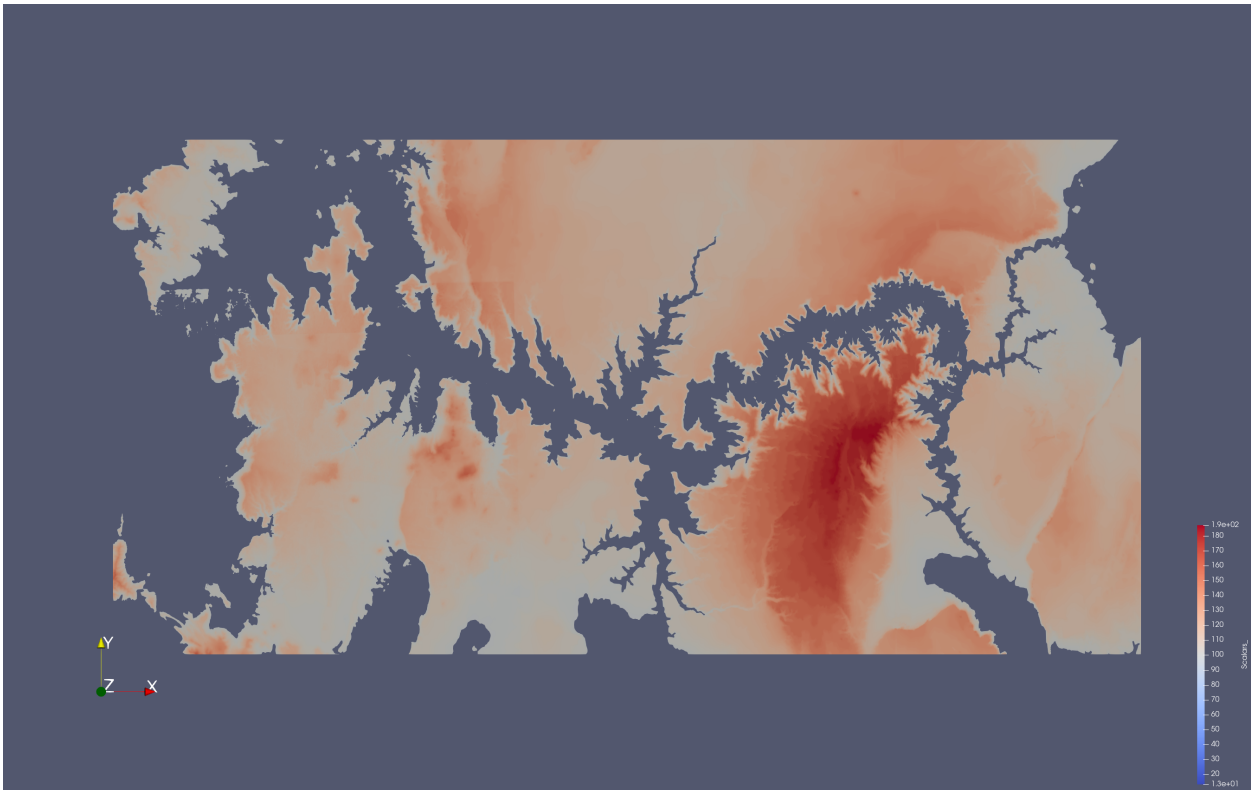
Question 1

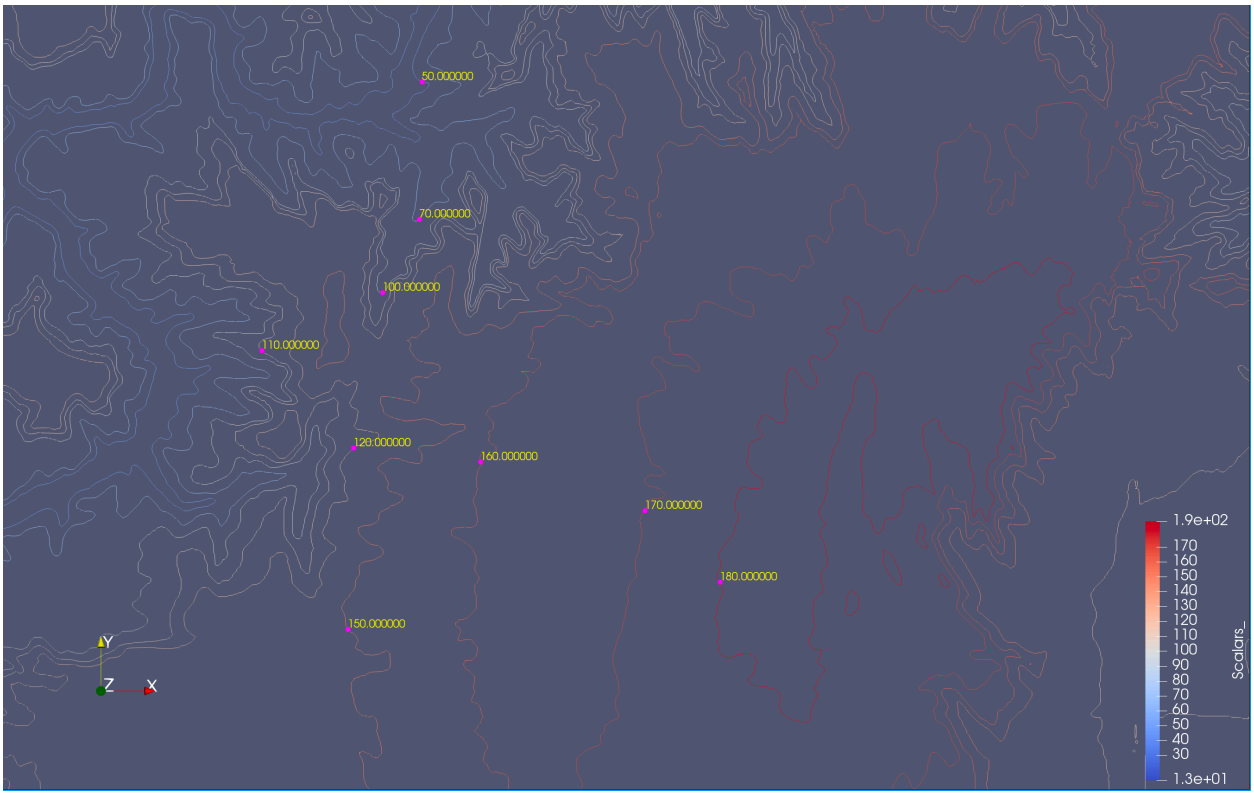


As the question suggests, the above is a thresholded image of the canyon where the non-selected pixels represent the river bed and the selected pixels represent the canyon.

By using a histogram with a bin count of 175, I have chosen a threshold whose minimum was 95 and maximum was 187. As the question hints, I have chosen this minimum threshold since it is slightly lesser than a bin which has a count of roughly ~200k. Further, around this value there seems to be a sudden jump in the histogram implying that terrain maybe flattening at this value. The thresholded image has 5964877 points.

I tried doing this various thresholds, however I noticed that on reducing the minimum threshold I may be missing some shallow crevices of the river bed and on increasing minimum the threshold I may be removing too much of the canyon itself.

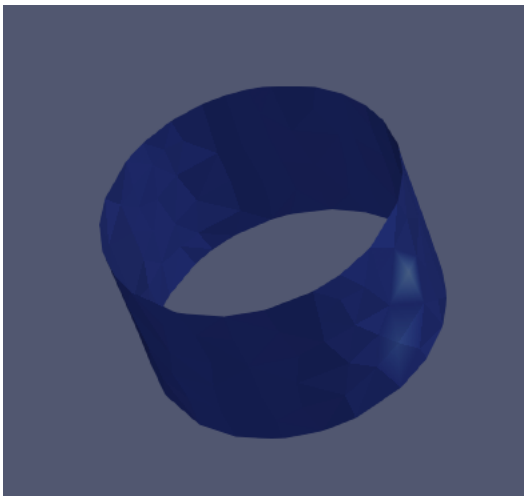
Finally, I would like to comment by saying with this approach one can't be too sure if they have captured the canyon exactly since it is always possible that the canyon may have lower altitude at some points and these need not be a river bed. In such a case, the threshold filter will not deliver the required result.



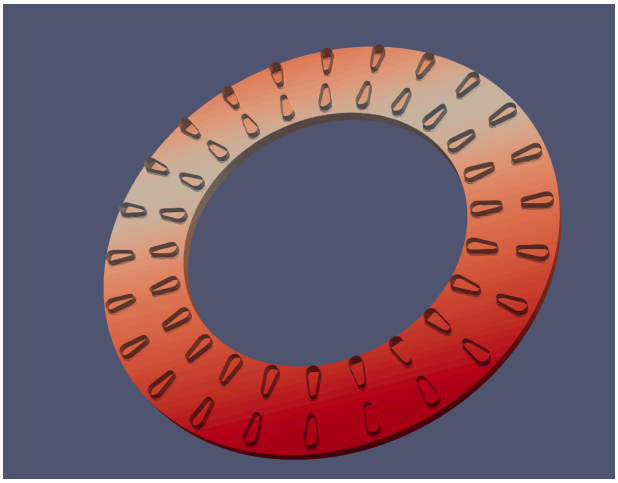
The above is the labelled contour plot of this data set. The contour values chosen is the same as the image in the question.

All of my results can be seen in the state file.

Question 2



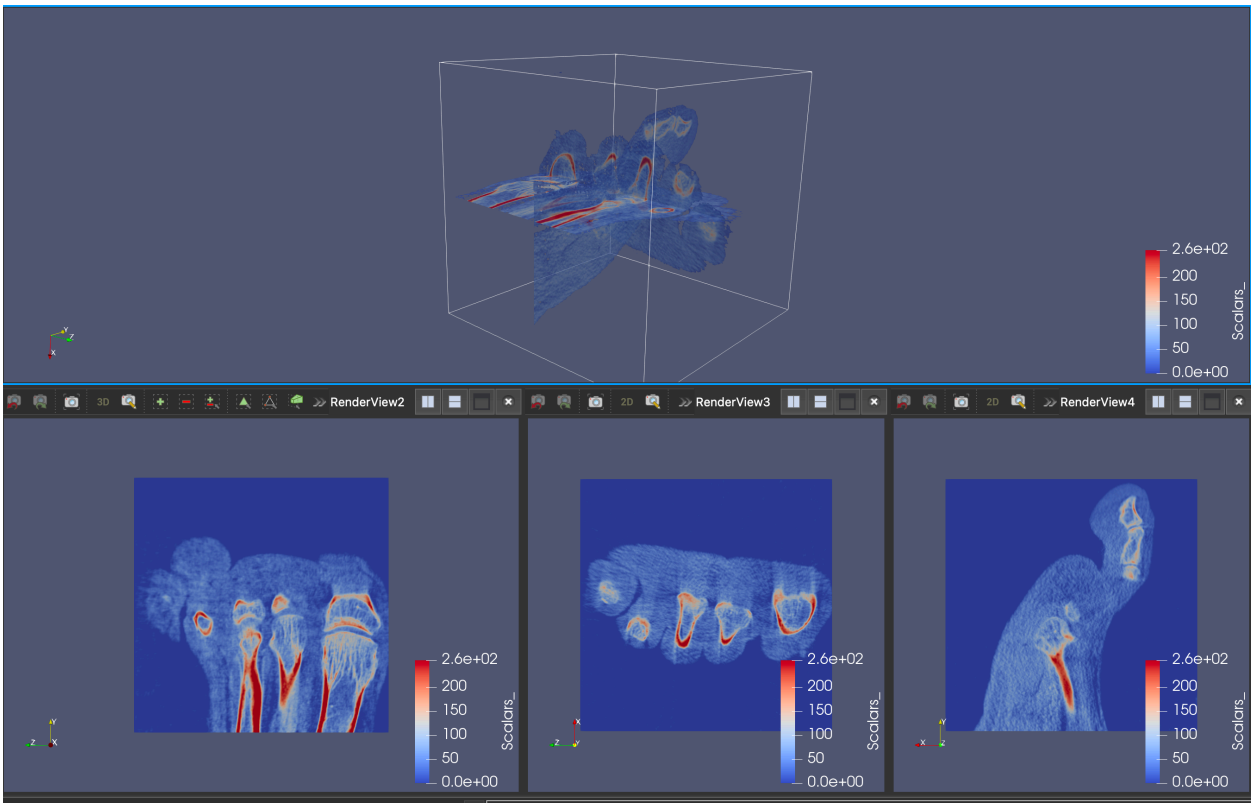
The above image is that of the single cylinder associated with the bolt's cylinder. It has been obtained by using minimum threshold of 0 and a maximum threshold of 30.



The above is the clipped image(along the y-axis) which shows the ventilation slots. From this image there seem to be 48 of these slots.

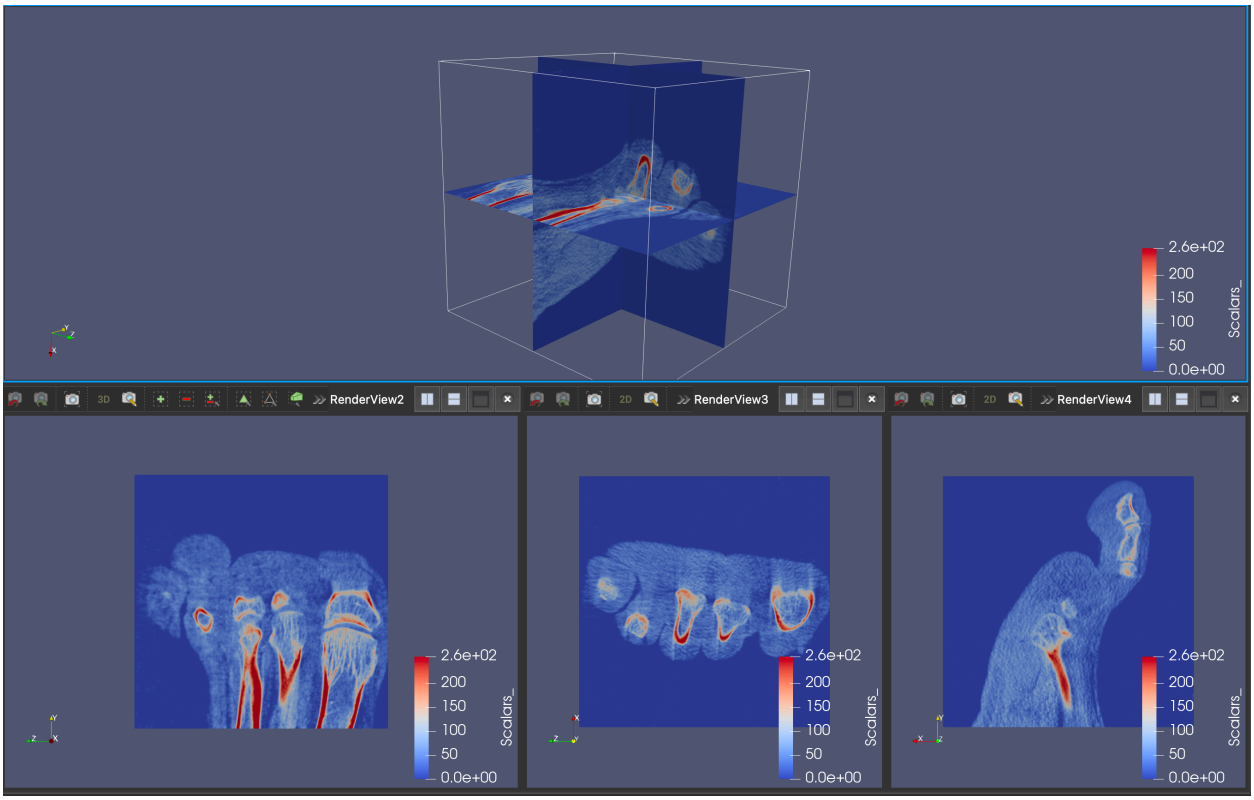
These images can been in my state file.

Question 3



The above image is linked view a 3D dataset where the bottom three views show the slice along a respective axis and the the top view is multi-slice view with a threshold to remove the background and opacity mapping.

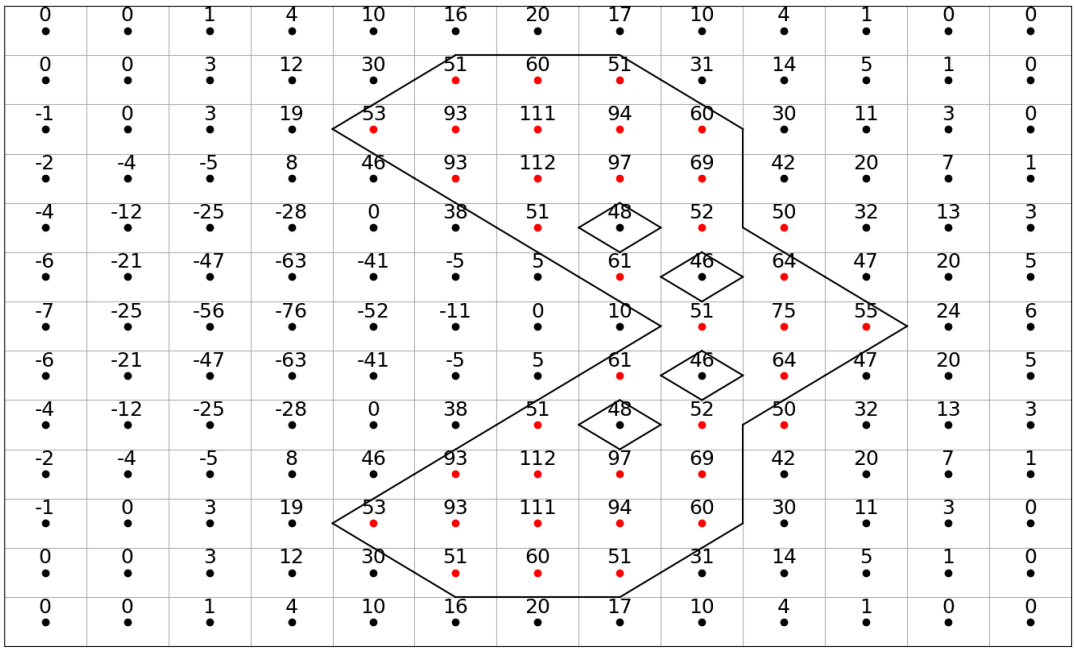
The following is a screen shot without applying threshold and opacity to the multi-slice view.

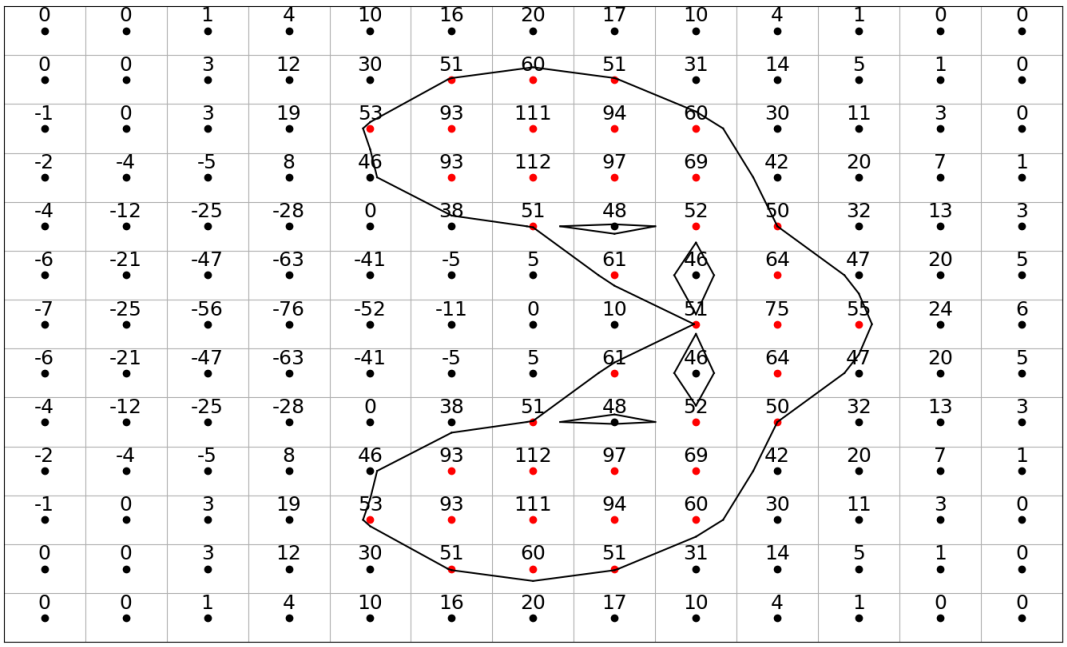


These are all present in the state file.

Question 4

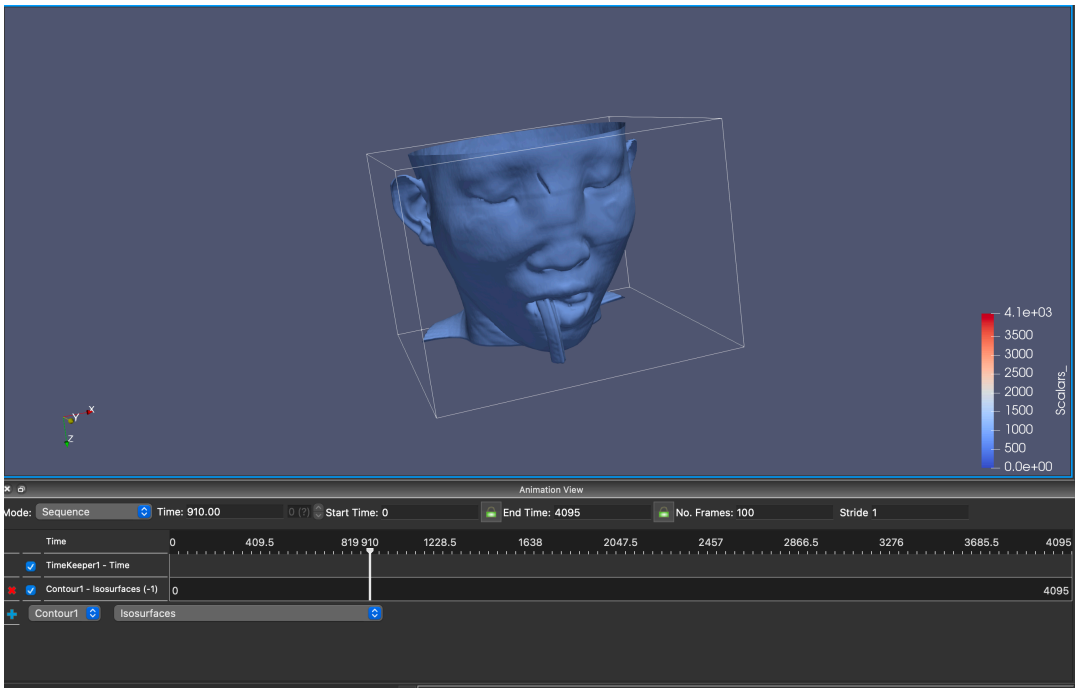
0	0	1	4	10	16	20	17	10	4	1	0	0
0	0	3	12	30	51	60	51	31	14	5	1	0
-1	0	3	19	53	93	111	94	60	30	11	3	0
-2	-4	-5	8	46	93	112	97	69	42	20	7	1
-4	-12	-25	-28	0	38	51	48	52	50	32	13	3
-6	-21	-47	-63	-41	-5	5	61	46	64	47	20	5
-7	-25	-56	-76	-52	-11	0	10	51	75	55	24	6
-6	-21	-47	-63	-41	-5	5	61	46	64	47	20	5
-4	-12	-25	-28	0	38	51	48	52	50	32	13	3
-2	-4	-5	8	46	93	112	97	69	42	20	7	1
-1	0	3	19	53	93	111	94	60	30	11	3	0
0	0	3	12	30	51	60	51	31	14	5	1	0
0	0	1	4	10	16	20	17	10	4	1	0	0

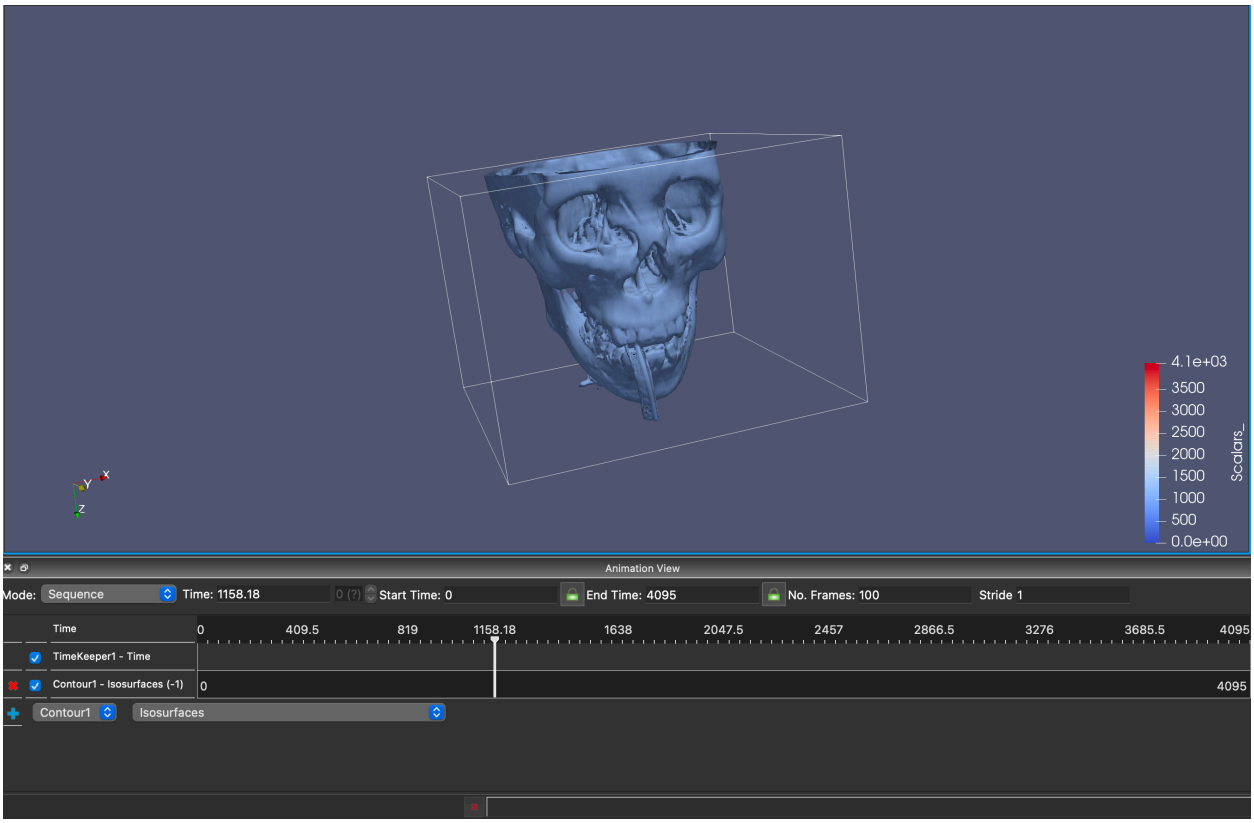


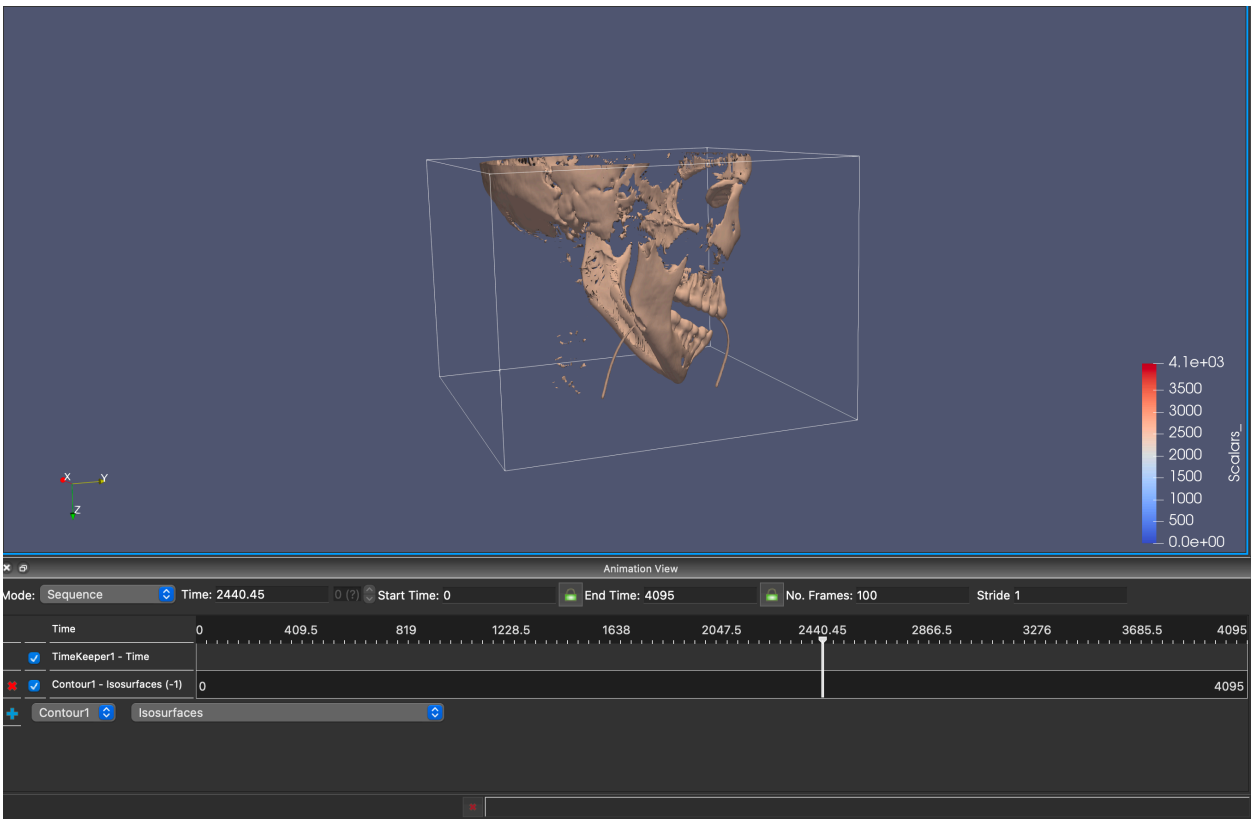


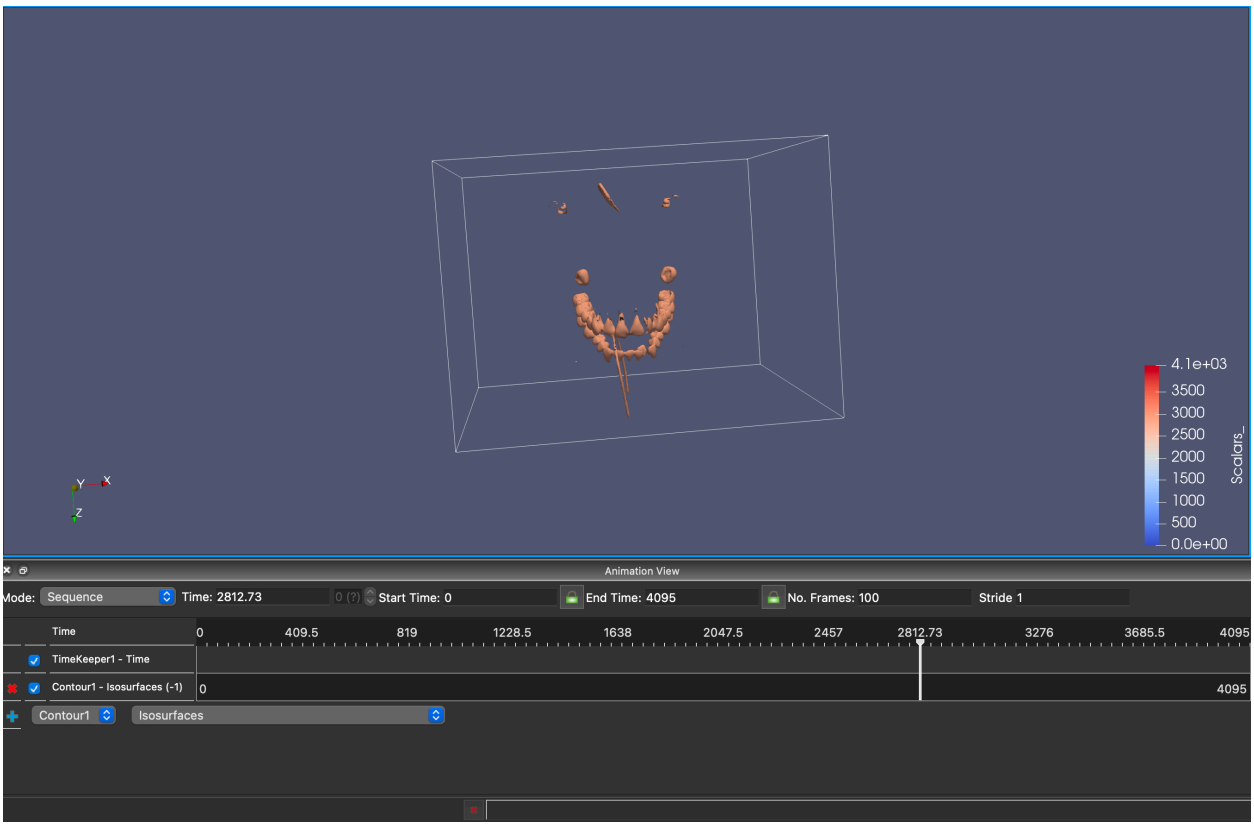
The first image depicts points in red as those above (or equal) to the isovalue = 50 and the points below the isovalue in black. The second shows the isocontour with the midpoint approach and the third shows the isocontour with linear interpolation.

Question 5




4.1e+03
3500
3000
2500
2000
1500
1000
500
0.0e+00
Scalars_
Animation View
Mode: Sequence
Time: 1158.18
Start Time: 0
End Time: 4095
No. Frames: 100
Stride 1
Time
0
409.5
819
1158.18
1638
2047.5
2457
2866.5
3276
3685.5
4095
TimeKeeper1 - Time
Contour1 - Isosurfaces (-1)
0
4095
Contour1
Isosurfaces


4.1e+03
3500
3000
2500
2000
1500
1000
500
0.0e+00
Scalars_
Animation View
Mode: Sequence
Time: 2440.45
Start Time: 0
End Time: 4095
No. Frames: 100
Stride 1
Time
0
409.5
819
1228.5
1638
2047.5
2440.45
2866.5
3276
3685.5
4095
TimeKeeper1 - Time
Contour1 - Isosurfaces (-1)
0
4095
Contour1
Isosurfaces



From the first 2 images it is clear that the transition between the skin and skull happens in the range of [819.910,1158.18]. From the third figure it is evident that the spine vanishes at 2440.45 and from the fourth figure the isovalue of the teeth is approximately 2812.73.

Reading Questions

1) Time-varying data is essentially data that changes with time. This data is majorly obtained from time-varying simulations with high temporal resolution and large number of time steps. In order to facilitate exploratory visual data analysis, it is important for the isosurfaces to be computed at an interactive speed as the time varies and it is this demand of high search performance while maintaining a low storage overhead that poses to be quite a challenge.

2) Most of the isosurface extraction algorithms such as ISSUE and NOISE are quite efficient but they incur a overhead for storing extra search indices. In the case of a single tilmestep this maybe affordable but for a lot of time steps this can accumulate to become quite large. Further, in time-varying data the user may want to look at various time steps back and forth and for possibly different isovalues and this would need a lot of disc I/O for getting indices at different time steps if there is

not enough memory for all the time steps. So the gain from the efficient isosurface extraction will be offset by the I/O overhead. To tackle these issues, the temporal hierarchical index tree is needed to keep the storage overhead small and the efficiency of the isosurface extraction high.

3) Given a query of an isosurface at a timestep `t', the temporal hierarchical index tree is traversed till the node `t'. It is to be noted that the tree is constructed such that every cell in the field lies in one of the nodes of the traversed path. So, within each node in the path, the lattice search index which was built is used to locate the relevant cells. In particular, given an isovalue, the lattice element with integer coordinates containing (isovalue,isovalue) is found in the span space. Then the isosurface cells are found in the upper left corner wrt this element.

Conclusions

All the state files, images and codes can be found in assignment 2 of this repository.

https://github.com/DhruvMeduri/CS6635.git