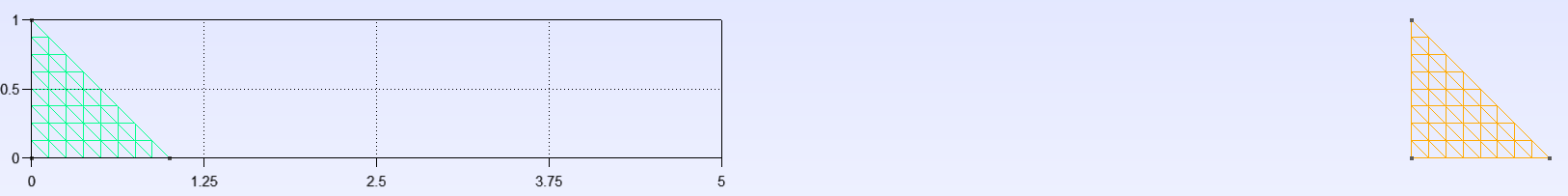
# 7/1/2021 ACA Fast Solve Verification

The prior strategy for solving problems in the acoustic integral equation code did not employ compression schemes to accelerate solving the matrix equation , and instead used an LU decomposition of **Z** to solve for **J** via effecting .

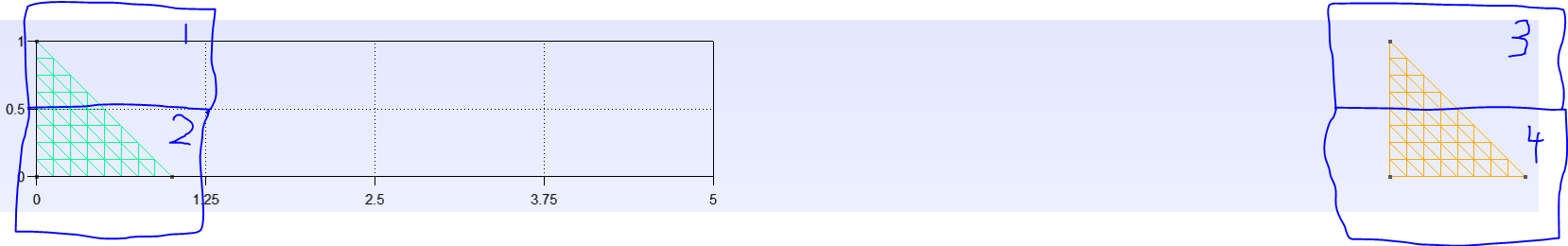
The fast solver method implemented uses ACA (Adaptive Cross Approximation) compression of sub-blocks of the **Z** matrix that are rank deficient to accelerate the overall operation. This method uses an iterative solver instead of the LU decomposition of **Z** to find the unknowns **J**. The mesh is first organized into nodes of an octree structure so that nodes that are physically well-separated can have their interactions approximated by the compressed sub-Z matrix given by ACA and close nodes will have their sub-Z matrix fully computed used the traditional direct way. Testing this algorithm was done with the following five “unit” tests:

**Test Problem 1:**

The geometry consists of two disjoint triangles with the discretization as shown in the image below. The distance between the triangles is ~80 elements edge lengths.



The octree used has three levels therefore the nodes containing element centroids look something like this (not drawn to scale necessarily).



This first test uses the direct computation of all sub-**Z** matrices (no ACA). The excitation is a spherical wave with and the resulting sources, **J**, are compared with the ones obtained using the old solver utilizing LU decomposition. In essence this test does a direct comparison between the LU decomposition method and the iterative solver (gmres).

*Results:* The l2-norm of the relative error between the two source vectors is ~1.3e-7

**Test Problem 2:**

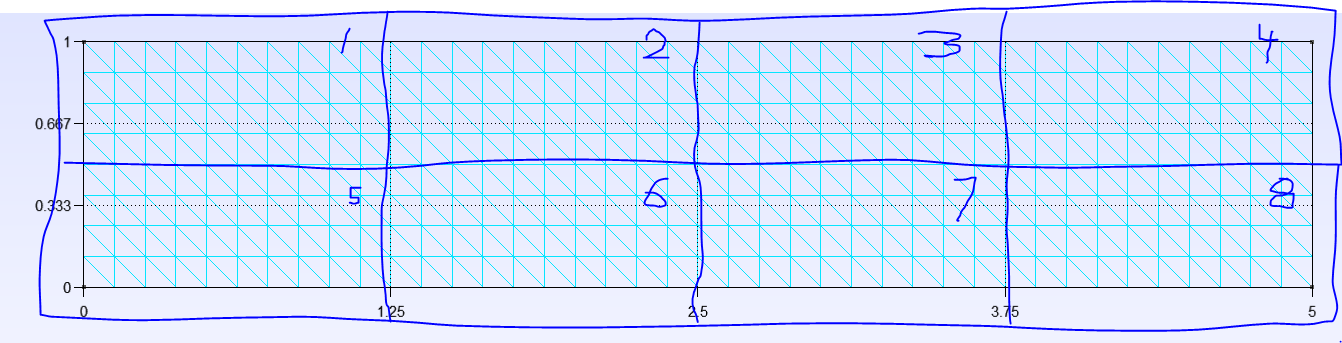
Everything about this problem remains the same as test problem 1, however, the sub-Z matrices containing interaction for nodes of separate triangles use ACA compression (e.g. node 1 interacting with 3 and 4 and node 2 interacting with 3 and 4, etc.). The tolerance for ACA is .

*Results:* The l2-norm of the relative error between the two source vectors is ~1.5e-7

These results are very close to test problem 2 because ACA works very well for these interactions since the nodes are far apart and the elements all lie in the XY-plane.

**Test Problem 3:**

A rectangular strip is the geometry used as shown below with the nodes drawn overtop.



The excitation is a spherical wave with . ACA is used for all interactions between nodes that don’t share an edge or corner (e.g. 1 interacting with 3, 4, 7 and 8 are compressed, but not interacting with 2, 5 or 6). The ACA tolerance is .

*Results:* The l2-norm of the relative error between the two source vectors is ~4.2e-3

**Test Problem 4:**

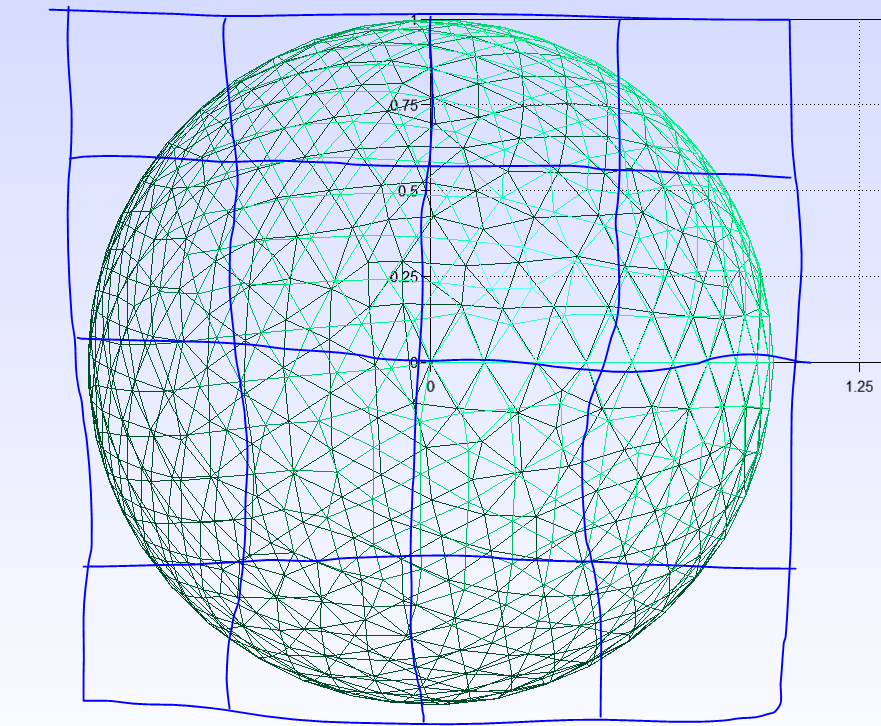
The same geometry and setup are the same as test problem 3 except the ACA tolerance is turned down to. To give a rough idea of whether this thing convergences or not (this small glimpse is by no means definitive).

*Results:* The l2-norm of the relative error between the two source vectors is ~3.4e-7

Looks like converging behavior. It appears that gmres will only go down to the 1e-7 range without any special tuning or options being passed into it.

**Test Problem 5:**

The geometry here is a sphere with 1266 elements and one 2D face of the octree structure is drawn on top (leaving visualization of the third dimension to the imagination) as shown below.



The excitation is a spherical wave with . ACA is used for all interactions between nodes that don’t share an edge or corner. The ACA tolerance is .

*Results:* The l2-norm of the relative error between the two source vectors is ~2.1e-4

This problem is the only one that has elements that do not all lie in the same plane. This increases the rank of the sub-Z matrices between nodes thus making ACA less accurate.

**Conclusions:**

The results presented above lead the conclusion that the algorithm is working correctly. Further convergence investigations could be pursued for further verification and to also give insight into appropriate input parameters for the solver.