

# An Examination of Coarse Sun Sensor Contingencies in Attitude Determination and the Sun Vector Calculation

Brennan Coffey, Ray Welch, Brad Burt  
Jet Propulsion Laboratory, California Institute of Technology,  
4800 Oak Grove Drive, Pasadena, CA, USA 91109-8099

## Abstract:

Satellite pointing is vital to the success of a mission. One element of that entails describing the position of the sun relative to the frame of the satellite. Coarse Sun Sensors (CSS) are typically used to provide the information to calculate the sun's position in Safe Modes or contingency operations. In the OCO-2 configuration there are 13 CSS total, which provide redundant 4  celestial coverage. Failures of the individual CSS elements can introduce holes in the celestial coverage resulting in potential loss of sun knowledge. These failures must be analyzed to determine if the contingency plan is sufficient to assure mission success. First the static case was looked at and determined that at a maximum, 3 CSS failures can be sustained on the body and 1 on the array without causing coverage holes. Also array sensors are more important to mission success. The Sun Vector calculation has been transcribed to MATLAB code and failure scenarios are being examined to determine the maximum error given a set of failure scenarios. This activity indicated that if there is a loss of the sun, the sun-searching algorithm could be modified to use XZ rotation as that is guaranteed to find it whereas the design using the YZ rotation misses the sun if it is at the + or - Y orientation.

## Introduction:

Astrodynamics is the study of motion of celestial bodies, such as satellites. It is important to know how satellites and other pieces of equipment are oriented in space because a lot of the science that can be conducted has stringent requirements on pointing. For example, if a satellite is conducting science, which requires fine pointing, then it becomes very important to be sure that where the analysis team thinks we are looking, and where we are actually looking is the same location. If this is off by even a little, then the integrity of the data, and mission are at risk. This relative pointing problem is solved through attitude control system (ACS), which is a series of sensors and equipment that measure, report and change the orientation of the vehicle<sup>1</sup>. This orientation can be changed in many different ways, one of which is through reaction wheels, which use momentum to move the satellite. But not only does the ACS have to maximize the pointing for the science objective, it also has to determine how to maximize the power that is received through the solar arrays on the satellite. Ideally the solar arrays are pointed normal to the sun so as to receive maximum power, but that is not always the case, especially if failure scenarios are being considered. Thus the ACS has to figure out where the sun is very precisely to determine how to orient the arrays to get this maximal power.

The function of figuring out the location of the sun is passed to a very simple device called a coarse sun sensor. The coarse sun sensor measures the current produced by incident sunrays on its surface and then passes that information to be used to calculate the sun vector<sup>1</sup>. Coarse sun sensors (CSS) are typically one of the most robust systems on a satellite, but the vendor for the CSS on OCO-2 has recently been called into question as to whether their product is reliable. Once this question had been raised it became important to assess the mitigation that was in the algorithm for failures of CSS as well as how well the mission could perform with a certain amount of failed CSS.

On the OCO-2 mission, there are 13 CSS total: 8 on the body and 5 on the arrays, 3 of which input into the sun vector approximation (Fig 1). Orbital Sciences Corp. (OSC) has put extensive analysis into the static case where all of the CSS are working, however, not much work has been done in assessing the robustness of the algorithm and how the Fault Detection Logic (FDC) handles failed CSS.

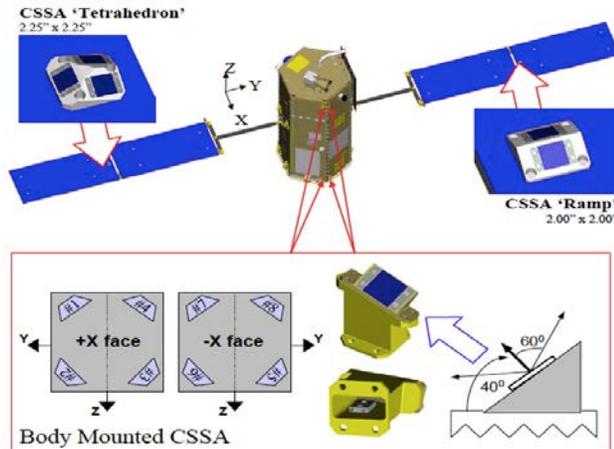


Figure 1. The above shows all 13 of the CSS that are present on the OCO-2 spacecraft. The “CSSA ‘Ramp’” is the pair of array sensors that do not input into the sun vector, while the “Tetrahedron” is the pair that does input into the sun vector calculation

#### Static Case with Failed CSS:

To assess the robustness of the flight software, a tool was developed using MATLAB, which allowed the spacecraft team to be able to select which sensors were and were not working. Using this tool, the team is able to visualize whether or not one sun sensor has redundant coverage as well as see the entire celestial sphere coverage. The script was adapted from a similar version used by the SMAP satellite and now works for a rotating body.

Using this field of view generator, we compared the analysis that OSC did for verification and all results agreed with theirs. Among just body sensors, the fields of view indicate none of the CSS can be lost without producing holes in the celestial sphere coverage. (Fig. 2)

The next goal was to determine if the rotation scheme (YZ) per the ACS presentation was the best method to reacquire the sun in a sun acquisition state. The graphs of the rotations and their coverage are very similar except for a small point that lies directly on the Y-axis (Fig 3). In the scenario where the sun was located at the pole, the YZ rotation scheme would miss this point entirely. The YZ scheme was being compared with another rotation scheme, XZ, which theoretically would not miss these poles because the orientation of the satellite puts the Y-axis at the end of the arrays and rotating about the X would sweep out the area that is uncovered at the poles.

A Monte Carlo scheme was developed where a maximum of 4 CSS were failed. From the analysis conducted, it appears that the best rotation method would be the XZ scheme. While having the sun nominally located elsewhere in the celestial sphere, the YZ works, the safest solution is to account for both situations as the XZ does not exhibit any loss in coverage of the celestial sphere but accounts for the potential hole at the pole.

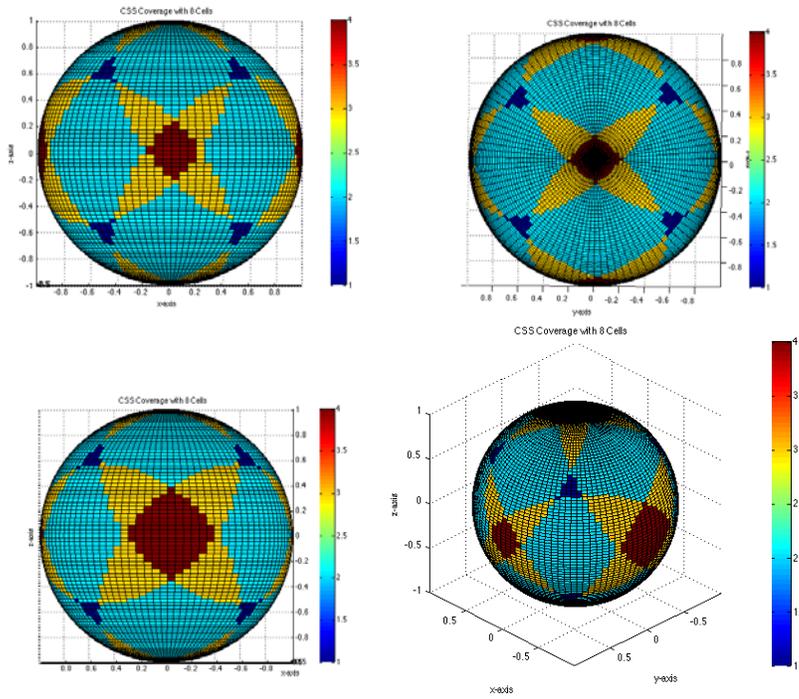
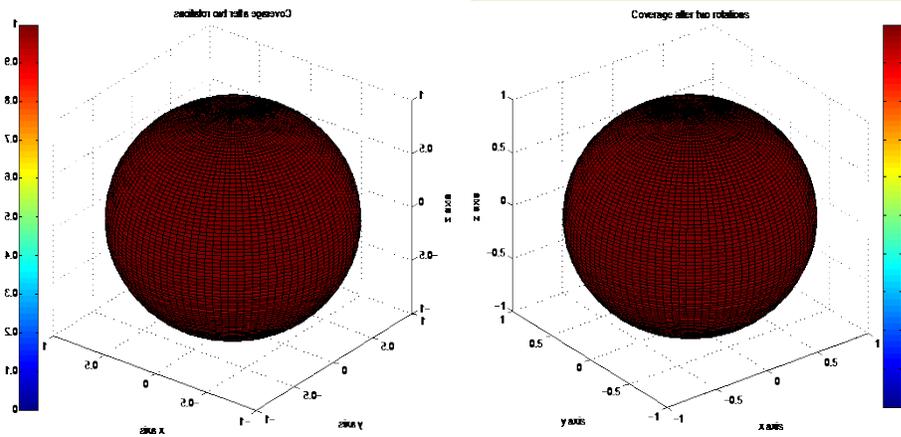


Figure 2. The above shows a series of plots, which display the base field of view of the satellite. In these graphs the 8 body sensors are the only ones being examined



Formatted: Font: Times, 10 pt

Figure 3. The above shows the different rotation schemes (a) YZ, (b) XZ, and the resulting coverage of the celestial sphere and the red spots indicate that at some point in the rotation that piece of sky was being viewed by at least one CSS. The hole in the celestial sphere would be directly on the Y-axis in the right graph.

The case examined next was the static case with various failed scenarios. The assessment was done based on 4 failed sun sensors at a given point in time and the percentage of sky not seen was then calculated. The results are in Table 1. In the description of this problem one of the conditions that was held true was the fact that a maximum of 7 CSS are viewing the sky at a given point in time (4 on one side of the body, and 3 on the array, all pointed in the same direction).

Percent Sky Seen	Number of CSS Lost								
	0	1		2		3		4	
		Min	Max	Min	Max	Min	Max	Min	Max
0	0.00	0.20	0.00	6.28	0.00	12.76	0.00	34.32	
1	2.29	13.02	3.86	27.79	6.23	31.25	9.65	35.65	
2	24.28	30.37	22.45	37.02	10.04	43.10	8.81	49.57	
3	20.71	26.22	10.51	29.17	9.84	29.06	9.81	34.47	
4	15.13	22.69	12.57	24.23	11.43	25.29	6.31	18.98	
5	11.54	13.01	9.52	14.20	0.00	14.20	0.00	14.19	
6	6.42	7.93	0.00	7.93	0.00	7.93	0.00	7.92	
7	0.00	6.30	0.00	6.30	0.00	6.30	0.00	6.30	

Table 1. The ranges for the Max and Min values are for the entire set of permutations run for that scenario. The percent sky seen is the percentage of the sky that is viewed by that number of CSS and it is the max over the entire set, not an individual failure.

From the table it is evident that as the number of lost CSS goes up, so does the range of percent sky not seen. When a max of 4 CSS are failed, the highest percentage of sky left uncovered is 34.32% which could cause large problems if the sun were placed in that position. The jump in sky left uncovered from 3 to 4 failed CSS is also noteworthy. At 3 failed CSS the failure is 12.76% and at 4 that value almost triples. From this it is evident that once the threshold of 3 sensors is reached, the potential errors in the sun search performance increase dramatically.

To understand the distribution of how many sensors can the mission feasibly sustain, the distribution of the percent sky left uncovered for each failure scenario was examined (Fig. 4).

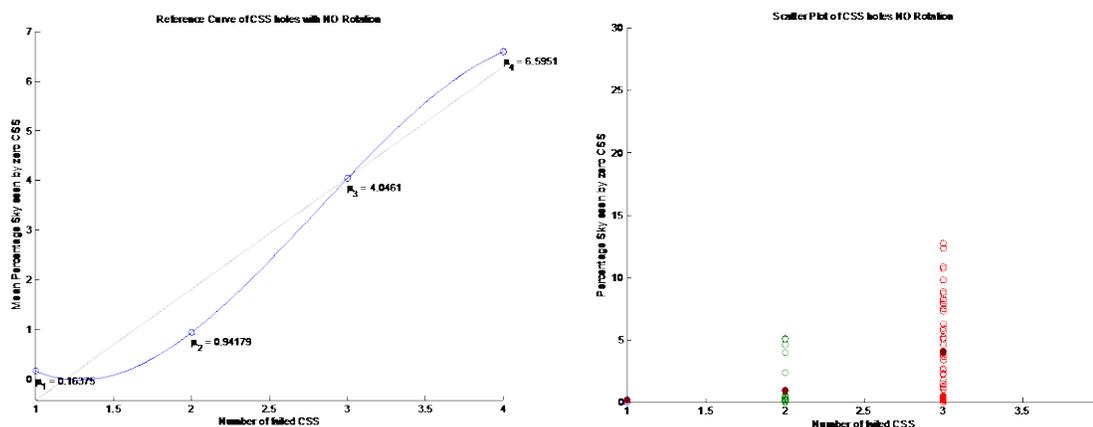


Figure 4. (A) The graph on the left is a graph of the averages of the percent sky left uncovered per a certain number of failed CSS scenarios, and has a polynomial interpolation laid over it as well as a linear regression. (B) On the left shows the distribution of the percentage of sky left uncovered overlaid with the average of each data set.

Figure 4a shows what the distribution looks like for all of the sensors in a static case. This was generated by looking at all of the permutations of 8 different failed sensors, excluding the 2 array sensors, which do not input into the sun vector calculation. The first two failure scenarios, one and two failed, do not differ significantly from one another, but at the third failure the average percent of sky left unseen dramatically increases from 0.94% to 4.04%. This also is followed with a larger distribution of percentage sky left unseen in figure 4b. Another important feature of this figure is the distribution of errors in 4b above 2 failed CSS. The majority of the errors seem to be concentrated between 0 and 1; however, there is a small distribution above that point. This indicates that while, on average 2 CSS failed have a low percent sky not seen, there is another factor to be examined before making a final statement. For the static case then, it is a good approximation to say that the maximum amount of failures that the mission can accept for failures is 3 with some dependencies.

### Sun Vector Estimation:

The assessment of the failure scenarios with the performance of the sun search algorithm is only a part of the total scope of a failed CSS; as the CSS are used to calculate the sun vector and then are also used in a lot of the attitude control of the spacecraft, another implication to be understood is how a failure scenario affects the sun vector estimation.

The flight algorithm for the computation of the sun vector was transcribed into MATLAB code so that a Monte Carlo analysis similar to the permutations observed for the rotation scheme determination could be run.

In order to effectively model the computation of the sun vector and permutation of a satellite in orbit, a few parameters were necessary to provide. One of them is the current observed in the sun sensors as a product of being lit by the sun. As the current seen through a sun sensor is closely approximated by a cosine curve<sup>1</sup>, I chose to model the nominal current by taking the dot product of the true sun vector with the sight of each individual sun sensor. This results in a value between 0 and 1 which gives a proportion of the maximum current that each of the sun sensor can see, which is then used to determine the modeled current in each sun sensor per the ACS table described in the flight software. These currents in each of the sun sensors are then used in determining the sun vector.

Once the sun vector estimation had been transcribed and understood, I ran my permutation routines of the sun calculation algorithm with different failure scenarios. To model this, I left the satellite body in a static position throughout all of the perturbations and only rotated the sun through the Y plane. Each rotation moved the true sun vector one degree around the satellite and then the estimated sun vector was calculated per my orbit modeling software. After each permutation, the error between the calculated and true sun vector was determined and recorded. Throughout the routine, the error is defined to be the inverse cosine of the dot product of the calculated and true sun vector:

$$\delta_{true} = \cos^{-1}(\mathbf{V}_{calc} \cdot \mathbf{V}_{true})$$

As this value includes imaginary numbers very frequently, the actual error recorded was the norm of  $\delta$  and a logical check if it exceeded a small number:

$$\delta_{used} = \begin{cases} \|\delta_{true}\| & \|\delta_{true}\| > 1 \times 10^{-5} \\ 0 & \|\delta_{true}\| \leq 1 \times 10^{-5} \end{cases}$$

### 3 Array Sensors:

The analysis run was a Monte Carlo type analysis, which examined up to 8 failed CSS and includes only 3 array sensors inputting into the sun vector estimation. In the scenario I also defined the eclipse to be from 0° – 120° rotation, so that region in all of the analysis is defined as a NaN. The data set was pre-filtered of the NaN values before statistics were computed. Generally what is observed is that average error increases linearly from one failed CSS all the way up to eight failed CSS (Fig 5.). Most of the error seen in the sun vector estimation is contained in the relative motion of the sun to the satellite (e.g.

when one sensor is lit and becomes unlit as the sun moves out of its normal plane). The error step is not gradual in this case if it happens to move into a combined sun vector approximation at that point. With the case of one failed CSS, the error in the sun vector is always  $0^\circ$  as this implies that another set, either the body or the array sensors, are at the optimal lit case and a sun vector can always be computed from this scenario. When 2 CSS have failed, the possible combinations imply that the error in the sun vector estimation is also zero, assuming the arrays are pointed normal to the sun, if they are not, then error begins to present itself in the estimation with the case that only two body sensors are lit. The more extreme cases ( $>3$ ) present significant errors in the sun vector estimation and can safely be called unsustainable.

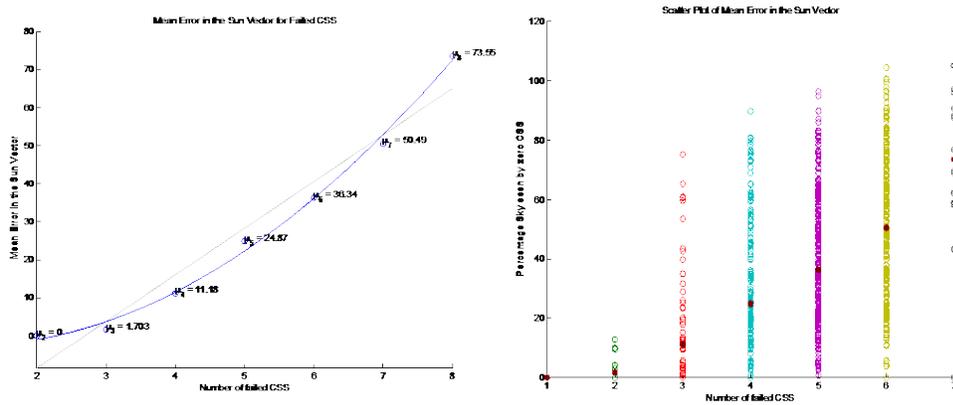


Figure 5. (A) Below left shows a graph of a linear regression of the data set. (B) Below right shows the scatter plot of all of the errors observed in the data set of all rotations of that number of failed CSS (e.g. 1 on the X is the set of all 1 CSS failures and the means are plotted on the Y)

An important note about the algorithm used in this simulation is that the calculation used the previous combined vector estimation when one could not be formed i.e. when the sights were too closely aligned. This is not exactly how the flight code works, as when this case presents itself, the algorithm assigns the new combined vector approximation to be 0; however, it was unclear from the flight software how the calculation proceeds. It is very simple to change the output to be  $\vec{0}$  and rerun the permutations.

An important feature of Figure 6b is the relative maximums and minimums in a given orbit. The on board computer attempts to drive the error in the estimation to zero, however, if during a rotation the error is within the peaks of the graph, then it will never reach the true minimum value as whenever the error began to change, the computer would drive it to the relative minimum and ignore the global minimum.

One interesting behavior of the sun vector estimation performance is a fan-like output, which can be seen in Figure 6. This is caused by having only 2 functioning CSS viewing the sky at a given point in time and would be classified as a death-grip scenario. The calculation of the sun vector in 3D space requires 3 vectors to define a point, otherwise the problem is underspecified and it leaves one free parameter. What it comes down to that, the algorithm can only define a plane in which the sun vector truly is, but can go no further until a third CSS is lit. Potentially, when entering an eclipse, and drifting while eclipsed this can occur with 2 failed body sensors. If those two body sensors are on the same side, and the arrays are pointed antiparallel to the sun, then only two cells will be lit and will only be able to calculate the sun plane, rather than the sun vector. If this is the case, the on-board computer will try and drive that error to zero, but will never be able to because of the oscillation of the error that is produced by this scenario. The only way out of this scenario is to drift into having a third CSS lit so the sun vector can be properly calculated. Even then, there is a possibility of having a significant error in the calculation if the combined vector is the one that is the primary estimate.

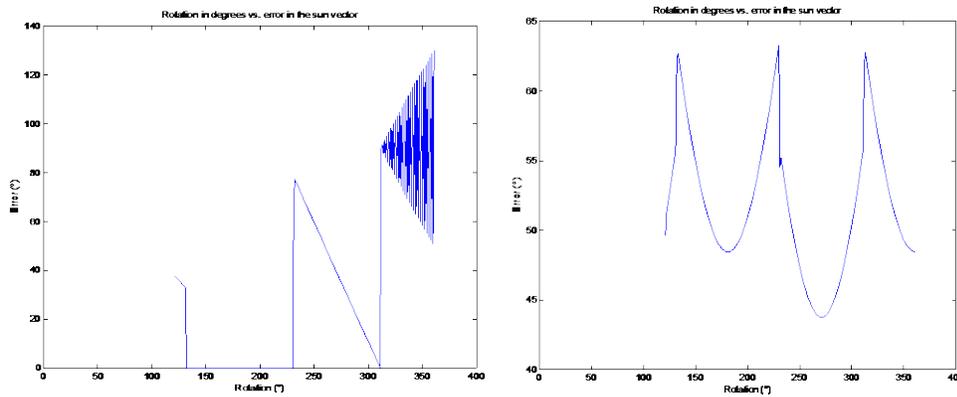


Figure 6. (A) Graph showing a case, which begins appearing at 3 failed CSS, where the sun is estimated to be in a plane and causes an oscillation, which appears as a “fan-like” region on the graph. (B) Shows a typical profile for failures > 3 which has large valleys and peaks which are potential traps for the on board computer driving the error to zero. Both of these figures are for a 9-failure scenario, but the principle remains as this pattern begins appearing at 2 failed CSS, which can be seen in the outliers of the scatter pots generated for that case.

### 5 Array Sensors:

This scenario is similar to the 3-array sensor one, except in this I considered the case where instead of 3 sensors inputting into the sun vector estimation, I am using the other 2 for a total of 5 sensors on the arrays. The approach for using them as input into the sun vector calculation is interpolated from the flight software and is very similar to the 4 body lit case.

Overall, the performance seems to be similar, though the errors stay lower for longer. The mission can sustain a loss of 2 failed CSS without dependencies if it uses 5 array sensors. On average, each of the permutations shows having an error over the entire set that is 10 degrees lower than the 3-array cell scenario (Fig 7.). So adding the extra two sensors into the calculation reduces the overall error, on average, by 3°. The “fan-like” behavior as described above still does appear at the 3-failed CSS range (Fig 6.) and is what the sparse distribution above the 2 failed CSS can be attributed to. This distribution above the 2 is very similar to the distribution above the 2 failed CSS in Figure 3. The fan behavior is what causes both of these occurrences in the data set.

While adding the extra sensors into the sun vector estimation appears to add no real benefit given the possibility of the fan case, if that case is ignored (for as unlikely it is) the gain appears to be small for any CSS below the 5 failed range (See Table 2).

3 Array Sensors	5 Array Sensors	
1	0	0

2	0	0
3	1.703	9.24
4	11.18	20.09
5	24.88	27.99
6	36.34	34.07
7	50.49	40.46
8	73.55	52.73
9	--	63.06

Table 2. The above is a comparison of the average errors in the estimation of the sun vector given a certain number of failed CSS. Note that all permutations for the 3-array case were examined while far fewer of the 5 array sensor case were examined due to cost in permuting over large data sets.

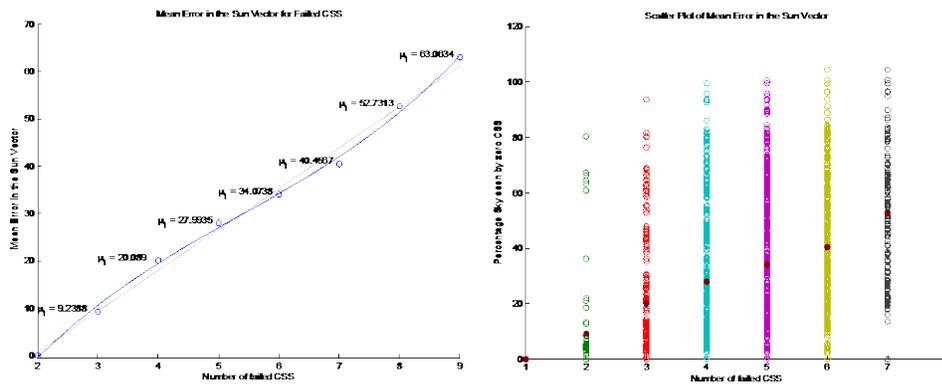


Figure 7. (A) Above, left, is a plot and polynomial fit to the averages of the error in the sun vector estimation for various failed CSS when using 5 array sensors inputting into the sun vector estimation. (B) Above right is a scatter plot for the error estimation of various permutations of the 5-array case, notice the outliers on the 3-array case.

The gain in robustness of the spacecraft with regards to failed CSS is minimal when excluding the fan case and small otherwise until 6 CSS are marked as failed.

### Robustness Discussion and Conclusions:

Overall it has been determined that the best solution for the sun search given various failure scenarios of the CSS would be a rotation about the X-axis then the Z-axis, as that would capture any potential holes residing directly on the Y-axis. Another important observation to make is that given up to 3 failed CSS, the satellite can rotate about the axis to cover the entire celestial sphere. This has some dependencies given the configuration of the failed CSS and it is important to note that the growth of percentage sky left uncovered is quite radical at a range > 3. Thus the upper limit for sustainability of failed CSS should be 2.

When assessing the sun vector approximation, the 3 array case inputting into the calculation of the sun vector was examined at first and it was determined that the threshold for sustainable failed CSS was 2, with some dependencies, such as the fan behavior which presents itself beginning at a failure level of 2. I cannot speak to the likelihood of that occurring but it is necessary to include it into fault protection. The 5

array case inputting into the sun vector estimation was also run and it was determined that the maximum benefit of using two more CSS to input into the sun vector calculation would be minimal, as the fan behavior is still the limiting factor to consider. Something important to note in the routines run here, the data indicate that for a low number of failures, the 3-array sensor is actually performing better than the 5 array sensor case. This is most likely a fault in the permutations which I chose to run, as I could not permute every, scenario as it is very expensive computationally, not all permutations were taken into account and therefore the data set is incomplete. Ignoring the fan scenario, the benefit is only gained when the number of failed CSS reaches a very high amount (5+).

With my study of the algorithm I do have several recommendations to improve the robustness with respect to the sun vector approximation. When the fan like behavior presented itself in the estimation, I implemented flag to test whether or not the sun vector changed dramatically in a short period of time. I recommend using something similar to verify that the CSS are performing as expected. This would allow ground control to be able to monitor the state of the spacecraft and determine if there are any scenarios at play, which need to be dealt with.

Another possible problem with the algorithm is the flagging of opposing sensors being lit. The algorithm denotes these sensors as being part of a "failed" scenario, but from what I can tell, does nothing to determine the state, or cause of the opposing lit scenario. In this scenario, the spacecraft would be indicating that the sun is surrounding 360° of the satellite and at least one sensor is flawed at that point, or both are. There is no test within the algorithm other than the minimum current threshold test and albedo check, to determine the state of a sensor and whether they should be inputting into the sun vector estimation. Thus a test of the state of the sensors would be something to implement into the code as well as a strategy of how to handle it. In addition to the state of the lit sensors, if the case presents itself where the two lit sensors are too closely aligned, the algorithm sets a value to 0, stating that, I assume, would be a skipped cycle. I could not understand how that instance was handled, whether or not the lit combination vector was presumed to be a zero vector, or whether or not the old approximation was used in lieu. In the routines run here, the old approximation was used when this case presented itself, but it was difficult to see what the handling strategy for such a scenario was within the code, or if there was one at all.

Another item worthy of mention is the use of the eclipse counter. From what I can tell, it uses AC cycles as the time scale, and sampling at 5 Hz, counts these cycles for the nominal eclipse time. Instead of just using a counter to check and verify if an eclipse state is currently in effect, using the ephemeris estimate would be something to implement into the flight algorithm.

Lastly, I noticed a discrepancy between the ACS look directions for each of the array mounted CSS, which corresponds to a stowed configuration, and the look directions in the OSC flight code, which correspond to a deployed configuration. The two different documentations of the configuration could cause some confusion and present interesting, and potentially dangerous situations.

#### References:

- 1 - Hall, Chris. "Attitude Determination." *Spacecraft Attitude Dynamics and Control*. N.p.: n.p., n.d. 1-23. Department of Aerospace Engineering Virginia Tech, 18 Mar. 2003. Web. 5 Aug. 2012. <<http://www.dept.aoe.vt.edu/~cdhall/courses/aoe4140/attde.pdf>>.
- 2 - Hashmall, Joseph A. "Acomodating Sensor Uncertainty in the Cones Method: Polycones and Fuzzycones." *American Institue of Aeronautics and Astronautics* (n.d.): 1-10. Web. 5 Aug. 2012.

#### Acknowledgements:

This work was performed at the Jet Propulsion Laboratory of the California Institute of Technology under contract from the National Aeronautical and Space Administration for the mission Orbiting Carbon Observatory – 2.