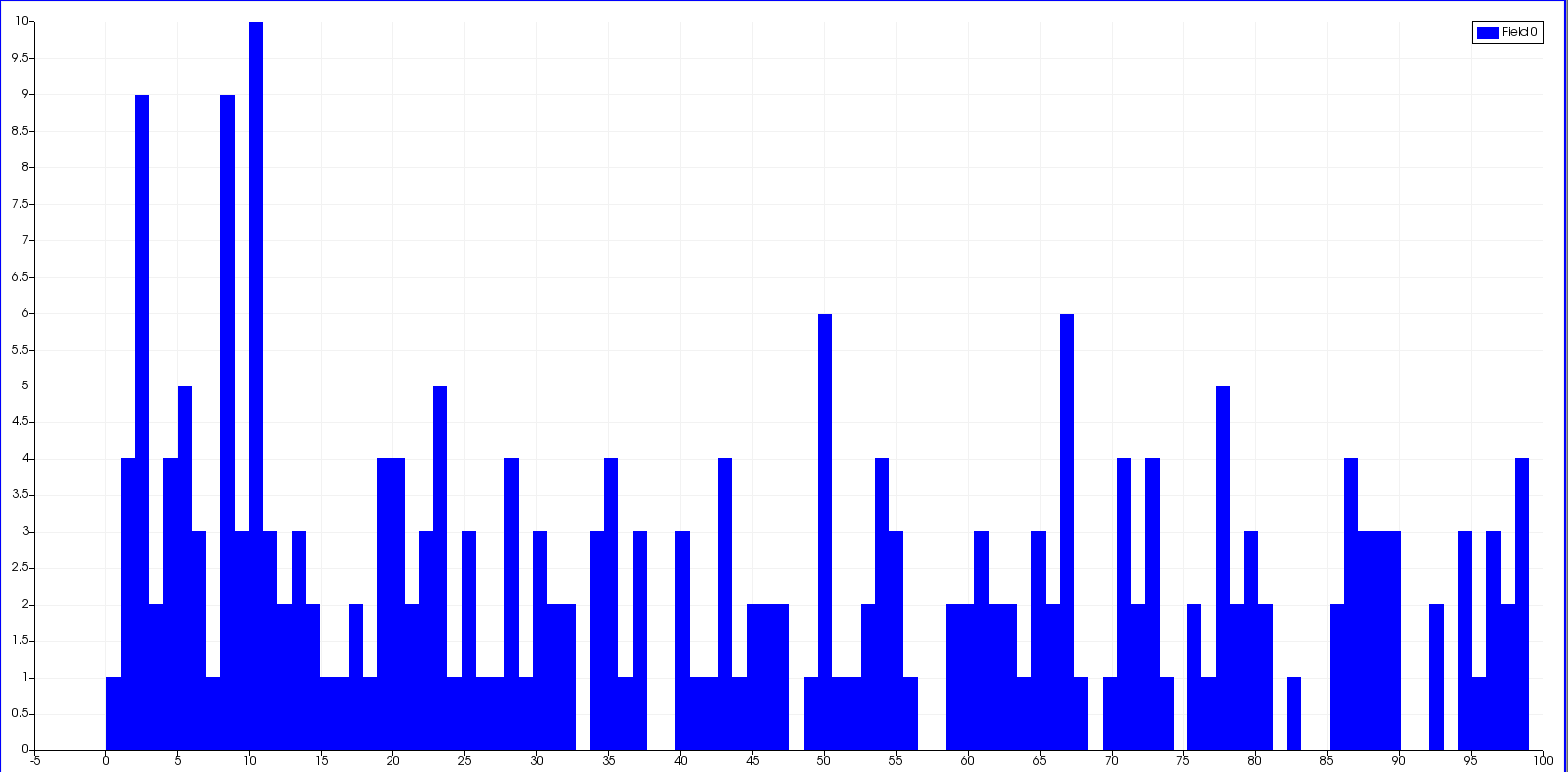
**CS 5635/6635 Spring Semester 2019 Assignment 2**

**Rajath P Javali**

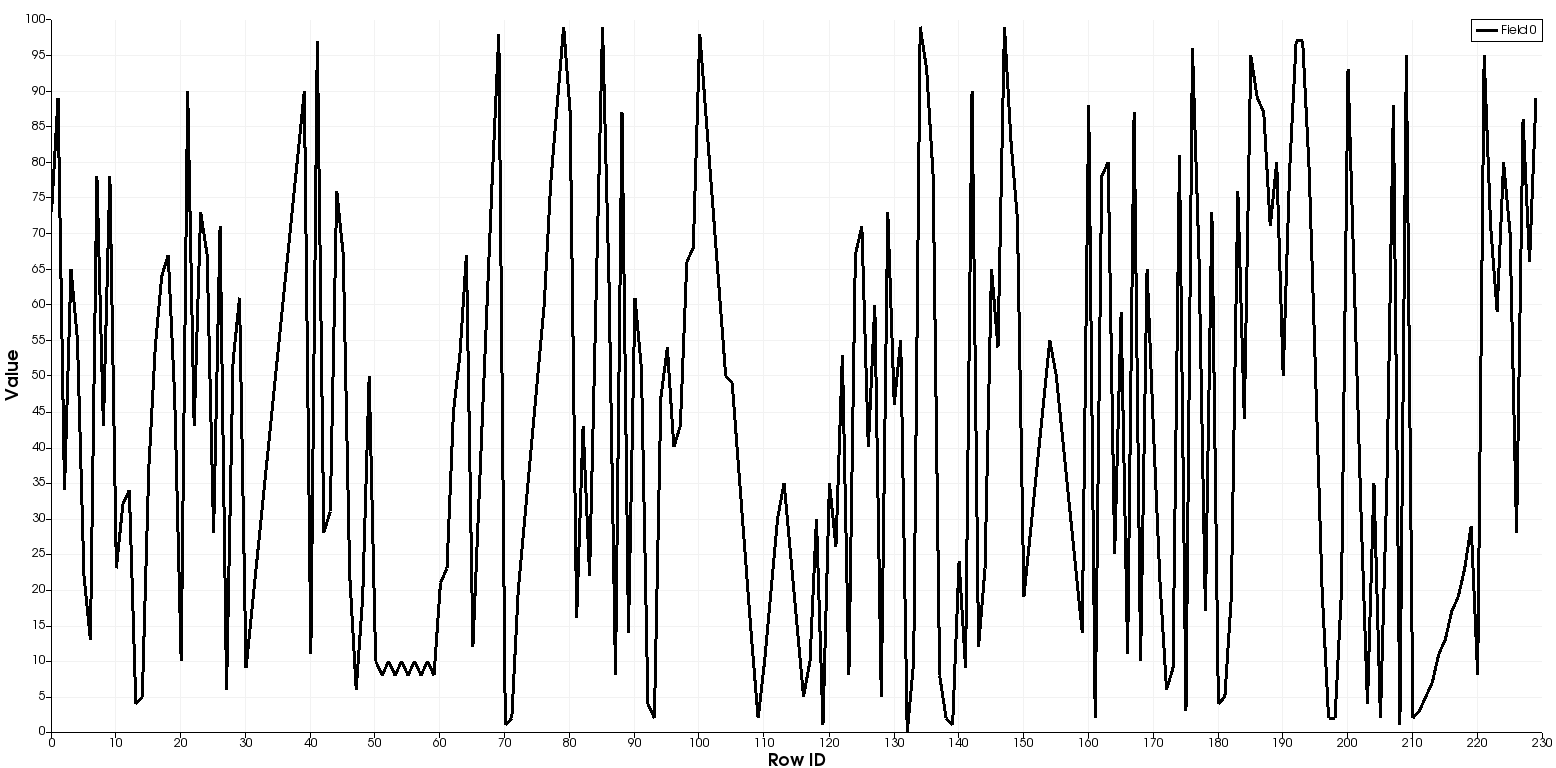
**u1140594**

Part 1: Load the data

Q1. Visualization of Statistics for 1-D Data



Histogram View



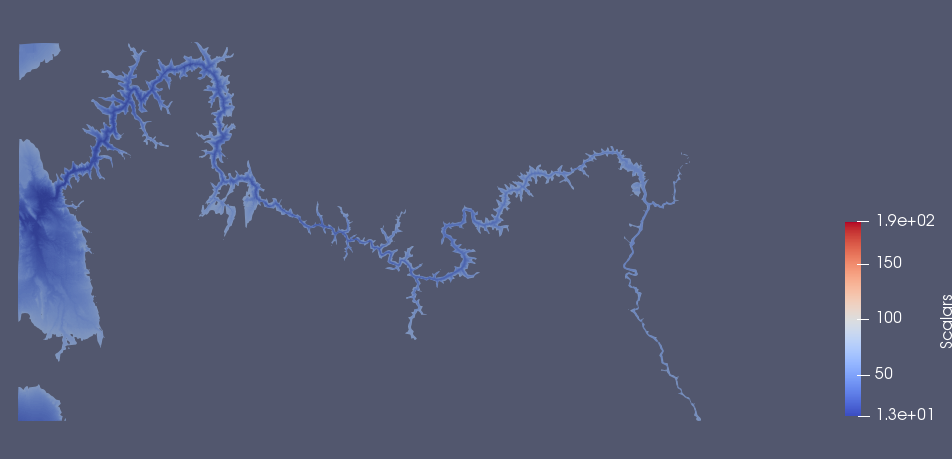
Line Chart View

1. Histogram View
   1. Number which occurred the most frequently is bin 10.395 with an occurance of 10.
   2. 14 numbers were never used by the class.
2. Line Chart View

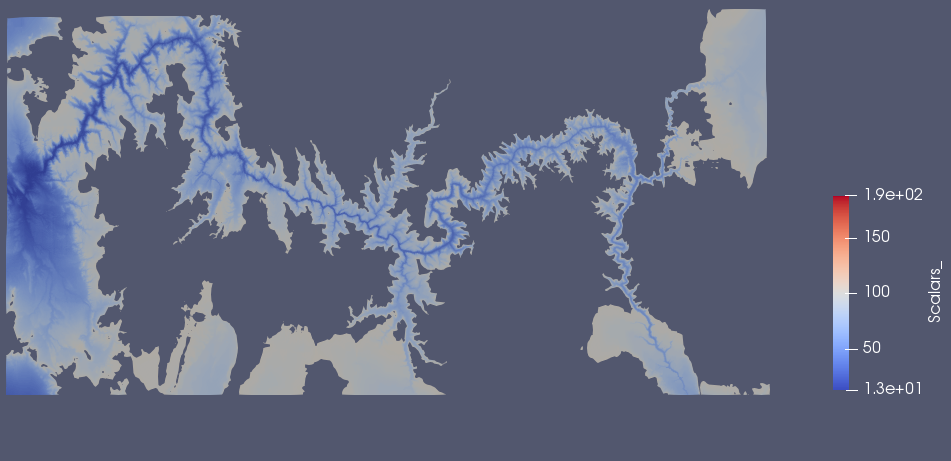
Q2. Visualization of 2d Image

By trial error, found that threshold from 0 to 68 gets the complete river basin clearly. Increasing the lower threshold value cuts of the river basin where the depth is maximum. Increasing the max threshold value includes the lowlands of the grand canyon adjacent to the river basin.

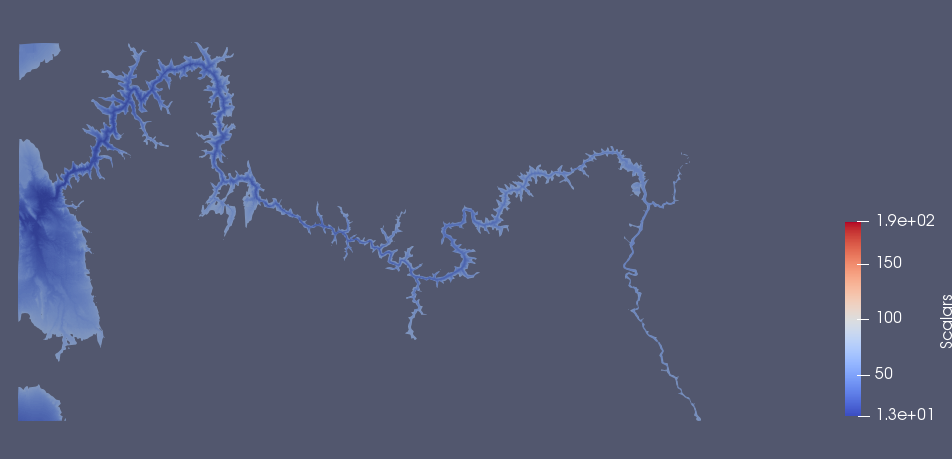
1. Threshold (0 - 68)



1. Threshold (0 - 100)

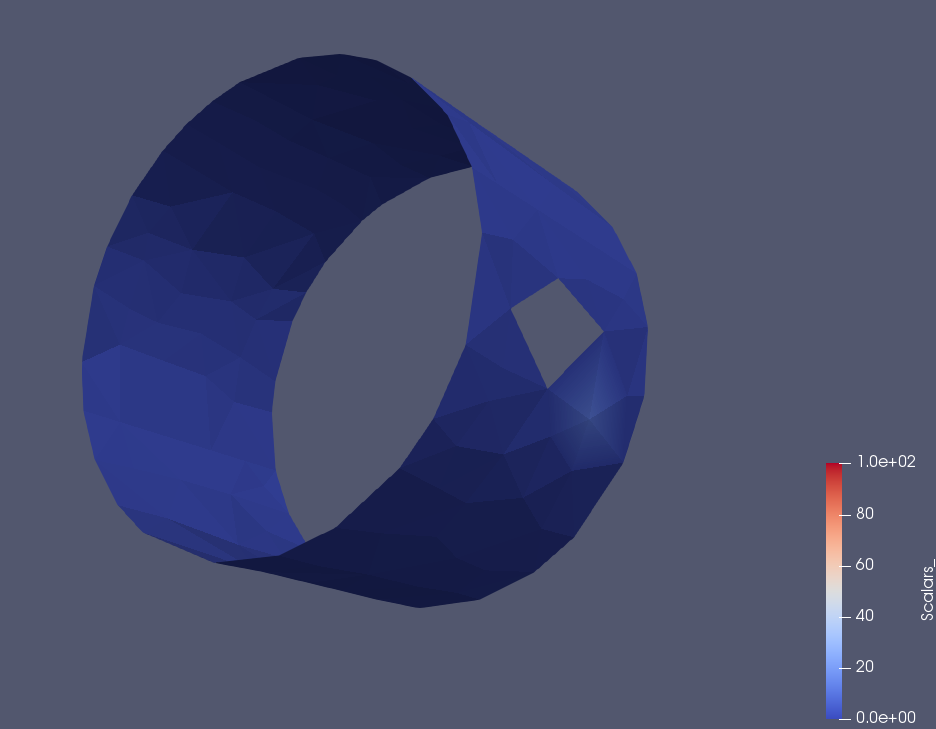


1. Threshold (13 - 68.68)

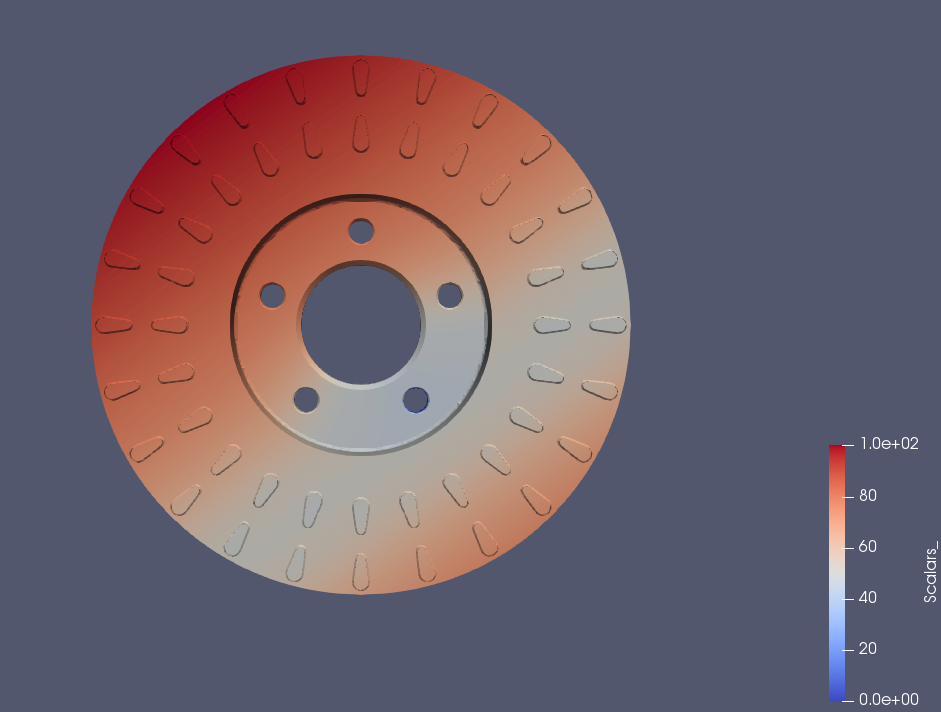


Part B: The number of points in the threshold images are 3143708.

Q3. Exploring Data on Polygonal Meshes

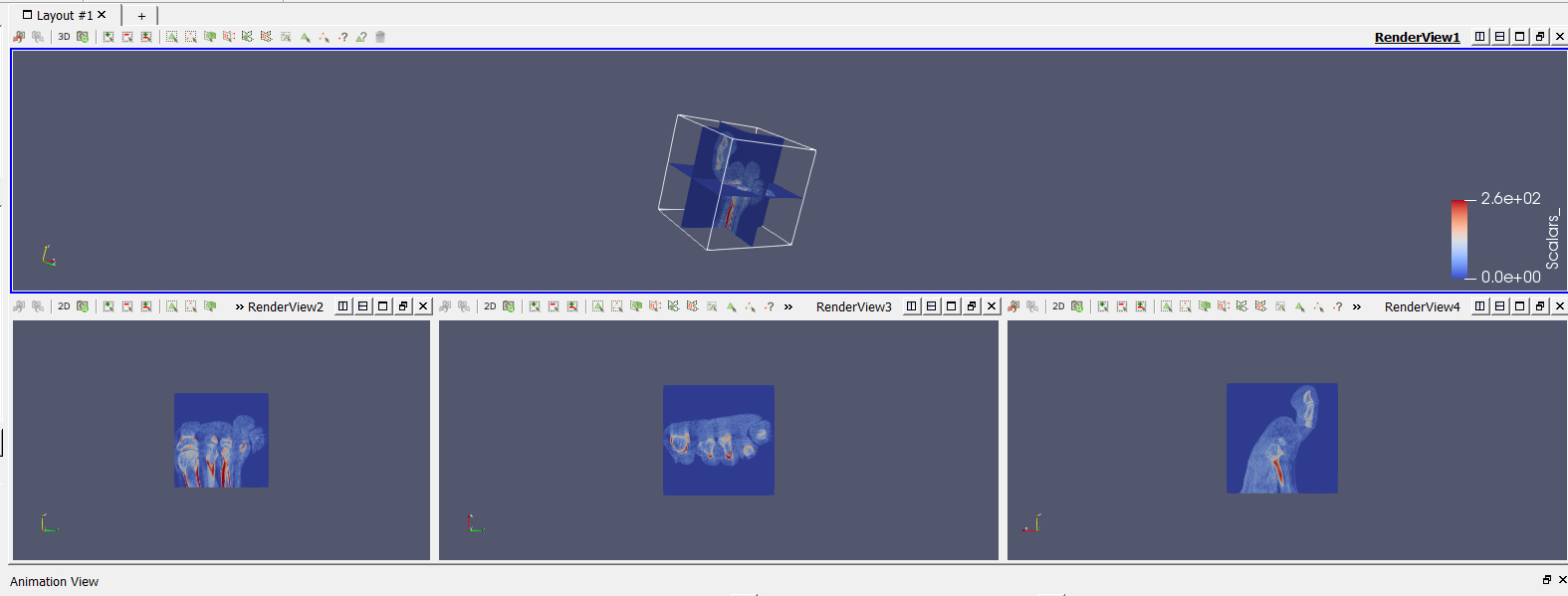


The minimum value of 0 and maximum value of 20 - 40 range best captured the single cylinder associated with the bolt’s cylinder.



There are total 48 ventilation slots.

Q4. Visualization of 3D Images

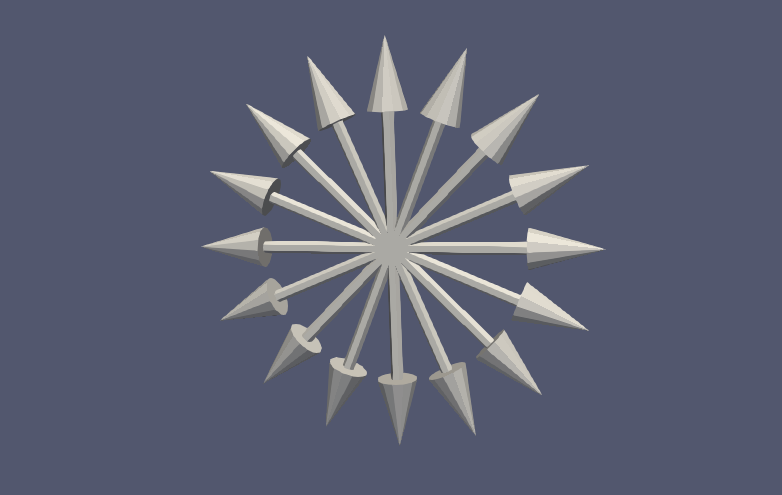


Rending of the 3D data and then cliping the data along all 3 axes gave the 1st render. Then the screen is split horizontally and then vertically into 3 sections with each section displaying one slice per window. Which results in a screen resembling as the one above.

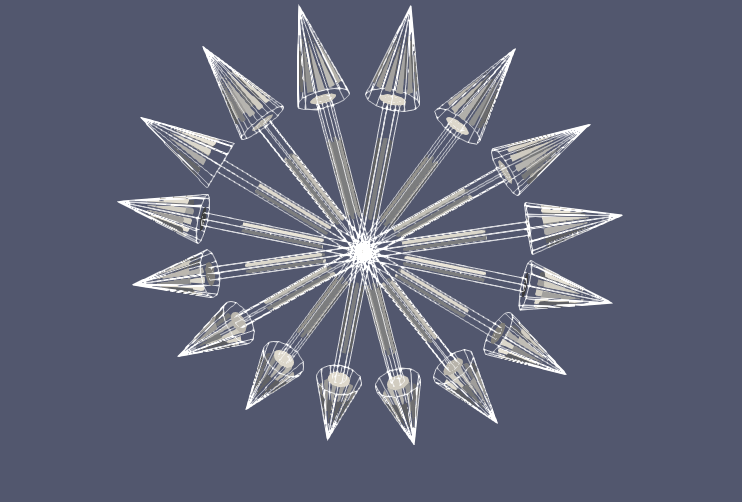
Part 2: Code with Python Script

Q1. Use batch script to create a pipeline

1. Generating 16 arrows placing at 360/16 degrees from each other.

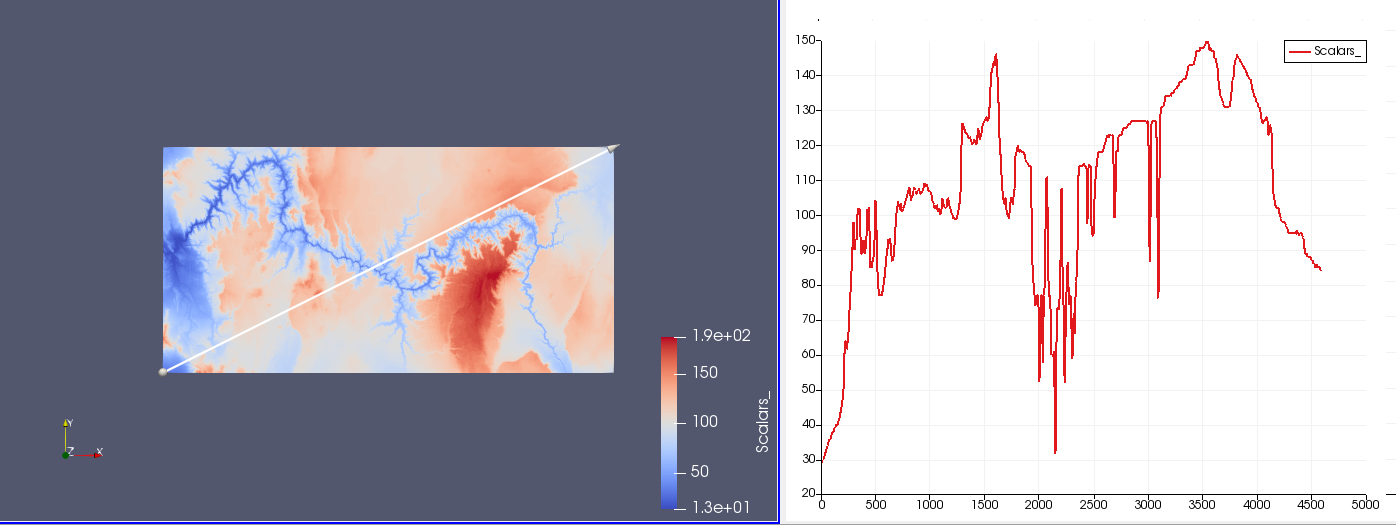


2. Shrink filter and extract edge filter on arrows.

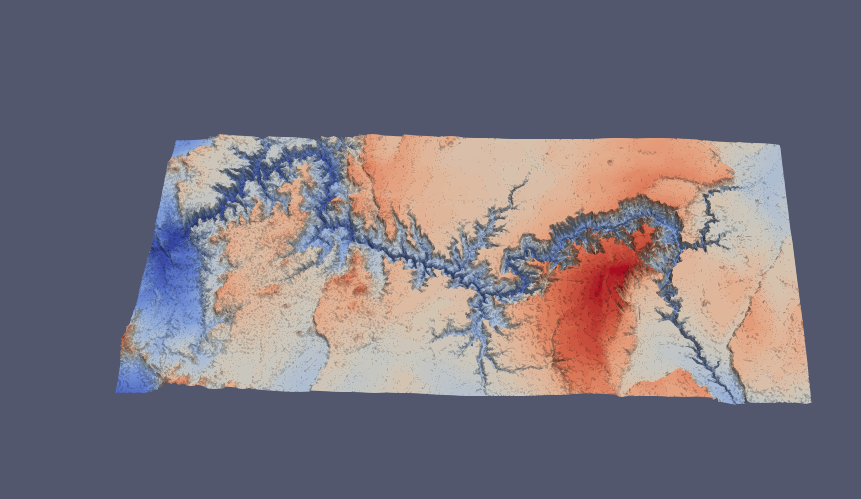


Q2. Read file and process it with Python script

1. Split window view on 2d grand canyon view along with plotOverLine graph on the right.



2. Converting 2D data to 3D using the warp filter.

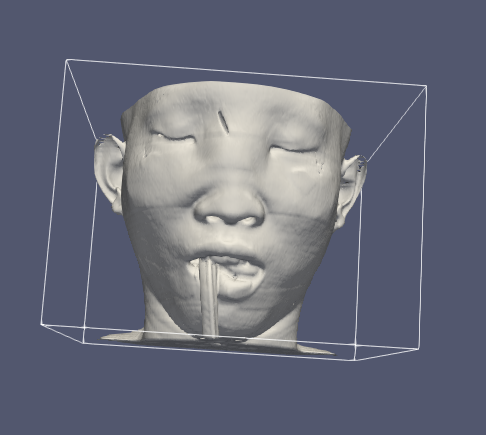


Part 3: Time-Dependent Isosurface Extraction

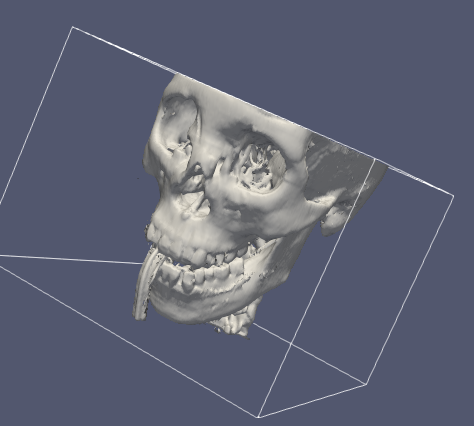
Q1. Isosurface Animation

1. Isosurface range from 950 to 1200 - transitions between skin and skull.

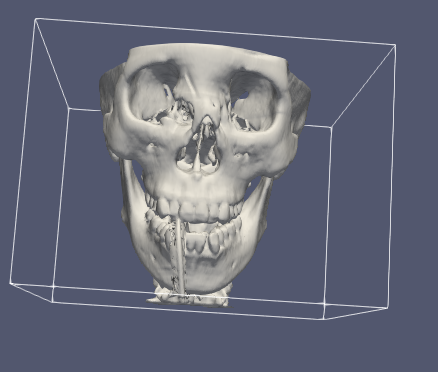
Isovalue - 950 - we see the skin disappearing on the nose



Isovalue - 1150 - still skin on lower sphine

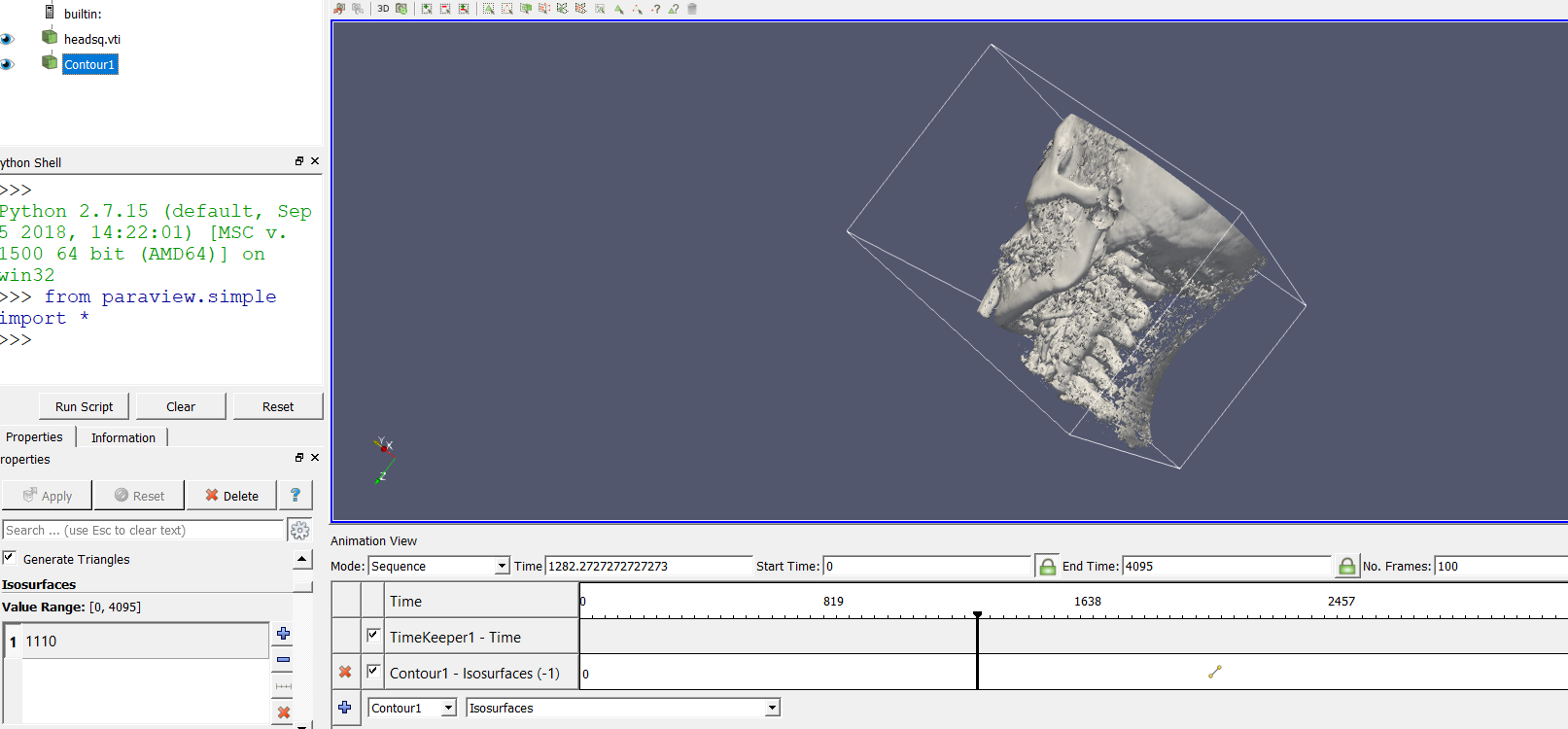


Isovalue - 1200 - complete skull

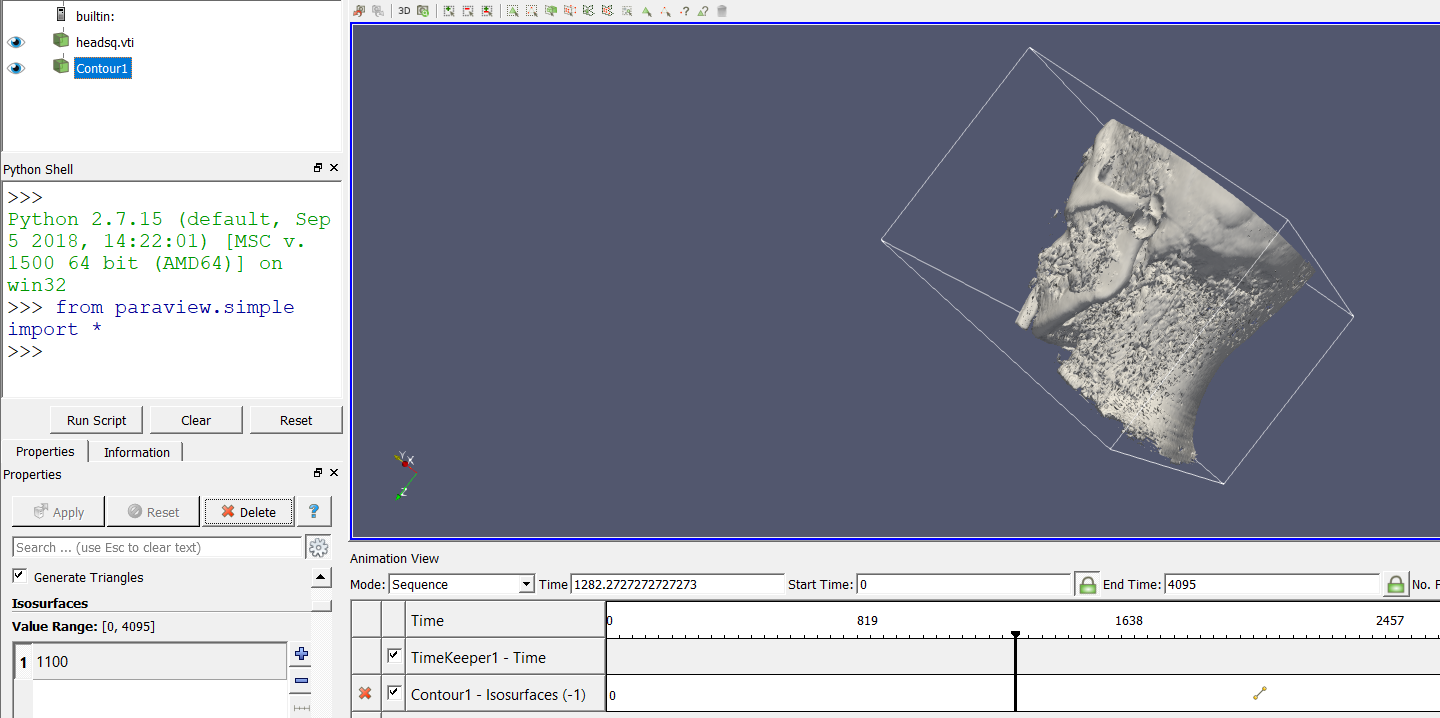


2. Spine vanishes completely at 1100 isovalue

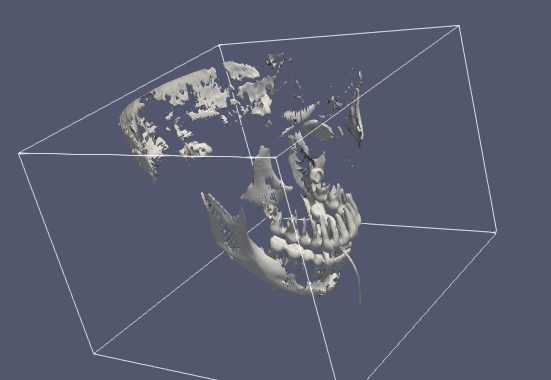
Isovalue 1110



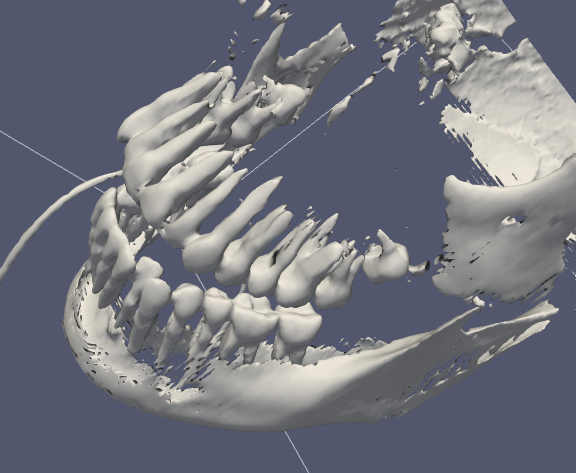
Isovalue 1100



If it means spine disappearing totally then at 2550 spine completely vanishes.



3. Teeth visible at 2500 Isovalue. Lower than that makes parts of the teeth to disappear



Q2. Reading Questions

1. Time-varying dataset is data collected over a fixed region or domain over time, representing the changes to the domain or object over time.

In many scientific simulations, the temporal variation and analysis of features are important. Visualization and visual analysis of time series data is still a significant challenge because of the large volume of data. Irregular and scattered time series data sets are even more problematic to visualize interactively.

To analyze complex dynamic phenomena from a time-varying dataset, it is necessary to navigate and browse the data in both the spatial and the temporal domain, select data at different resolutions, experiment with different visualization parameters, and compute and animate selected features over a period of time. To facilitate exploratory visual data analysis, it is very important that the visualization software be able to compute, animate, and track desired features at an interactive speed.

2. Temporal hierarchical index tree data structure provides an adaptive scheme that can reduces the storage overhead incurred by the search index for isosurface extraction in time-varying fields without sacrificing the performance too much. This helps classify the cells according to the amount of variation in the cell’s values over time.

Cells with small amount of variation are placed in a single node of the tree that covers the entire time span. Cells with a larger variation are placed in multiple nodes of the tree multiple times, each for a short time span.

When generating an isosurface, a simple traversal will retrieve the set of nodes that contains all cell index entries needed for a given time-step. The cells in each node can be organized using existing algorithms developed for generating isosurfaces from a steady dataset.

3. Given a temporal hierarchical index tree we run 2 phase algorithms on it. First is a simple traversal method to retrieve the sets of nodes that contain all cell index entries needed for a given time-step and then apply the isosurface cell search algorithm.

The isosurface for a given query is computed by first traversing the tree and locating the node containing the required isosurface cells. This is done in terms of traversing every child nodes of the root at each step until reaching the leave nodes. At every node along the traversal path the lattice search indexes which are built at the nodes is used to locate the isosurface cells as per the query.

The candidate isosurface calls as per the query can be collected in 3 ways:

* For every list in row *R, R* = *I* + 1..L-1 which are sorted on cells min value, the cells are collected from the beginning to the point where the min value of the cell is greater than the iso-value.
* For the list in a particular row *I* which was sorted on max values of the cells, we collect all the cells to the point where the max value of the cell drops below the iso-value.
* Collect the isosurface cells from the interval tree built at lattice element [*I,I*].

Once all the candidate isosurface cells are located we use the actual data of the cells' at the time-step of the query to perform triangulation.