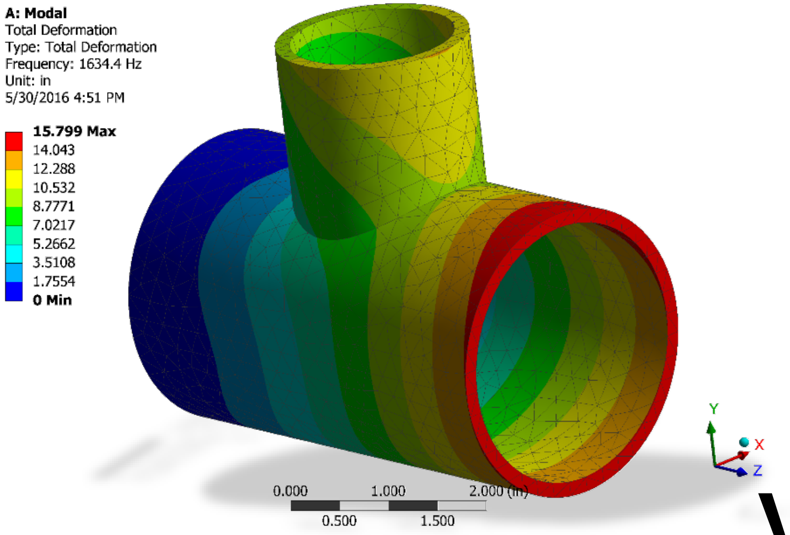
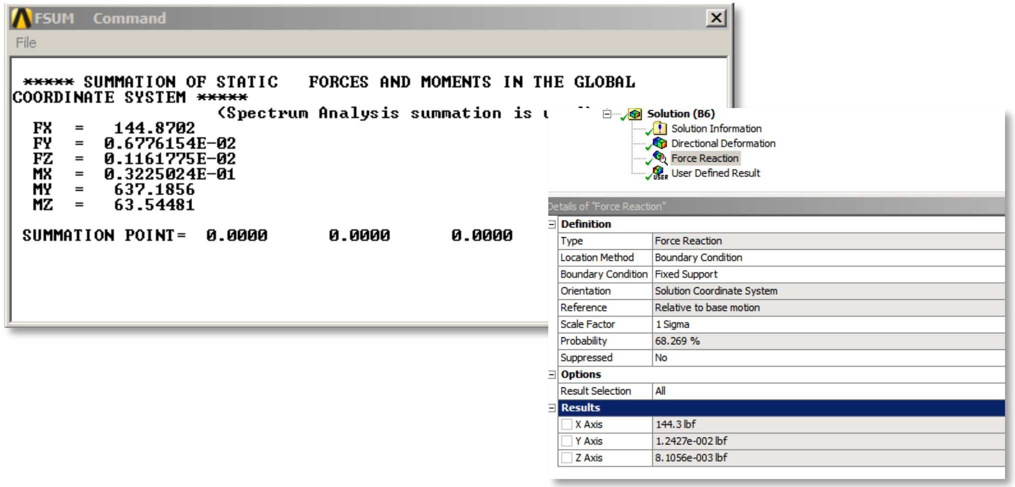


Estimating Structural Response to Random Vibration: Reaction Forces



```
 ***** SUMMATION OF STATIC   FORCES AND MOMENTS IN THE GLOBAL
COORDINATE SYSTEM *****
                    (Spectrum Analysis summation is u
  FX   =   144.8702
  FY   =  0.6776154E-02
  FZ   =  0.1161775E-02
  MX   =  0.3225024E-01
  MY   =   637.1856
  MZ   =   63.54481

 SUMMATION POINT=  0.0000      0.0000      0.0000
```

Definition	
Type	Force Reaction
Location Method	Boundary Condition
Boundary Condition	Fixed Support
Orientation	Solution Coordinate System
Reference	Relative to base motion
Scale Factor	1 Sigma
Probability	68.269 %
Suppressed	No
Options	
Result Selection	All
Results	
□ X Axis	144.3 lbf
□ Y Axis	1.2427e-002 lbf
□ Z Axis	8.1056e-003 lbf

Alex Grishin, Consulting Engineer

PADT, Tempe AZ

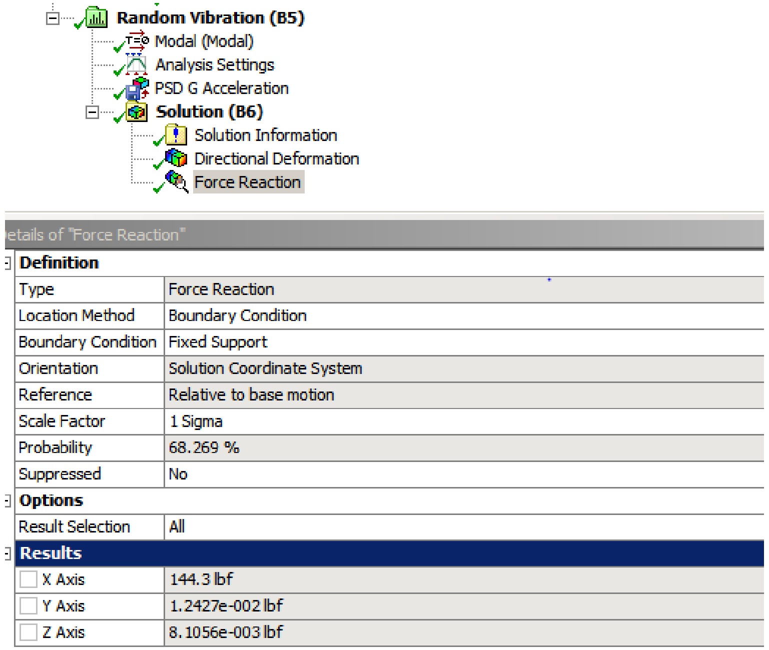
ANSYS Release 17.0 (January, 2016)

- Among the various enhancements, users may now estimate reaction forces directly in Mechanical (using the force reaction probe)!

  - In the past, users had to be content with inserting APDL commands (as described in: http://www.padtinc.com/blog/the-focus/retrieving-accurate-psd-reaction-forces-in-ansys-mechanical)

  - The ability to estimate accurate reaction forces for random vibration loading has existed in MAPDL since release 11 (2007)

  - What took so long?

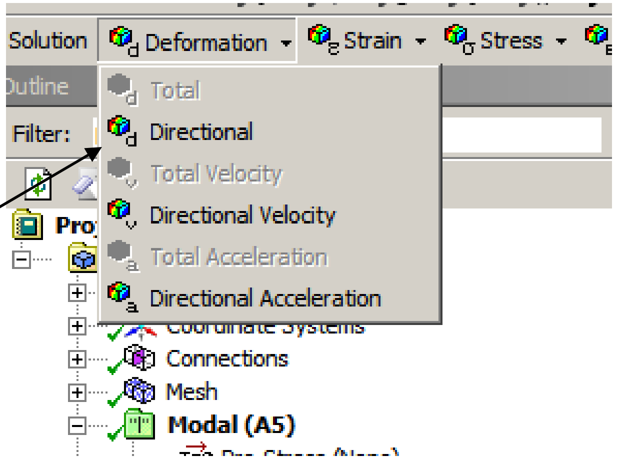


etails of "Force Reaction"

Definition	
Type	Force Reaction
Location Method	Boundary Condition
Boundary Condition	Fixed Support
Orientation	Solution Coordinate System
Reference	Relative to base motion
Scale Factor	1 Sigma
Probability	68.269 %
Suppressed	No
Options	
Result Selection	All
Results	
□ X Axis	144.3 lbf
□ Y Axis	1.2427e-002 lbf
□ Z Axis	8.1056e-003 lbf

- In release 17, Workbench users can still not select "Total" displacement in a random vibration environment (here again, users can invoke MAPDL commands if necessary –or even create a user-defined result)

- Again, it's reasonable to ask "why isn't this available?"



Where's the total displacement?

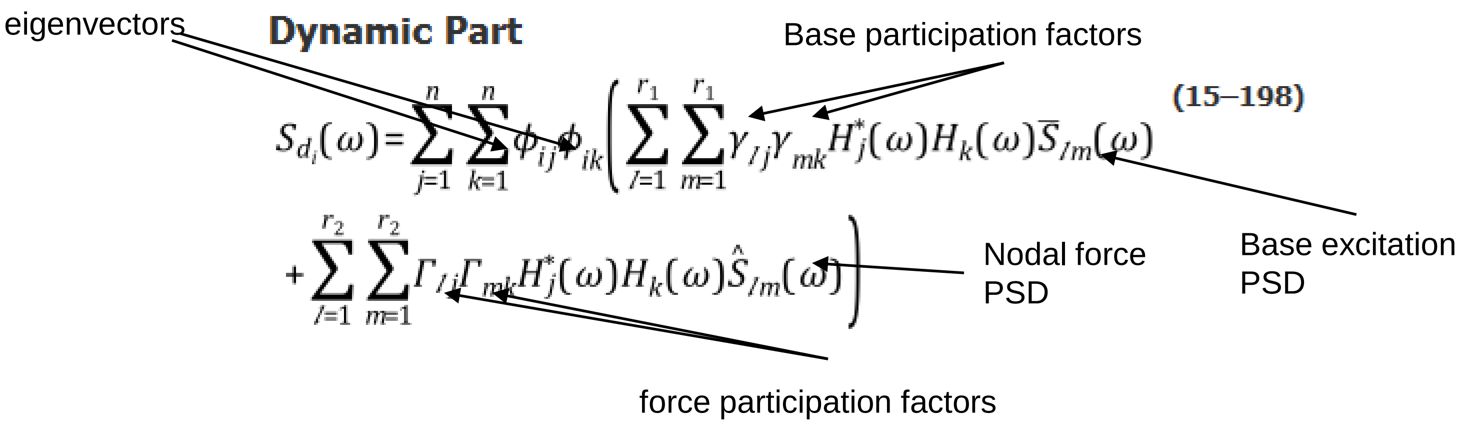
A Brief (recent) History of Random Vibration Analysis in ANSYS

- **ANSYS has had random vibration analysis capability for decades. Although PADT doesn’t know precisely when it was first offered, the following are some more recent milestones in which PADT played a part:**
  - August, 2001: ANSYS 5.7. PADT was made aware of ‘incorrect’ von Mises stress and principle stress calculations by a large aerospace customer (ANSYS 6.0 was released later that year). Customer also complains about slow solution times and large results files
  - February, 2002. PADT offers an external software ‘workaround’ to the problems, called x-PSD. Among the innovations, x-PSD utilizes a clever method of calculating the one-sigma von Mises stress (which we dub the ‘Segalman-Reese’ method after authors of the paper)
  - February, 2003. PADT stops officially developing and supporting x-PSD. Becomes a free download
  - May 2004. ANSYS releases version 8.1. Officially adopts the Segalman-Reese method for calculating one-sigma von Mises stress. Makes principle stresses unavailable in a random vibration environment
  - March, 2006. Another large aerospace customer alerts PADT that PSD reaction force calculations are ‘wrong’.
  - May, 2006. International ANSYS Conference. PADT demonstrates covariance approach to calculating one-sigma reaction forces
  - June, 2007. ANSYS releases version 11, which includes new methodology to calculate reaction forces

Random Vibration Analysis: Background and Simplifications

- Users may review the ‘clever’ Segalman-Reese method first introduced in ANSYS 8.1 by reading the ANSYS APDL Theory manual (section 15.7.11.1). However, the procedure for estimating the reaction forces is a little more mysterious. To understand the basic problems involved, some background is necessary

Let’s start with the Theory Manual. The displacement response PSD (RPSD) is:



eigenvectors

Dynamic Part

Base participation factors

$$S_{d_i}(\omega)=\sum_{j=1}^{n}\sum_{k=1}^{n}\phi_{ij}\phi_{ik}\left(\sum_{l=1}^{r_1}\sum_{m=1}^{r_1}\gamma_{lj}\gamma_{mk}H_j^*(\omega)H_k(\omega)\bar{S}_{lm}(\omega)+\sum_{l=1}^{r_2}\sum_{m=1}^{r_2}\Gamma_{lj}\Gamma_{mk}H_j^*(\omega)H_k(\omega)\hat{S}_{lm}(\omega)\right) \qquad (15\text{–}198)$$

Nodal force PSD

Base excitation PSD

force participation factors

Background

- In the previous equation, H is the complex frequency response (or ‘Transfer’) function*:

$$H_j(\omega)=\frac{1}{\omega_j^2-\omega^2+i(2\xi_j\omega_j\omega)} \qquad \textbf{(15–201)}$$

To keep the discussion simple (without losing anything important), let’s assume that we always have a single base excitation, so that (15-198) reduces to:

Dynamic Part

$$S_{d_i}(\omega)=\sum_{j=1}^{n}\sum_{k=1}^{n}\phi_{ij}\phi_{ik}\left(\sum_{l=1}^{r_1}\sum_{m=1}^{r_1}\gamma_{lj}\gamma_{mk}H_j^*(\omega)H_k(\omega)\overline{S}_{lm}(\omega)\right) \qquad \textbf{(15–198)}$$

Response PSD

Input PSD

*We’re dropping units and assuming the output PSD units are the same as the input PSD units (we’re ignoring PSDUNIT and PSDRES)

Background

- The mean square response (or *variance*), σ^2, is equal to:

$$\sigma_{f_i}^2 = \int_0^\infty S_{d_i}(\omega)d\omega \qquad (15\text{–}204)$$

- In what follows, it would be very helpful to re-write (15-204) in terms of the *modal coordinate covariance matrix, C*. And since we're only looking at a single base excitation, we'll eliminate some indices, and use the Einstein Summation Convention*:

$$\sigma^2 = \phi_i \phi_j C_{ij} \qquad \text{(a-1)}$$

where

Participation factor, $g_j = f_{jk}e_k$

$$C_{ij} = \int_0^\infty \left(\frac{\phi_{ik} e_k}{(\omega_i^2 - \Omega^2 + \mathbf{i}(2\omega_i \Omega \xi_i)} \right) \left(\frac{\phi_{jk} e_k}{(\omega_j^2 - \Omega^2 - \mathbf{i}(2\omega_j \Omega \xi_j)} \right) S d\omega$$

* https://en.wikipedia.org/wiki/Einstein_notation

Background

- Note that (a-1) represents a double-sum over the extracted modes. It also has a very flexible structure called a *bilinear form*. Among other things, the quantities ϕ_i and ϕ_j may be scalars, vectors, or matrices –it doesn't matter!

- It's important to realize that when creating contour plots for a particular result quantity, we are obtaining n-sigma quantities, n σ, where:

$$n\sigma = n\sqrt{\sigma^2}$$

- MAPDL stores these values on a nodal and element basis in the general postprocessor in load steps 3 through 5

- Workbench Mechanical also accesses these values

Background

it’s now possible to see what the issue is (was).

- Prior to ANSYS 8.1, if one wanted the one-sigma von Mises stress, ANSYS would first store the component stresses according to equation (a-1), where ϕ_i and ϕ_j correspond to the modal eigenstresses (mass-normalized stresses for modes i and j). It would then perform a standard von Mises calculation, such as:

$$\sigma' = \frac{1}{\sqrt{2}}\left[(\sigma_x-\sigma_y)^2+(\sigma_y-\sigma_z)^2+(\sigma_z-\sigma_x)^2+6(\tau_{xy}^2+\tau_{yz}^2+\tau_{zx}^2)\right]^{1/2}$$

- But, since each stress component is calculated according to (a-1), it is a root-mean-square quantity. In other words, always positive!

- The issue here is that one-sigma quantities are calculated first, then the equivalent stress is calculated based on these quantities

Background

So, what to do?

- The answer lies in first realizing that the eigenquantities in (a-1) need not be calculated on a component-wise basis, as was originally done

- Segalman and Reese realized that if they could re-write the von Mises stress as a *vector product* (also referred to as a *quadratic form*), then equation (a-1) could easily accommodate it. Thus, the variance (mean square) of the von Mises stress of mode j can be re-caste as:

$$\sigma_{sj}^2 = \boldsymbol{\eta}_j^T \cdot \mathbf{A} \cdot \boldsymbol{\eta}_j$$

where

$$\boldsymbol{\eta} = \begin{Bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \sigma_{xy} \\ \sigma_{yz} \\ \sigma_{xz} \end{Bmatrix} \qquad \mathbf{A} = \begin{pmatrix} 1 & -1/2 & -1/2 & 0 & 0 & 0 \\ -1/2 & 1 & -1/2 & 0 & 0 & 0 \\ -1/2 & -1/2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}$$

Background

And so, (a-1) can be directly employed (and stored as a one-sigma quantity in the results file):

$$\underset{\text{This is a scalar…}}{\sigma_s^2} = \underbrace{\underbrace{\boldsymbol{\eta}_i^T \cdot \mathbf{A} \cdot \boldsymbol{\eta}_j}_{\text{This is a scalar…}} C_{ij}}_{\text{…and this is still a double sum formula…}} \qquad \text{(a-2)}$$

- This is the solution that ANSYS adopted in 2004, starting with release 8.1

- Here I'm mixing vector and indicial notation. Although a bit ambiguous, this makes the presentation much more palatable.

Reaction Forces (and beyond…?)

- So, how does ANSYS calculate the reaction forces?
- For the purpose of discussion, suppose we have nodal component of size N, and we want a vector sum of its nodal forces. The reaction force for each node in a particular direction (x, say), can be expressed as a vector in *nodal coordinates*:

$$\mathbf{f}_x = \begin{Bmatrix} f_{1x} \\ f_{2x} \\ \cdot \\ \cdot \\ \cdot \\ f_{Nx} \end{Bmatrix}$$

- Following Segalman and Reese, the mean square of this vector can be calculated according to:

$$\sigma_{fx}^2 = \mathbf{f}_{xi}^T \mathbf{f}_{xj} C_{ij} \qquad \text{(a-3)}$$

Reaction Forces (and beyond…?)

- But notice this is not the same thing as the mean-square of the SUM of the components of **f** (which is what we're after)!
- As we'll see shortly, prior to ANSYS release 11, FSUM produced an even worse estimate of the sum of **f** than would be obtained by (a-3): it simply computed each component of f according to (a-1), took the square root and summed the resulting (one-sigma) values!

Reaction Forces (and beyond…?)

- Recall what we're after. We want the total (one-sigma) vector sum (in the x-direction in this case) of the nodal forces of the component. The mean square quantity of (a-3) does not produce this. By adding the square of each nodal force, we lose the sign, and produce a result which, in general, is way too high!

- To find the solution to this, let's do a thought experiment. Let's define a new quantity, called 'cov':

We'll explain what 'cov' stands for shortly…

$$\mathrm{cov}(\mathbf{f}_x) = \underbrace{\underbrace{\mathbf{f}_{xi}\mathbf{f}_{xj}^T}_{\text{This is an N x N matrix}} C_{ij}}_{\text{Still a double sum over number of modes}} \qquad \text{(a-4)}$$

- The quantity to the left of C is a symmetric N x N matrix (the outer product of $\mathbf{f}_i$ and $\mathbf{f}_j$ instead of the dot product).

Reaction Forces (and beyond…?)

- To see how (a-4) is useful, let’s expand it…

$$\operatorname{cov}(\mathbf{f}_x) = \begin{pmatrix} f_{1x}f_{1x} & f_{1x}f_{2x} & \cdot & \cdot & \cdot & f_{1x}f_{Nx} \\ f_{2x}f_{1x} & f_{2x}f_{2x} & \cdot & \cdot & \cdot & f_{2x}f_{Nx} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ f_{Nx}f_{1x} & f_{Nx}f_{2x} & \cdot & \cdot & \cdot & f_{Nx}f_{Nx} \end{pmatrix}$$

- Note first that off-diagonal terms *retain their sign relative to the square-root of that row’s diagonal member!*

Reaction Forces (and beyond…?)

- Now, let's focus on a single row. We'll pick an arbitrary row of this matrix and divide by the square root of its diagonal term (first making sure that's not zero!). For illustration, let's take row 2. We divide each component of row 2 by f_{2x} (since that's the square roof of the diagonal term)

$$\sigma(\mathbf{f}) = \left(f_{1x}, f_{2x}, \ldots, f_{Nx}\right) \qquad \text{(a-5)}$$

- This gives us back the original terms! Note that it doesn't matter which row we pick. We'll always get result (a-5) as long as we divide by the square root of the diagonal term.

- But *what* have we got exactly? Well, note that the diagonal values of (a-4) are simply the mean square values (variances) of each member of **f**. It turns out that the quantity (a-4) is the *covariance* of **f**

Reaction Forces (and beyond…?)

- Dividing any row by the square root of its diagonal member should produce components whose magnitudes are each equal to their one-sigma value (by the definition of the covariance matrix), and whose sign is relative to the divisor. This allows us to produce a correct one-sigma vector SUM.

- This can be made more efficient. For example, knowing that every row of the covariance matrix will produce the same result, the calculation (a-4) thru (a-5) (and then taking the resulting sum) can be carried out on just one row. We can write this as:

$$\sigma(\Sigma f) = \frac{1}{\sigma(f_{N,N})} \mathbf{e}_N^T \cdot \mathbf{f}_i \, \mathbf{f}_j^T C_{ij} \cdot \mathbf{1} \quad \text{(a-6)}$$

Where $\mathbf{e}_N^T$ is a unit vector containing the index of the row (N) to be kept, and 1 is an N x 1 matrix of ones*

* https://en.wikipedia.org/wiki/Matrix_of_ones

Reaction Forces (and beyond…?)

- So, the main trick to finding accurate one-sigma vector sums in a random vibration environment is to first calculate the covariance matrix of the vector in question, as in (a-4)
- For instance: Suppose one needs signed one-sigma stresses (for a principle stress calculation, for example). All one needs is the covariance of the stress vector, $\boldsymbol{\eta}$

$$\mathrm{cov}(\boldsymbol{\eta}) = \boldsymbol{\eta}_i \boldsymbol{\eta}_j^T C_{ij} \quad \text{(a-7)}$$

This is a 6 x 6 matrix

Still a double-sum over the number of modes…

…and then extract the signed one-sigma $\boldsymbol{\eta}$:

$$\sigma(\boldsymbol{\eta}) = \frac{1}{\sigma(\eta_{N,N})} \mathbf{e}_N^T \cdot \boldsymbol{\eta}_i \boldsymbol{\eta}_j^T C_{ij} \quad \text{(a-8)}$$

One-Sigma stress vector

Validation by Example

- The relations (a-4) thru (a-8) are new to most engineers, but they can be easily proven from the definition of covariance matrix and arithmetic properties of the expectation operator

- Verifying these calculations numerically on complex model problems is a bit harder. In principle, this would require comparing results of these calculations to statistical properties of equivalent time histories

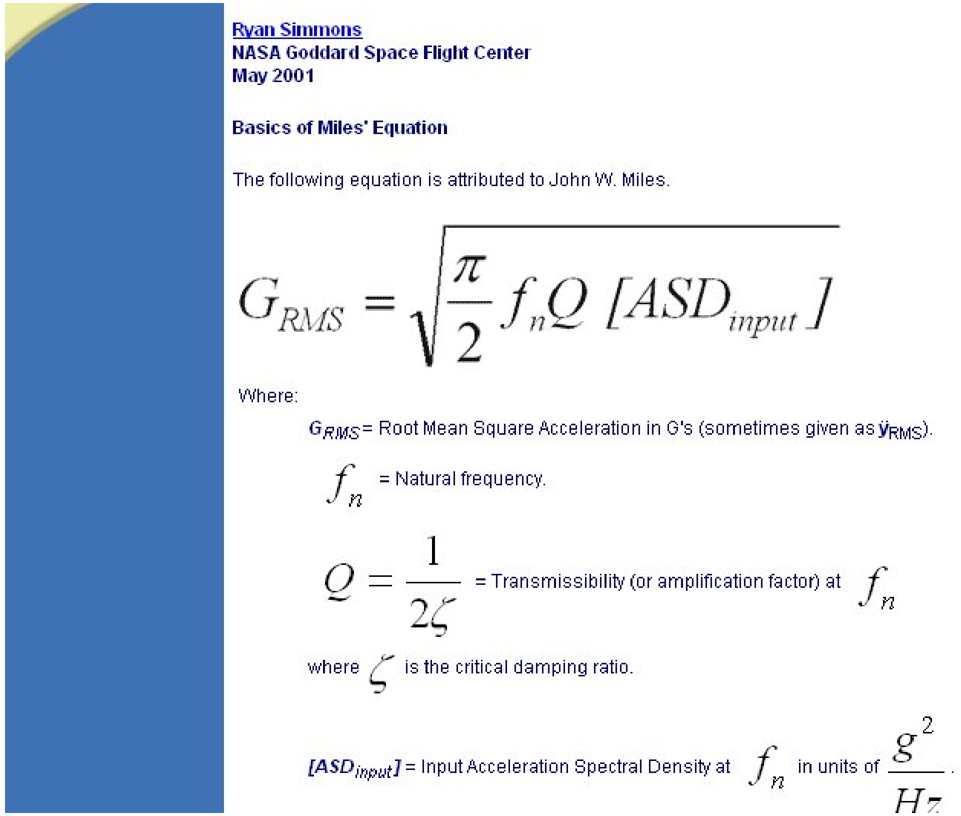
- However, for simple systems, we can compare these results to single DoF approximations. There are two basic techniques*:
    1. Miles' Equation
    2. Single Mode Coefficient

*We might mention in passing a third technique: One could constrain a group of nodes on which a reaction force is required to a single mass element. The node of this element will contain the correct force value

Validation by Example: Miles’ Equation

$$F = M_e * G_{rms} * 386.4$$

Modal effective mass



Ryan Simmons
NASA Goddard Space Flight Center
May 2001

Basics of Miles' Equation

The following equation is attributed to John W. Miles.

$$G_{RMS} = \sqrt{\frac{\pi}{2} f_n Q\ [ASD_{input}]}$$

Where:

G_{RMS} = Root Mean Square Acceleration in G's (sometimes given as $\ddot{y}_{RMS}$).

f_n = Natural frequency.

$Q = \frac{1}{2\zeta}$ = Transmissibility (or amplification factor) at f_n

where ζ is the critical damping ratio.

$[ASD_{input}]$ = Input Acceleration Spectral Density at f_n in units of $\frac{g^2}{Hz}$.

http://femci.gsfc.nasa.gov/random/MilesEqn.html

Validation by Example: Scale mode by ANSYS Mode Coefficient

- After a random vibration analysis has been successfully performed, ANSYS allows one to extract a 'Mode Coefficient', *MC*. These values are simply the diagonals of the *Modal Coordinate Covariance Matrix*, C:

$$MC(i) = C_{ii}$$

No summation implied by repeated indices…

```
*get,mci,mode,i,mcoef
```

APDL for getting mode coefficient i

- To a single-DoF approximation, the ith-mode one-sigma response is given by:

$$\sigma(\phi_i) = \phi_i \sqrt{C_{ii}}$$

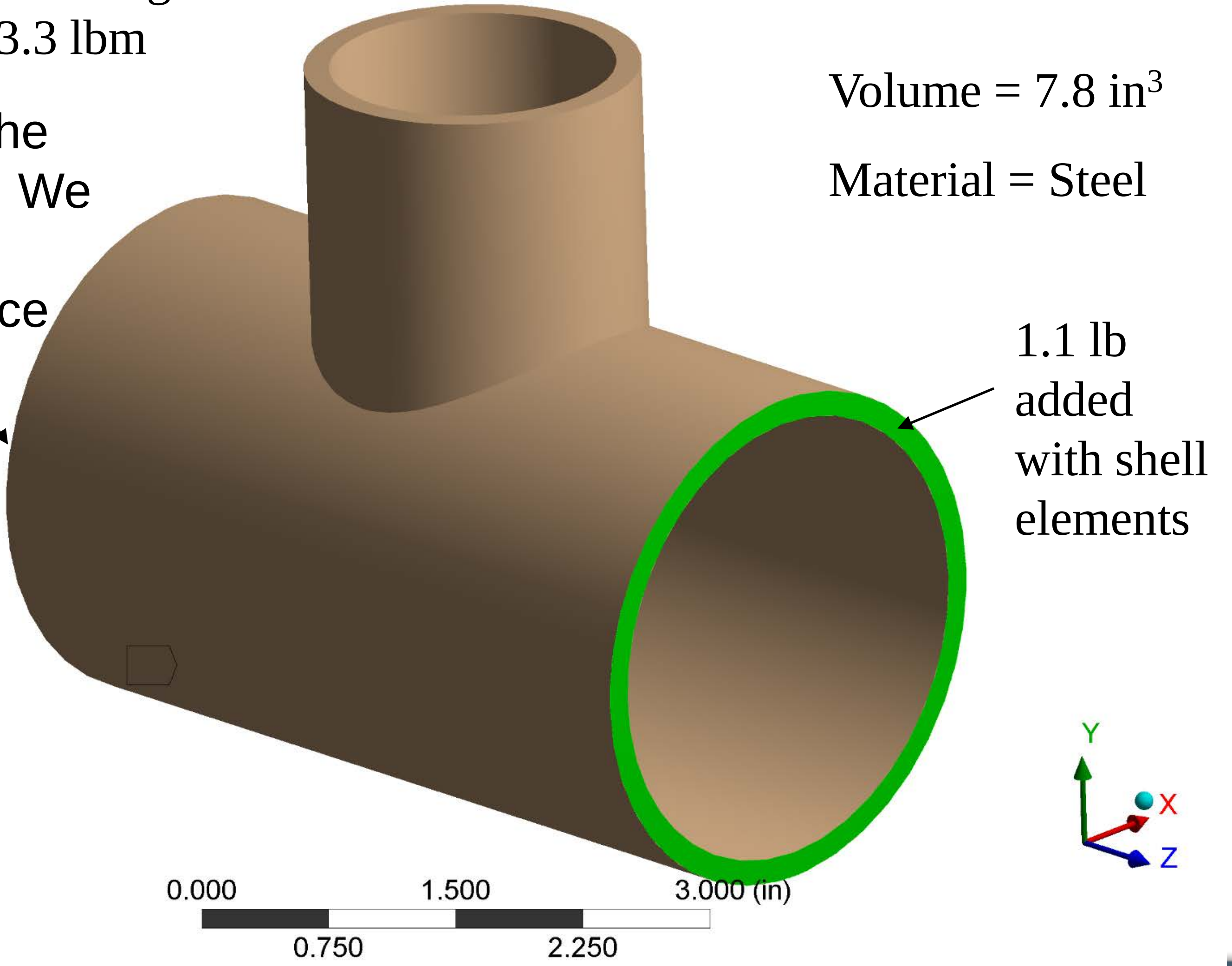
- In APDL, this can be easily achieved by scaling the ith mode by the mode coefficient:

```
set,1,i,sqrt(mc)
```

APDL for scaling ith mode by mode coefficient.

- FSUM will now produce accurate reaction force values

Validation by Example: Reaction Force For A Simple Valve Body



Total Weight = 3.3 lbm

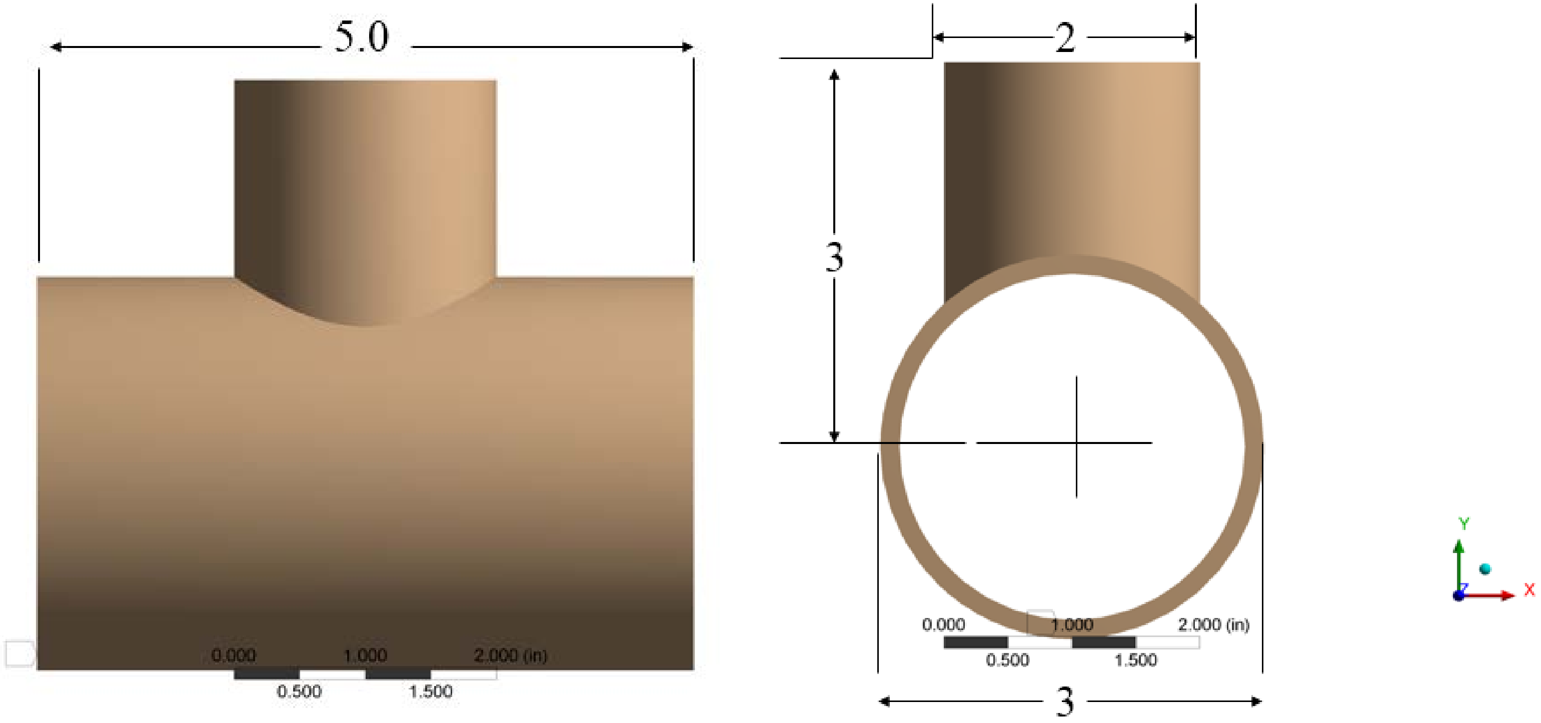
This face (the base) fixed. We want the reaction force here

Volume = 7.8 in^3

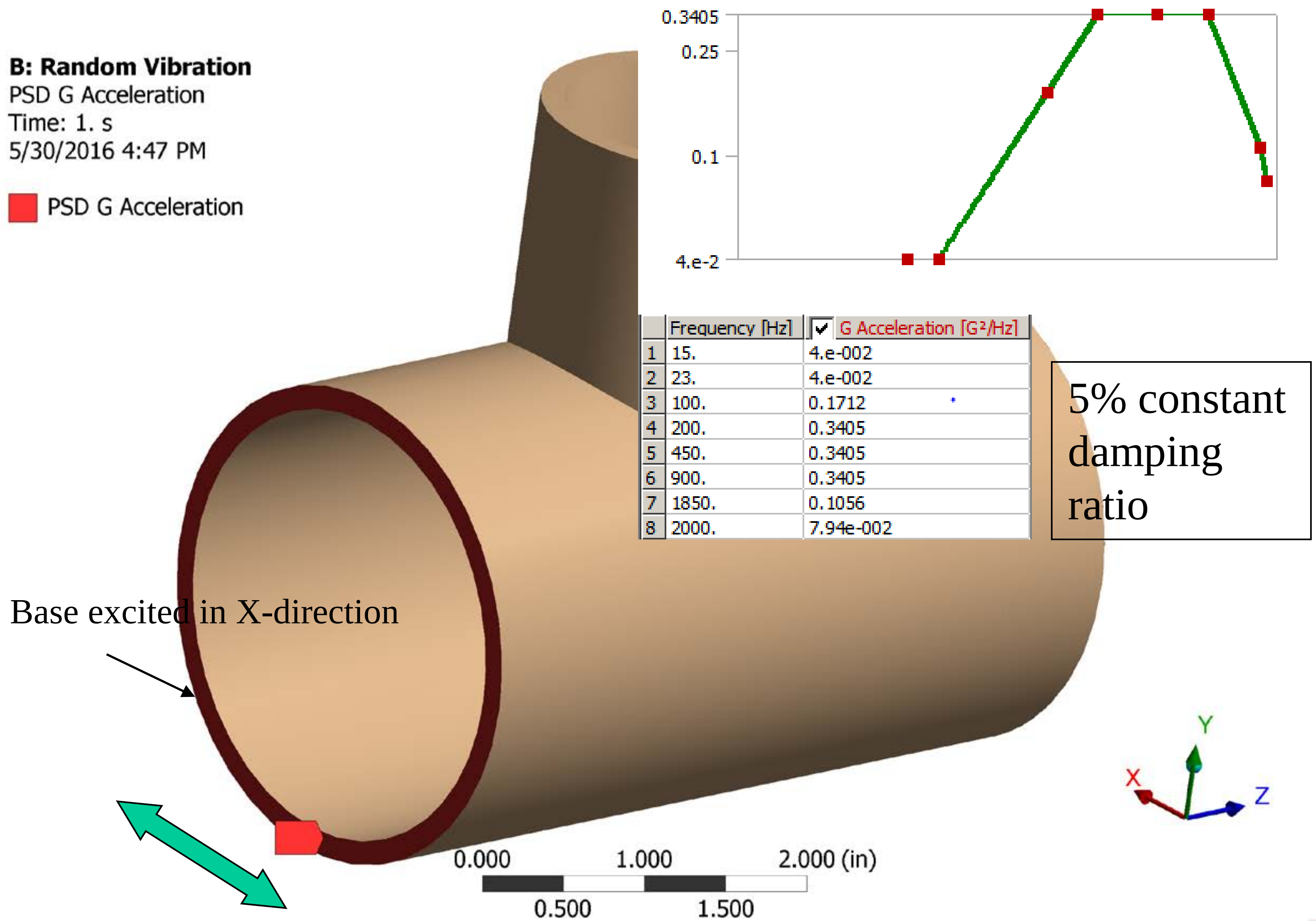
Material = Steel

1.1 lb added with shell elements

Validation by Example: Reaction Force For A Simple Valve Body



Validation by Example: Reaction Force For A Simple Valve Body



B: Random Vibration
PSD G Acceleration
Time: 1. s
5/30/2016 4:47 PM

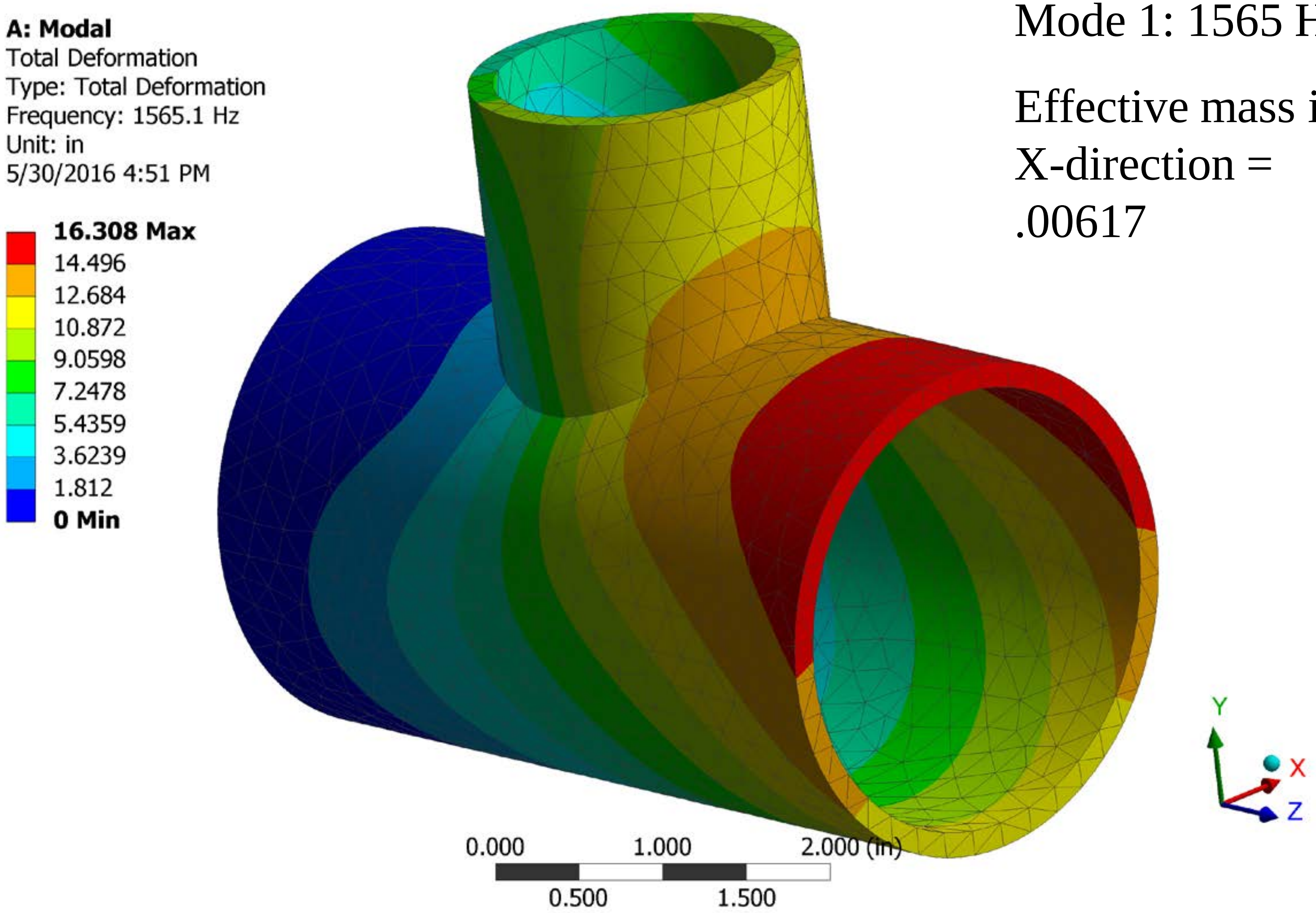
🟥 PSD G Acceleration

	Frequency [Hz]	☑ G Acceleration [G²/Hz]
1	15.	4.e-002
2	23.	4.e-002
3	100.	0.1712
4	200.	0.3405
5	450.	0.3405
6	900.	0.3405
7	1850.	0.1056
8	2000.	7.94e-002

5% constant damping ratio

Base excited in X-direction

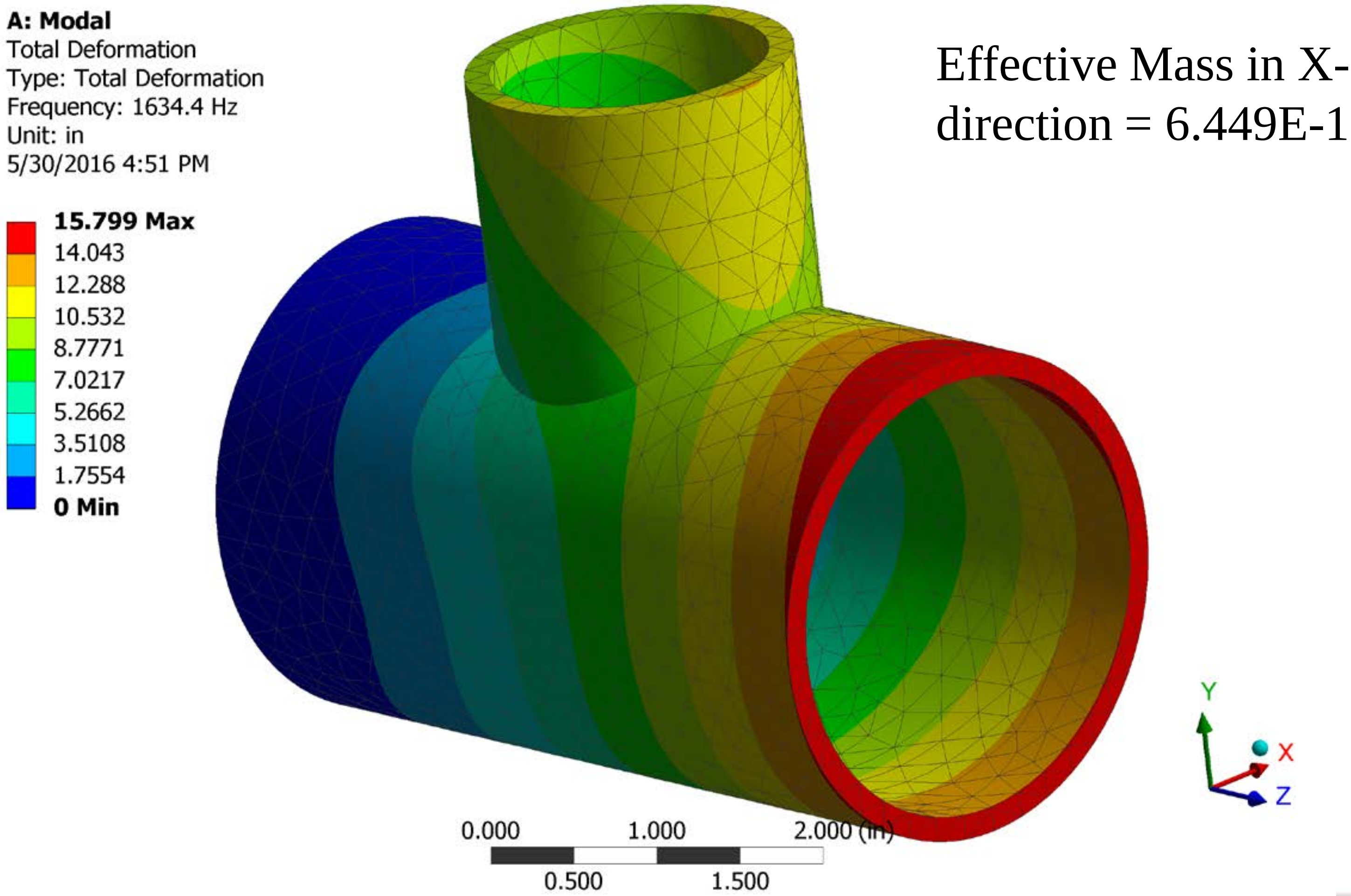
Validation by Example: Reaction Force For A Simple Valve Body



Mode 1: 1565 Hz

Effective mass in X-direction = .00617

Validation by Example: Reaction Force For A Simple Valve Body

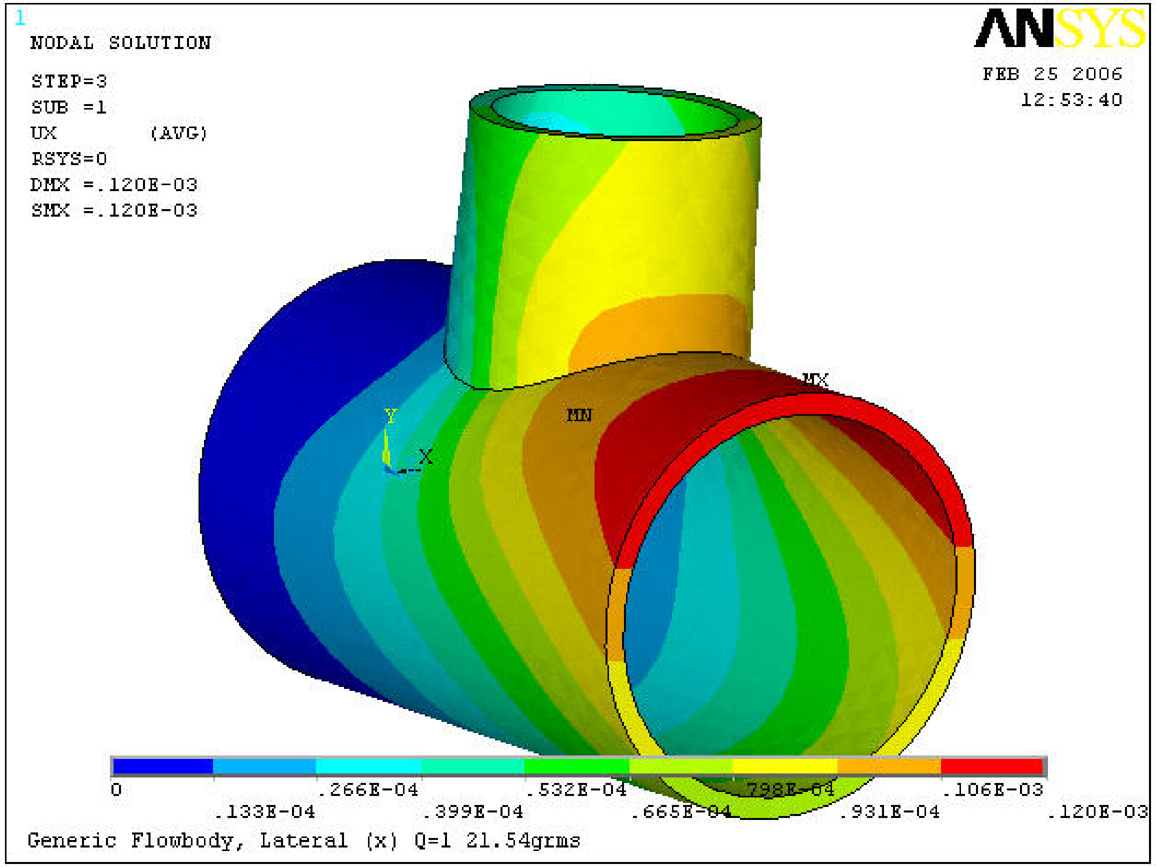


Mode 2: 1634 Hz

Effective Mass in X-direction = 6.449E-10

Validation by Example: Reaction Force For A Simple Valve Body

Solution 1: FSUM on Load Step 3 Prior to Release 11



Fsum Results
For Base:

FX	=	278.594
FY	=	194.053
FZ	=	616.79
MX	=	-89.102
MY	=	0.1197
MZ	=	0.7650

Validation by Example: Reaction Force For A Simple Valve Body

Solution 2: Miles' Equation

$$G_{rms} = \text{sqrt}(\text{pi}/2*\text{f1}*Q*A)$$

$$G_{rms} = \text{sqrt}(3.14159/2*1570*10*.1378$$

$$G_{rms} = 58.295\text{g}$$

$$F_{base} = Me*G_{rms}*386.4$$

$$F_{base} = 0.00617*58.295*386.4$$

$$\boxed{F_{base} = 138.98 \text{ lbf}}$$

About ½ the value calculated by legacy versions (<8.1)

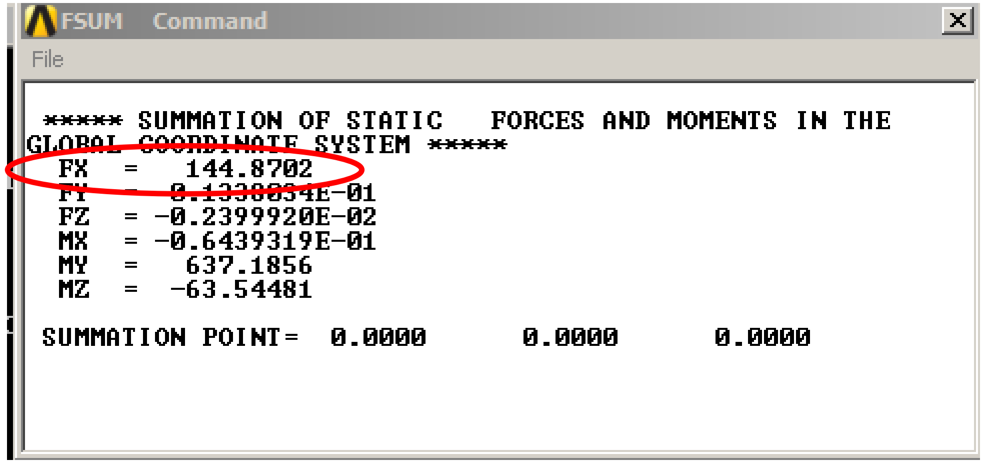
Validation by Example: Reaction Force For A Simple Valve Body

Solution 3: Scale Mode 1 by Mode Coefficient

MC(1) = 3.58753644E-10 …Using *get,mc1,mode,1,mcoef

Sqrt(MC(1)) = 1.894079313E-5

```
cmsel,s,nbase
set,1,1,sqrt(mc1)
FSUM
```



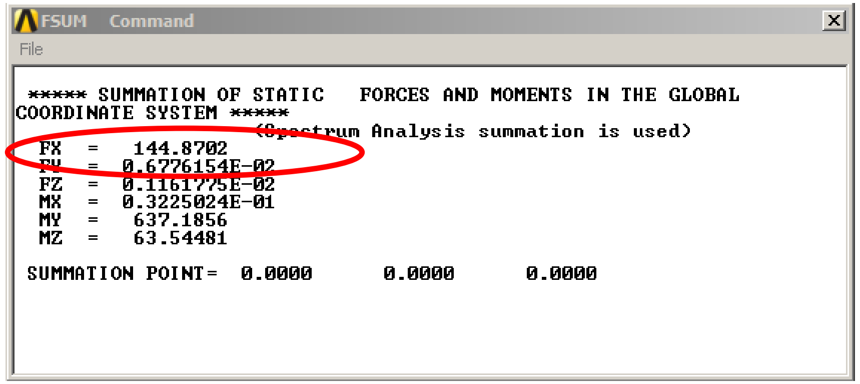
```
 ***** SUMMATION OF STATIC   FORCES AND MOMENTS IN THE
GLOBAL COORDINATE SYSTEM *****
  FX   =    144.8702
  FY   =  0.1330034E-01
  FZ   = -0.2399920E-02
  MX   = -0.6439319E-01
  MY   =    637.1856
  MZ   =   -63.54481

 SUMMATION POINT=  0.0000       0.0000       0.0000
```

$$\boxed{F_{\text{base}} = 144.87\ \text{lbf}}$$

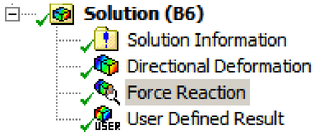
Validation by Example: Reaction Force For A Simple Valve Body

Solution 4: FSUM in ANSYS releases > 11

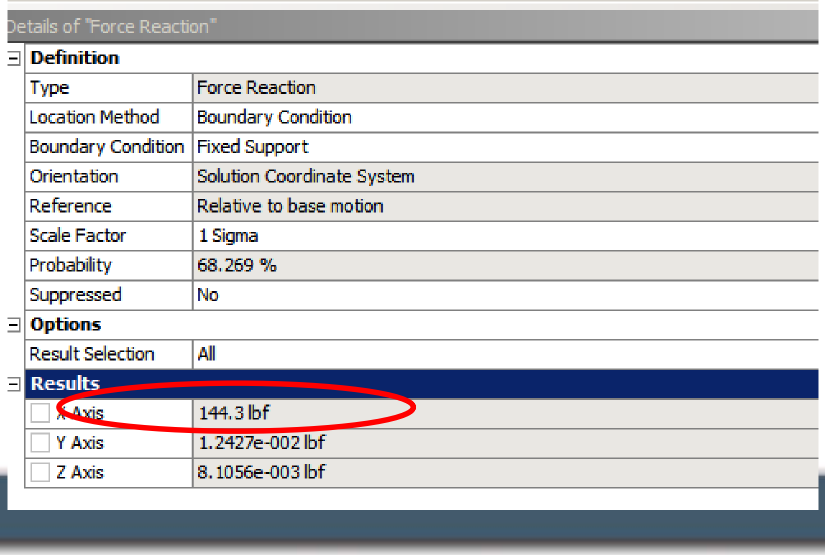
```
cmsel,s,nbase
set,3,1
FSUM
```



```
 ***** SUMMATION OF STATIC   FORCES AND MOMENTS IN THE GLOBAL
COORDINATE SYSTEM *****
                    (Spectrum Analysis summation is used)
  FX   =   144.8702
  FY   =  0.6776154E-02
  FZ   =  0.1161775E-02
  MX   =  0.3225024E-01
  MY   =   637.1856
  MZ   =   63.54481

 SUMMATION POINT=  0.0000      0.0000      0.0000
```



…And at release 17.0, this agrees (more or less) with the reaction probe in ANSYS Mechanical



Details of "Force Reaction"

Definition	
Type	Force Reaction
Location Method	Boundary Condition
Boundary Condition	Fixed Support
Orientation	Solution Coordinate System
Reference	Relative to base motion
Scale Factor	1 Sigma
Probability	68.269 %
Suppressed	No
Options	
Result Selection	All
Results	
□ X Axis	144.3 lbf
□ Y Axis	1.2427e-002 lbf
□ Z Axis	8.1056e-003 lbf

Conclusions and Summary

- For scalar engineering quantities of interest, one-sigma random quantities were always calculated according to (15-204) and (a-1)
- Historically, confusion arose over the difficulty of performing arithmetic operations on quantities calculated this way (the sign is lost, etc.). If Equations (15-204) and (a-1) were all that there was to random vibration, that would be the end of the story. However…
- This confusion evaporates once one realizes that (15-204) and (a-1) calculate the variance ONLY!
- Calculating the full covariance matrix of any result quantity allows us to recover arithmetic operations on all engineering quantities –resulting in correct one-sigma results for those quantities. ANSYS currently utilizes this epiphany for reaction forces, but no other quantities of interest (there’s room for improvement)