# M1M3 Shape in Surface Sag and Surface Normal

## Contents



## 1. Introduction

When it comes to determining the surface shape of a mirror as a function of actuator forces, we use the Finite Element Analysis (FEA). The FEA can tell us, given a set of actuator forces, how each node on the surface is deflected in 3D space. But it doesn't tell us directly how the surface sag changes as a function of mirror coordinates. Defining the x-y plane to be the back surface of the mirror, and the z-axis pointing toward M2 and camera, the surface sag defines where the rays coming along –z direction first hit the glass. This is the surface that is relevant for ray tracing purposes, therefore the surface that matters for the image quality on the focal plane. On the other hand, for optical testing using interferometry, the incoming rays from the interferometer are always along the surface normal. In this technical note, we discuss how we project the 3D surface deflection onto surface sag and surface normal. We also give the M1M3 bending modes in surface sag and surface normal.

Figure 1 shows how the 3D surface displacement is related to its various projections. The upper blue curve is the real mirror surface. The lower blue curve is the ideal mirror surface as designed. For ray tracing programs, the important thing is the surface sag (AD). While for interferometer test of the mirror, we need to know the displacement AF, which is along surface normal. This technical note describes how we start from the raw output of the Finite Element Analysis (FEA), which are 3D vectors (AB in Figure 1), to get surface sag (AD) and surface normal displacements (AF).

The mathematics behind the generation of bending modes can be found in Document-15312. In that note, George Angeli explains the general framework of generating bending modes, and the need to use balanced unit load force vectors. In Document-15338, Ed Hileman describes the technical details in calculating the LSST M1M3 bending modes using the axial component (z-projection) of the 3D surface deflection.

The FEA data that we start with are the same data that were used in Document-15338, i.e., for each of the balanced unit load force vectors, we have the x, y, and z displacement for the 5256 FEA nodes. In addition, we need the balanced unit load force vectors themselves, and the x and y coordinates of each FEA node. For visualization purposes, we also need the position of each actuator. These actuators positions are not required for the analysis itself, as long as we always order the actuators the same way.

The analysis described in this note assumes the mirror has been polished to its perfect shape (as defined by the LSST optical model) in outer space, and then placed under the effect of gravity and the actuators. This means there is zero polishing error, a zenith print through is not polished into the mirror, no forces is needed to bend out the low-frequency modes due to polishing, and no additional forces is necessary to correct the mirror shape if the supporting actuators during polishing has not be optimized. The assumptions have no effect on the surface sag and surface normal projection calculations, nor do they affect the bending modes. They do have effects on the optimization of the forces, therefore the optimized surface shape.

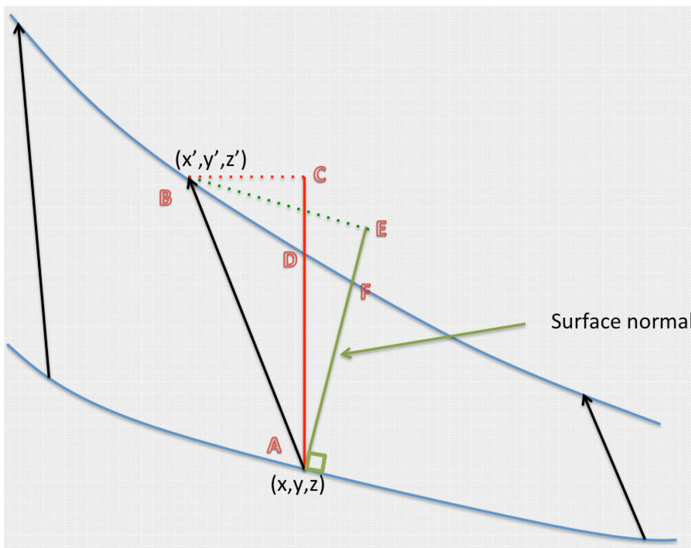


**Figure 1 Schematic drawing of the various projections of the 3D deflection vectors.**

## 2. M1M3 Shape in Surface Sag

### 2.1. Calculation of Surface Sag

Since we have 5256 3D coordinates (x', y', z') on the real mirror surface, in principle, we know everything about this surface within the given resolution. Therefore, the most straightforward way to calculate the surface sag is to interpolate the real surface to (x, y) positions. There are some limitations to this method, however. First, the interpolation is relatively computationally expensive. Second, the discontinuity at the edges of the substrate and especially between M1 and M3 could present a challenge to the interpolation algorithm.

The method we use, is to analytically calculate the difference in z at (x, y) and (x',y'), on the ideal mirror surface (see Figure 1), then subtract it from z'. This requires minimal computation, and doesn't lose much accuracy. Since for each FEA node, we know without ambiguity whether it is on M1 or M3, the right surface slope can always be used. The underlying assumption in this method is that the real mirror surface can be considered parallel to the ideal surface on the scale defined by (x'-x, y'-y). This is valid, considering that our mirror is 8.4m in diameter, and the typical surface deformation we deal with is on the level of nanometers. Nevertheless, we will provide one verification example in Section 2.2. Alternatively, we could choose to assign the new coordinates (x',y') to the surface sag at each node calculated as described above. In this case, there is no approximation involved. However, when we calculate the bending modes, each balanced unit load case would have a set of different (x',y'). We therefore choose to ignore the difference between (x,y) and (x',y') in defining the surface sag.

The optical prescription of the ideal M1M3 surface is

$$z = \frac{cr^2}{1+\sqrt{1-(1+k)c^2r^2}} + \sum_{i=1}^{8} \alpha_i r^{2i} \tag{1}$$

For M1, the curvature of radius c=1/19835 mm$^{-1}$ †, the conic constant k= -1.215,

$$\alpha_i = \begin{matrix} -1.38\times10^{-24} & \text{for i=3} \\ 0 & \text{else} \end{matrix} \tag{2}$$

For M3, the curvature of radius c=1/8344.5 mm$^{-1}$, the conic constant k=0.155,

$$\alpha_i = \begin{matrix} -4.5\times10^{-22} & \text{for i=3} \\ -8.2\times10^{-30} & \text{for i=4} \\ 0 & \text{else} \end{matrix} \tag{3}$$

The difference between the surface sag and the z-component of the 3D vector is determined by the x- and y- node displacements, together with the local slope of the mirror surface. We sometimes refer to this calculation as the "slope-correction" approach.

## 2.2. Verification

To numerically verify the assumption on the local parallelism is good enough, and to verify our surface sag calculation, we take one mirror state as the example, where the 3D node deflections are known for each surface node. We then compare the difference between the surface sag using our slope-correction approach to what we get by directly interpolating the surface determined by (x',y',z') to the (x,y) coordinates. For this verification example we give here, the telescope is at zenith pointing. The 256 actuator forces have been adjusted to achieve the minimal RMS of the z-displacements. More about this mirror state is discussed in Section 2.4. Here we focus on the calculation of the surface sag itself, without changing the actuator forces to optimize the surface.

Figure 2 left shows the correction, i.e., the difference tween the surface sag and the z-displacement. Figure 2 right shows the difference in surface sag calculated using the two approaches. The surface map on the left has RMS of 48.7nm, while the one on the right has RMS of 1.4nm. The difference is about 2.9% to the total correction. We therefore consider our slope-correction approach to be accurate.

† This is the radius of curvature for operation at the summit. Same for the M3 radius of curvature given. The numbers for manufacturing at SOML are slightly different (19835.5mm vs. 19835 for M1, and 8344.7mm vs. 8344.5mm for M3).

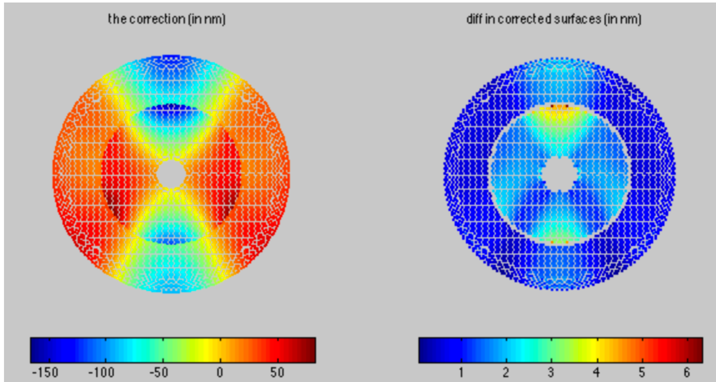


**Figure 2 For zenith pointing z-optimized surface, Left: the difference between Surface sag and the z-component of the 3D deflection vectors; Right: the difference between the two methods of calculating the surface sag.**

## 2.3. Surface Sag Bending Modes

To get the surface sag bending modes, we take the force vectors for the 256 balanced unit loaded cases (a 256x256 matrix), and the 3D displacements of each of the 5256 surface nodes under the influence of each balance force vectors (three 5256x256 matrices). The general procedure for obtaining the surface sag bending modes is the same as described in Document-15312 by George Angeli and Document-15338 by Ed Hileman. The only exception is that Instead of using only the z-component of the 3D node deflections and ignoring those for x and y, we first use the 3D vectors to calculate the surface sag of each of the 5256 surface nodes under the influence of each balance force vectors.

Note that we calculate the surface sag before we do the Singular Value Decomposition (SVD). If we choose to do the SVD first, the SVD will be on a 15768x256 matrix. Bending modes obtained that way are more suitable to use if we care about moving the mirror back to its original position in 3D. In reality, the surface sag directly contributes to the optical path difference of the incoming rays therefore the image quality. It is all we need to care about.

These surface sag bending modes are shown in Figure 3. Document-16389 gives the bending modes in Matlab .mat format. The variables inside the .mat file are defined as following.

- x and y are the FEA node coordinates in meter;
- Urt3norm gives the surface sag bending modes in meter, with each bending mode normalized to RMS of 1um;
- Vrt3norm gives the force vectors that produces the Urt3norm bending modes;
- G3 is the influence matrix, G3=Urt3norm*pinv(Vrt3norm)[‡];
- zRef and zpRef are z-coordinates on the ideal M1M3 surface at (x,y) and (x',y'), respectively They each have 256 columns, each corresponding to one of the 256 balanced unit force vectors.

These bending modes and the influence matrix is good for use in optimizing the M1M3 surface sag while other components of the Active Optics System (AOS) are not included in the

[‡] pinv() is the Matlab function for pseudo-inverse.

optimization, such as the camera and M2. The reason is because piston and tip-tilt (PTT) have been removed from the balanced unit load surface sags, assuming that adjustment of other AOS components will cancel them out. In controlling the AOS all together, the PTT need to be included. Because these bending modes, together with the degrees of freedom (DOFs) of other AOS components, will all be part of the input to the AOS influence function. We need a mechanism to let the system know that correcting M1M3 shape is introducing additional PTT, which need to be compensated using other AOS components.

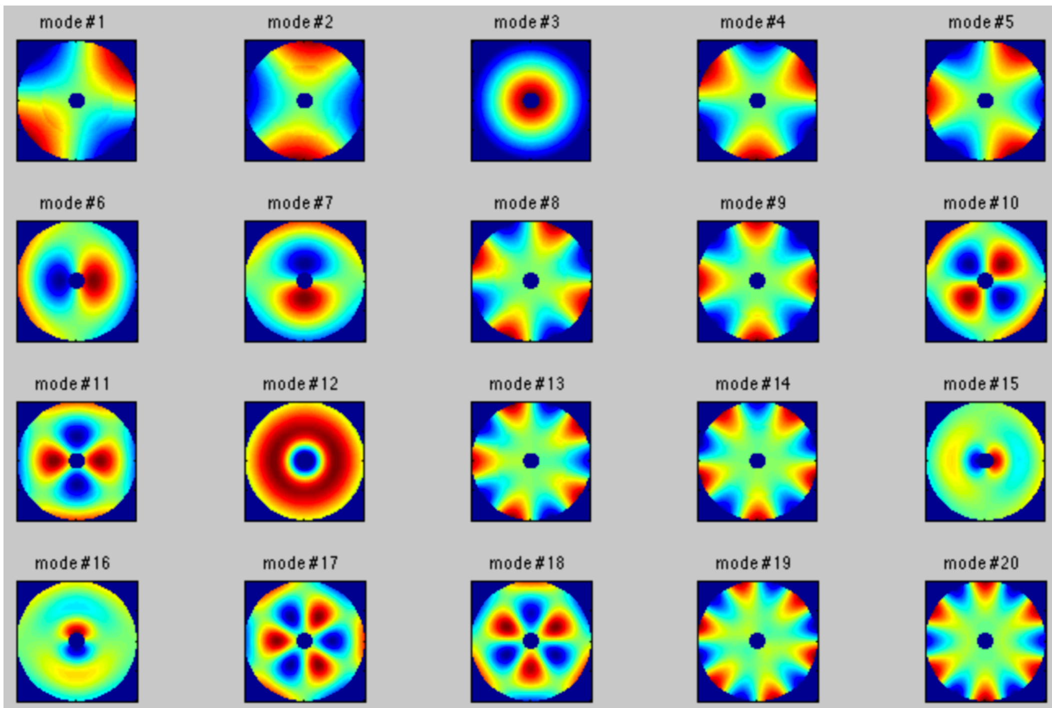


**Figure 3 Surface sag bending modes with 256 actuators and PTT removed.**

## 2.4. Using Surface Sag and the Bending Modes

When we control the actuators to optimize the M1M3 surface sag, to first order, we are also minimizing the z-component of the 3D node deflections. When the z-component of the 3D vectors is small, the x- and y- components are often not negligible compared to the z-component. Therefore, in this case, the slope-induced correction that relates the z-component of the 3D vector and the surface sag does play a significant role.

Figure 4 shows the actuator forces (Fy and Fz) that are needed to produce the optimal M1M3 surface at zenith pointing as defined by the z-component of the 3D node deflections. This optimal surface in z is shown in Figure 5 left, where PTT have been removed. The surface sag calculated using the 3D node deflections is shown in Figure 5 right. The PTT have also been removed. In Figure 5, the surface map on the left has RMS of 10.06nm, while the one on the right has RMS of 48.60nm.

The additional forces that are needed to further correct the above surface sag are shown in Figure 6. These are calculated using the surface sag bending modes which we described in Section 2.3. The linear analysis using the bending modes also predicts that the surface sag after putting these additional forces will be the one shown in Figure 7, with a surface RMS of

10.30nm. As a sanity check, the updated forces are given to Ed Hileman, whose then put the forces directly into the FEA and output the new 3D deflections. The same surface sag calculations are then done on these new 3D vectors. With no bending modes involved, the surface sag calculation gives almost identical surface map as the one shown in Figure 7, within the limits of numerical noise.

The same analysis as described above for the zenith pointing is also performed for several cases of horizon pointing. These include

(1) Optimized forces, where all 256 actuator forces are allowed to change freely to achieve the best surface shape;
(2) 2300N clipping, where the 100 lateral forces have been clipped to 2300N manually and fixed during the optimization;
(3) 1800N clipping, where the 100 lateral forces have been clipped to 1800N manually and fixed during the optimization.

In each of the above test cases, when the mirror surface appears to be optimized based on the z-component of the 3D deflection vectors, the actual surface sag has RMS in the range from 40 to 100 nm. In order to bring the surface sag to RMS of about 7-10nm, hundreds of newtons of addtional forces are needed.

Another kind of sanity check we can do on the bending modes is the following. In each of the three tests we did above for horizon pointing, we always start from a set of forces and the 3D node deflections it results in. So we have been given multiple initial states of the same linear system. Starting from each of the three initial states, if we let all the actuator forces to change freely to achieve the optimized surface sag, they should converge to the same system state, the one with minimal surface sag. We have done this test for each of the three cases. We always get identical surface sag map and identical force maps, all within limits of numerical noise. This is another proof that the system is indeed linear, and the surface sag bending modes and corresponding force vectors have been determined properly.

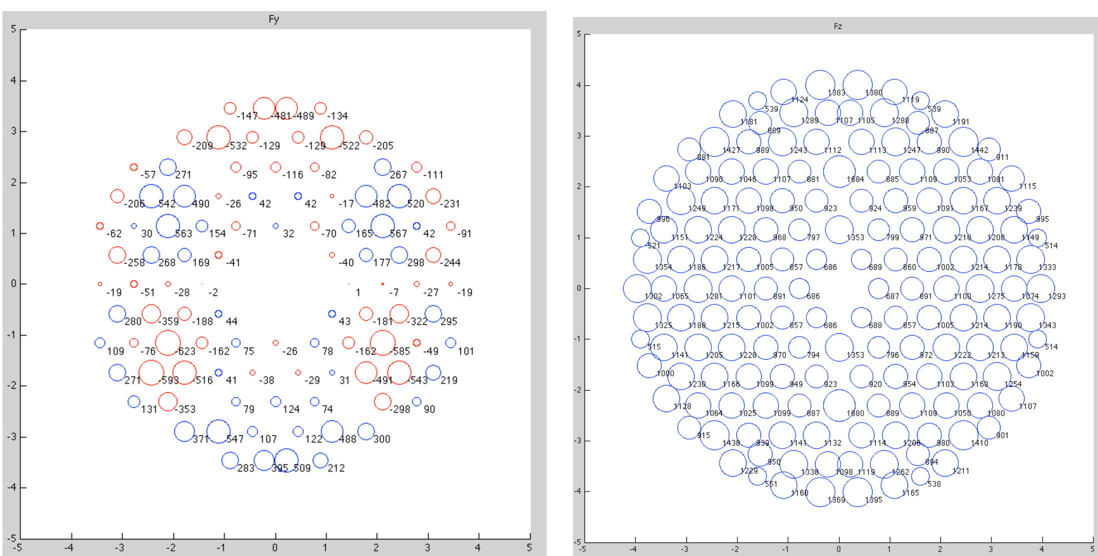


**Figure 4 Actuator forces (left: Fy, right: Fz) that produce the optimal M1M3 surface shape as defined by the z-component of the 3D node deflections (Figure 5 left). This is for zenith pointing.**

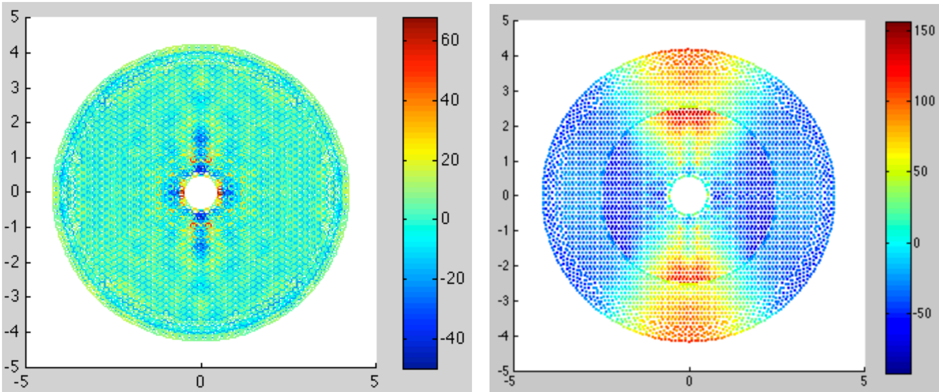

**Figure 5 Surface shape as defined by the z-component (left) and corresponding surface sag (right) produced by the force map in Figure 4. This is for zenith pointing.**

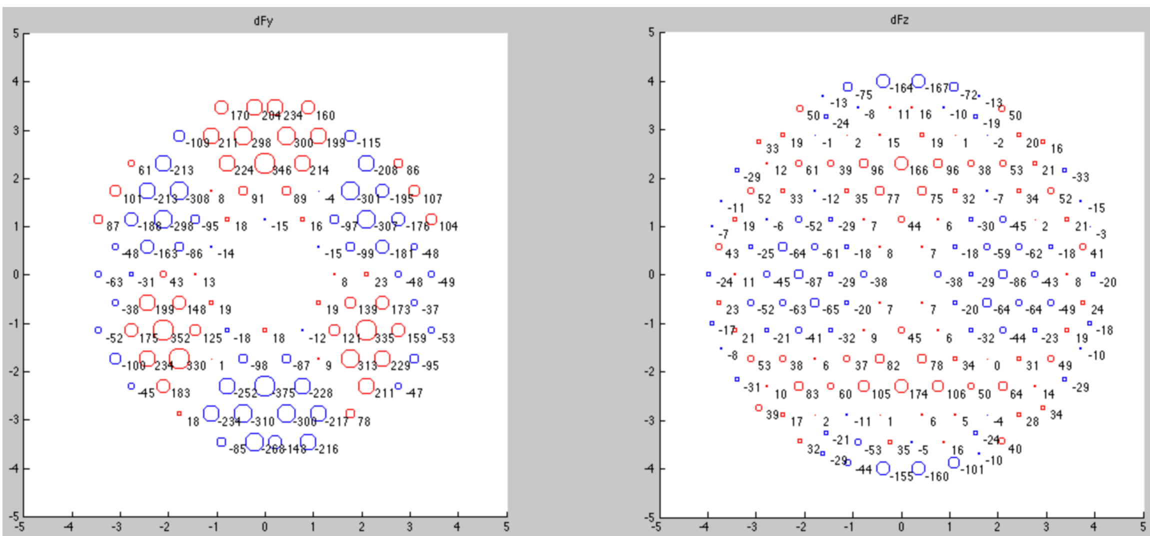


**Figure 6 Additional forces (Fy on the left and Fz on the right) that are needed to further correct the surface sag shown in Figure 5 right.**

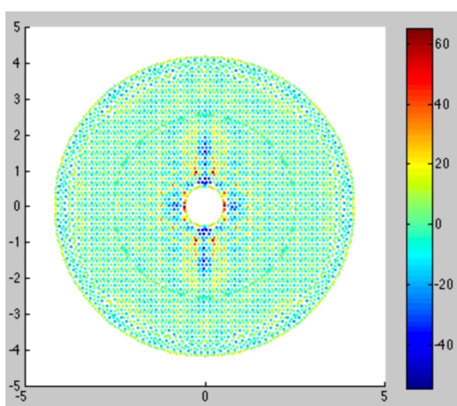

**Figure 7 Optimized zenith pointing surface sag. The corresponding forces are those shown in Figure 4 plus those shown in Figure 6.**

# 3. M1M3 Shape in Surface Normal

## 3.1. Calculating the Surface Normal Projection

Looking at Figure 1, the surface displacement along the surface normal direction is AF. Given that in Sections 2.1 and 2.2 we have proven that the real surface and the ideal surface can be assumed to be parallel on the spatial scale defined by (x'-x, y'-y), we can calculate AE and use AF=AE. AE is simply the projection of the 3D vector AB onto the surface normal direction.

$$AE = AB\cos(\angle BAE) = \overrightarrow{AB} \bullet \overrightarrow{dn} \qquad (4)$$

The surface normal unit vectors $\overrightarrow{dn}$ are shown in Figure 8.

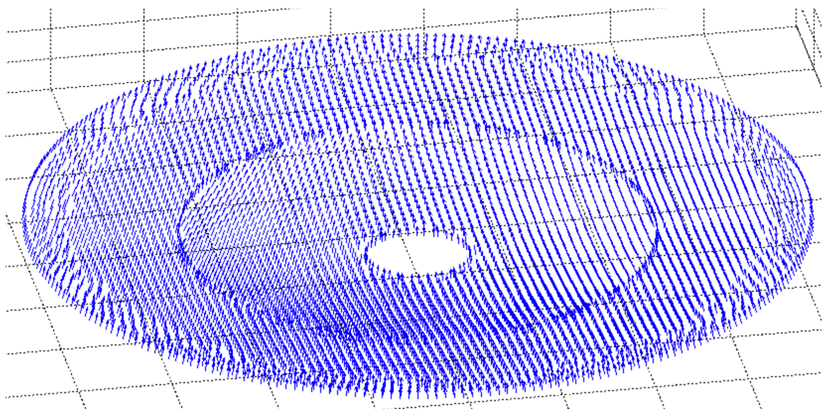

Figure 8 Surface normal unit vectors on the M1M3 surface.

## 3.2. One Example of the Surface Normal Projection and Verification

To exercise the surface normal projection, we look at a thermal compensation surface map used in the correction of M1M3 optical testing data. The map is created by FEA to determine the surface change due to deviations of the mirror temperature from the ambient temperature. This is for one of the many data sets, and is very preliminary at the time this note is being written. However, this can serve as a good example to show how much a surface defined in surface normal can be different from one defined using the z-component, given the same mirror state.

Figure 9 shows the raw 3D node deflections that come out the FEA. In this example, the x displacement is positive when x<0, negative when x>0. The same is true for the y displacement. This indicates that the overall motion of the mirror is a contraction. The angles between the 3D vectors and the surface normal direction are close to $90^{o}$. This explains why the surface normal projection is much smaller in magnitude compared to the z-displacements of the nodes, as shown in the bottom two plots in Figure 9.

A verification of the surface normal projection is also done using this example. Since we have calculated AE, we interpolate the real surface as defined by (x',y',z') to the (x,y) of point E. Then we know how far along z-axis point E is above the real surface. This distance, multiplied by the cosine of the angle between z-axis and the surface normal, gives an estimate of the difference between AE and AF. This is again found to be about 2-3% of the difference between AE and AB.

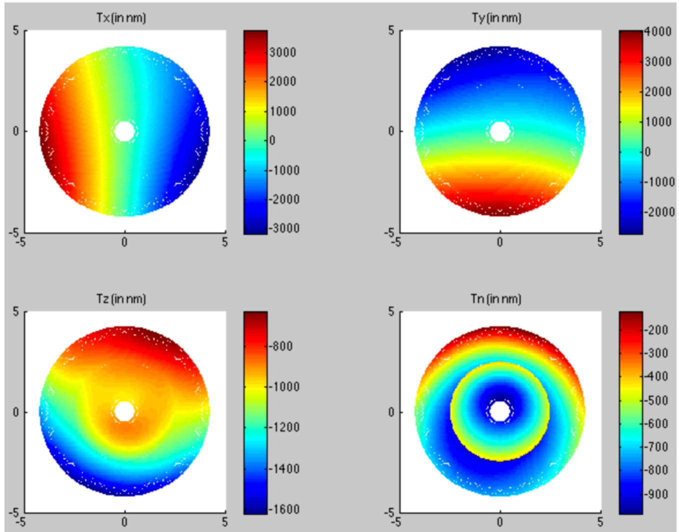


**Figure 9 An example of 3D FEA node deflections and the surface normal projection, when M1M3 temperature deviates from ambient. Top row: the x- and y- components of the 3D node deflections. Bottom row: the z-component of the 3D deflections and the projection in surface normal.**

### 3.3. Surface Normal Bending Modes

To get the bending modes in surface normal, we again start from the force vectors for the 256 balanced unit loaded cases (a 256x256 matrix), and the 3D displacements of each of the 5256 surface nodes under the influence of each balance force vectors (three 5256x256 matrices). The general procedure for obtaining the surface normal bending modes is the same as described in Document-15312 by George Angeli and Document-15338 by Ed Hileman, except

(1) Instead of using only the z-component of the 3D node deflections and ignoring those for x and y, we first use the 3D vectors to calculate the surface normal projection for each of the 5256 surface nodes under the influence of each balance force vectors.

(2) After the SVD, PTT are removed from the bending modes for M1 and M3 separately. This is because these bending modes are to be used in M1M3 interferometer data processing, where measurements on M1 and M3 are performed separately. As a result, the relative PTT between the M1 and M3 surface maps are unknown. Using the bending modes with PTT locally-removed on M1 and M3 to fit the PTT locally-removed M1 and M3 maps is mathematically equivalent to making the fit to the stitched M1M3 surface using bending modes that has local PTT, but is more practical because it doesn't involve stitching.

These surface normal bending modes are shown in Figure 10. Document-16385 gives the bending modes in Matlab .mat format. The variables inside the .mat file are defined as following.

- x and y are the FEA node coordinates in meter;
- Udn3norm gives the surface normal bending modes before the local PTT subtraction, in meter, with each bending mode normalized to RMS of 1um;
- Udn3norm_rmLocalPTT gives the surface normal bending modes in meter. Note that these bending mode are no longer normalized to RMS of 1um, due to the subtraction of PTT on M1 and M3 separately;
- Vdn3norm gives the force vectors that produces the Udn3norm bending modes;

- G3 is the influence matrix, G3=Urt3norm*pinv(Vrt3norm);
- dnx, dny, and dnz are the x-, y-, and z- components of the 5256 surface normal unit vectors.

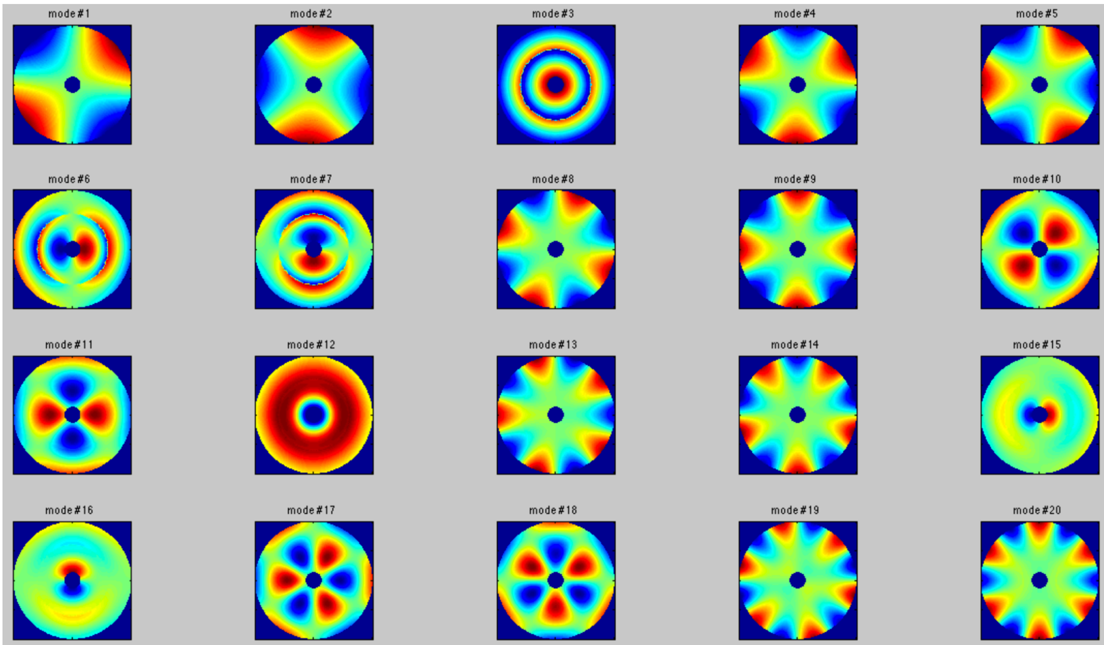


**Figure 10 Surface normal bending modes with 156 actuators and PTT separately removed from M1 and M3.**

## 4. Conversion between Surface Normal to Surface Sag

The conversion between surface sag and surface normal is trivial, but easy to forget. For example, the polishing error surface map we get out of the M1M3 optical testing data processing is always in surface normal. Before plugging it into a ray tracing program, we need to convert it into surface sag.

Because the surface normal unit vector is well defined at each FEA node, and the surface sag is always along z-axis, the angle between them is well defined. If we use Tz' and Tn for surface sag and the displacement along surface normal, and let α be the angle between the two vectors, we have

$$Tn = Tz' * \cos\alpha \quad (5)$$

## 5. Summary

In this technical note, we have discussed how we calculate surface sag using the raw 3D node deflections from FEA, and how to project it onto surface normal. Verifications of the calculations are given. We also showed examples of their applications and how different the actual surface shapes are from the simple z-projection of the 3D displacements. Bending modes in surface sag and surface normal are calculated, and provided in Matlab .mat format in Document-16385 and Document-16389. We also talked about how to convert a surface map defined in surface normal into surface sag and vise versa.