

Extracting Relative Displacements in ANSYS Mechanical

Alex Grishin

3/1/2018



We Make Innovation Work
www.padtinc.com

Strain-Based Displacement

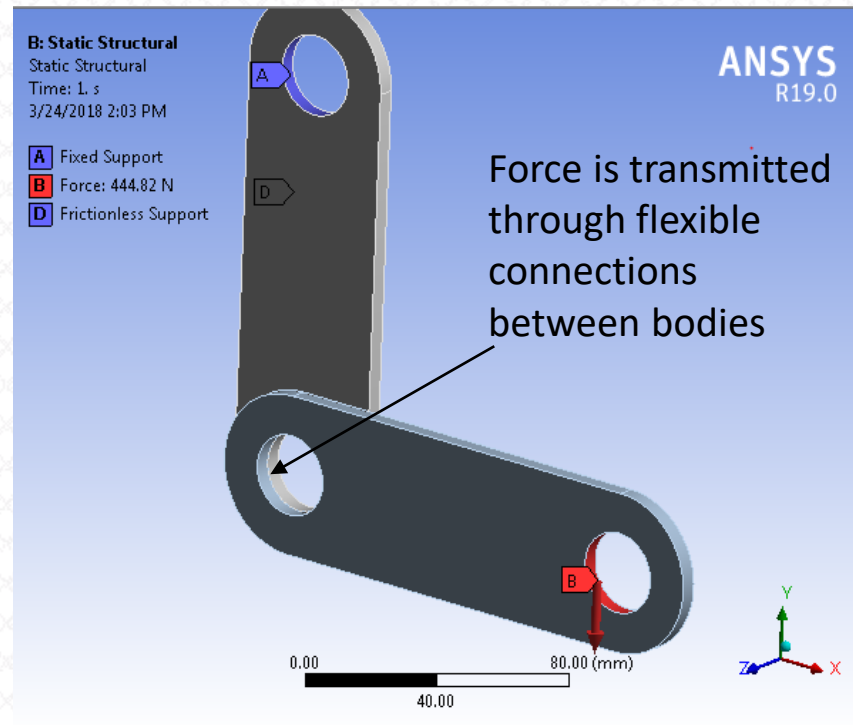
- A recurring theme in ANSYS Technical Support queries involves the separation of rigid-body from material deformations without performing an additional analysis. Many users simply assume this capability should exist as a simple post-processing query(or that in any case, this shouldn't be a difficult operation)
- “Rigid-Body” displacements implies a transient dynamic analysis (as such displacements should not occur in static analyses), but as we'll see, there are contexts within static structural environments where this concept DOES play an important engineering role
- In static structural contexts, such rigid-body motion implies motion transmitted across multiple-bodies. There are two important and loosely related contexts we'll look at



Strain-Based Displacement: 2 Use Cases

- In the first context, we seek the displacements of a body within a larger mechanism that are due purely to nonzero strains within that body

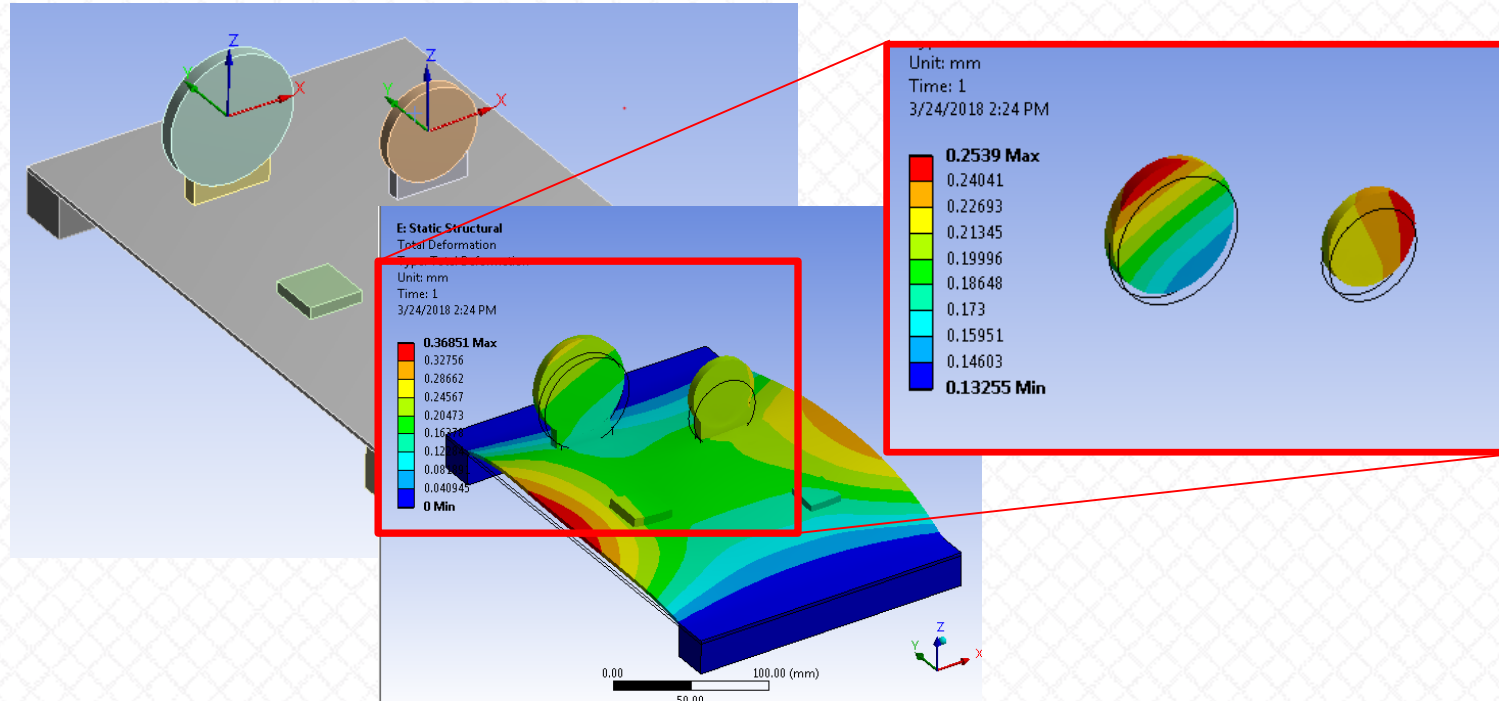
Case 1: Relative displacement of a single body in multi-component mechanisms



Strain-Based Displacement: 2 Use Cases

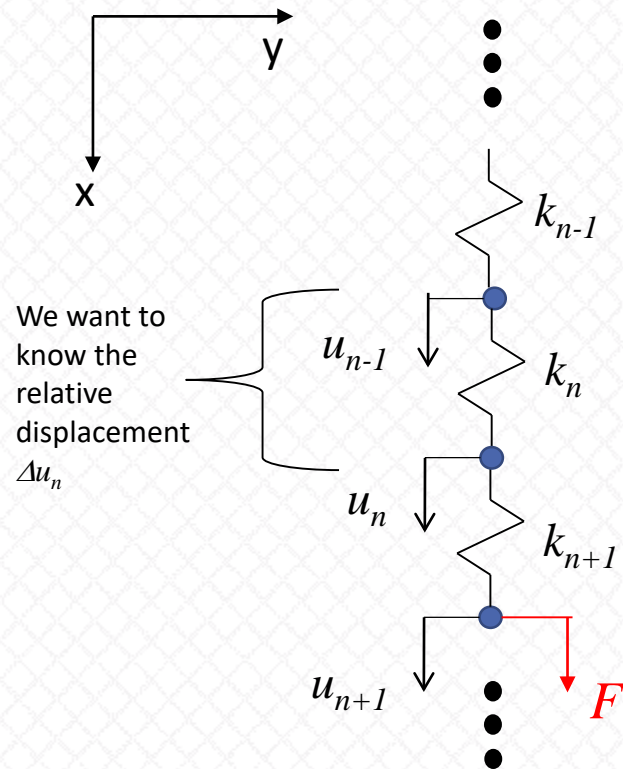
- In the second context, a body's total displacement may be completely due to nonzero strains –but we want to extract the mean translations AND rotations of the CG

Case 2: Mean motion of a body's Center of Gravity

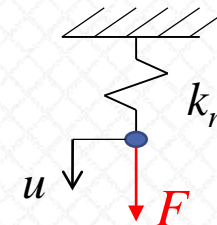


Use Case 1

- It's quite clear to most user's that the relative displacement (due only to nonzero internal strains) of a body in series with others may be determined by performing an additional analysis --fixing only the body of interest and applying the transmitted load to it (in fact, we'll do this to verify our solution). However, we seek a technique that DOES NOT require an additional analysis.

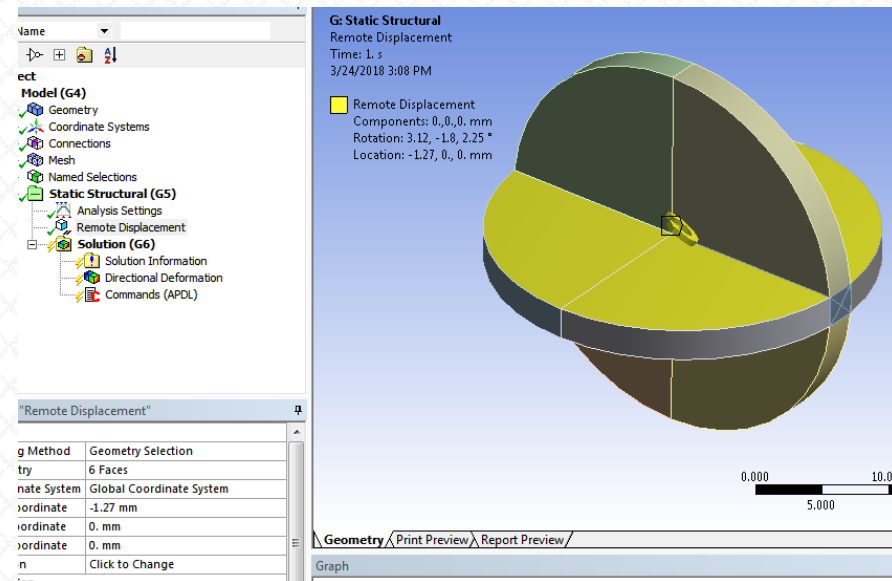


- We want to know the “relative” displacement of n th spring (k_n): $\Delta u_n = u_n - u_{n-1}$
- But since the springs are in series, the force carried by each is equal to F
- So, $\Delta u_n = F/k_n = u$
- This is the same displacement k_n would experience if the spring were detached, $u_{n-1} = 0$, and F applied at the node at u_n



Use Case 2

- In the second use case, users may resort to ANSYS' RBE3 technology* to create a node at a body's CG (the master) and constrain it to all nodes of the body (the slaves)
- The closest way to achieve this in Workbench (using only a minimum number of scripts) is to attach a remote point to the body's external surfaces (using the 'deformable' option), and then query the resulting motion of the remote point
- Note that this doesn't 'quite' give us the second use case, since remote points can only be attached to surfaces –not entire bodies. Still, we'll use this concept to test our technique



*For a nice tutorial, see here: <https://www.simutechgroup.com/tips-and-tricks/fea-articles/147-fea-tips-tricks-rotation>

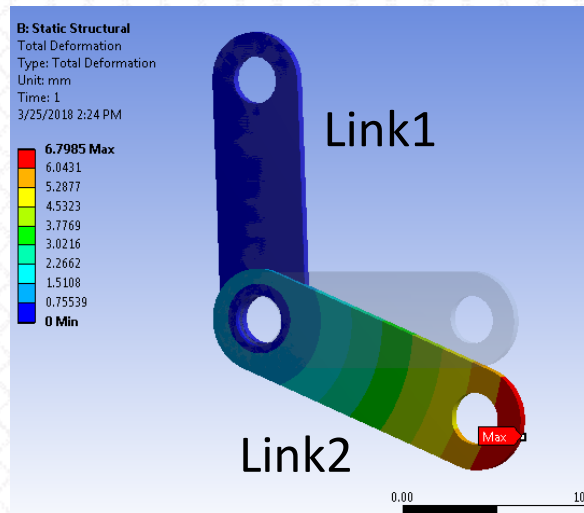
What we're after

- So, there ARE more-or-less 'standard' (or at least recommended) techniques of obtaining reliable results for the 2 use-cases
 - However, we'd like ONE solution for both use-cases
 - In addition, we'd like the solution to be a post-processing solution ONLY. Notice that the recommended solution for use-case 1 requires an additional analysis, while the recommended solution for use-case 2 requires the addition of special elements (we have to modify an existing model).
-
- So, our proposed solution consists of two parts:
- Part 1: Extract the effective translation and rotation of the CG of a named selection without modifying the model
 - Part 2: Calculate a resultant displacement field that subtracts out the motion calculated in 1



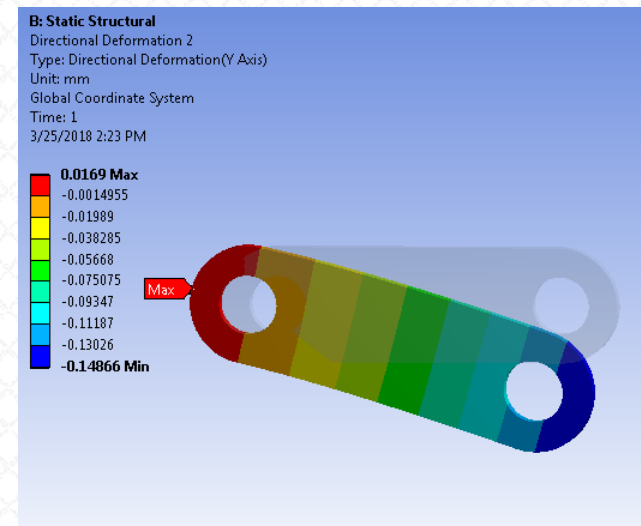
Part 1: Extract effective translations and rotations

- Both use cases we want to consider require first that mean translations and rotations be calculated for named selections
- However, the named selection for which this CG motion is calculated may be different



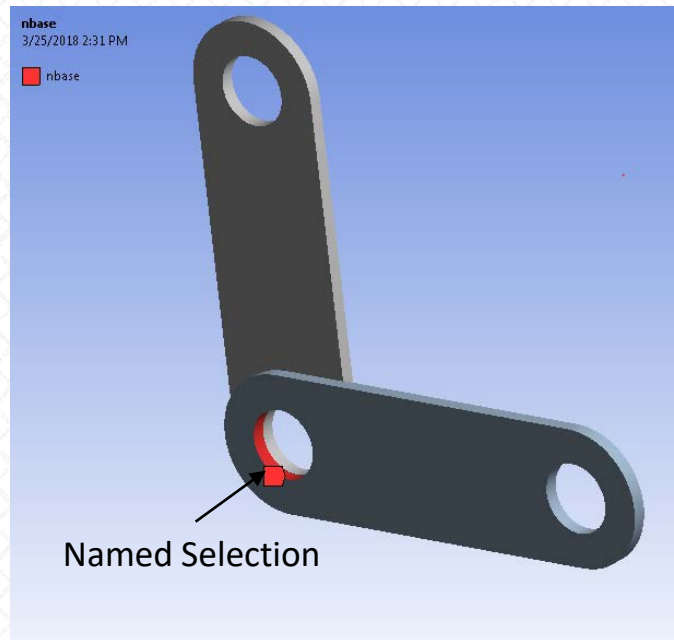
Use Case 1: Two links connected via revolute joint. External load applied on link2

We want the purely strain-based (relative) displacement of link2

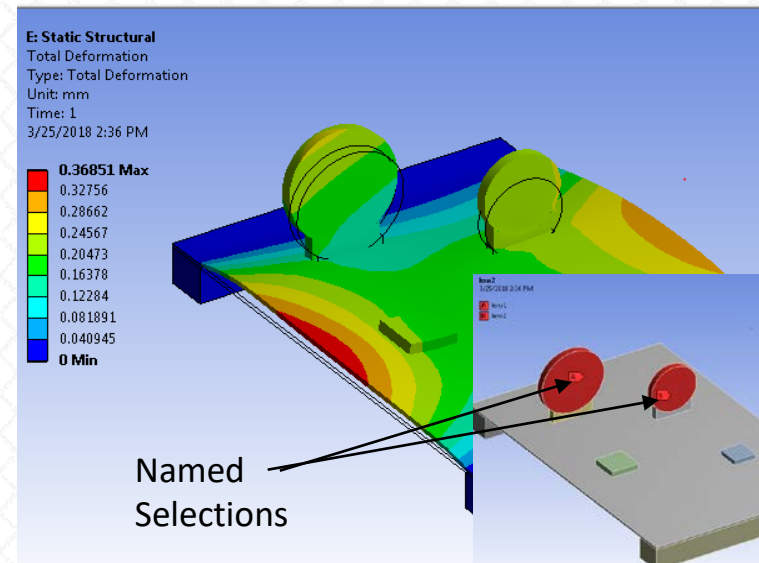


Part 1: Extract effective translations and rotations

- For use case 2, we'll most often want to calculate the effective CG motion of an entire body (volume) of interest. In use case 1, however, doing so would likely over-estimate the rigid motion. This is because we have no control over how much of a displacement field over a given body is dominated by rigid or deformable motion. However, we CAN restrict our inquiry to a named selection associated only with a connection



Use case 1: Named selection for CG motion restricted to link connection surface



Use case 2: Named selection encompasses entire volume(s) of interest

Part 1: Extract effective translations and rotations

- This means that for most problems that fall under use case 1, the named selection over which we want to calculate CG motion will be surface-based, which in turn means that we must perform our calculations over a nodal component in MAPDL
- For most problems that fall under use case 2, on the other hand, we'll want to use volume-based named selections, which correspond to element components in MAPDL
- The two calculations will slightly differ:

Calculating CG translations for nodal components

- Nodal component CG calculations will, in general, be less accurate than volume-based components because they are not weighted by local element volumes. This means (among other things) that differences in mesh refinement across the component are not accounted for in such calculations.
- Still, accuracy can be improved with mesh refinement

Nodal CG

$$\bar{\mathbf{x}} = \frac{\sum_{i=1}^N \mathbf{x}_i}{N}$$

Nodal CG
Translation

$$\bar{\mathbf{u}} = \frac{\sum_{i=1}^N \mathbf{u}_i}{N}$$

Part 1: Extract effective translations and rotations

Calculating CG translations for element components

- The element CG values are volume-weighted, as shown below

Element
component CG
Location

$$\bar{\mathbf{x}} = \frac{\sum_{i=1}^N \mathbf{x}_i \cdot dv_i}{\sum_{i=1}^N dv_i}$$

Element
component CG
Translation

$$\bar{\mathbf{u}} = \frac{\sum_{i=1}^N \mathbf{u}_i \cdot dv_i}{\sum_{i=1}^N dv_i}$$

Part 1: Extract effective translations and rotations

Calculating CG Rotations

- In all cases, CG rotations are calculated over the nodes contained in the named selection (whether a surface or body-based named selection)
- This is done with a least-squares procedure
- The small-rotation matrix dR is calculated by minimizing (1)

$$\left(dR \cdot (\mathbf{x} - \bar{\mathbf{x}}) - (\mathbf{u} - \bar{\mathbf{u}}) \right)^2 \quad (1)$$

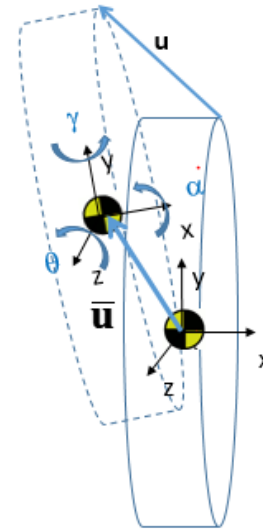
$$dR = \begin{pmatrix} 0 & -\theta & \gamma \\ \theta & 0 & -\alpha \\ -\gamma & \alpha & 0 \end{pmatrix}$$

\mathbf{u} Nodal displacement vector

$\bar{\mathbf{u}}$ CG displacement vector

\mathbf{x} Nodal position vector

$\bar{\mathbf{x}}$ CG position vector



Angle calculation:

- α (alpha): angle in yz plane
- γ (gamma): angle in xz plane
- θ (theta): angle in xy plane

Part 1: Extract effective translations and rotations

Calculating CG Rotations

- Solving for a small rotation matrix, dR allows us to keep the algorithm simple and linear
- Equation (1) will only yield reliable results for small angles (on the order of 5 degrees or less)
- Equation (1) says “when the effective CG translations are subtracted from the nodal displacements, what’s left is a rigid-body rotation about the CG”.
- But for use case 2, it will provide a ‘mean’ rotation of the CG (which is what we want in that case).
- But it also clearly states the need to select a nodal component associated ONLY with a connection for use case 1 (only then can we expect this statement to be useful)



Part 2: Subtract (or plot) rigid-body displacement field

- The second macro we need is something that allows us to visualize the rigid-body rotations calculated in 1 (use case 2), OR the displacements that result when they are subtracted from the full displacement field (use case 1)
- Use Case 2: Calculate the rigid-body displacement field, \mathbf{u}_{cg}

$$\mathbf{u}_{cg} = \mathbf{dR} \times (\mathbf{x} - \bar{\mathbf{x}}) + \bar{\mathbf{u}} \quad (2)$$

- Use Case 1: Calculate relative displacement field, \mathbf{u}_r

$$\mathbf{u}_r = \mathbf{u} - \mathbf{u}_{cg} \quad (3)$$

Macros mk_breldisp and mk_uvects

- The procedures we've outlined are captured in two general-purpose macros (APDL code), which may be used in Workbench as command objects. The two macros are:

First macro: mk_breldisp.mac

Purpose: To calculate the CG location and displacements (including rotation) as outlined in Part 1 previously

This macro is also issued first. The CG locations and displacements are both stored as variables and written out to file "ouput.txt"

Second macro: mk_uvects.mac

Purpose: To calculate and plot the resulting deformation field due to CG displacement calculated by mk_breldisp as outlined in Part 2. This contour plot may be written out to a png file viewable in Workbnnench



Macros mk_breldisp and mk_uvects

- A detailed description macro usage, as well as all inputs and outputs may be read in the headers of each macro
- For Workbench users, PADT recommends that both macros be stored in the user files folder of any WB project which needs them

Mk_breldisp.mac header

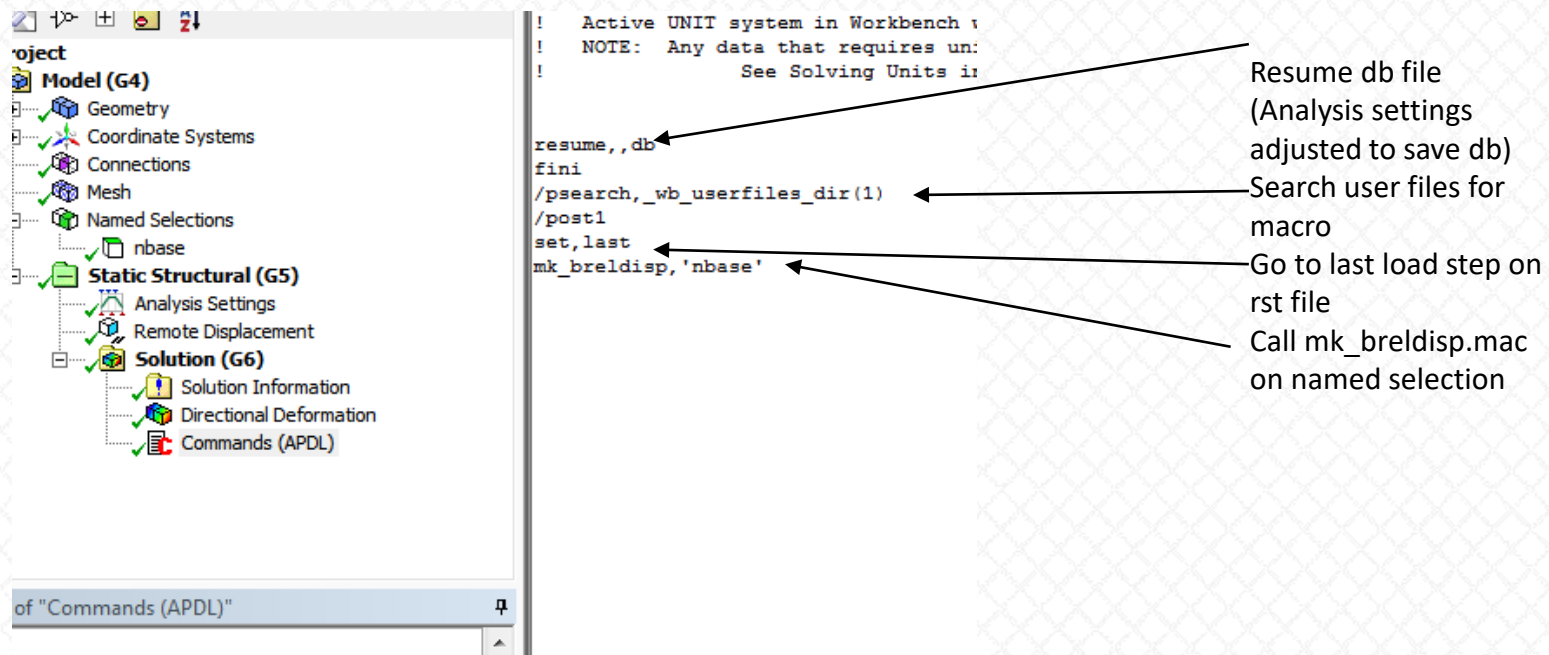
```
! Context: Post-processing
!
! macro usage: mk_breldisp,arg1
! Example: mk_breldisp,'nbase'
!
! Inputs
! argument      Description
! -----
! arg1          component name string (can be either nodal or element-based)
!               representing domain to calculate CG displacements for
!
! Outputs
! Variable      Description
! -----
! centX         Centroid x-location of arg1 (scalar)
! centY         Centroid y-location of arg1 (scalar)
! centZ         Centroid z-location of arg1 (scalar)
! CGUx          Centroid x-translation of arg1 (scalar)
! CGUy          Centroid y-translation of arg1 (scalar)
! CGUz          Centroid z-translation of arg1 (scalar)
! alpha         Centroid rotation about global x (yz plane) (scalar)
! gamma         Centroid rotation about global y (xz plane) (scalar)
! theta         Centroid rotation about global z (xy plane) ((scalar)
!
! Actions
! macro calculates output variables for domain represented
! by input variable and writes them to file "output.txt"
!
! Prerequisites
! None. Macro must be issued in general post-processor ('Solution' in WB)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Mk_uvects.mac header

```
! Context: Post-processing
!
! macro usage: mk_uvects,arg1,arg2
!
! Inputs
! argument      Description
! -----
! arg1          component name string representing volume over which to calculate
!               displacement field
! arg2          integer toggle: Whether to output rigid-body field or relative disp field
! 0             Relative displacements
! 1             rigid-body displacements
!
! Outputs
! Variable      Description
! -----
! urig_         N x 1 vector of rigid-body displacements. N is number of nodes in model
! udif_         N x 1 vector of relative displacements. N is number of nodes in model
!
! Actions
! macro calculates output variables given the input component name,
! type of output, and existence of alpha, gamma, theta. To ensure these
! exist, it is assumed the user has already run mk_breldisp.
! Macro then plots the requested variable (urig_ or udif_) in MAPDL
!
! Prerequisites
! Macro must be issued in general post-processor ('Solution' in WB)
! The following variables must be nonzero (typically calculated by mk_breldisp)
! Variable      Description
! -----
! alpha         Rotation of arg1 to breldisp about global x (yz plane)
! gamma         Rotation of arg1 to breldisp about global y (xz plane)
! theta         Rotation of arg1 to breldisp about global z (xy plane)
! centX         X-location of centroid of arg1 to breldisp
! centY         Y-location of centroid of arg1 to breldisp
! centZ         Z-location of centroid of arg1 to breldisp
! CGUx          X-displacement of centroid of arg1 to breldisp
! CGUy          Y-displacement of centroid of arg1 to breldisp
! CGUz          Z-displacement of centroid of arg1 to breldisp
!
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```


Examples

- In what follows, we demonstrate the use of these macros for some instructive examples in Workbench (R19). To follow our own recommendation, we place these macros in the user files folder of the Workbench Project we're using in all cases
- An additional detail for Workbench users: Adjust the Analysis Settings (under Analysis Data Management) to always 'Save the MAPDL db' file (check 'Yes')
- We'll demonstrate the use of these macros with an example of use case 1 and 2. But first we want to show that the macro's angle calculation is accurate



Example 1. Angle Verification

- Problem Statement: Rotation is applied via remote displacement to surface named selection shown

Project Tree:

- Project
 - Model (G4)
 - Geometry
 - Coordinate Systems
 - Connections
 - Mesh
 - Named Selections
 - mbase
 - Static Structural (G5)
 - Analysis Settings
 - Remote Displacement
 - Solution (G6)
 - Solution Information
 - Directional Deformation
 - Commands (APDL)

Details of "Remote Displacement"

Scope	
Scoping Method	Geometry Selection
Geometry	6 Faces
Coordinate System	Global Coordinate System
<input type="checkbox"/> X Coordinate	-1.27 mm
<input type="checkbox"/> Y Coordinate	0. mm
<input type="checkbox"/> Z Coordinate	0. mm
Location	Click to Change

Definition	
Type	Remote Displacement
<input type="checkbox"/> X Component	0. mm (ramped)
<input type="checkbox"/> Y Component	0. mm (ramped)
<input type="checkbox"/> Z Component	0. mm (ramped)
<input type="checkbox"/> Rotation X	3.12 ° (ramped)
<input type="checkbox"/> Rotation Y	-1.8 ° (ramped)
<input type="checkbox"/> Rotation Z	2.25 ° (ramped)

3D Model View:

Remote Displacement
Components: 0., 0., 0. mm
Rotation: 3.12, -1.8, 2.25 °
Location: -1.27, 0., 0. mm

Rotation X: 3.12°
Rotation Y: -1.8°
Rotation Z: 2.25°

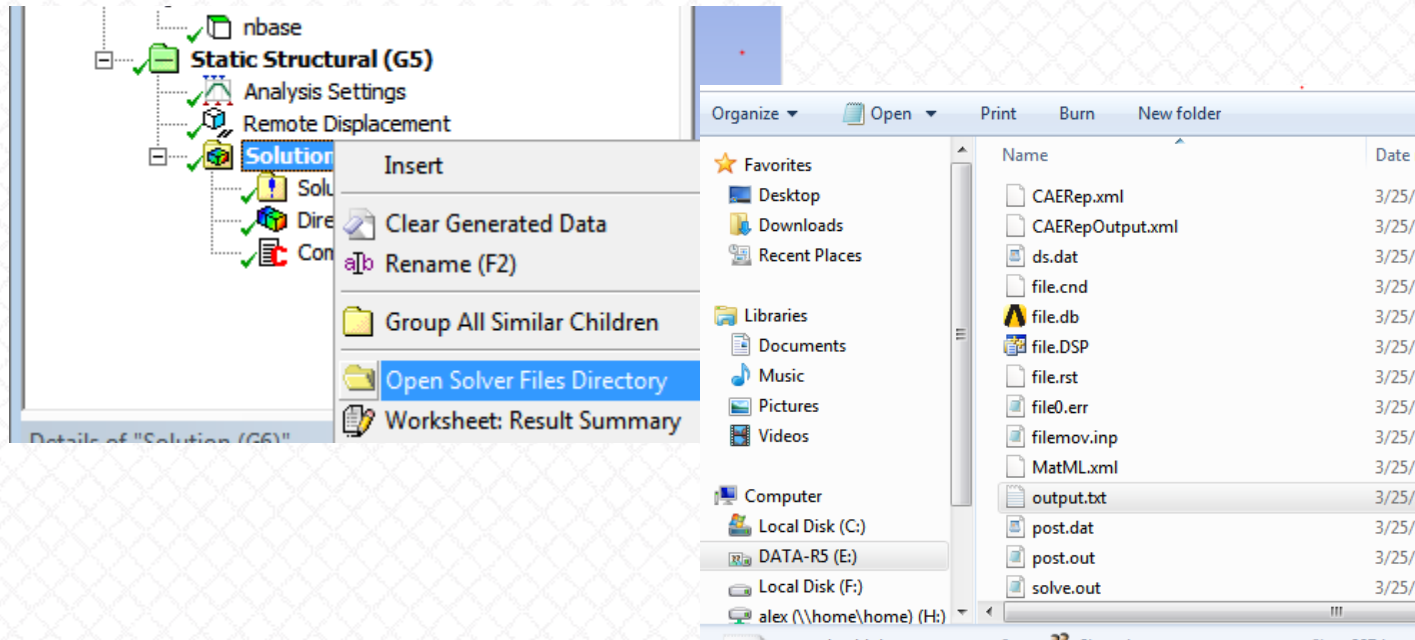
Mk_breldisp.mac macro called (from user files location)

Graph:

1

Example 1. Angle Verification

- When run: mk_breldisp creates output (output.txt) summarizing CG calculations. This resides in the Solver Files folder



- Calculated angles match applied values to within available precision

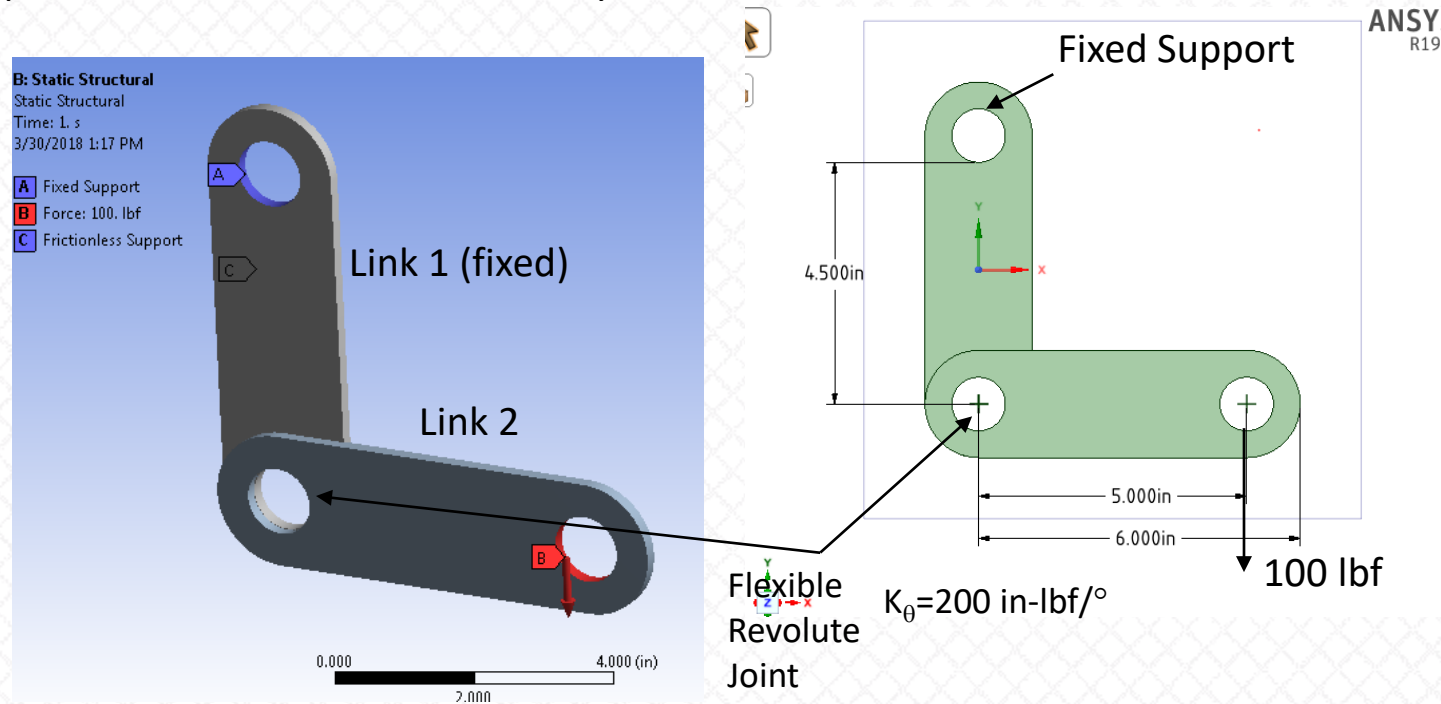
output.txt - Notepad

Comp. Name	centX	centY	centZ	CGux	CGuy	CGUZ	alpha	gamma	theta
nbase	-1.270	0.067	0.045	-.4066E-02	-.2458E-02	0.3672E-02	3.120	-1.800	2.250

Rotation X points to alpha, Rotation Y points to gamma, Rotation Z points to theta

Example 2. Use Case 1: Link Analysis

- Next, we'd like to use the two macros to determine the purely strain-based displacement of link 2 in a 2-link analysis

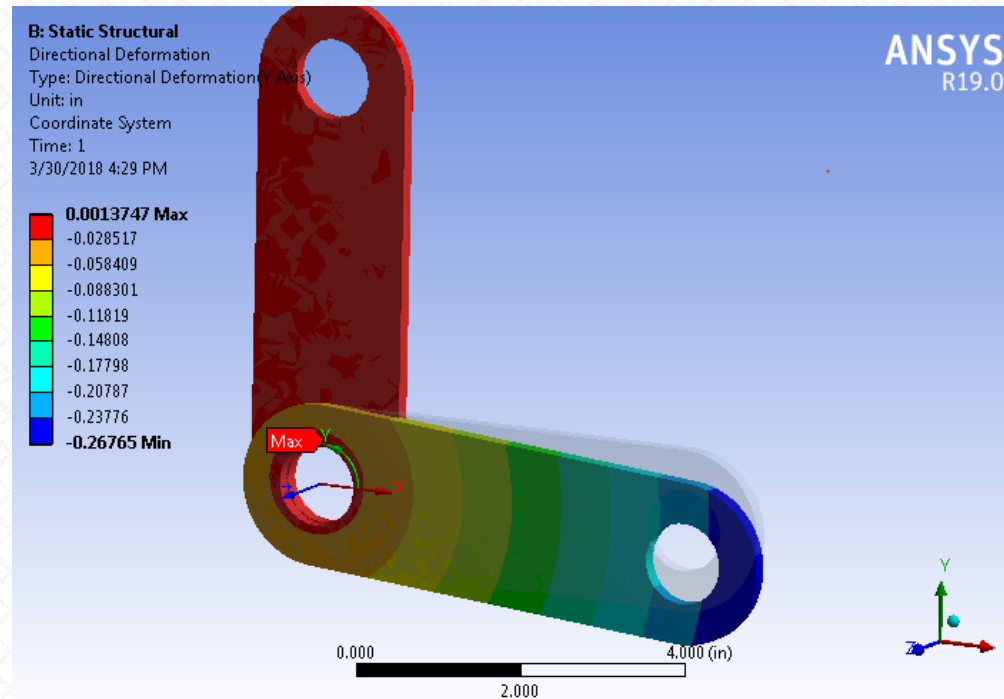


We want to know the 'relative' displacement of link 2 (relative to the connection to link 1)

- In this example, we can quickly estimate the rigid angular deflection to be $\theta = -100 \cdot 5.0 / 200 = -2.50^{\circ}$

Example 2. Use Case 1: Link Analysis

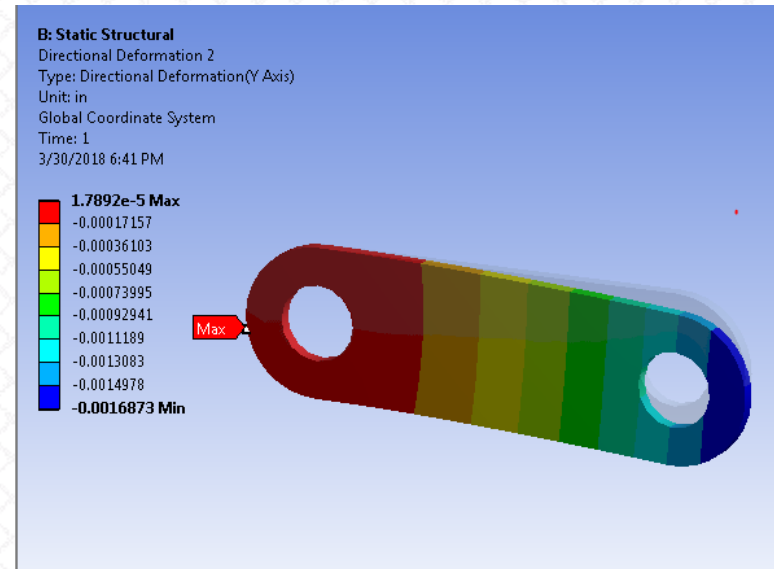
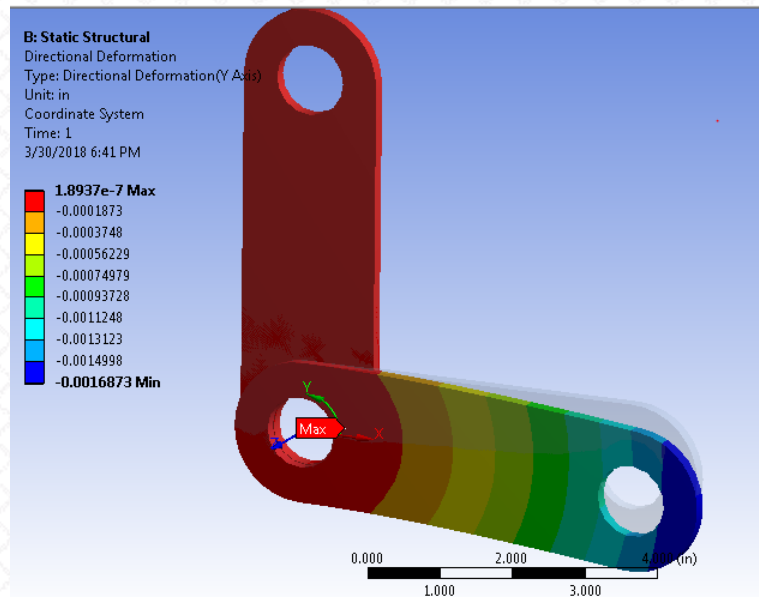
- When we perform the analysis, we're off by ~2.3% (0.0567 degrees)
- This shouldn't be surprising, but it should lead us to suspect that 2.3% of the rotation comes from strain-based deflection (mostly bending, in this case)
- We'll check this two different ways



$$\theta \approx \sin^{-1}(-.26765/6.0) = -2.56^\circ$$

Example 2. Use Case 1: Link Analysis

- As a first check, we'll perform another analysis to get an estimate along the lines recommended in slide 5
- We'll suppress the revolute joint, and instead fix the attachment point of link 2
- Then re-run the analysis...

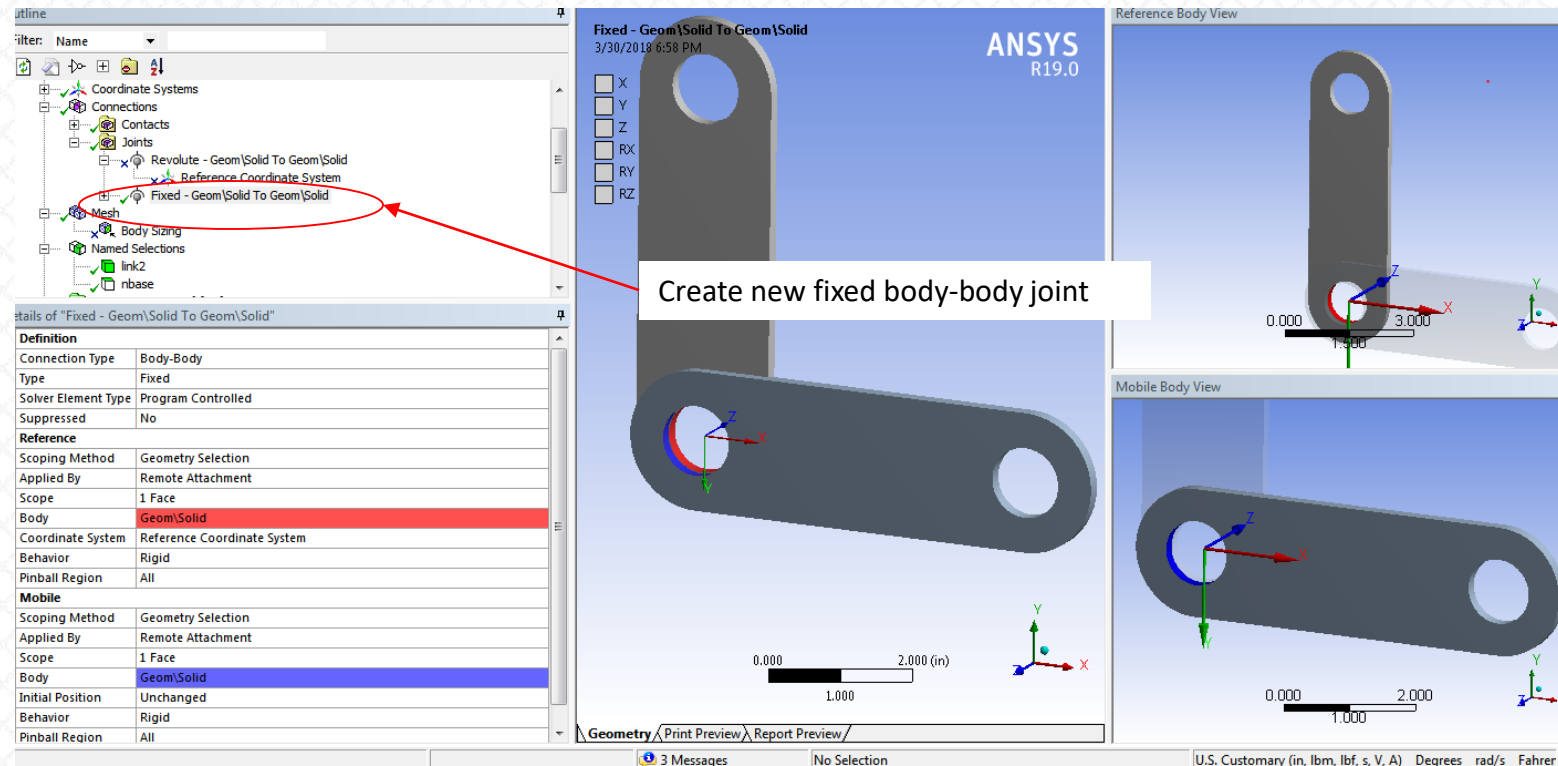


$$\theta \approx \sin^{-1}(-.0016873/6.0) = -0.01611^\circ$$

- This still doesn't quite account for the difference between the rigid-body rotation estimate and the total (-2.5° vs. -2.56°). What's going on?

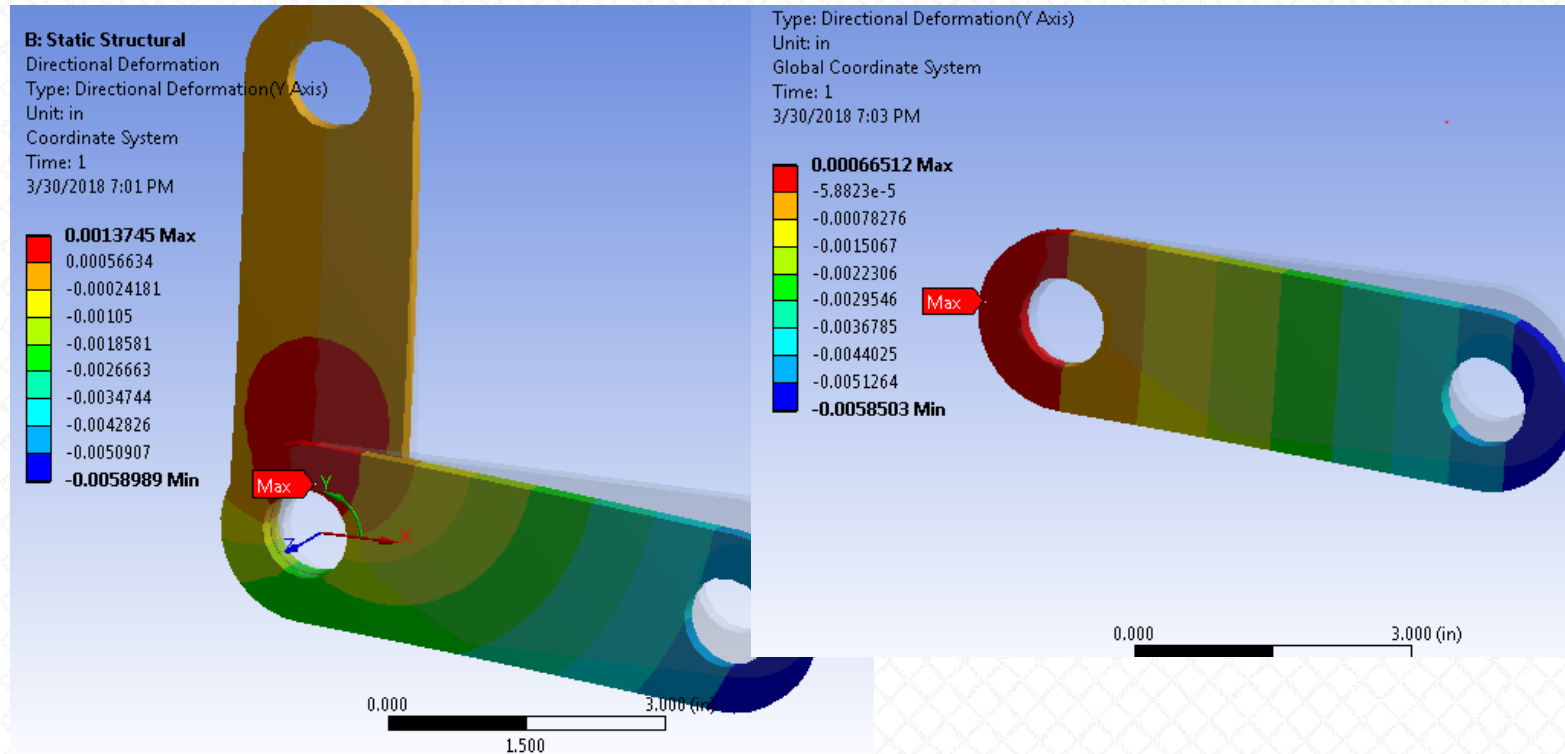
Example 2. Use Case 1: Link Analysis

- To find out what's going on, first suppress the fixed support at the connection surface of link 2 and create a new 'fixed' body-to-body joint between the two link connection surfaces as shown (make sure the behavior is 'rigid')
- ..and re-run the analysis



Example 2. Use Case 1: Link Analysis

- Note the difference in deflection (with slide 22)
- To recap: This is the difference between using a fixed body-body joint and fixing the connection surface of link 2



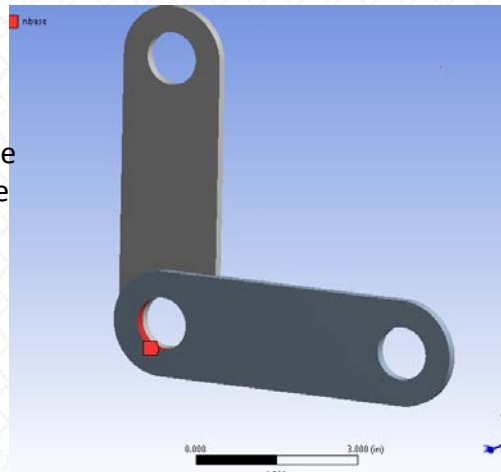
- It appears that using joints vs. directly applied fixity conditions accounts for the difference...!

$$\theta \approx \sin^{-1}(-.00585/6.0) = -0.05586$$

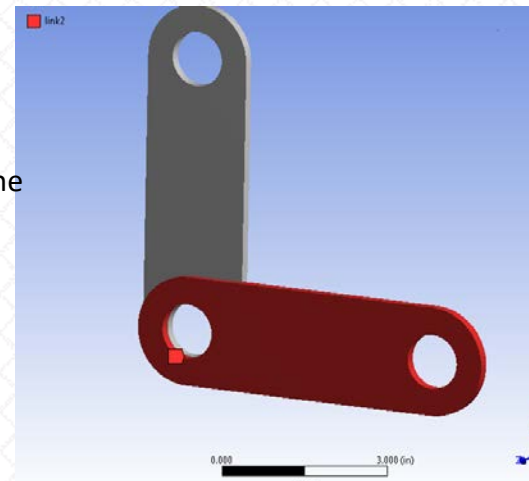
Example 2. Use Case 1: Link Analysis

- The previous slide demonstrates that using joints and other connections between bodies introduces additional flexibility (perhaps not always accounted for). We can conclude from the previous slide that SOME non-zero rotation must still be occurring in the connection surface of link 2 when the fixed body-body joint is applied. Let's check this with our macros!
- First, let's suppress the fixed body-body joint and un-suppress the revolute again
- Next, we'll make a named selection for the connection surface of link 2. We'll calculate the rigid-body motion purely from this surface (so as to eliminate the strain-based deflection from the estimate)
- Then, we'll make a named selection out of the link 2 body. We'll call this as an element component in the 'uvects' macro to make a plot of the strain-based displacement field.

Make a named selection out of the connection surface (we'll call it 'nbase')

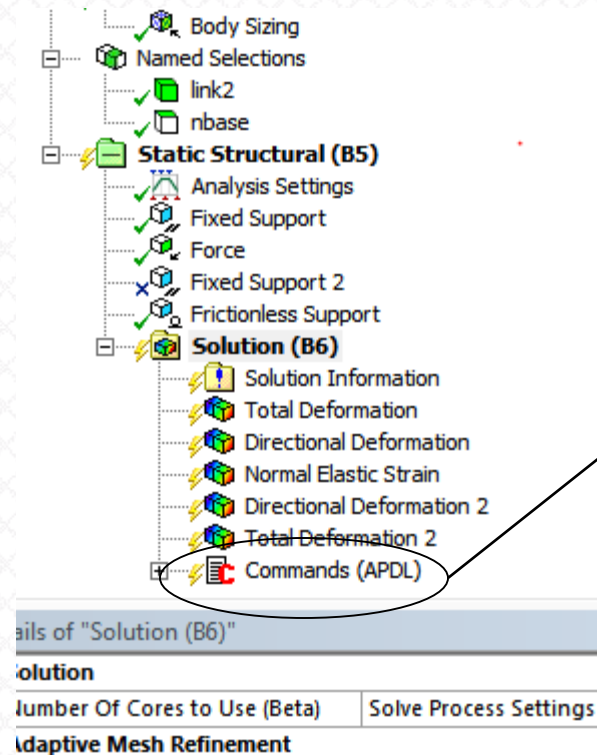


Make a named selection out of the link 2 body (we'll call it 'link2')



Example 2. Use Case 1: Link Analysis

- Then, call the two macros in a command object in WB...

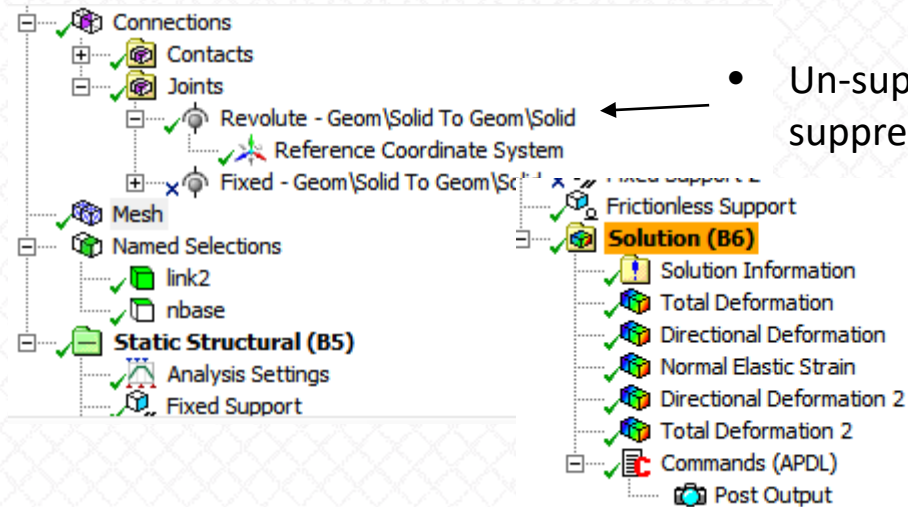


```
resume,,db
fini
/psearch,_wb_userfiles_dir(1)
/post1
set,last
mk_breldisp,'nbase'
mk_uvects,'link2',0
/show,png
plnsol,u,y
/show,foo
/show,term
```

- You can cut-and-paste the script above into your command object
- Remember: mk_breldisp and mk_uvects must reside in the user_files folder to work this way

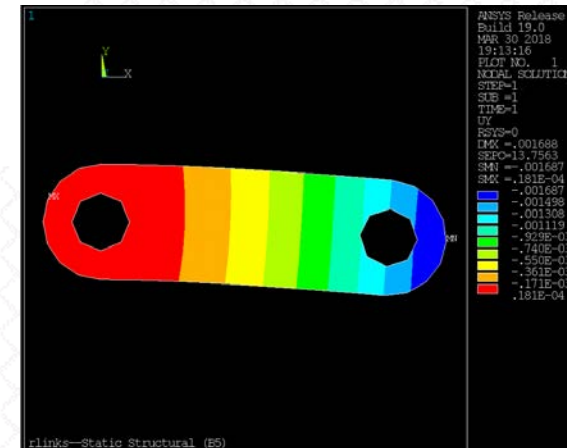
Example 2. Use Case 1: Link Analysis

- Re-run the analysis. Note that the command object we inserted creates a new image called 'Post Output'. This is the relative (strain-based) displacement calculated by mk_uvects
- Click on it to see this displacement field...



- Un-suppress the revolute joint and suppress the fixed body-body joint

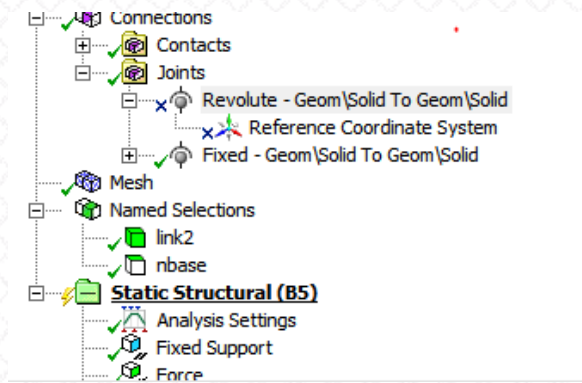
- Min. displacement is - 0.001687"



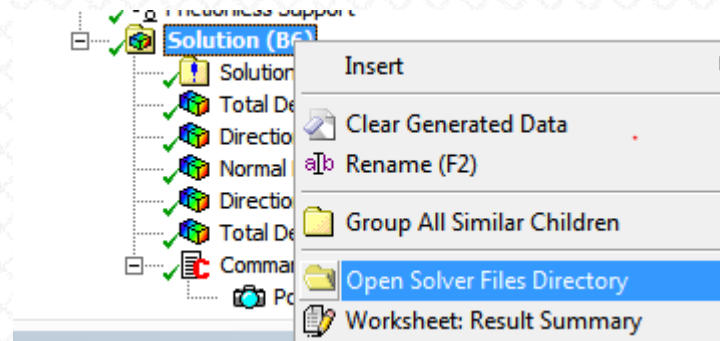
- Note that this is the same displacement we calculated by fixing the link 2 connection surface directly!
- That's encouraging! We don't want our estimates to depend on the nature of body-body connections (and we don't have to perform a second analysis to obtain this result).

Example 2. Use Case 1: Link Analysis

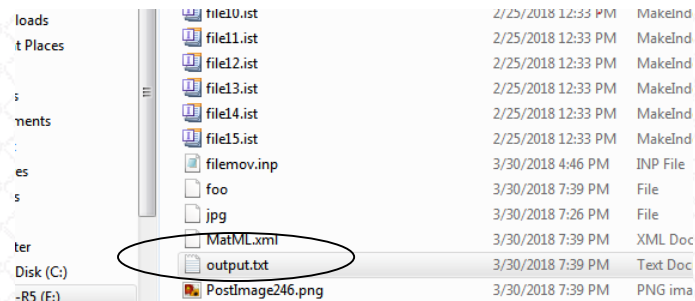
- Let's do a further check. This time, we want our macro to estimate the amount of additional rotation introduced by the flexibility of the fixed body-body joint
- Suppress the revolute joint and un-suppress the fixed body-body joint
- Re-run



- Right-click on 'Solution' in the tree outline...

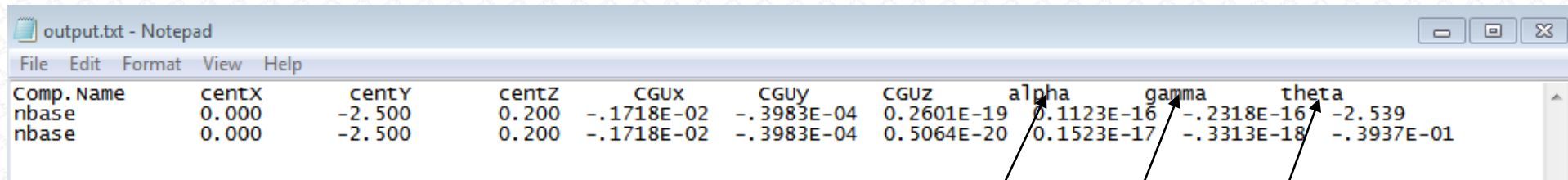


- ...and open the file 'output.txt'. This is the output of macro mk_breldisp



Example 2. Use Case 1: Link Analysis

- The output file contains the output macro mk_breldisp for the last two runs (it appends to the same file after every run). Reading from left-to-right, we see the calculations for the x, y, and z centroid location and displacements, followed by the three orthogonal (infinitesimal) rotation components for the named selection associated with link 2's connection surface.



Comp. Name	centX	centY	centZ	CGUX	CGUY	CGUZ	alpha	gamma	theta
nbase	0.000	-2.500	0.200	-.1718E-02	-.3983E-04	0.2601E-19	0.1123E-16	-.2318E-16	-2.539
nbase	0.000	-2.500	0.200	-.1718E-02	-.3983E-04	0.5064E-20	0.1523E-17	-.3313E-18	-.3937E-01

Rotation X

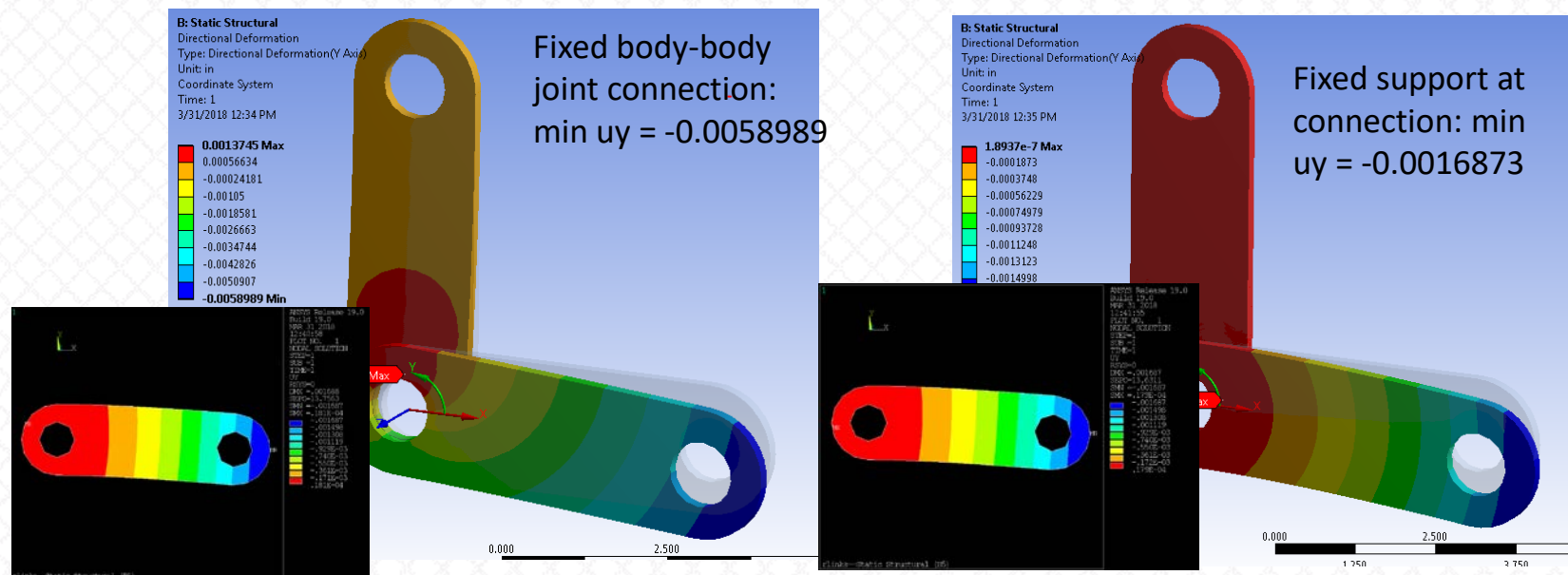
Rotation Y

Rotation Z

- The first row corresponds to the results of the analysis with the flexible revolute joint. The second row corresponds to the results with the fixed body-body joint
- Notice the absolute magnitude of the calculated angle (about the z-axis) in both cases is 0.039° higher than it should be (we couldn't be sure with a single data point if this was purely due to the joint stiffness. However, the two points data points taken together provide strong evidence of this)!

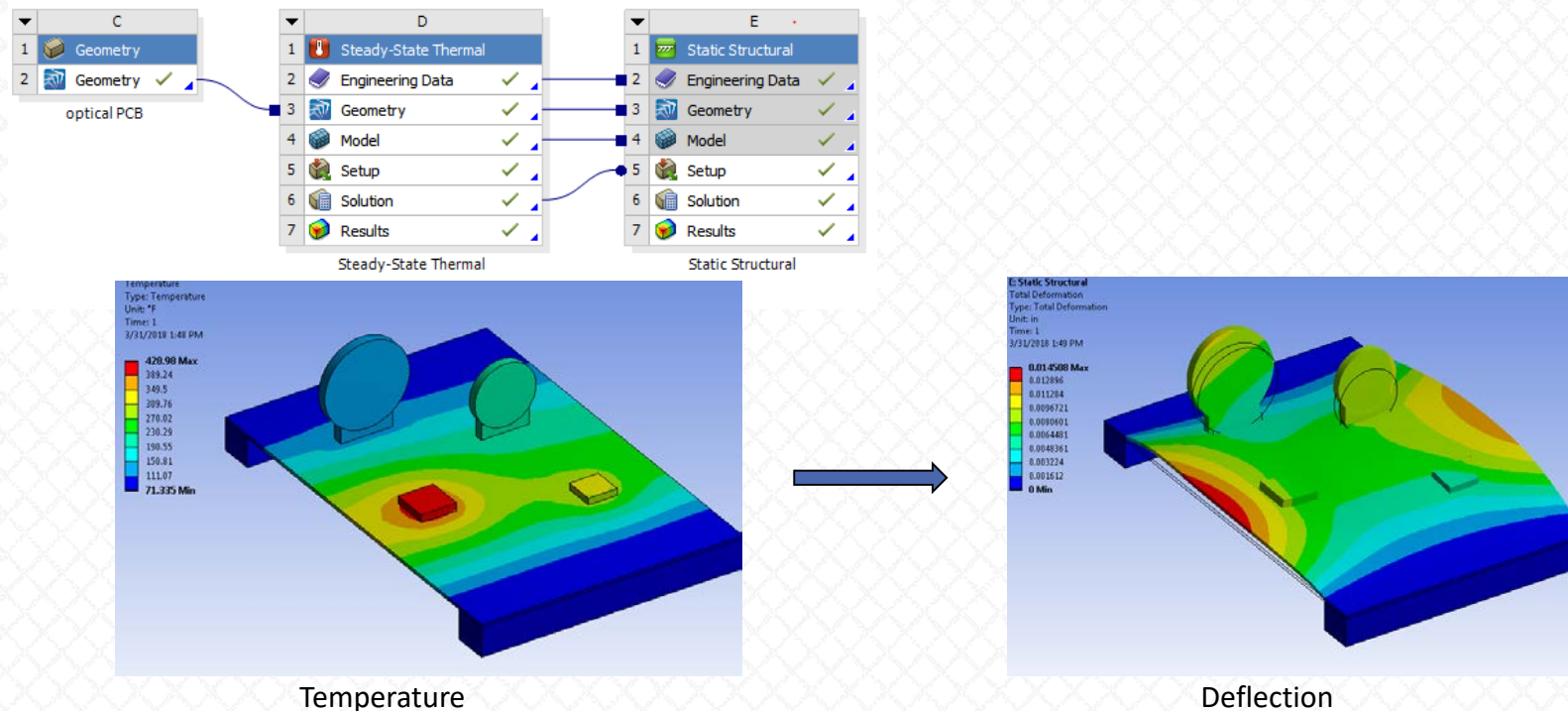
Example 2. Use Case 1: Link Analysis Summary

- The use of macro mk_breldisp has enabled us to discover an additional, typically unaccounted for, stiffness inherent in all body-body connections (even 'rigid' ones)
- A rough estimation of the resultant rotation for named selection 'nbase' for a fixed body-body connection would have yielded zero, while the flexible revolute joint should result in a rotation of -2.5°. Instead, we get -0.039° and -2.539°, respectively.
- This may seem like a small (and in many cases, acceptable) error, but notice the difference this makes in the Workbench deflection plots below.
- Still, macro mk_bredisp and mk_uvects calculate the 'correct' angles and relative displacement EVERY TIME (to within the available precision)



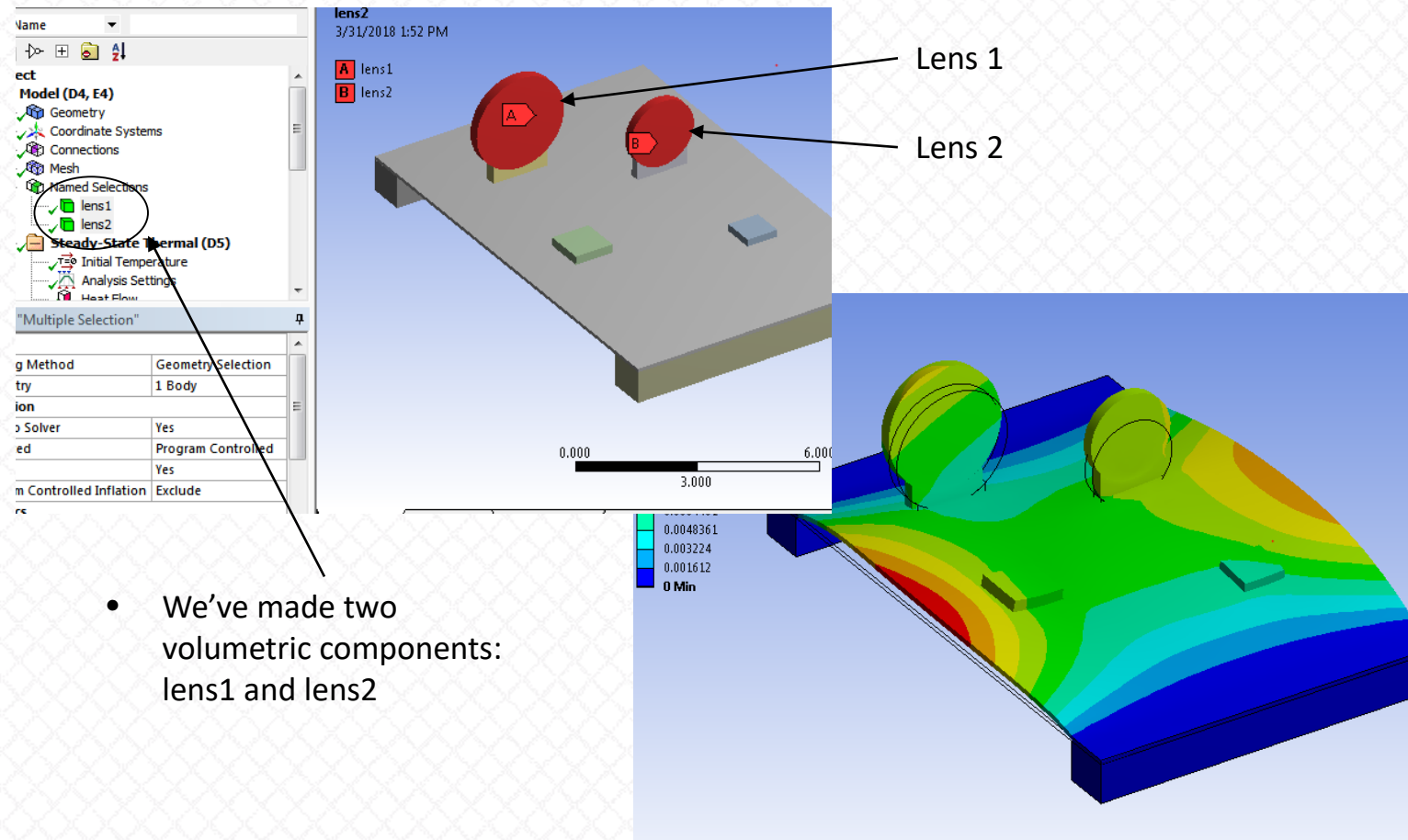
Example 3. Use Case 2: Mean Rotation

- Our third example involves use case 2: the determination of a body's mean, or equivalent (centroidal, or volume-weighted) rotation
- We have an assembly (in this case, some approximation of an electronics circuit board) fitted with two lens components (let's pretend this is some sort of optical device). The board is subjected to thermal loading. This results in deflection (warpage) due to disparities in the Coefficient of Thermal Expansion (CTE) in the various components.



Example 3. Use Case 2: Mean Rotation

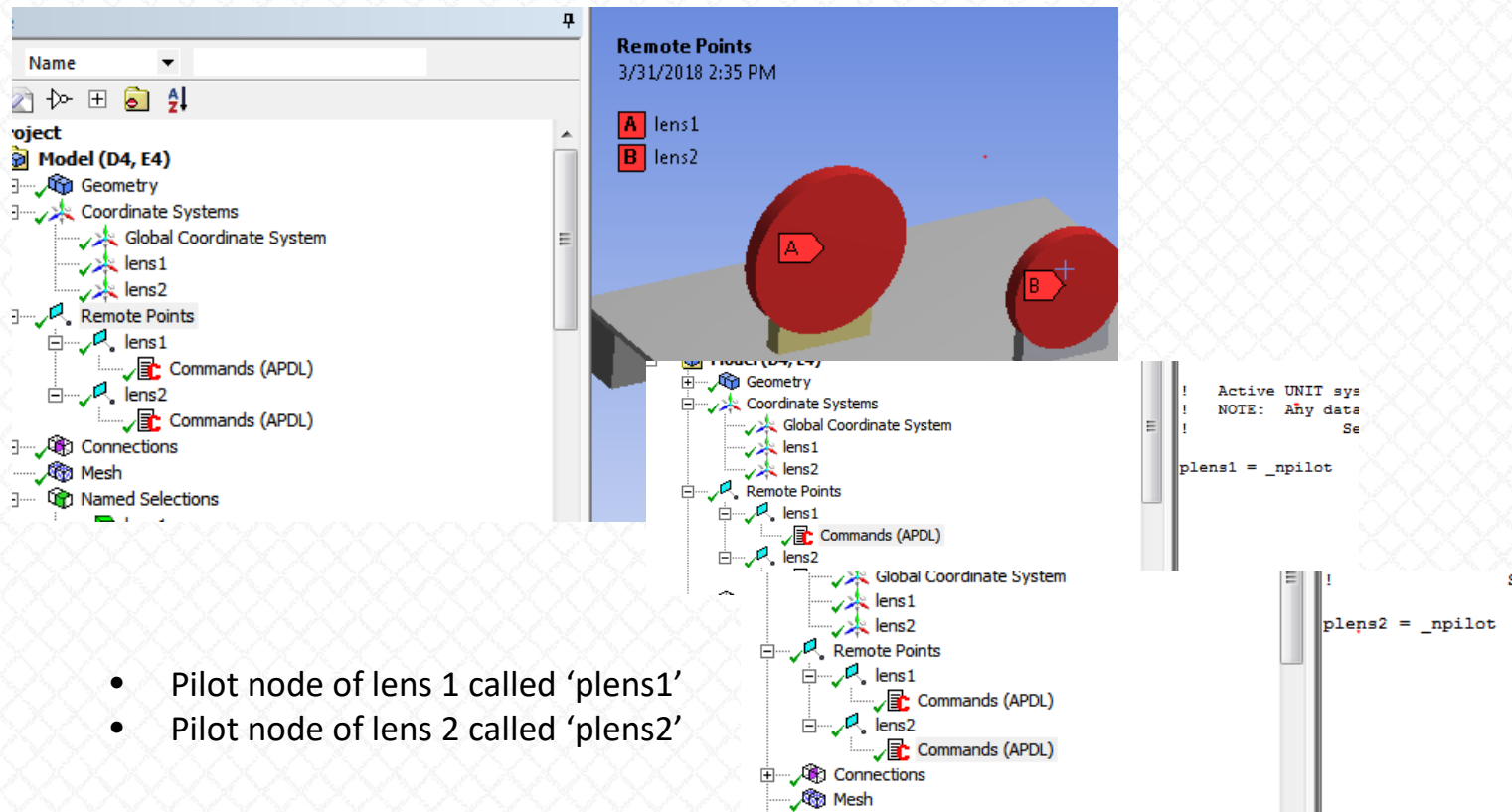
- We're interested in the mean rotation of the two lens components shown below



- We've made two volumetric components: lens1 and lens2

Example 3. Use Case 2: Mean Rotation

- This time, we'll use the ANSYS RBE3 technology to check our result
- This is achieved in WB by inserting a deformable remote point at the center of each lens and attaching it to the lens surfaces as shown below
- We insert command objects to capture the node numbers of the pilot node on each remote point



- Pilot node of lens 1 called 'plens1'
- Pilot node of lens 2 called 'plens2'

Example 3. Use Case 2: Mean Rotation

- Remote points use the RBE3 constraints via the MPC option (they are actually defined using surface-to-surface contact in ANSYS)
- The rotational degree of freedom of the pilot node is obtained in a way identical to what macro mk_breldisp does (least squares)
- We expect some differences, however, because the pilot is attached to surfaces rather than an entire volume. In most cases, this difference will be small (and will diminish with increasing refinement). However, if the mean rotation of a composite body is sought, these differences can become great
- Still, in our case, we expect this solution to yield results very similar to ours

From the R19 help documentation:

RBE3 creates constraint equations such that the motion of the master is the average of the slaves. For the rotations, a least-squares approach is used to define the "average rotation" at the master from the translations of the slaves. If the slave nodes are colinear, then one of the master rotations that is parallel to the colinear direction can not be determined in terms of the translations of the slave nodes. Therefore, the associated moment component on the master node in that direction can not be transmitted. When this case occurs, a warning message is issued and the constraint equations created by **RBE3** are ignored.

Applying this command to a large number of slave nodes may result in constraint equations with a large number of coefficients. This may significantly increase the peak memory required during the process of element assembly. If real memory or virtual memory is not available, consider reducing the number of slave nodes.

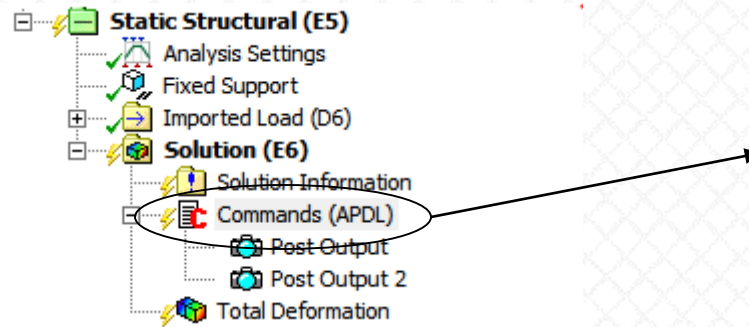
As an alternative to the **RBE3** command, you can apply a similar type of constraint using contact elements and the internal multipoint constraint (MPC) algorithm. See [Surface-based Constraints](#) for more information.

This command is also valid in SOLUTION.



Example 3. Use Case 2: Mean Rotation

- We'll insert a command object which both runs our macros (mk_breldisp and mk_uvects) AND checks the result of the pilot node rotations on the remote points for confirmation



- Users may freely cut-and-paste this code into their command object. Remember to place mk_breldisp and mk_uvects in the user_files folder

```
resume,,db
fini
/psearch,_wb_userfiles_dir(1)
/post1
set,last
mk_breldisp,'lens1'
mk_uvects,'lens1',1
/show,png
/view,1,1,1,1
/auto,1
plnsol,u,sum
mk_breldisp,'lens2'
mk_uvects,'lens2',1
/view,1,1,1,1
/auto,1
plnsol,u,sum
/show,foo
/show,term
```

```
pi = 3.1415926
my_lens1rx = ROTX(plens1)*180/pi
my_lens1ry = ROTY(plens1)*180/pi
my_lens1rz = ROTZ(plens1)*180/pi
```

```
my_lens2rx = ROTX(plens2)*180/pi
my_lens2ry = ROTY(plens2)*180/pi
my_lens2rz = ROTZ(plens2)*180/pi
```


Example 3. Use Case 2: Mean Rotation

- When the run the analysis (and the command object), we get the following results

Details of "Commands (APDL)"		
<input type="checkbox"/> ARG5		
<input type="checkbox"/> ARG6		
<input type="checkbox"/> ARG7		
<input type="checkbox"/> ARG8		
<input type="checkbox"/> ARG9		
Results		
<input type="checkbox"/> my_lens1rx	-0.14086	Lens 1 Rotation X
<input type="checkbox"/> my_lens1ry	3.9311e-002	Lens 1 Rotation Y
<input type="checkbox"/> my_lens1rz	-2.335e-002	Lens 1 Rotation Z
<input type="checkbox"/> my_lens2rx	-4.7873e-002	Lens 2 Rotation X
<input type="checkbox"/> my_lens2ry	-4.0349e-002	Lens 2 Rotation Y
<input type="checkbox"/> my_lens2rz	1.4229e-002	Lens 2 Rotation Z

Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL
Input Arguments	
<input type="checkbox"/> ARG1	
<input type="checkbox"/> ARG2	
<input type="checkbox"/> ARG3	
<input type="checkbox"/> ARG4	

- Compare with the results of macro `mk_breldisp`. Our expectations hold (close, but not exactly the same) !

- The results of the rotations at the pilot node of lens1 and lens2 show up in the details view of the command object
- Recall that WB brings back variables calculated in MAPDL scripts according to a 'Search prefix' (the default is 'my_')

							Rotation Y		
							Rotation X	Rotation Y	Rotation Z
Comp. Name	centX	centY	centZ	CGUX	CGUY	CGUZ	alpha	gamma	theta
lens1	2.520	9.292	1.588	0.2526E-03	0.4819E-02	0.5758E-02	-.1406	0.3927E-01	-.2335E-01
lens2	5.537	7.512	1.197	0.7519E-03	0.2059E-02	0.8899E-02	-.4784E-01	-.4009E-01	0.1427E-01

Conclusions

- The macros used in this study: mk_breldisp.mac, and mk_uvects.mac, allow ANSYS users to calculate mean displacements of named selections as a post-processing operation
- Users CAN always make similar estimates with RBE3 technology (as described here: <https://www.simutechgroup.com/tips-and-tricks/fea-articles/147-fea-tips-tricks-rotation>), but this requires the creation of additional specialized (MPC) elements
- The macro mk_uvects provides users with the unique capability of obtaining contour plots of just the (rigid) deformation field associated with CG motion of a body, OR the relative (total minus CG displacement) deformation field of a named selection
- The macros may be inserted into any WB project. If the Analysis Settings have been adjusted to save the MAPDL db file, no additional analysis need be performed: the two macros operate in the general post-processor and require no additional solutions.

