SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

PROJECT REPORT

on

Data visualization of time-varying multivariate volume data using matrix of isosurface similarity maps.

Vellore-632 014, Tamil Nadu, India.
www.vit.ac.in

15BCE0082

VOLETI RAVI

CONTENTS

I. ABSTRACT

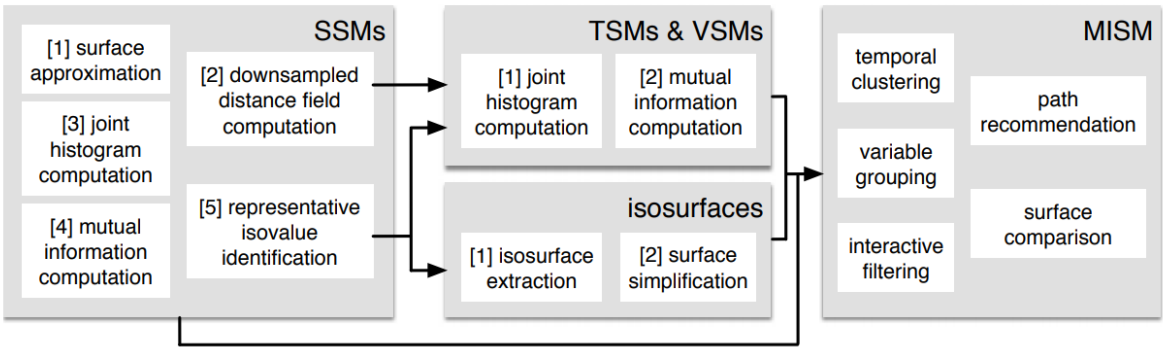
Isosurfaces are an important visual representation of volumetric data sets and isosurface extraction and rendering remains one of the most popular methods for volume visualization. Previous works identify a small set of representative isosurfaces from a set of sample ones, providing a concise description of the underlying volume. The project presents a novel visual representation and interface named the matrix of isosurface similarity maps for effective exploration of large time-varying multivariate volumetric data sets. MISM synthesizes three types of similarity maps (i.e., self, temporal, and variable similarity maps) to capture the essential relationships among isosurfaces of different variables and time steps. Additionally, it serves as the main visual mapping and navigation tool for examining the vast number of isosurfaces and exploring the underlying time-varying multivariate data set. The project present temporal clustering, variable grouping, and interactive filtering to reduce the huge exploration space of MISM. In conjunction with the isovalue and isosurface views, MISM allows users to identify important isosurfaces or isosurface pairs and compare them over space, time, and value range. More importantly, the project introduces path recommendation that suggests, animates, and compares traversal paths for effectively exploring MISM under varied criteria and at different levels-of-detail. A silhouette-based method is applied to render multiple surfaces of interest in a visually succinct manner. The project demonstrates the effectiveness of approach with case studies of several time-varying multivariate data sets and an ensemble data set, and evaluate this work with two domain experts.

II. INTRODUCTION

There are two dominant techniques to visualize volumetric data sets: **direct volume rendering** and **isosurface rendering**. Direct volume rendering relies on the specification of a transfer function that maps scalar values to colours and opacities for rendering. The visualization can highlight particular values or value ranges with high opacity while deemphasizing non-interesting ranges with low or zero opacity. Isosurface rendering explicitly extracts one or multiple isosurfaces from the volume data and renders these surfaces as the visual data representation. The rendering can reveal important data features or structure if the corresponding isovalues are salient or representative. In this work, the project focus on the less-explored surface-based analysis in the context of time-varying multivariate data visualization. This goal is to investigate volumetric data evolution over space and time and across multiple variables and ensembles by extracting a range of isosurfaces, computing their similarities, identifying representative surfaces, and presenting a visual interface to organize, summarize, and explore the underlying

time-varying multivariate data set. Several key challenges are involved with this approach. First, studying the similarities for all the surfaces across time steps and variables leads to a very large number of similarity maps. For instance, for a data set of 100 time steps and ten variables, the total number of similarity maps amounts to 55,000. These include 1000 self-similarity maps, 49,500 temporal similarity maps, and 4,500 variable similarity maps. Without an effective way to organize and visualize them, examining such a large number of similarity maps would become a daunting task. Second, even though similarity maps can help us see the summarized information, the project still need to examine related isosurfaces in the original spatial view to identify where and how they are (dis)similar. However, since a single similarity map typically compares tens or even hundreds of isosurfaces, it is not practical for users to manually select the related isosurfaces for further examination. This task will be even more exhausting when the temporal development of a variable is studied as thousands of isosurfaces may be involved. Third, rendering just a few surfaces may already lead to serious occlusion and clutter which prevents clear observation and comparison. This calls for new techniques that render a number of surfaces with minimal occlusion while preserving their spatial relationships or context. Finally, it could take hours to compute pairwise isosurface similarities for only a single volume if no approximation, down sampling, or acceleration solutions are considered. This makes it impractical to apply the same computation process to generate similarity maps for a typical time-varying multivariate data set, although the process is only done once. Therefore, it is imperative to seek a cost-effective solution to make the similarity computation scalable.

The project tackles the above challenges and present the matrix of isosurface similarity maps (MISM), a new approach for comparative visualization of time-varying multivariate volume data. During pre-processing, self-similarity maps are first computed for each volume of the data. For each self-similarity map, the project then selects a set representative isovalues and compute the corresponding isosurfaces. Finally, each of these isosurfaces is simplified to save storage space and support runtime interaction. Furthermore, the information acquired from self-similarity map computation will be used to derive temporal and variable similarity maps. At runtime, all similarity maps (self, temporal, and variable) will be used to construct MISM. The project provide a list of functions for users to explore the collection of similarity maps and compare the underlying isosurfaces.



The diagram of the framework. Self-similarity maps (SSMs) are computed for each volume of the data, from which the project compute representative isovalues and isosurfaces and further derive temporal similarity maps (TSMs) and variable similarity maps (VSMs). At runtime, SSMs, TSMs, and VSMs are used to construct MISM for users to explore the underlying data set.

III. DESIGN REQUIREMENTS

1. **OVERVIEW** The system should provide two levels of overview. First, it should allow users to observe the overall relationships between variables and time steps at the volume level, helping them answer questions such as which variables are more similar, which time steps are more similar, and which variables exhibit more frequent changes than the others? Second, it should allow users to understand the relationships at the isovalue level. Similar questions should be answered at this level with more detail: which ranges of isovalues lead to the relationships discovered at the volume level?

2. **IDENTIFICATION OF REPRESENTATIVES.** The system should quantify the similarities among time steps, variables, and isovalues and derive the representative ones. This provides users more quantitative evidence to verify the relationships they observe from the overview and to reduce the search space for detailed exploration.

3. **RELATIONSHIP-CENTRIC EXPLORATION**. The system should provide multiple modes to investigate various types of relationships, including the relationships among value ranges for a given variable at a specific time step, the relationships among variables at a specific time step, the relationships of the same variable at different time step, the temporal evolution of the relationships of two variables, and the relationships of a specific feature and others (which may not reside in the same variable), etc. The system should be able to quantify each type of relationship and the interface should be optimized to allow exploring each type of relationships efficiently.

4. **MULTI-STEP COMPARISON PATH**. The dynamic relationships among variables may require comparisons of hundreds of surfaces to be demonstrated. The system should decompose these complicated relationships into a path of multiple comparison steps. Each step should reveal a facet of the relationships with a reasonable number of surfaces that can be effectively rendered for clear visual comparison. In addition, the system should also maintain the consistency between surfaces in consecutive steps to preserve the user's mental map.

5. **PATH CUSTOMIZATION**. Users should be able to interact with the system to customize paths focusing on certain variables of interest or a specific type of relationship described in R3. The customized path should answer questions such as how does a feature evolve over time, what are the surfaces related to the selected one over time, and what are the most different surfaces from the selected one over time?

6. **PATH ANIMATION**. The system should produce an animation that concisely describes the comparison path with isosurface rendering. The animation should answer the questions in (1) with more detail: it not only indicates which isosurfaces are (dis)similar, but also describes how they are (dis)similar by providing the isosurfaces in the spatial view. For effective visual comparison, the isosurfaces at each animation frame may need to be rendered in different styles, allowing the more important ones to be observed clearly.

REQUIREMENTS

- GCC/G++ Version 6+
- GCC Version 5 for CUDA part
- CUDA
- ThorSerializer (https://github.com/Loki-Astari/ThorsSerializer)
- SDL2
- Boost (program_options, regex, system, filesystem)
- Tested with
- GCC/G++ 6.2.0
- GCC 5.2.0
- Cuda 8.0
- Boost 1.63

PROGRAM MODULES

The built executable contains several modules. To run, use ./isosurfaces

The following modules are available:

multi - computes the VSM & TSM

single - computes a single ISM

distance - computes a single distance equalization

script - computes a set up ISM or Distances

stitch - stitches together several Json files

There are further modules in the program, but they are for testing purposes only and left undocumented.

MODULE INPUT

Different modules need different input files.

Single & Distance need a file specifying a single volumetric data point (a certain time step & variable) (see configs/single.json)

Script needs a json file as input, that describes a volumetric data set (see configs/ionization-distance-all.json)

Multi uses a short input file specifying where to read the output from an ensemble of "Single" runs (see configs/multi.json)

Stitch uses a list of json files as input. The first one is the so-called "host file" and should contain most entries. The entries of the other json files will be added to this one, and the result will be saved in a separate json file.

IV. DATA SET

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 could not simply stream into the universe because they were capable of ionizing H and He into ions and free electrons. Instead, they 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 radiation waves at first propagates supersonically through space, leaving gas undisturbed in its wake, but then slows as it recedes from the star. The hot ionized gas drives a shock that overtakes and pushes past the front, and the radiation wave is subsonic thereafter. The shock driven by the radiation front snowploughs ambient gas into a dense layer that can erupt in violent dynamical instabilities as

shown in the figure below from. There, the initially planar I-front sweeps over a spherical blob of gas and is deformed by it. The dimple destabilizes and fragments into a jet, a phenomenon known as a shadow instability. I-front instabilities endow nebulae in the galaxy today with spectacular morphologies (such as the "pillars of creation' in the Eagle Nebula) and recent numerical work confirms they were also present in those of the first stars.

Besides being aesthetically pleasing, instabilities in primordial radiation fronts clumped gas that might have later collapsed gravitationally into the next generation of stars, accelerating the formation of the first galaxies. While we have numerical proof of their existence, much remains unknown about these beautiful structures in the primeval universe and how they may have fostered subsequent star formation. The scientists need your help as visualization experts to unravel the role of I-front instabilities in structure formation in the early universe. [16]

DATA FORMAT

COORDINATE SYSTEM

In this simulation, the region is treated as a flat Cartesian box. This volume is divided into regular squares with a 0.001 parsec spacing in all directions in the pre-decimated mesh, creating a 600x248x248 point regular mesh. This results in a mesh consisting of 36.9 million regular cubes. The output data is saved as single file per time step.

Time: The original simulation stored the state of the entire volume approximately every 126.8 years. This was calculated for a total of 25,370 years for a total of 200 time steps. The actual time corresponding to each time index is stored here; the values range from around 126 years to around 128 years between steps.

DATA LAYOUT

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 are two types of data here: scalar (temperature, mass density, chemical species), and vector (velocity). All of the scalar fields for one-time step are stored in a single gzip-compressed ASCII file named multifield.XXXX.txt.gz, where XXXX is the time step index (starting at 0000).

Line format: The values for each grid cell are laid out in a line that ends with a single ASCII newline (character 10, 0a in hexadecimal) except the last line, which has no terminating character. Each line has ten values, separated by a single space.

There is no space between the last character of the last value and the newline ending the line. Each value is represented in scientific notation an optional '-' sign, then one digit, then a decimal point, then three more digits, then 'E', then + or -, then a three-digit exponent. An example line (the first line from the first line from multifield.0030.txt) is shown here:

8.563E+002 2.051E+004 4.180E-004 7.596E-001 5.260E-005 6.898E-002 1.710E-001 8.053E-011 2.686E-013 8.650E-012

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).

The order of the ten columns on each line is:

total particle density (# of particles/cm^3)

gas temperature (degrees Kelvin)

H mass abundance

H+ mass abundance

He masse abundance

He+ mass abundance

He++ mass abundance

H- mass abundance

H_2 mass abundance

H_2+ mass abundance

DENSITY DATA SET

This is the total number of particles (of any type) per cubic centimetre. This and all scalar fields are sampled at the cell centre.

GAS TEMPERATURE

Temperature is stored in degrees Kelvin. Ambient gas is very cool (72 K). Shocked gas is around 2000-3000 K. Ionized gas is much hotter: 20,000 Kelvin). Temperature thus indicates where shock waves and radiation are present. This and all scalar fields are sampled at the cell centre.

CHEMICAL SPECIES DATA SETS

The chemical-species data values are relative abundancies of the eight different chemical species being simulated (H, H+, He, He+, He++, H-, H_2, and H_2+). These values are normalized per grid cell so that they sum to 1. This is the fraction of total mass composed of each of these elements. This and all scalar fields are sampled at the cell centre.

VELOCITY DATA SET

The velocity data set is stored in a separate gzip-compressed file per time step, named velocity.XXXX.txt.gz, where XXXX is the time-step index (starting at 0000).

There are three components of velocity, X, Y, and Z in km/s. They are stored sequentially in the line. They are also separated and denoted as the values in the scalar-field file. Although the scalar data is cell-centred, the velocity data has each component centred on its respective lower face of the cell (x-velocities are centred on the lower x face, y-velocities on the lower y face, and z-velocities on the lower z face). An example line (from velocity.0030.txt) follows:

8.555E-012 -2.576E-040 9.431E-016

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

Note that since velocities are face-centred in ZEUS-MP the derivative terms above (and therefore the components of the curl) are cell-centred. The vorticity (magnitude of the curl) is thus also cell centred.

V. IMPLEMENTATION

OVERVIEW

The project requires a set of interactive functions for navigating the huge matrix of similarity maps. Firstly, cluster temporal sequences and group variables to reduce the exploration space. The project also highlights representative cells in a similarity map to attract user attention and guide the exploration. A typical workflow to explore a data set using MISM is as follows: Users will start by examining the matrix view in the evolution mode, which shows the SSMs and

VSMs of all the representative time steps and variables. This provides users an overall understanding of the data set and guides them to discover the time steps and variables of interest. Then, users can simply click and drag to form a path that reveals the temporal development of the selected variables and time steps. Users can further create waypoints to edit the traversal path so that the connections to additional features can be discovered. In the evolution mode, users can easily click an SSM to enter the single variable mode or click a VSM to enter the single-pair mode for detailed exploration. They can specify a time step to explore the relationship among all variables in the all-pairs mode as well. Finally, they can create paths to investigate the isovalues in these detailed modes. Next, the project introduces temporal clustering and variable grouping, path recommendation and graph construction, and silhouette-based rendering and animation.

CLUSTERING AND GROUPING

The project reduces the size of MISM along the temporal and variable dimensions through clustering temporal sequences and grouping variables, respectively. This not only allows users to quickly identify the representative time steps or variables for exploration, but also produces more compact paths and animations for efficient knowledge discovery. The project defines a similarity measure between volumes using TSMs or VSMs and use it to derive the similarity between two time steps or two variables. The project applies affinity propagation to cluster the temporal sequences based on the similarity measure. Affinity propagation automatically determines the number of clusters, which naturally reflects how frequently the variables change along time in a data set. For variables, the project use k-means clustering to group the variables into the desired number of groups. In the following, the project only describes our similarity measure for temporal clustering, as the detail for variable grouping is the same. Similarity measure. The project evaluates the similarity of two volumes based on the similarities among their isosurfaces. Two volumes are considered to be similar if for each isosurface in one volume, a similar isosurface can be identified in the other volume. Our similarity measure is analogous to the mean of closest point distances defined on two curves by considering each volume as a curve and each isosurface of the volume as a point on the curve. Formally, the similarity of an isosurface S 0 and a volume V is defined as:

$$\mathscr{S}(\mathbf{S}', \mathbf{V}) = \min_{\mathbf{S} \in \mathbf{V}} \mathscr{S}(\mathbf{S}', \mathbf{S}),$$

where the similarity between two isosurfaces, S (S 0, S), can be looked up from TSMs or VSMs. The similarity of two volumes Vi and Vj is defined as the

weighted average of the similarity of each isosurface in one volume to the other volume, i.e.

$$\mathscr{S}(\mathbf{V}_i, \mathbf{V}_j) = \frac{\sum_{\mathbf{S}_p \in \mathbf{V}_i} w_{\mathbf{S}_p} \mathscr{S}(\mathbf{S}_p, \mathbf{V}_j)}{\sum_{\mathbf{S}_p \in \mathbf{V}_i} w_{\mathbf{S}_p}} + \frac{\sum_{\mathbf{S}_q \in \mathbf{V}_j} w_{\mathbf{S}_q} \mathscr{S}(\mathbf{S}_q, \mathbf{V}_i)}{\sum_{\mathbf{S}_q \in \mathbf{V}_j} w_{\mathbf{S}_q}},$$

where the weight wSp for an isosurface Sp is derived based on its representativeness ranking, which is used for selecting the most representative isovalues. Note that the similarity between any pair of volumes is in [0,1], which has the same range as the similarity between any pair of isosurfaces. Temporal clustering. Temporal clustering evaluates the combined similarity between two time steps as the summation of similarities calculated for each of the variables using Equation 2. Affinity propagation is applied to cluster the time steps based on their combined similarities. For each cluster, the clustering algorithm identifies one exemplar, which is considered as the representative time step for that cluster.

PATH RECOMMENDATION

MISM has a two-tiered structure: maps at the coarse matrix level and cells at the fine map level. The map-level captures the overall relationships among volumes, while the cell-level allows the detailed relationships among isosurfaces to be discovered. Given two user specified maps (cells) as the start and end points, path recommendation identifies a series of intermediate maps (cells) to construct a path for traversal. The project creates an animation for the generated path, showing the isosurfaces corresponding to a map (cell) at one animation frame. Users can adjust the weights of different terms to define the desired path. They may visit the maps or cells to discover affinity relationships or compare distinct features. The differences between frames can be minimized or maximized for generating a smooth animation or increasing information gain. They can specify the start and end points of a path in the matrix view, and drag any point along the path to add waypoints. The project identifies a path that minimizes the total cost between the user-specified points. The project introduces the following types of paths: (1) map-level paths for the evolution mode, (2) cell-level paths for all the four modes, and (3) variable traversal paths for only the all-pairs mode. All three types of paths are built with similar constructions. In the following, the project discusses map-level paths in detail. For the other two, the project only explains their differences with respect to map-level paths.

The project denotes a similarity map between two variables A and B at time step τ as $M_{AB,\tau}$ and their corresponding volume as $V_{A,\tau}$ and $V_{B,\tau}$. Note that $M_{AB,\tau}$ can refer to an SSM when A = B or a VSM when A 6= B. However, it does not

refer to a TSM (which is not displayed in the matrix view) as the indices do not specify two different time steps. The project use MAB,τ [i, j] to denote a cell in MAB,τ at the i-th row and the j-th column, and SA,τ,i and SB,τ, j to denote the two isosurfaces corresponding to the i-th row and j-th column, respectively. The project may ignore τ in the notation for simplicity when time is not discussed. The project defines the difference D (,) between two elements as 1−S (,). Map-level path. The project identifies the map-level path between two similarity maps as the shortest path in a map-level graph. A map-level graph is a directed graph whose nodes are all SSMs and VSMs and whose edges are all the possible transitions from one map to another, Specifically, the project considers two kinds of transitions:

(1) variable transition between similarity maps at the same time step sharing at least one common variable, as shown by the blue arrows; and

(2) temporal transition between similarity maps at neighbouring time steps, as shown by the red arrows.

Weighing the edges appropriately is critical to obtaining a path that shows the desired features. In our approach, the weight of an edge corresponding to a transition is a linear combination of the target cost Ctg and the transition cost Cts raised to a user-specified power exponent α: (wtg Ctg +wts Cts) α, where wtg and wts are the weights of Ctg and Ctg, respectively. The project includes α to further distinguish the edge costs, so that the shortest path is less likely to end up with a path with a larger average cost but a smaller number of edges. Consider the transition MAB → MBC. The target cost C (MBC) is the difference D (VB, VC) between the two corresponding volumes VB and VC. When the target cost is weighed positively, the shortest path is more likely to visit the maps corresponding to variables with similar structures by minimizing the cost. When the target cost is weighed negatively (the project use 1 − C (MBC)), the resulting path tends to visit the variables with different behaviours for the most surprise. The transition cost is the penalty when the project replaces the isosurfaces to display for MAB to those for MBC. It can be weighed positively in the linear combination, so that the shortest path minimizes the transition cost and maintains a smooth animation between neighbouring frames; otherwise, it can be weighed negatively using 1−C (MAB → MBC) to visit the maps with diverse information. A variable transition from a VSM MAB,τ to another MBC,τ indicates that the focus of analysis shifts from one pair of variables (A and B) to another pair (B and C). After the transition, the project replaces the isosurfaces of A with the ones of C in the isosurface view. Therefore, the project weigh the corresponding edge by the difference of their volumes, i.e., D (VA,τ ,VC,τ). The edge weight for

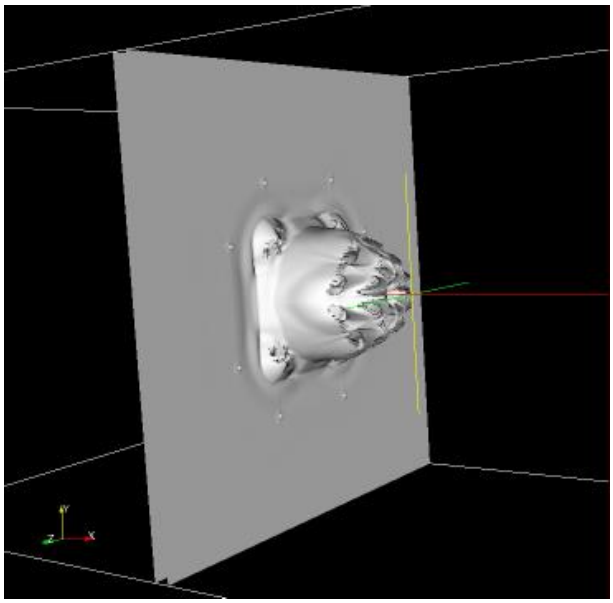
transitions between a VSM and an SSM can be defined similarly by letting A = B. For coherence, variable transitions should take place only if necessary. To avoid the paths from jumping back and forth among the variable pairs, the project includes a sufficiently large jumping cost in the variable transition. A temporal transition between two VSMs MAB,τ and MAB,τ+1 describes the evolution of the two variables A and B over time. The corresponding edge weight is defined as (D (VA,τ ,VA,τ+1) + D(VB,τ ,VB,τ+1))/2.

VI. RESULTS

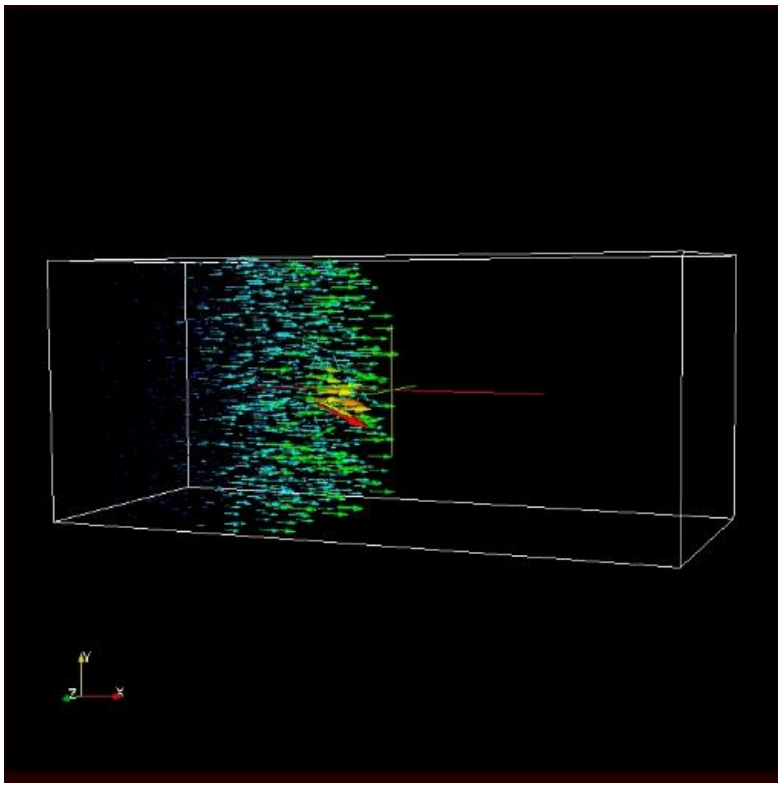
Unfortunately, there was no, proper data set available in the form of JSON file for visualization using isosurface analysis.

Therefore, I recreated the isosurface analysis using the vtk files on Para View.

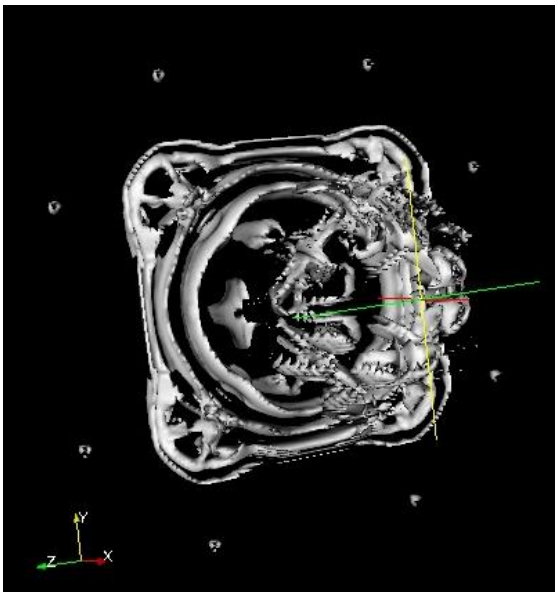
The image below shows a view of the isosurface of the density data set above taken at the density level 2000 at time step 30.



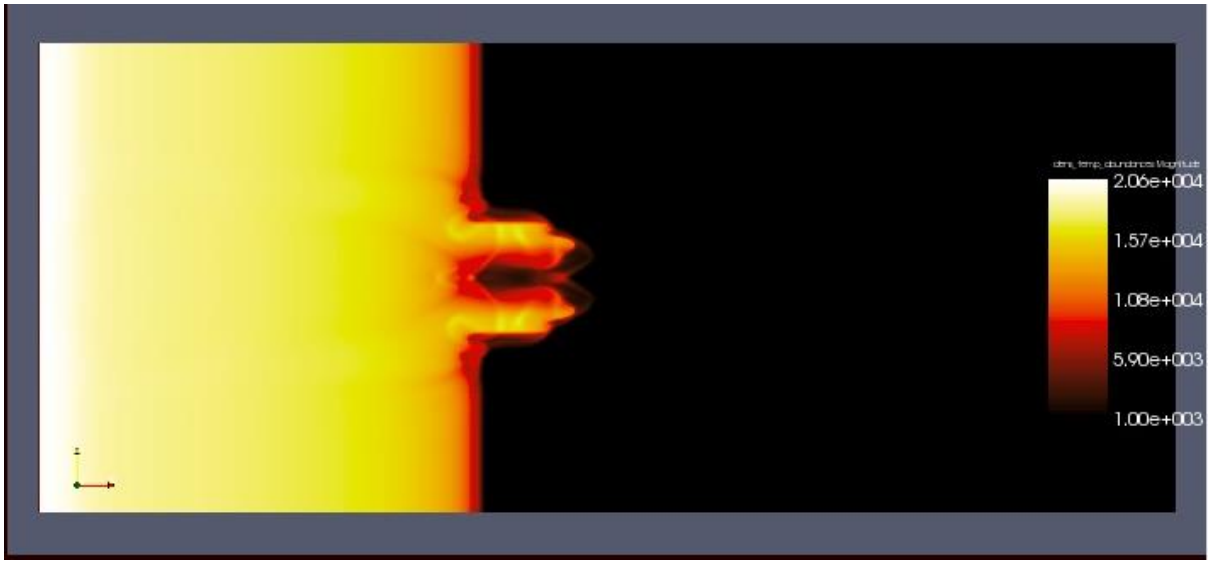
The image below shows the results from applying the glyph filter in Para view 2.4 to the loaded binary file, using the default settings, at time step 30.



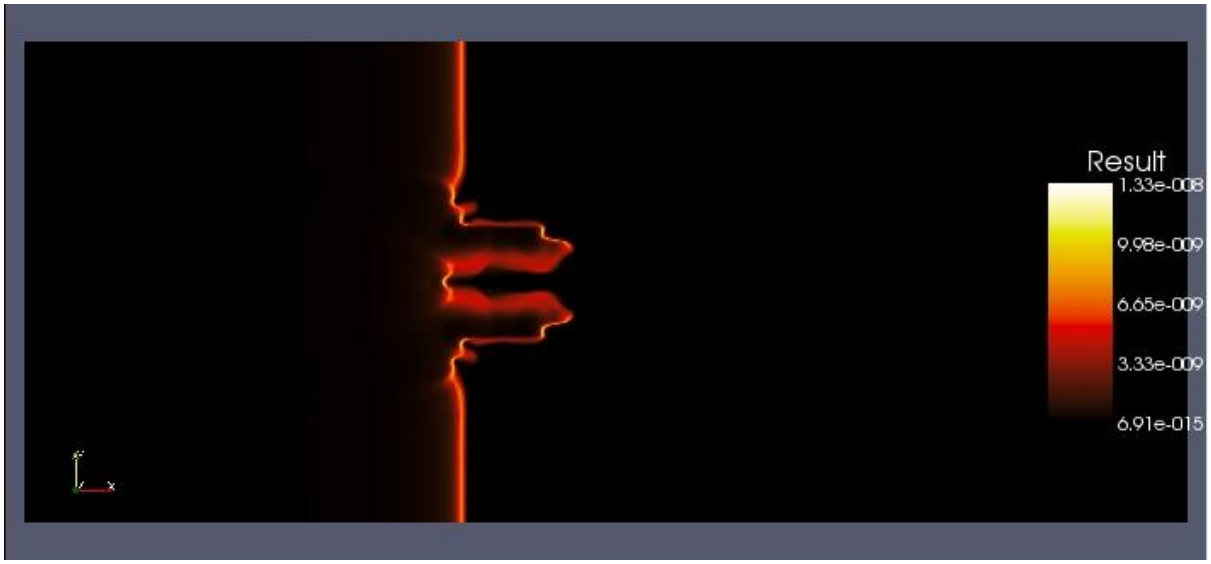
The image below shows the results of applying an isosurface at value 3500 to the VTK curl magnitude file calculated using the curl-magnitude-calculation program on the velocity field for time step 0030.



The image below shows the 0th (density) field from a Z slice taken at slice 124 from time slice 0030 using the derived data example above. It is shown using the blackbody radiation spectrum:



The image below shows the 9th (H_2+) field from a Z slice taken at slice 124 from time slice 0030 using the derived data example above:



VII. REFERENCES

[1] H. Akiba and K.-L. Ma. A tri-space visualization interface for analysing time-varying multivariate volume data. In Proceedings of Eurographics - IEEE VGTC Symposium on Visualization, pages 115–122, 2007.

[2] H. Akiba, C. Wang, and K.-L. Ma. AniViz: A template-based animation tool for volume visualization. IEEE Computer Graphics and Applications, 30(5):61–71, 2010.

[3] D. Angus, A. Smith, and J. Wiles. Conceptual recurrence plots: Revealing patterns in human discourse. IEEE Transactions on Visualization and Computer Graphics, 18(6):988–997, 2012.

[4] A. Biswas, S. Dutta, H.-W. Shen, and J. Woodring. An information-aware framework for exploring multivariate data sets. IEEE Transactions on Visualization and Computer Graphics, 19(12):2683–2692, 2013.

[5] S. Bruckner and T. Moller. Isosurface similarity maps. ¨ Computer Graphics Forum, 29(3):773–782, 2010.

[6] G. H. Bryan and J. M. Fritsch. A benchmark simulation for moist nonhydrostatic numerical models. Monthly Weather Review, 130(12):2917–2928, 2002.

[7] I. Demir, J. Kehrer, and R. Westermann. Screen-space silhouettes for visualizing ensembles of 3D isosurfaces. In Proceedings of IEEE Pacific Visualization Symposium, pages 204–208, 2016.

[8] B. J. Frey and D. Dueck. Clustering by passing messages between data points. Science, 315:972–976, 2007.

[9] S. Frey, F. Sadlo, and T. Ertl. Visualization of temporal similarity in field data. IEEE Transactions on Visualization and Computer Graphics, 18(12):2023–2032, 2012.

[10] M. Glatter, J. Huang, S. Ahern, J. Daniel, and A. Lu. Visualizing temporal patterns in large multivariate data using textual pattern matching. IEEE Transactions on Visualization and Computer Graphics, 14(6):1467–1474, 2008.

[11] M. Gleicher, D. Albers, R. Walker, I. Jusufi, C. D. Hansen, and J. C. Roberts. Visual comparison for information visualization. Information Visualization, 10(4):289–309, 2011.

[12] Y. Gu and C. Wang. Trans Graph: Hierarchical exploration of transition relationships in time-varying volumetric data. IEEE Transactions on Visualization and Computer Graphics, 17(12):2015–2024, 2011.

[13] Y. Gu and C. Wang. iTree: Exploring time-varying data using index able tree. In Proceedings of IEEE Pacific Visualization Symposium, pages 137–144, 2013.

[14] H. Guo, H. Xiao, and X. Yuan. Scalable multivariate volume visualization and analysis based on dimension projection and parallel coordinates. IEEE Transactions on Visualization and Computer Graphics, 18(9):1397–1410, 2012.

[15] M. Haidacher, S. Bruckner, and E. Groller. Volume analysis using multi-¨ modal surface similarity. IEEE Transactions on Visualization and Computer Graphics, 17(12):1969–1978, 2011.

VIII. APPENDIX

ISOSURFACE SIMILARITY MAPS

To fulfil the design requirement, the project create three types of similarity maps to capture different types of relationships among the isosurfaces.

SELF-SIMILARITY MAP.

Bruckner and Moller introduced the isosurface similarity map (ISM) to evaluate the similarities among isosurfaces generated from a volumetric data set. For a given n sampled isovalues, the ISM is a symmetric n×n matrix M where each cell M [i, j] records the similarity value between isosurfaces Si and Si. For every surface Si, the project computes a distance field Di which records the distance from each voxel in the volume to the closest point on Si. The similarity between Si and Sj is then computed as the mutual information between distance fields Di and Dj, using the joint histogram of their distance distributions. The project refers to such an ISM as the self-similarity map (SSM) as it is only concerned with isosurfaces generated from a single volume. For a time-varying multivariate data set of t time steps and v variables, the project need to compute t × v SSMs. Since each SSM may take hours to compute given a reasonable number of sampled isosurfaces (e.g., the project use n = 256), the project applies the GPU-accelerated approximation solution to speed up the computation while maintaining the fidelity of the resulting SSM. Specifically, the project considers the following:

- First, instead of generating the actual isosurface, the project record voxels that contain the isosurface and use that information as an approximation for distance field computation. The error of using this approximation to compute

the distance from a voxel to a surface is bounded by √ 3/2 (i.e., half the diagonal length of the grid cell). This error is acceptable as down sampled distance fields are often suggested to compute the similarity map.

- Second, for distance field computation, the project need to search for each voxel in the volume, the closest point on the isosurface (in our case, the closest voxel containing the isosurface). To reduce the search space, the project uses the down sampled distance field. The project leverage the bounding volume hierarchy (BVH)-trees to speed up the search process. Specifically, the project use bounding boxes for a lower construction time and apply Karras's algorithm to build BVH-trees in parallel. This is essential for performance gain since a BVH-tree needs to be built for every isosurface approximation. Since the BVH-trees store approximation points from the original volume, this guarantees that small features can still be preserved for accurate distance field computation.
- Third, the most time-consuming component in the SSM computation is the construction of the joint histogram of two distance fields since it has to be performed for each pair of isovalues. The project therefore computes the down sampled distance field and use it for subsequent computation. It has been shown that the resolution can be considerably reduced (e.g., 8× in each dimension) without substantial changes in the resulting similarity map [5]. For each SSM, the project identifies m representative isovalues using a greedy strategy that recursively partitions the set of isovalues and selects representative ones based on a priority queue scheme. Typically, m is much smaller than n, e.g., the project use m = 16. The project leverage a GPU to compute isosurface approximations, down sampled distance fields, and joint histograms. With that, the project is able to reduce the average time to compute a single SSM from hours to a few minutes using a single GPU.

ISOSURFACE COMPUTATION.

After the representative isovalues are identified for each volume, the project implements a GPU version of the marching cubes algorithm to compute the actual isosurfaces and simplify the resulting surfaces. Both steps are performed on the CPU during a one-time pre-processing stage. The simplification can significantly reduce the space for storing isosurfaces without sacrificing much of the surface quality. This is important as the project need to generate a total of met ×v representative isosurfaces for the entire time-varying multivariate data set. Using simplified isosurfaces alleviates I/O burden, making it possible for us to achieve interactive visualization and comparison of a number of isosurfaces while still maintaining good visual quality. With our CUDA-accelerated solution, the time cost to compute the SSM is improved by a factor of 43× (72.30s vs. 3,161.70s).

The difference indicates that surface simplification yields a close rendering result with a much less number of triangles (3,519,952 vs. 235,030, or nearly 15× reduction).

TEMPORAL AND VARIABLE SIMILARITY MAPS.

Our goal is to investigate not only the SSMs corresponding to individual volumes, but also the similarity maps between the volumes of different time steps and the volumes of different variables. Therefore, we also compute two other kinds of isosurface similarity: temporal similarity and variable similarity, extending the work of multimodal surface similarity from only a single pair to all pairs of variables, and from steady to time varying data. These similarities are computed between representative isosurfaces from the same variable at different time steps (temporal similarity), and from different variables at the same time step (variable similarity). We call the resulting similarity maps temporal similarity maps (TSMs) and variable similarity maps (VSMs), respectively. A TSM or VSM is not symmetric anymore as the representative isovalues come from different volumes. Each TSM or VSM is much smaller in size compared with an SSM (m×m vs. n×n), but the numbers of TSMs (i.e., v×t (t −1)/2) and VSMs (i.e., t ×v(v−1)/2) are much larger than that of SSMs (i.e., t ×v) as we need to compute TSMs and VSMs for different pairs of time steps and variables.