocity and multiplies it by the element area . The sum this product over all the elements of a user defined panel, multiplied with a constant, yields the ERP over a panel. A detailed explanation of the ERP calculation can be found in the MD Nastran 2010 Release Guide. ERP sensitivities are computed by a direct design sensitivity method (MSC Nastran Design Optimization User's Guide).

Input

A new design response ERP is added to DRESP1 bulk data entry as below

Response	Response Attributes		
Type (RTYPE)	ATTA (Integer>0)	ATTB	ATTi
ERP (see Remark)	ERP Item Code	real (for freq value) or characters for function name See QRG Remarks 15 and 20	Blank or SET3 ID≥0

Remarks:

1. `DRESP1 =ERP's PTYPE` field is extended to include `PTYPE`= "`ERPPNL`" (Equivalent Radiated Power Definition bulk data entry)
2. `ATTA` - ERP output code (item code). Valid numbers are 2 to 4 where 2 – ERP value, 3-ERP fraction, and 4-ERP(DB)
3. `ATTB` - real (for frequency value) or characters for function name; i.e. `MAX` or `AVG`.
4. `ATTi` – Blank or Integer≥0. Blank or SET3 ID (default=Blank=0 is ALLPANEL)

Example Input

Case Control

```
ERP(PUNCH,Filter=0.0,rhocp=2.0E9,ERPRHO=1.189E-12,ERPC=3.43E5)=ALL
DSAPRT(FORMATTED, END=SENS) = ALL (for sensitivity computation only)
```

Example ERP Panel Definition Bulk Data

```
ERPPNL,ROOF1,103,ROOF2,203,ROOF3,303
SET3,103,PROP,100
SET3,203,PROP,200
SET3,303,ELEMENT,114,124,134,214,224,234,
,314,324,334
```

Example ERP Design Response Definition Bulk Data

```
DRESP1,700,ERP1,ERP ,ERPPNL,,2,,103
DRESP1,710,ERP2,ERP ,ERPPNL,,3,,203
DRESP1,720,ERP2,ERP ,ERPPNL,,4,,303
DRESP1,730,ALLPANEL,ERP ,ERPPNL,,2,,
```

Output

If P2=8 (or sum of 8) is present on the DOPTPRM bulk data entry, the output file `*.f06` has the ERP design response prints such as Listing 10-1. The ERP sensitivity formatted prints are shown in Listing 10-2.

Guidelines and Limitations

- ERP, ERP sensitivity, and optimization are calculated currently for linear 3- and 4-node shell elements only. If desired, the user can generate a layer of linear shells on top of quadratic solids.
- PSHELL and PCOMP are supported
- ERP, ERP sensitivity, and optimization are supported in direct and modal frequency response only.
- ERP sensitivity is not supported by the adjoint method. Thus, it will be very time consuming for problems that have many design variables (for example, topology, topometry, and topography optimization) since a direct method is used for ERP sensitivity computation.

Test Cases

The following test cases are available in the TPL in directory `/tpl/erpopt`:

`erp_opt_usecase.dat`, `erpopt0.dat`, `erpopt1.dat`, `erpopt2.dat`, `erpopt3.dat`, `erpopt4.dat`, `erpopt5.dat`, `erpopt6.dat`, `erpopt7.dat`, `erpopt8.dat`

TPL Example Problem `erpopt1.dat`

Test problem `erpopt1.dat` is a simple fluid bound by two panels (based on ERP analysis TPL example problem erp_base1.dat, (see *MD Nastran 2010 Release Guide,* 2010). The excitation is on one panel as shown in Figure 7-1.

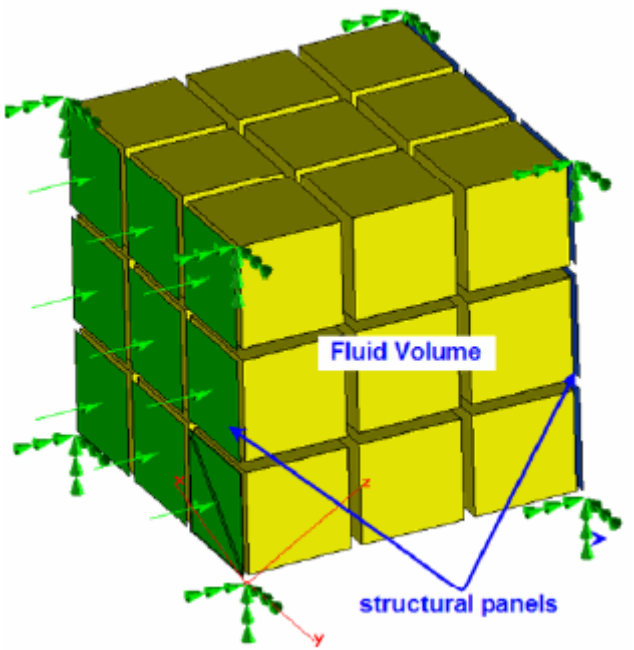


Figure 7-1 Example `erpopt1.dat` Geometry

The input for `erpopt1.dat` is a standard modal frequency response with a pressure loading and including fluid-structure interaction. The case control and bulk data entries required for ERP and sensitivity calculation are as follows:

Case Control

```
ERP(PRINT,PUNCH,FILTER=0.0)=ALL
DSAPRT(FORMATTED, END=SENS) = ALL
```

Example ERP Panel Definition Bulk Data

```
ERPPNL,ERPX0,103,ERPX3,203,erpeid3,303
set3,103,prop,100
set3,203,prop,200
set3,303,element,114,124,134,214,224,234,
,314,324,334
```

Example ERP Design Response Definition Bulk Data

```
DRESP1,700,ERP1,ERP ,ERPPNL,,2,,103
DRESP1,710,ERP2,ERP ,ERPPNL,,2,,203
DRESP1,720,ERP2,ERP ,ERPPNL,,2,,303
DRESP1,730,ALLPANEL,ERP ,ERPPNL,,2,,
```

Example ERP Design Constraint Definition Bulk Data

```
DCONSTR,100,700,,10.0
DCONSTR,100,710,,10.0
DCONSTR,100,720,,10.0
DCONSTR,100,730,,30.0
```

Specify Design Variables, Relate Linearly to PSHELL Thickness

```
DESVAR, 1,       T1,      .01,     .0001,     1.0
DESVAR, 2,       T2,      .01,     .0001,     1.0
DVPREL1 101      PSHELL   100      4        .01                                  +00
+00      1       1.0
DVPREL1 102      PSHELL   200      4        .01                                  +01
+01      2       1.0
```

Optimization Control Parameter Definition Bulk Data

```
DOPTPRM, P1, 1, P2, 8, DESMAX, 20
```

The design task for `erpopt1.dat` is to minimize the structural weight with constraints on ERP and frequency velocity responses. There are two PSHELL thickness design variables defined by `DESVAR` and `DVPREL1` entries. ERP design responses are printed in `erpopt1.f06` shown in Listing 7-1.

Listing 7-1 TPL Example ERPOPT1 ERP Design Response Print

```
                                    -----    WEIGHT RESPONSE    -----

---------------------------------------------------------------------------------------------------------------------------------
    INTERNAL    DRESP1    RESPONSE      ROW       COLUMN    LOWER                                  UPPER
       ID         ID        LABEL        ID          ID        ID          BOUND          VALUE        BOUND
---------------------------------------------------------------------------------------------------------------------------------
        1        200       WEIGHT         3           3                   N/A        1.9400E-03        N/A

                          I N I T I A L   A N A L Y S I S    S U B C A S E =    1000

                               ----  EQUIVALENT RADIATED POWER RESPONSES ---

---------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE      SET3    ITEM CODE                              LOWER                  UPPER
      ID        ID       LABEL        ID          NO.          FREQUENCY      BOUND    VALUE      BOUND
---------------------------------------------------------------------------------------------------------------------------------
        2       700      ERP103       103           2   1.0000E+01          N/A         6.1034E+00  1.0000E+01

                               -----    FREQUENCY VELOCITY RESPONSES    -----

---------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE    GRID    COMPONENT                              LOWER                  UPPER
      ID        ID       LABEL      ID        NO.          FREQUENCY      BOUND    VALUE      BOUND
---------------------------------------------------------------------------------------------------------------------------------
        3       612  G608          121         1     1.4000E+01       N/A          2.8299E+00    5.0000E+00
        4       613  G805          131         1     1.4000E+01       N/A          5.2953E+00    5.0000E+00 V

                               ----  EQUIVALENT RADIATED POWER RESPONSES ---

---------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE      SET3    ITEM CODE                              LOWER                  UPPER
      ID        ID       LABEL        ID          NO.          FREQUENCY      BOUND    VALUE      BOUND
---------------------------------------------------------------------------------------------------------------------------------
        5       700  ERP103        103         2     1.4000E+01       N/A          5.9479E+01    1.0000E+01 V
        6       710  ERP203        203         2     1.4000E+01       N/A          7.7890E+01    1.0000E+01 V
        7       720  ERP303        303         2     1.4000E+01       N/A          7.7890E+01    1.0000E+01 V
        8       730  ERP303          0           2     1.4000E+01       N/A          1.3737E+02    3.0000E+01 V

                          I N I T I A L   A N A L Y S I S    S U B C A S E =    2000

                               ----  EQUIVALENT RADIATED POWER RESPONSES ---

---------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE      SET3    ITEM CODE                              LOWER                  UPPER
      ID        ID       LABEL        ID          NO.          FREQUENCY      BOUND    VALUE      BOUND
---------------------------------------------------------------------------------------------------------------------------------
        9       700  ERP103        103         2     1.0000E+01       N/A          6.1034E+00    1.0000E+01

                               -----    FREQUENCY VELOCITY RESPONSES    -----

---------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE    GRID    COMPONENT                              LOWER                  UPPER
      ID        ID       LABEL      ID        NO.          FREQUENCY      BOUND    VALUE      BOUND
---------------------------------------------------------------------------------------------------------------------------------
       10       612  G608          121         1     1.4000E+01       N/A          2.8299E+00    5.0000E+00
       11       613  G805          131         1     1.4000E+01       N/A          5.2953E+00    5.0000E+00 V

                               ----  EQUIVALENT RADIATED POWER RESPONSES ---

---------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE      SET3    ITEM CODE                              LOWER                  UPPER
      ID        ID       LABEL        ID          NO.          FREQUENCY      BOUND    VALUE      BOUND
---------------------------------------------------------------------------------------------------------------------------------
       12       700  ERP103        103         2     1.4000E+01       N/A          5.9479E+01    1.0000E+01 V
       13       710  ERP203        203         2     1.4000E+01       N/A          7.7890E+01    1.0000E+01 V
       14       720  ERP303        303         2     1.4000E+01       N/A          7.7890E+01    1.0000E+01 V
       15       730  ERP303          0           2     1.4000E+01       N/A          1.3737E+02    3.0000E+01 V
```

The Case Control Command DSAPRT(FORMATTED, END=SENS) = ALL will result in sensitivity computation only and the sensitivity coefficients are presented with headings and labels as shown in Listing 7-2.

Listing 7-2 TPL Example ERPOPT1 ERP Design Sensitivity Print

```
                                    *************************************************************************
                                    *                                                                       *
                                    *   D E S I G N    S E N S I T I V I T Y    M A T R I X    O U T P U T    *
                                    *                                                                       *
                                    *   R E S P O N S E    S E N S I T I V I T Y    C O E F F I C I E N T S   *
                                    *                                                                       *
                                    *************************************************************************

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      200             RESPONSE TYPE= WEIGHT                                                           SEID=        0
       RESP VALUE                DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
        1.9400E-03                    1 T1         1.0002E-01        2  T2         9.0022E-02

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      612             RESPONSE TYPE= FRVELO       GRID ID=      121     COMP NO=        1       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  2.8299E+00  1.4000E+01        1 T1        -9.9617E+03        2  T2        -5.0551E+03
   2000  2.8299E+00  1.4000E+01        1 T1        -9.9617E+03        2  T2        -5.0551E+03

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      613             RESPONSE TYPE= FRVELO       GRID ID=      131     COMP NO=        1       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  5.2953E+00  1.4000E+01        1 T1        -1.4141E+04        2  T2        -1.3616E+04
   2000  5.2953E+00  1.4000E+01        1 T1        -1.4141E+04        2  T2        -1.3616E+04

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      700             RESPONSE TYPE= ERP          SET3 ID=      103     COMP NO=        2       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  6.1034E+00  1.0000E+01        1 T1        -6.8398E+03        2  T2        -1.4668E+03

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      700             RESPONSE TYPE= ERP          SET3 ID=      103     COMP NO=        2       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  5.9479E+01  1.4000E+01        1 T1        -1.8879E+05        2  T2        -2.8652E+04
   2000  6.1034E+00  1.0000E+01        1 T1        -6.8398E+03        2  T2        -1.4668E+03
   2000  5.9479E+01  1.4000E+01        1 T1        -1.8879E+05        2  T2        -2.8652E+04

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      710             RESPONSE TYPE= ERP          SET3 ID=      203     COMP NO=        2       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  7.7890E+01  1.4000E+01        1 T1        -1.3324E+05        2  T2        -2.7518E+05
   2000  7.7890E+01  1.4000E+01        1 T1        -1.3324E+05        2  T2        -2.7518E+05

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      720             RESPONSE TYPE= ERP          SET3 ID=      303     COMP NO=        2       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  7.7890E+01  1.4000E+01        1 T1        -1.3324E+05        2  T2        -2.7518E+05
   2000  7.7890E+01  1.4000E+01        1 T1        -1.3324E+05        2  T2        -2.7518E+05

-------------------------------------------------------------------------------------------------------------------------------
DRESP1 ID=      730             RESPONSE TYPE= ERP          SET3 ID=        0     COMP NO=        2       SEID=        0
SUBCASE  RESP VALUE  FREQ/TIME   DESIGN  VARIABLE  COEFFICIENT   DESIGN  VARIABLE  COEFFICIENT
-------------------------------------------------------------------------------------------------------------------------------
   1000  1.3737E+02  1.4000E+01        1 T1        -3.2202E+05        2  T2        -3.0383E+05
   2000  1.3737E+02  1.4000E+01        1 T1        -3.2202E+05        2  T2        -3.0383E+05
```

To run ERP optimization, simply remove the Case Control Command DSAPRT(FORMATTED, END=SENS) = ALL in erpopt1.dat (i.e., this is erpopt8.dat) and rerun the job. The optimization results are presented in Listing 7-3. It is seen that there is a significant reduction in the ERP response for a very small change the panel thicknesses.

Listing 7-3 TPL Example ERPOPT8.dat ERP Design Optimization History Table

```
                        *******   ANALYSIS RESULTS BASED ON THE FINAL DESIGN   *******

                                  -----------------------------------------------
                                  |       R E S P O N S E S    I N    D E S I G N    M O D E L        |
                                  -----------------------------------------------

                                     (N/A - BOUND NOT ACTIVE OR AVAILABLE)
                                  (*** VIOLATED RESPONSES MARKED WITH V ***)
                                  (*** ACTIVE RESPONSES MARKED WITH A ***)

                                       -----    WEIGHT RESPONSE    -----

-------------------------------------------------------------------------------------------------------------------------------
    INTERNAL     DRESP1     RESPONSE      ROW       COLUMN          LOWER                          UPPER
       ID          ID        LABEL        ID         ID            BOUND          VALUE           BOUND
-------------------------------------------------------------------------------------------------------------------------------
         1         200      WEIGHT        3          3              N/A         1.9711E-03         N/A

                                  F I N A L    A N A L Y S I S     S U B C A S E =    1000

                                     ----  EQUIVALENT RADIATED POWER RESPONSES ---

-------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE       SET3    ITEM CODE                    LOWER                          UPPER
      ID        ID      LABEL         ID        NO.       FREQUENCY       BOUND          VALUE           BOUND
-------------------------------------------------------------------------------------------------------------------------------
         2       700   ERP103         103         2     1.0000E+01        N/A          5.3160E+00     1.0000E+01
         3       700   ERP103         103         2     1.4000E+01        N/A          9.9843E+00     1.0000E+01 A
         4       710   ERP203         203         2     1.4000E+01        N/A          5.2394E+00     1.0000E+01
         5       720   ERP303         303         2     1.4000E+01        N/A          5.2394E+00     1.0000E+01
         6       730   ERP303           0         2     1.4000E+01        N/A          1.5224E+01     3.0000E+01

                                  F I N A L    A N A L Y S I S     S U B C A S E =    2000

                                     ----  EQUIVALENT RADIATED POWER RESPONSES ---

-------------------------------------------------------------------------------------------------------------------------------
   INTERNAL   DRESP1   RESPONSE       SET3    ITEM CODE                    LOWER                          UPPER
      ID        ID      LABEL         ID        NO.       FREQUENCY       BOUND          VALUE           BOUND
-------------------------------------------------------------------------------------------------------------------------------
```

7	700	ERP103	103	2	1.0000E+01	N/A	5.3160E+00	1.0000E+01
8	700	ERP103	103	2	1.4000E+01	N/A	9.9843E+00	1.0000E+01 A
9	710	ERP203	203	2	1.4000E+01	N/A	5.2394E+00	1.0000E+01
10	720	ERP303	303	2	1.4000E+01	N/A	5.2394E+00	1.0000E+01
11	730	ERP303	0	2	1.4000E+01	N/A	1.5224E+01	3.0000E+01

****
S U M M A R Y O F D E S I G N C Y C L E H I S T O R Y
****

(HARD CONVERGENCE ACHIEVED)

(SOFT CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED 7
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS 6

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY

CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		1.940000E-03		6.789004E+00
1	1.976649E-03	1.976640E-03	4.476062E-06	3.760033E-02
2	1.970862E-03	1.970851E-03	5.434307E-06	2.967129E-02
3	1.958435E-03	1.958432E-03	1.902180E-06	4.177151E+00
4	1.963510E-03	1.963513E-03	-1.185786E-06	3.864629E-01
5	1.969974E-03	1.969973E-03	8.273284E-07	4.618206E-02
6	1.971119E-03	1.971119E-03	1.181211E-07	-1.572514E-03

DESIGN VARIABLE HISTORY

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	1	2	3	4	5
1	1	T1	1.0000E-02	1.0248E-02	1.0035E-02	1.0184E-02	1.0000E-02	1.0044E-02
2	2	T2	1.0000E-02	1.0131E-02	1.0304E-02	1.0000E-02	1.0261E-02	1.0284E-02

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	6	7	8	9	10	11
1	1	T1	1.0052E-02					
2	2	T2	1.0288E-02					

8 Aeroelasticity

- Support for MONPNT2, MONPNT3 and MONSUM in Solution 146

Support for MONPNT2, MONPNT3 and MONSUM in Solution 146

This ability to provide bulk data input for MONPNT2 and MONPNT3 responses has been added to SOL 146 (Dynamic Aeroelastic Response) as has support for MONSUM's. The result is that these quantities can now be output as part of a standard dynamic aeroelastic analysis. SOL 146 performs dynamic aeroelastic analysis (including gust analysis) using methods based on modal frequency response analysis with the presence of Fourier Transforms enabling the input of loads and the output of results in the time domain.

Monitor points were originally implemented in SOL 144 (static aeroelasticity). They have since been implemented in SOL's 101,103,108,109,111,112 and SOL 200. MONPNT1 and MONDSP1 monitor responses were implemented in SOL 146 as part of the *MD Nastran 2005R3* release. For these two types of responses, results can be provided on both the structural and aerodynamic meshes. MONPNT2 and MONPNT3 results are only available for the structural mesh. The addition of MONPNT2/MONPNT3/MONSUM capability to SOL 146, therefore, closes a hole in the ability to use monitor points in the linear solutions sequences.

Benefits

With MONPNT2 and MONPNT3 responses, the user can obtain targeted results from the analysis. For example, it might be critical to view the bending response in a particular beam element that may be critical in assessing fatigue life. The MONPNT3 can provide a view of the total loads of a particular section or component. This is valuable in establishing the total forces acting on the structure and provides insight into where the critical internal loads occur. The MONSUM is applied when the user wants to construct a particular response that is not directly available. A simple example is when a units conversion is desired. A more complicated example could entail constructing a tailored response that combines several MONPNT1 results to produce a response that is important in the vehicle design. Another easily understood use of the MONSUM is to difference aerodynamic and structural monitor points to see if the net result is zero, implying that the forces have been correctly transferred from the aerodynamic model to the structural model.

This complements the existing capability to output MONPNT1 and MONDSP1 results in SOL 146.

User Interface

The existing MONPNT2, MONPNT3, MONSUM bulk entries are utilized for the SOL 146 application. The following SOL 146 comments are added to the *MSC Nastran Quick Reference Guide* descriptions of these entries. For the MONPNT3:

Partial exclusion flags are not supported in SOL 146 so that the only supported XFLAG values are blank or SMAD.

For the MONSUM, the following sentence is added to Remark 5 which defines "similar types" in the context of the MONSUM:

> In SOL 146, the MONPNT3 is not regarded as a similar type to the MONDSP1 and MONPNT1, and hence, should not be used on the MONSUM entry.

Test Cases

A number of test cases have been added to the TPL library in location `tpl/s146m2m3`. A sampling of these are:

Deck Name	Features
S146m2t	Creates MONPNT2 responses in the time domain as part of a gust analysis
S146m23t	Creates MONPNT2 and MONPNT4 responses in the time domain as part of a gust response analysis.
S146m23f	MONPNT2 and MONPNT3 responses in the frequency domain as part of a gust response analysis
S146msml	Includes a MONSUM which combines a MONPNT1 and a MONPNT3 in the frequency domain for a gust response analysis.
S146msmd	Includes MONSUMs that combine structural and aerodynamic MONPNT1's and structural and aerodynamic MONDSP1 in the time domain for a gust response analysis.

Guidelines and Limitations

As mentioned above, MONPNT3 support in SOL 146 does not support partial exclusion flags so the `XFLAG` value must be either blank or `SMAD`.

There is no SOL 146 support in any postprocessor so that user have to rely on the results in the `.f06` file or create their own postprocessing GUI to display the results in the frequency or time domain.

9 Miscellaneous

Monitor Point Enhancements (2013.1)

MSC has a long standing capability designated "Monitor Points" that enables the user to recover results at a particular location that are in addition to standard data recovery. The ability now extends to the major solution sequences (101,103,108,109,111,112,144,146 and 200) with bulk data entries MONDPS1 (monitored displacements), MONPNT1 (monitored loads), MONPNT2 (monitored element responses) and MONPNT3 (monitored grid point forces). The MONSUM entry can be used to combine monitor points of similar type using a scalar coefficient applied to each. Two further enhancements are provided to extend the support for monitor points:

a. An output coordinate system is added to the MONPNT3
b. A new MONSUM1 entry is provided that specifies the location of the summed quantity. This enables the summation of monitor points from disparate points to a single location and supports the transfer of moments.

A defect correction that first appears in this release is that the label information provided on a MONSUM entry is now printed to the `.f06` file. Previously, this label was omitted when the new input file processor was used.

Benefits

The coordinate system the results are most conveniently expressed in may not be the same as those used in computing the monitor point results so that an output coordinate system facilitates communication among groups and makes the MONPNT3 consistent with the MONPNT1 and MONDPS1 in terms of supporting an output coordinate system.

The ability to transfer moments allows the specification of a monitor point location that is apart from the finite element model and thereby facilitates communication between the loads group and the stress group in the aircraft development process, MONSUM

Feature Description

The existing MONPNT3 bulk data entry is enhanced by the addition of an output coordinate system on a second optional continuation entry (see the MONPNT3) in the *MSC Nastran Quick Reference Guide*.

A new MONSUM1 entry is added with the user input and remarks shown in the 2013.1 *MSC Nastran Quick Reference Guide*. Relative to the MONSUM, the MONSUM1 allows the user to specify where the summed output is requested. MSC Nastran uses this information plus the locations of the referenced monitor points that are being summed to perform a coordinate transformation as part of the summation.

Example

The MSC Nastran TPL has a subdirectory `monsum1` that contains a series of simple examples that apply the new MONSUM1 entry across a variety of solution sequences. A very simple example is `monsum1101`. This is a cantilevered beam example where the beam is of length 10 units and has a load of magnitude 10.0 applied at the half-span.

The example includes a MONSUM and a MONSUM1, both of which are applied to one MONPNT1 that is calculated at the half span and another that is at the tip. The MONSUM simply adds these two MONPNT1's together resulting in a Z

force of 20.0 and a moment about the y axis of 50.0 (from the monpnt1 at the tip). The MONSUM1 computes the results at the tip so now the MONPNT1 at the half span contributes a moment so that the Z force is again 20.0 but the moment is 100.0 (=50. + 5.0 * 10.0)

The example also includes two MONPNT3's (HALF3X and HALF3XR) that are identical except latter has an output coordinate system that points in the opposite direction of the former. This is reflected in the Z output of the respective MONPNT3's.

Guidelines and Limitations

The new output coordinate system of the MONPNT3 and the new MONSUM1 entry are supported for SOL's 101,103,108,109,111,112,144,146 and 200.

As mentioned, MONDSP1's are not supported with the MONSUM1.

The CP,X,Y,Z, CD output is not meaningful in the typical case for the MONSUM. Therefore, a message has been added in the .f06 output:

```
    "CP,X,Y,Z AND CD ARE NOT DEFINED FOR MONITOR RESULTS CREATED FROM A
MONSUM"
```

MONSUM and MONSUM1 entries cannot have more than 24 continuation lines when IFPSTAR is used.

MONSUM1 cannot reference a MONDSP or MONSUM.

Metadata (2013.1)

A new Bulk Data entry, was added to allow text to be sent directly to post processors. Text on the METADATA entry will be written to a datablock named META and will be written to the op2 file if PARAM,POST is used to direct results to it. The datablock is qualified by SEID, so it allows separate METADATA for each superelement (using BEGIN SUPER). For more information please see METADATA (p. 2788) in the *MSC Nastran Quick Reference Guide*.