DATA VISUALIZATION

Assignment-1
2D Scalar and Vector Field Visualization

- **Tanishq Jaswani (IMT2018077)**

Introduction

Numerical simulations of the first stars in the universe reveal that they formed in isolation several hundred million years before the first primitive galaxies were assembled and that they were very massive: 100 - 500 solar masses. With surface temperatures greater than 100,000 K they were millions of times brighter than the sun, with most of their light in the form of hard (energetic) UV radiation. These UV photons advanced behind an abrupt wall of radiation known as an ionization front (or I-front) at well below the speed of light. The I-front itself is the extremely thin layer separating the hot (20,000 K), completely ionized gas from the cold (72 K), neutral gas beyond the front. The shock driven by the radiation front snowplows ambient gas into a dense layer that can erupt in violent dynamical instabilities. They want to understand the formation of galaxies, in particular the effect of "shadow instabilities", where radiation ionization fronts scatter around primordial gas.

Problem Statement

The problem statement is based on the 2008 IEEE Visualization Design Contest. For the given dataset, we have to choose sufficient timesteps to visualize the progression of the simulation. Following which we have to use the same time-steps for both multi-field and velocity files. After selecting one 2D plane we are supposed to study 3-5 scalar fields. For the vector field, we use the curl as the vector field to visualize.

Outputs

1. For scalar field visualization, use color mapping and contour mapping (or contour fill) for 5 contours. For vector field visualization, use quiver/arrow plots.

2. Experiment with different types of color palettes/spectrum (sequential, diverging, qualitative) using colorbrewer/matplotlib predefined palettes.
3. Experiment with combining 2 visualization techniques in a single view.

Dataset

The subject is an ionization front instability simulation data set submitted by Mike Norman and Daniel Whalen. All simulation data is saved in ASCII format. The data is saved in files with no headers. To enable proper interpretation of the data, we specify the x,y,z dimensions of the mesh and data types that the floating points represent, the units of the data, and the order in which the indices change. There is one such line for every grid point in each file. There is one file per time step in the simulation. The X indices value most rapidly, then Y, then Z; the first line in the file refers to element (0,0,0); the second to element (1,0,0); the last to element (599,247,247).

Line format: The values for each grid cell are laid out in a line. Each line has ten values, separated by a single space.

There are two types of data here: scalar (temperature, mass density, chemical species), and vector (velocity in km/s).

List of scalar data:

1. gas temperature (degrees Kelvin)
2. Total particle density (# of particles/cm^3)
3. H mass abundance
4. H+ mass abundance
5. He mass abundance
6. He+ mass abundance
7. He++ mass abundance
8. H- mass abundance

9. H_2 mass abundance
10. H_2+ mass abundance

The velocity data set is not of direct relevance to the questions being asked by the scientists, but the magnitude of the curl of the velocity field can be used as an estimator of turbulence, which is of direct interest. The formula for the three components of the curl vector field is:

- curl_x(i,j,k) = (vz(i,j+1,k) - vz(i,j,k) - vy(i,j,k+1) + vy(i,j,k)) / 0.001
- curl_y(i,j,k) = (vx(i,j,k+1) - vx(i,j,k) - vz(i+1,j,k) + vz(i,j,k)) / 0.001
- curl_z(i,j,k) = (vy(i+1,j,k) - vy(i,j,k) - vx(i,j+1,k) + vx(i,j,k)) / 0.001

Methodology

1. The data is first taken from the input files to produce images for each of the given time stamps.
2. The images then undergo pre-processing before they can be converted into a ".gif" using the pillow library.
3. The pillow library is finally used to generate the final output.

Common characteristics used:

1. Reading from the text files and extracting the relevant values.
2. Generating a 2D grid to store the values at each position.
3. The stored values are normalized to keep the scaling consistent across time frames.
4. Using Basemap from the matplotlib toolkits library.
5. Choosing the appropriate technique to overlap with Basemap.
6. Saving the plt instances generated.

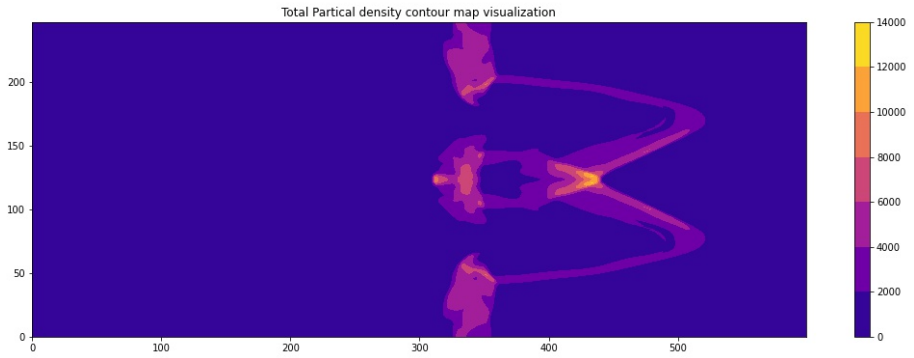
Visualizations

For scalar field visualization, we use contour mapping(or contour fill) and color mapping. For vector field visualization, we use a quiver plot. We found the contour without fill to be hard to understand but yet all plots gave the observer different levels of values over the

complete dataset. The minimum and maximum value of the colormaps have been fixed by taking the minimum and maximum value of the corresponding variable across all data files.

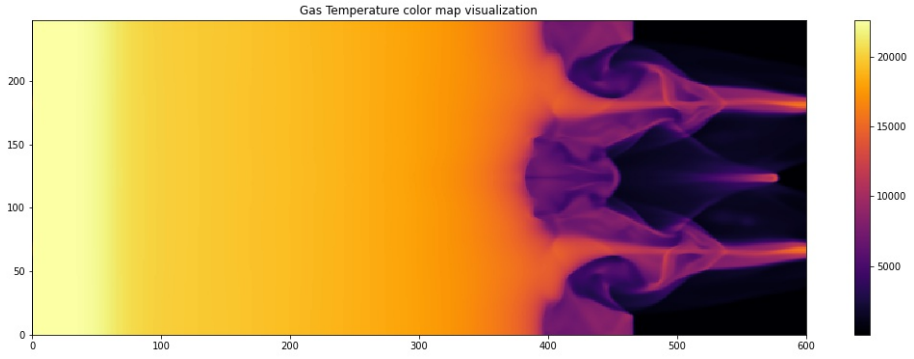
A. Total Particle Density

Technique used to portray the best visualization of total particle density was contour fill mapping. The Cmap we used was plasma and the color bar scale was kept as default.



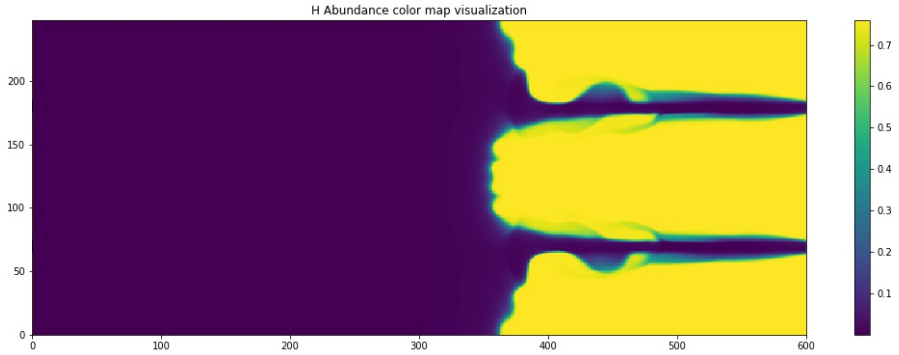
B. Gas Temperature

Technique used to portray the best visualization of gas temperature was color mapping. The Cmap we used was inferno and the color bar scale was kept as default.



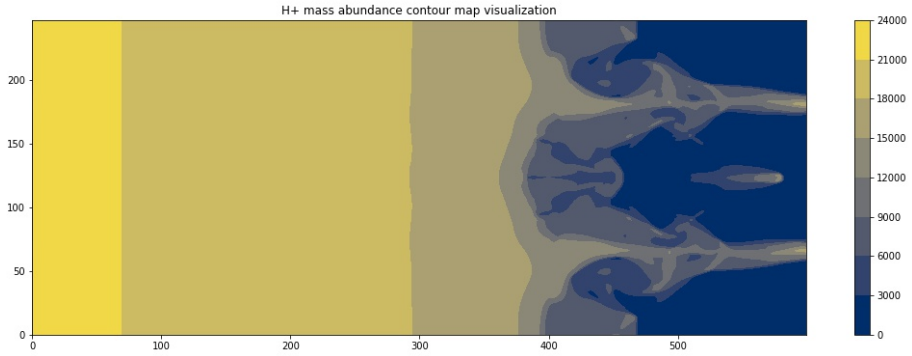
C. H mass Abundance

Technique used to portray the best visualization of H Mass abundance was color mapping. The Cmap we used was viridis and the color bar scale was kept as default.



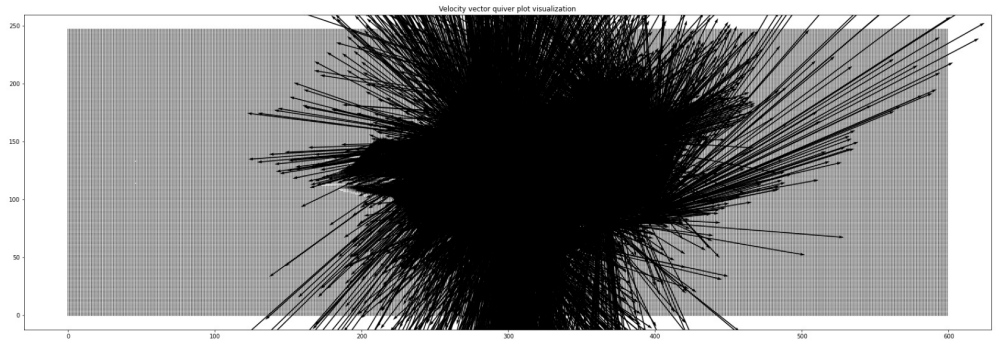
D. H+ mass Abundance

Technique used to portray the best visualization of H+ Mass abundance was contour mapping. The Cmap we used was cividis and the color bar scale was kept as default.

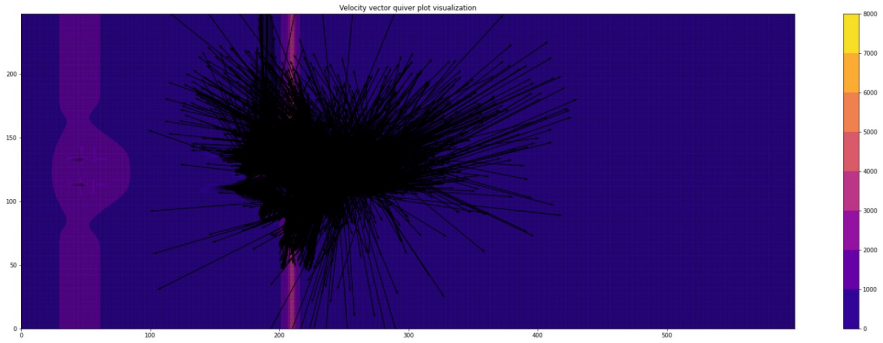


E. Velocity vector

Following our same methodology as above, we first read the data set by uploading a few txt files timestamps. We fill the velocity vector which we initialize by reading the dataset. The curl vector is used to improve the visualization as suggested on the website. We used a quiver plot on a 2D x-y plane and scaled it upto 10000.



Coming to the last part where we combine scalar visualization we did it in part a and vector visualization we did it in part b for every 20 time steps. Following is the magical visualization we get.



Conclusion

Q. How did you arrive at the timesteps?

Ans. Since the zslice multifield data was easily accessible to the time stamps and did not increase the run time of the code so we were able to easily visualize the scalar values. For the vector fields - velocity we used every 20 time stamps and uploaded the data to visualize. We did this to keep our run time to visualize less than 6hrs.

Q. How did you arrive at which plane you are going to explore? What is your rationale for the selection of variables?

Ans. For better and informative visualization we choose total particle density and gas temperature as 2 scalar values to visualize and two other types of hydrogen molecule H and H+ mass abundance. We took the z value constant and visualized x-y 2D plane for clearer visualization.

Q. For contours, will you use the same contour values for all time steps? For color-mapping, will you use the same min-max values to generate the color palette? Did any color palette outperform the others?

Ans. For color mapping we use Max-Min Normalization while in contour mapping we use 2 * {(val - min) / (max -min)} - 1 as a normalization technique. Yes we used the same contour values for all the time stamps for easier comparison visualization. We found the inferno in the counter-fill in the “gas temperature” very amazing.

This assignment made us learn new techniques to visualize scalar as well as vector values in 2D planes.