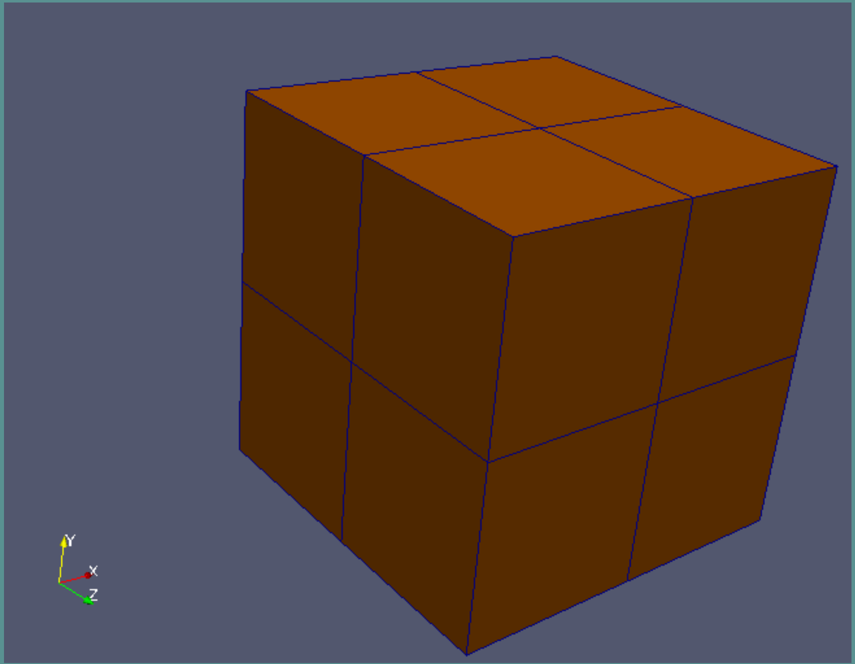
Computational Mechanics: Warp3D HDF5 Parallel I/O

Daniel Pledger
Rocco Febbo
Carlos Estrada
Mentor: Dr. Tim Truster

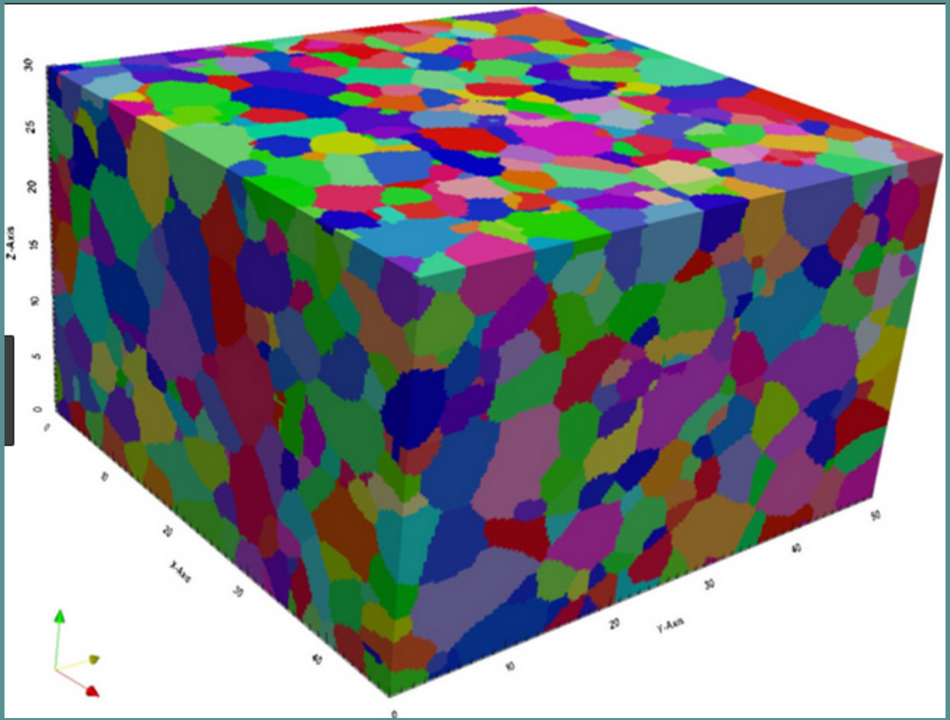
Project Overview

Create a workflow for several different programs to each other.

Use HDF5 to store output from Warp3D to be rendered by Paraview in parallel with OpenDIEL working with python GUI



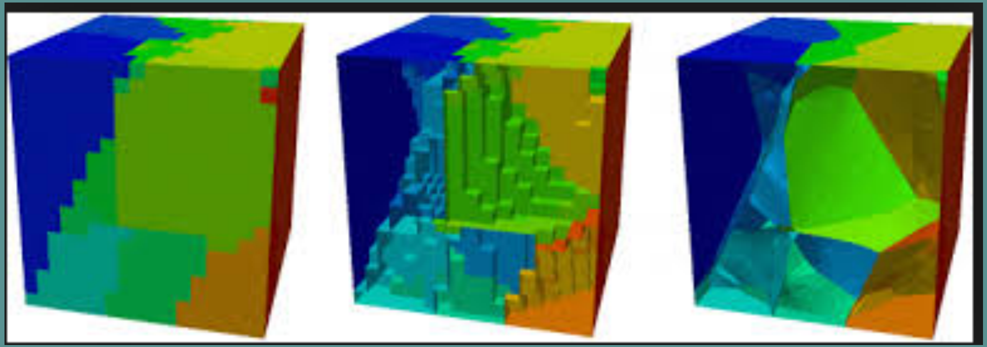
Dream3d



Description: Creates grain structures to perform Finite Element Analysis (FEA).

Purpose: Analyze fractures and crack growth in materials under loading.

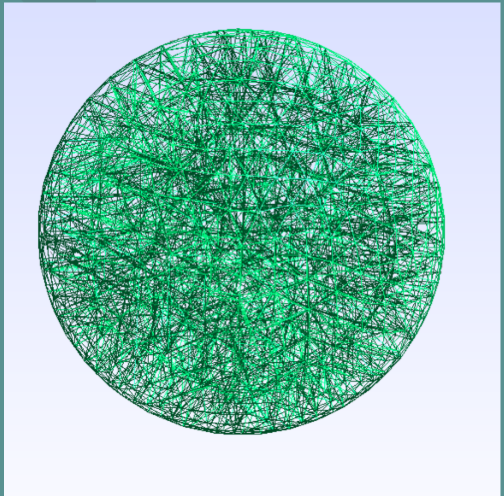
Voxel2Tet

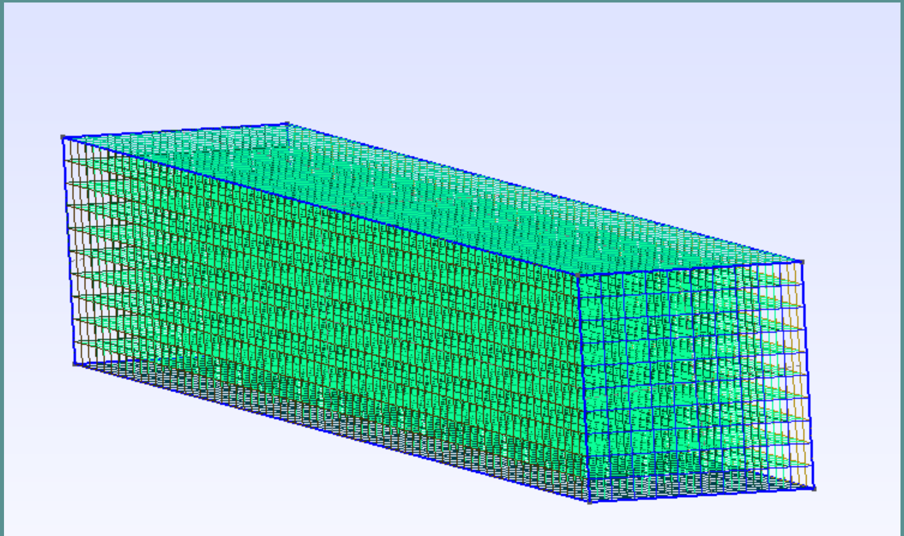


Description: Converts tetrahedral mesh to smoother interfaces.

Purpose: Create more realistic and defined meshes for more accurate FEA analysis.

Gmesh





Description: Gmsh is a free 3D finite element mesh generator with a built-in CAD engine

Purpose: Create meshes of individual grain structures for FEA.

Warp3D

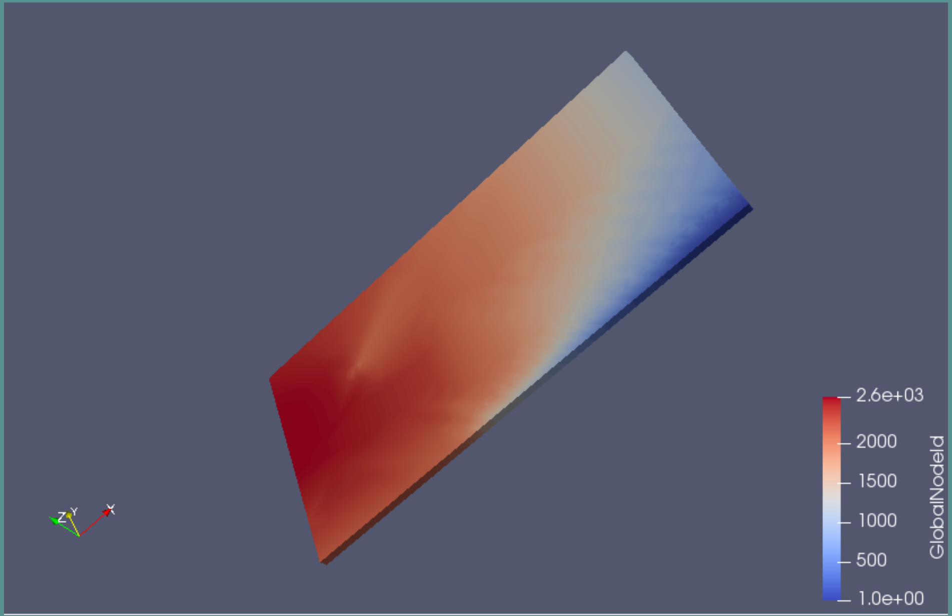
```
>> Running WARP3D on Linux (gfortran)...
   OpenMP + threaded MKL direct/iterative sparse solver
>> Number of threads for parallel execution:  24

    *******************************************************************
    **                                                               **
    **                                                               **
    **   W       W    AAAAA    RRRRRR    PPPPPP       33333  DDDD    **
    **   W       W   A     A   R     R   P     P     3     3 D   D   **
    **   W       W   A     A   R     R   P     P           3 D    D  **
    **   W       W   A     A   R     R   P     P           3 D    D  **
    **   W   W   W   AAAAAAA   RRRRRR    PPPPPP  ----   3333  D    D **
    **   W   W   W   A     A   R RR      P                 3 D    D  **
    **   W   W   W   A     A   R   RR    P           3     3 D   D   **
    **    WWW WWW    A     A   R     RR  P            33333  DDDD    **
    **                                                               **
    **      Linux 64-bit (gfortran)        -rel-    Release:  18.0.0 **
    **      Code Build Number: 3800                                  **
    **      Built on: Jul  1 2019 09:30:25                           **
    **      University of Illinois @ U-C.    Civil & Env Engineering **
    **      Today: Mon Jul  1 09:40:00 2019                          **
    **                                                               **
    **      NOTICE:  Use of Program Implies Agreement with Terms &   **
    **               Conditions Set Forth in File 'license_agreement' **
    **               Enter the Command 'license' to Display Text     **
    **                                                               **
    **      Limits (nodes, elements): none as of 17.9.3              **
    **                                                               **
    *******************************************************************
       c
       c       Nasa C(T), W = 2 ", a/W = 0.4, B = 0.09"
       c            CTOA growth w/ constant front
       c            CTOA = 5.6 degrees, Lc = 0.04", Le = 0.02"
       c            out-of-plane displacements prevented
       c            2 elements over half-thickness
       c
       c            The material is Al 2024-T3, a typical aluminum alloy
       c            used in aircraft.
       c
       c       The analysis uses CTOA growth with the constant front algorithm,
       c       which enforces uniform growth along the crack front. Measurement
       c       of the CTOA occurs at a distance (Lc) of 0.04" from the crack tip.
       c       The element size on the crack plane in the direction of growth (Le)
       c       is 0.02", thus generally providing two elements between the crack
       c       tip and the point at which the CTOA is evaluated.  When the CTOA
       c       at the master node reaches the critical value (5.6 degrees), then
       c       the crack advances by the distance Lc (in this case, 0.04", or
       c       roughly two elements).
```

Description: Open source code for 3D nonlinear analysis of solids primarily for fatigue and fracture simulations with static, impact, dynamic and thermal loadings

Purpose: Analyze material mechanics under stresses to improve designs.

Code: Written in Fortran (Late 1980's - Current)

Paraview



Description: 3D Material Viewer

Purpose: Render a 3D object and visually represent stresses from Warp3D

Program Use: Opens .exo file given by Warp3D or .xmf file used with .h file

HDF5

Description: File Type .h5

Purpose: Store files/data by efficient and compact means using a hierarchical format (similar to terminal file storage)

Code: Written in C, C++, Fortran

```
(warp3d) user1@REU1904:~/Warp3D/warp3d_distribution_18.0.0/src$ h5dump Dan_File_BDE.h5
HDF5 "Dan_File_BDE.h5" {
GROUP "/" {
   DATASET "BDE" {
      DATATYPE  H5T_STD_I32BE
      DATASPACE  SIMPLE { ( 4, 6, 4 ) / ( 4, 6, 4 ) }
      DATA {
      (0,0,0): 1, 2, 3, 4,
      (0,1,0): 5, 6, 7, 8,
      (0,2,0): 9, 10, 11, 12,
      (0,3,0): 13, 14, 15, 16,
      (0,4,0): 17, 18, 19, 20,
      (0,5,0): 21, 22, 23, 24,
      (1,0,0): 25, 26, 27, 28,
      (1,1,0): 29, 30, 31, 32,
      (1,2,0): 33, 34, 35, 36,
      (1,3,0): 37, 38, 39, 40,
      (1,4,0): 41, 42, 43, 44,
      (1,5,0): 45, 46, 47, 48,
      (2,0,0): 49, 50, 51, 52,
      (2,1,0): 53, 54, 55, 56,
      (2,2,0): 57, 58, 59, 60,
      (2,3,0): 61, 62, 63, 64,
      (2,4,0): 65, 66, 67, 68,
      (2,5,0): 69, 70, 71, 72,
      (3,0,0): 73, 74, 75, 76,
      (3,1,0): 77, 78, 79, 80,
      (3,2,0): 81, 82, 83, 84,
      (3,3,0): 85, 86, 87, 88,
      (3,4,0): 89, 90, 91, 92,
      (3,5,0): 93, 94, 95, 96
      }
   }
}
}
```

Measures Taken

```
<?xml version="1.0" ?>
<!DOCTYPE Xdmf SYSTEM "Xdmf.dtd" []>
<Xdmf Version="2.0">
        <Domain>
                <Grid Name="mesh1" GridType="Uniform">
                        <Topology Type="Hexahedron"
                                  NumberOfElements ="8">
                                <DataItem Format="HDF"
                                          Dimensions ="64">
                                          simpleCUBE.h5:/indicies
                                </DataItem>
                        </Topology>
                        <Geometry GeometryType="X_Y_Z">
                                <DataItem Dimensions="27"
                                          Format="HDF">
                                          simpleCUBE.h5:/X
                                </DataItem>
                                <DataItem Dimensions="27"
                                          Format="HDF">
                                          simpleCUBE.h5:/Y
                                </DataItem>
                                <DataItem Dimensions="27"
                                          Format="HDF">
                                          simpleCUBE.h5:/Z
                                </DataItem>
                        </Geometry>
                </Grid>
        </Domain>
</Xdmf>
```

Installed, compiled and ran all required programs separately

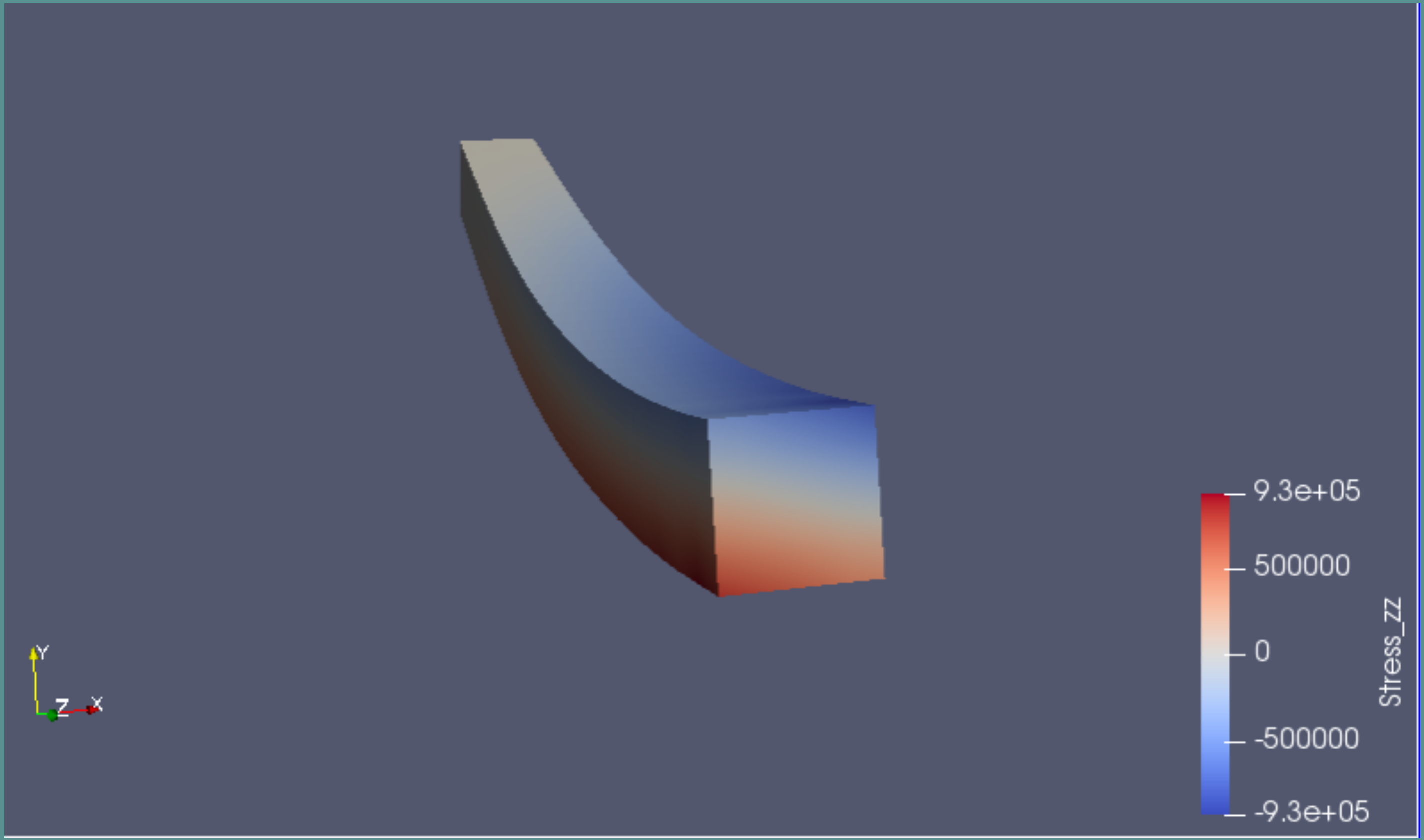
Wrote program to convert .geo to .h5 and opened with Paraview by giving a .xmf to represent formatting and attributes

Learning the structure of Warp3D source

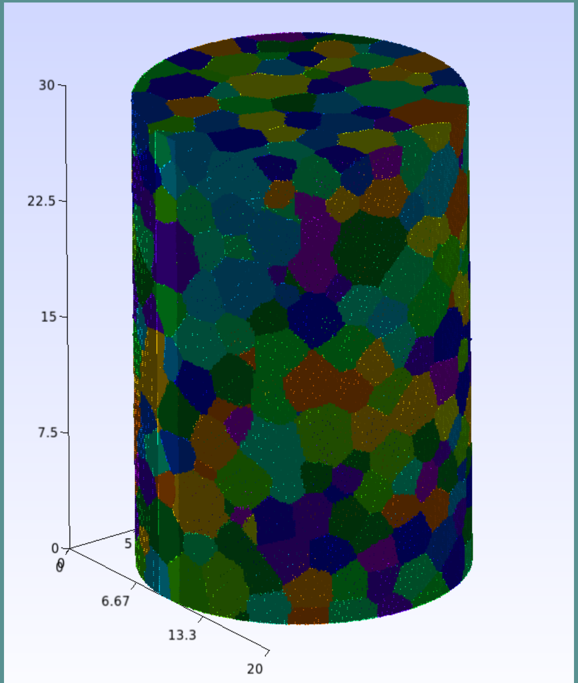
Tried manipulating Warp3D source code to write to .h5 from C function called in Fortran

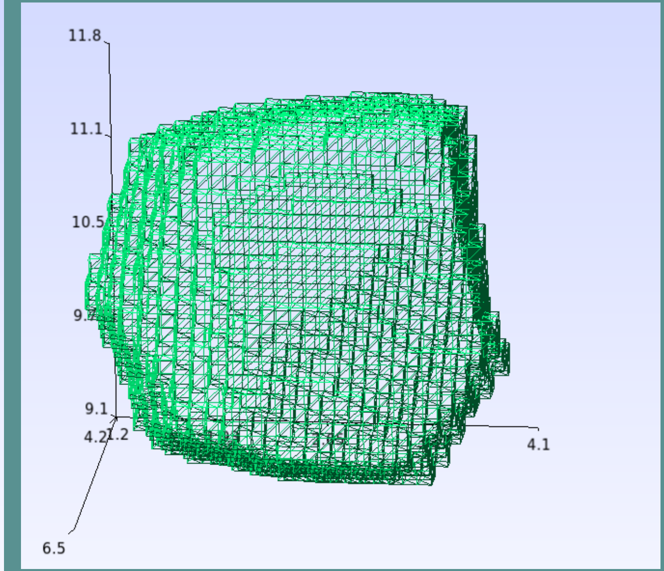
Wrote code to take from Patran Neutral Text File to .h5 and read to Paraview via .xmf file

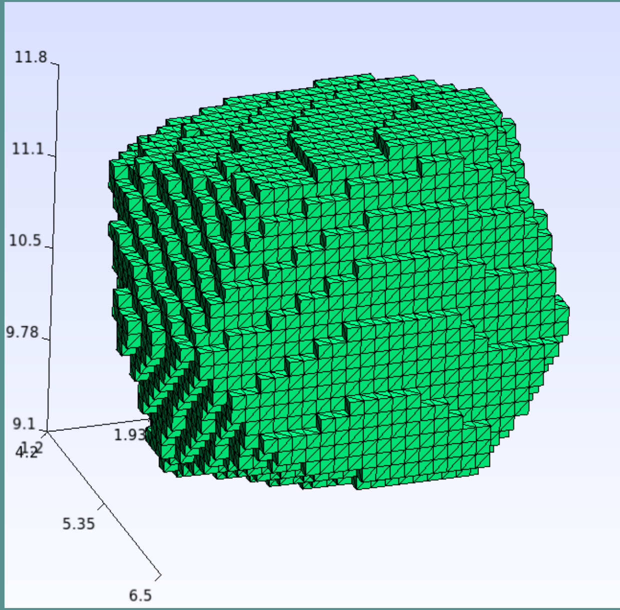
Gmesh, Warp3d, and Paraview



Dream3d and Gmesh





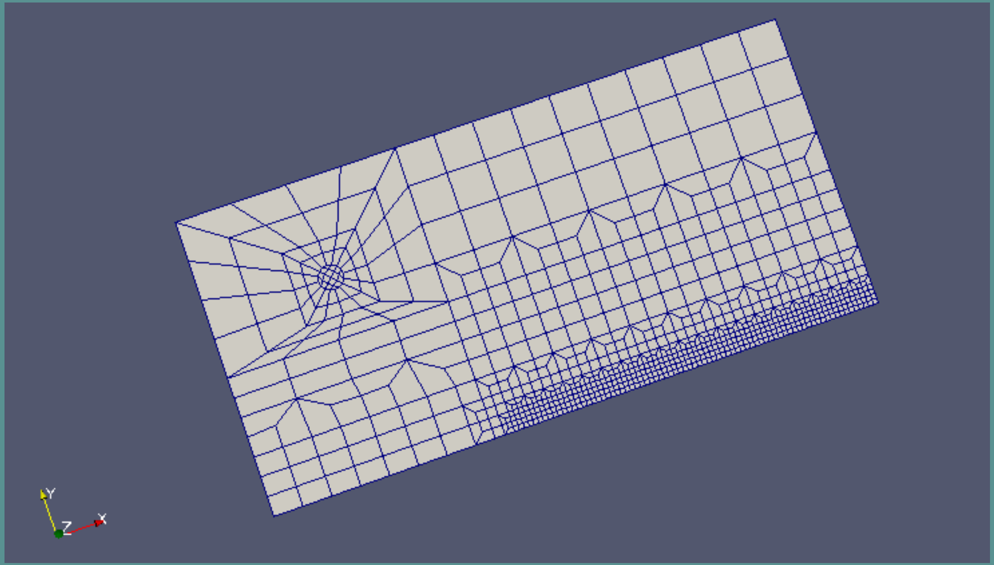


Terminal Goal

Write code in HDF5 C/C++ to store output and physics from Warp3D

Run Warp3D, HDF5 and Paraview in parallel with OpenDIEL and Python GUI

Manipulate object with Warp3D physics, Python GUI and Paraview to show calculated stresses on meshed nodes and elements



What's next?

- Continue developing workflow to get meshed 3d grain structures input to Warp3d for analysis.
- Continue work on getting Warp3d to output stress, strain and other data in HDF5 to be viewed in Paraview

Questions