A. Maries, T. Luciani, P.H. Pisciuneri, M.B. Nik, S.L. Yilmaz, P. Givi, G.E. Marai

Chapter X: A Clustering Method for Identifying Regions of Interest in Turbulent Combustion Tensor Fields

in: Visualization and Processing of Higher Order Descriptors for Multi-Valued Data.
Editors: Ingrid Hotz and Thomas Schultz

October 13, 2014

Springer

-

Chapter 1
A Clustering Method for Identifying Regions of Interest in Turbulent Combustion Tensor Fields

1.1 Introduction

U.S. energy consumption is dominated by the burning of fossil fuels such as coal, natural gas and petroleum [1]. Production of electricity and propulsion systems are two primary reasons for this energy use. Both processes, at their core, involve turbulent combustion. Turbulent combustion modeling is an important area of research driven by the effort to improve the efficiency of these processes, reduce fuel consumption and reduce pollution.

In order of decreasing fidelity and costs, three computational approaches to turbulence combustion are direct numerical simulation (DNS), large eddy simulation (LES) and Reynolds-averaged Navier-Stokes (RANS). Specific tensor quantities such as stress, strain, and turbulent stress play in these computational approaches, as part of the computational modeling process; these quantities are discussed in detail in our previous work [9]. DNS requires directly capturing the wide range of length- and time-scales. This severely limits the approach to simulations of relatively simple, canonical configurations. The complex nature of turbulent reacting flows can be attributed to the non-linear convection terms and scalar transport terms appearing in the coupled set of governing equations. In DNS these terms are accounted for without modeling. However, as discussed in more detail in our previous work [9], in both LES and RANS these terms create a closure problem and require modeling. As such, an important aspect of turbulence modeling and model validation involves comparison of the subgrid scale stress tensor (for LES) or the Reynolds stress tensor (for RANS) with available DNS data.

A three-dimensional LES features millions of grid points. A DNS of the same configuration would feature several orders of magnitude more grid points. The result is that a given snapshot of the flow variables at a particular instance in time for the entire domain would range on the order of gigabytes in the LES case to terabytes for the DNS case. An entire simulation is composed of tens of thousands to hundreds of thousands of time steps. Thus data is typically only retained at specified intervals. Even in adopting this approach total datasets become cumbersome to work with.

Moreover, as LES models are validated, they are in turn used for much larger flow geometries of industrial applications, resulting in snapshots that are tens or hundreds of gigabytes in size.

Producing data at this scale implies a few prerequisite conditions: that a highly scalable flow solver will be used for the simulation, and that the researcher has access to a large supercomputing environment. This introduces new complexities to the workflow. First, file I/O must be handled in parallel and is typically a costly operation relative to the time required to calculate a given time step of the simulation. Thus regularly outputting entire snapshots of simulation data will adversely impact the progress of the calculation. Second, this data is produced at a remote location. Ideally the researcher would transfer the data to a local machine for data analysis and interactive visualization. For large datasets transferring files can take a significant amount of time, adding a measurable bottleneck to the workflow.

To explore the physical phenomena from a volumetric dataset, the ability of a visualization tool to compute and track salient features is crucial. The large amount of data may severely affect the availability of the data for visualization (*i.e.*, the simulation may not be paused to output the data for visualization), the data transfer (bandwidth limitations), and the manipulation speed. These are major challenges to the interactive visualization of tensor data. Furthermore, tensor datasets tend to be very dense, leading to clutter and occlusion problems when visualizing such datasets with existing tools.

In situ visualization aims to address some of these issues. Such approaches enable the user to connect directly to a running simulation, examine the data, do numerical queries and create graphical output while the simulation executes, bypassing the need to write data to disk. The visualization and computation can be tightly coupled (memory sharing), loosely-coupled (communication over a network), or hybrid (the data is computationally reduced and then sent out for visualization).

The feature extraction approach is an emerging *in situ* hybrid method. This approach extracts the meaningful and interesting regions from the datasets, showing only those parts to the researchers. Typically, only a small percentage of datasets are of interest, and the feature can be described very compactly. These abstractions lead to a sharp reduction in the amount of data processed, making an effective visualization of very large datasets possible. Another advantage of feature extraction is that it helps the users highlight and focus on regions of particular characteristics that they are interested in.

As opposed to filtering approaches, which may require expert knowledge about the structure of the flow, in this work we present an unsupervised statistical approach for the segmentation, visualization and potentially the tracking of regions of interest in large tensor data. The approach employs a machine learning clustering algorithm to locate and identify areas of interest based on specified parameters such as strain tensor value.

1.2 Related Work

A great deal of research has been conducted in the problems of feature extraction and tracking. While initially developed in the field of computer vision [11, 19], Feature extraction and tracking have also been adopted for flow visualization—see Post et al. [15] for an extensive review. In this section we focus on the existing related feature-tracking work in flow visualization.

Most feature extraction and tracking techniques fall into one of three basic categories. The most widespread method is to extract features in each time step separately and then to track them through time. The first to employ feature extraction and tracking in flow visualization, Samtaney et al. [16] use feature attributes such as mass, centroid, volume, or moment of inertia to establish a correspondence of features across timesteps. Silver and Wang [17, 18] developed a volume tracking schema that requires that there is a certain amount of overlap between features in adjacent timesteps for them to be associated. Caban et al. [2] introduce a texture-based feature tracking technique that compares textural characteristics across timesteps to find the best match. A second approach to feature extraction and tracking is exemplified by the work of Muelder and Ma [12]. Instead of extracting features in each timestep and then establishing a correspondence, they use a prediction-correction method that makes a prediction about the location and size of the region in the next step. The region is then adjusted by growing or shrinking the border in order to extract the feature of interest. Ji et al. [5] developed a third approach to feature extraction and tracking, which uses isosurfacing in higher dimensions. Once again, as opposed to extracting isosurfaces in 3D for every timestep, their method tracks features by performing an isosurfacing process in 4D.

Other approaches use various machine learning techniques to aid in feature tracking. Tzeng and Ma [20] utilize neural networks to learn which transfer functions are most appropriate in tracking the features of interest. Noticing that tracking groups of features that exhibit similar behavior is more cost-effective than tracking the features individually, Ozer et al. [13] use a clustering algorithm to group features based on similarity measures. Our approach is similar to that of Ozer et al. in that we also utilize clustering analysis. The difference is that, rather than use it to group features, we use it to define regions of interest.

The visualization community has long been very concerned with the shock location problem. A number of techniques and algorithms for characterizing the regions of interest, detecting and visualizing shocks waves have been developed. Lovely and Haimes [7] designed an algorithm for extracting the shock surface that uses the fact that the shock surface normal is typically aligned with the pressure gradient vector. Thus, the algorithm computes the Mach number in the direction of the pressure gradient and builds the shock surface from the points where the Mach number equals one. Another widely-used algorithm utilizes the density gradient and consists of three steps [14]. It first computes the first and second derivatives of the density in the direction of the velocity. It then builds an isosurface where the second derivative equals zero and, finally, it picks the first derivative maxima, which correspond to the shock, and discards the minima. Ma et al. [8] make the distinction between shock

waves and expansion waves in the third step of the previous algorithm by using the normal Mach number rather than the first derivative of the density. Specifically, it picks regions where the Mach number is close to one. The method we present herein is novel in that it integrates machine learning with visualization for extracting and clustering regions of interest. It is thus a promising approach to apply to very large flow datasets.

1.3 Methods

Given the size and interaction challenges of combustion datasets, automated methods are particularly relevant to the problem of identifying regions of interest. Such an approach would allow pushing the feature extraction process *in situ*, to the same computational side that also processes the combustion simulation. Only the regions extracted would then be sent out to visualization, thus reducing both I/O and bandwidth usage.

Automated methods for feature identification can be provided through Machine Learning (ML), a branch of statistics and computer science which studies algorithms and architectures that learn from observed facts. Unsupervised ML algorithms are of particular interest for combustion tensor data: in an unsupervised setting, the objective is to cluster or discover structures in the data. Example algorithms and representations for unsupervised learning include K-means clustering, mixture models, hierarchical clustering, and PCA (Principal Component Analysis).

Clustering analysis is used to group data points that are similar to one another. There are various reasons for using clustering: one may wish to analyze points in the dataset that are close to one another, to reduce a high-dimensional dataset by replacing groups of dimensions with single labels, or to reduce the size of the dataset by replacing groups of data points with single labels.

From the class of clustering methods, we focus on K-means, a powerful yet computationally-effective approach. As in most Big Data applications, the ability to trade semantic meaning for performance is important in this context: more sophisticated methods like mixture models or hierarchical clustering are also significantly slower than K-means.

1.3.1 K-means Clustering

K-means clustering attempts to partition a set of N observations (the number of grid points in a simulation) into K clusters; in the resulting partition each observation belongs to the cluster with the nearest mean observation. The mean is referred to as the centroid of the cluster. The cluster centroid can later be used to describe all cluster members, thus attaining data reduction.

In our case, an observation is the computed value of a tensor at a given location. A tensor is an extension of the concept of a scalar and a vector to higher orders. Scalars and vectors are 0-th and 1-st order tensors, respectively. In general, a k-th order tensor can be represented by a k-dimensional array, *e.g.* a second order tensor is a 2D array (a matrix). For example, while a stress *vector* is the force acting on a given unit surface, a stress *tensor* is defined as the components of stress vectors acting on each coordinate surface; thus stress can be described by a symmetric 2-nd order tensor.

The velocity stress and strain tensor fields are manifested in the transport of fluid momentum, which is a vector quantity governed by the following conservation equation:

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j}, \qquad \text{for } i = 1,2,3 \tag{1.1}$$

where the Cartesian index notation is employed in which the index $i = 1,2,3$ represents spatial directions along the x, y, and z Cartesian coordinates, respectively; and the repeated index j implies summation over the coordinates. t is time, ρ is the fluid density, $\mathbf{u} \equiv [u_1, u_2, u_3]$ is the Eulerian fluid velocity, p is the pressure, and τ is the stress tensor defined as:

$$\tau_{ij} = 2\mu \left(S_{ij} - \frac{1}{3}\delta_{ij}\frac{\partial u_k}{\partial x_k} \right) \tag{1.2}$$

where μ is the dynamic viscosity coefficient (a fluid-dependent parameter) and S is the velocity strain tensor defined as:

$$S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{1.3}$$

We perform clustering on the six distinct values of the strain tensor in the combustion data, arranged in a six-dimensional vector $x^{(i)}$, where i ranges over the points in the dataset. Tensors used in turbulence modeling are rank 2 tensors, which for our purposes are 3×3 matrices. Additionally, strain tensors are symmetric, $s_{ij} = s_{ji}$. This means that there are a total of 6 distinct tensor values for each point in the grid. We compute mean values and distances using a simple, squared Euclidean distance metric. We note that in our previous work [9] the alternative approach of working in a dimensionally-reduced space—such as the space of eigenvalues and eigenvectors—revealed that in turbulent combustion modeling these reduced descriptors are small, fairly uniform and non-distinctive throughout the volume, and thus of limited value for cluster analysis. Similar prior experiments [9] have shown that reduced descriptors such as trace and determinant can act as valuable flow filters; proposing and using such descriptors requires, however, expert knowledge about the nature of a particular flow configuration.

The K-means method follows two alternative steps, one initialization step, and one assignment step. In the first step, the cluster means are initialized (for example, with K random observations $x^{(i)}$ from the set). In the second step, each of the N

points is assigned to the cluster whose mean is most similar to the point. The cluster means are repeatedly recomputed based on the points assigned to each cluster, and the N points are reassigned, until convergence:

Tensor K-Means

- Randomly initialize K cluster centroids
  $\mu_1, \mu_2, .. \mu_K \in R^6$

- Repeat {

  //cluster assignment step
  For i = 1 to N
  $c^{(i)} := index\ (from\ 1\ to\ K)\ of\ cluster\ centroid\ closest\ to\ x^{(i)}$

  //centroid move step
  For k = 1 to K
  $\mu_{(k)} := average\ (mean)\ of\ points\ assigned\ to\ cluster\ k$
  }

- $J(c^{(1)}, c^{(2)}, .. c^{(N)}, \mu_1, \mu_2, .. \mu_k) = 1/N \sum ||x^{(i)} - \mu_{c(i)}||^2$

Convergence is assessed using the clustering error function J, given by the mean distance of all points to their assigned cluster centroid. The problem is NP-hard and thus computationally challenging; but the iterative approach described above can converge to local optima.

Unfortunately, ML clustering algorithms do not scale well. In our experiments, we found that datasets larger than $450,000$ points cannot be clustered using K-means, and datasets larger than $250,000$ points cannot be clustered using greedy agglomerative clustering in less than 24 hours (Intel duo CPU at 2.26GHz and 4GB RAM). To reduce such run-times, it is necessary to preprocess the dataset.

A common approach for preprocessing large datasets is to use canopy clustering [10] as a pre-clustering algorithm. This pre-clustering is followed by a clustering algorithm such as K-means, hierarchical clustering or expectation maximization. The preprocessing step produces initial estimates for the dataset clusters, which are then used to speedup the clustering step. However, after a series of clustering experiments, we concluded that canopy clustering was unable to perform clustering in a reasonable amount of time. The problem was that in these tensor datasets the number of clusters of interest is typically below 10. Given this restriction and the fact that the majority of the data points in a cluster have to be in the same canopy, the size of the canopies would have had to be no smaller than 800,000 (8M/10), even for the smaller datasets. Clustering such large collections of data points is, however, unfeasible using K-means: in our further experiments we found that even

450K datasets require more than 24 hours runtime to converge (Quad core Intel 5, 3.3GHz, 16GB RAM).

To circumvent this obstacle, we used instead a pre-clustering step in which K-means clustering was run on a sub-sampled dataset to obtain good starting cluster centers. The first dataset was sampled every $4 \times 4 \times 4$ data points and the second every $4 \times 6 \times 4$ data points regularly throughout the grid. The sampling rate was empirically selected (lower rate along larger dimensions) so that the pre-clustering step could complete in minutes. The resulting starter centroids were then used in K-means over the full datasets. Using this pre-computed cluster centroid setup, the second clustering step converges in under 50 iterations for the mixing layer dataset (8M points) and 20 iterations for the shock dataset (21M points); both datasets are described in detail in the results section. The entire approach takes on average 15-20 minutes to compute four clusters (8M point dataset).

1.3.2 Cluster Analysis

As is the standard procedure in K-means, we repeat the clustering procedure for a varying number K of clusters, from 2 to 6, and select the K value that leads to the lowest clustering error J. In our experiments, the clustering run times for different K values are fairly similar, with deviation of at most one hour.

To more easily analyze the features of cluster centroids, we use a star-plot representation of each centroid. The star-plot is a high-dimensional visualization technique based on the parallel coordinate plot (PCP). In the PCP descriptor, dimensions are represented by parallel axes and data points are mapped to the axes; the data points are then connected by lines [4, 21]. The star-plot is a more compact representation of the PCP, in which axes are radii of a circle [3, 6].

Figure 1.1 shows star-plot descriptors for the centroids for $K = 5$, $K = 6$, $K = 7$, and $K = 8$; in each star-plot glyph the six centroid values are mapped to radial spokes, allowing for easy comparison of the centroid traits. Note how increasing K from 5 to 6 adds a distinctive cluster, while further increasing K to 7 and 8 introduces clusters which are fairly similar to clusters already identified. This type of analysis can be used to automate the selection of K.

To ensure consistent clustering along the time dimension for the shock dataset, the tensor centroids of the clusters can be tracked over time based on their similarity, with visual assistance where necessary to account for splitting, recombining, vanishing, and appearance phenomena.

1.4 Results

We evaluate this approach on two large datasets, a mixing layer dataset, and a shock dataset. Mixing layer configurations are common in combustion simulations, where

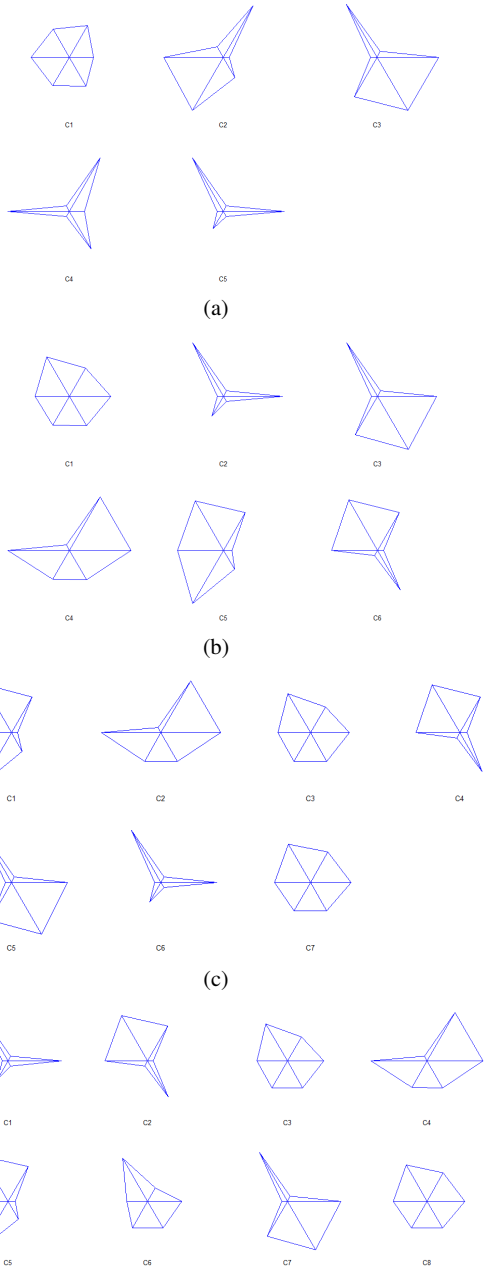


Fig. 1.1 Star-plot descriptors for the centroids for $K = 5$, $K = 6$, $K = 7$, and $K = 8$. Increasing K from 5 to 6 adds a distinctive cluster, while further increasing K to 7 and 8 introduces clusters which are fairly similar to clusters already identified.

two fluids flow over and against each other. Shock waves are important features in compressible flow datasets that are characterized by abrupt, nearly discontinuous changes in physical flow quantities such as density, pressure and velocity. Shock waves are of interest to researchers since they can increase drag and cause structure-failure in design problems in fluid dynamics.

1.4.1 Mixing-Layer Dataset

The first dataset is a temporal mixing layer and is a simple configuration where two streams of fuel and oxidizer flow over and against each other. The flow speeds are adjusted for a low Reynolds number yielding a narrow range of length scales, and this configuration can be easily tackled with DNS and then used as a benchmark. The data for the temporal mixing layer is at a snapshot in time and at the full DNS resolution over a grid of size 193 grid points in two Cartesian directions and 194 in the other (approx. 8M grid points).

The goal for the mixing layer dataset was to see if clustering can provide insight into the structure of the flow. We provided a senior combustion researcher with a 3D volume rendering of the divergence of the tensor (sum of components on the main diagonal, indicating fluid density changes), and a volume rendering of the 4-group clustering (Fig. 1.2). We then asked the expert for an evaluation of the clustering results. The researcher remarked that the clusters coincided with the interesting regions of the flow: the "mushroom" pattern around the shear layer at the mid-zone where the two fluids mix, in contrast with the less active outer zones. The domain expert is currently investigating an interpretation of the clusters.

1.4.2 Shocklet Dataset

The second dataset has a similar mixing-layer configuration, with flow from one direction in the top half and in the opposite direction in the bottom half. There are a few differences as well; one being that the dataset is significantly larger. The size of the grid in two of the three dimensions is 194 and along the third dimension it is 577, which brings the total number of grid points close to 21M. This simulation has been done up to time $t = 600$ in $12,900$ time steps. By this time, the flow is going through pairing and exhibits 3D effects. This is a supersonic flow, in which the flow field exhibits shocklets. Thus, the flow field variables such as Mach number, divergence of velocity and gradients of density, temperature and pressure change sharply across the shocklet surface. Figure 1.3 shows the regions of the flow field with Mach number close to one. The study of shock waves is critical in understanding of high-speed flows. An efficient and reliable shock wave detection and visualization method would significantly assist this task.

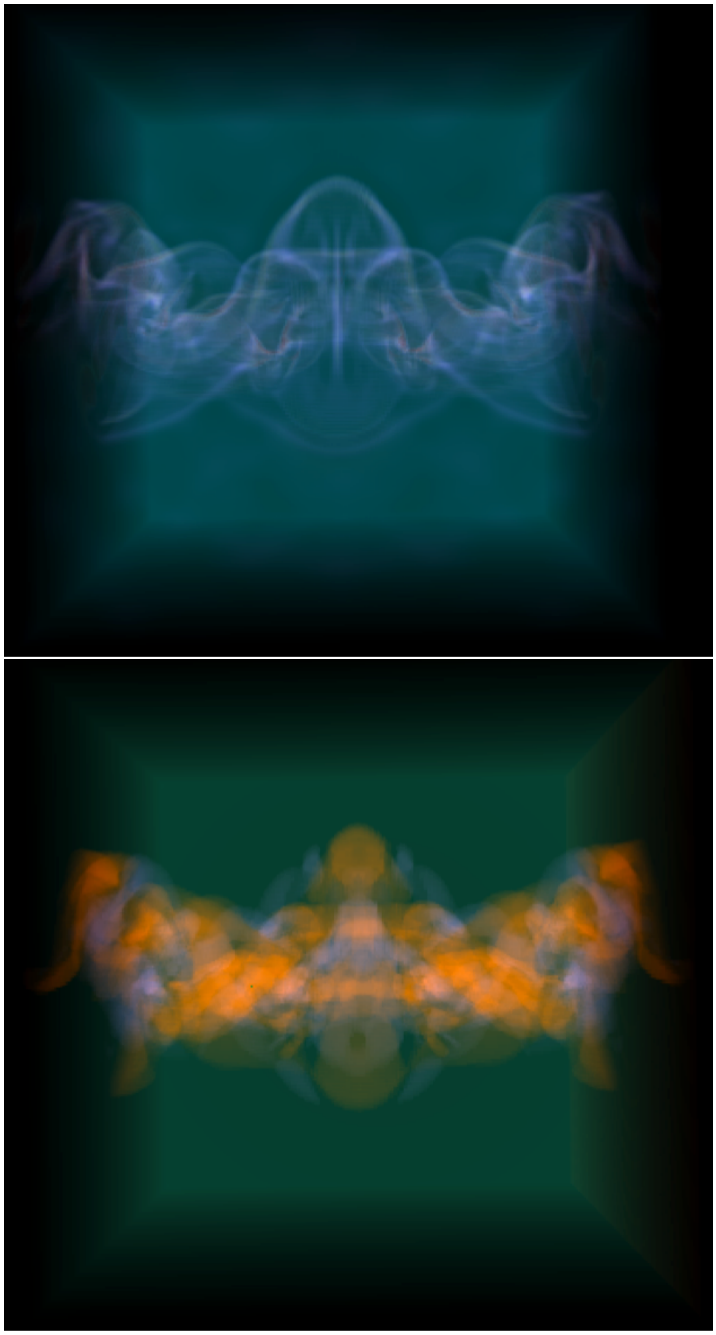


Fig. 1.2 Mixing-Layer Dataset: volume rendering of divergence (top), which can be calculated via the trace of the strain tensor, and 4-group strain tensor clustering (bottom), rendered by assigning each cluster an individual value and setting the transfer function monochromatically to each. The clusters correlate well with the mixing region of interest:the "mushroom" pattern around the shear layer at the mid-zone where the two fluids mix, in contrast with the less active outer zones.

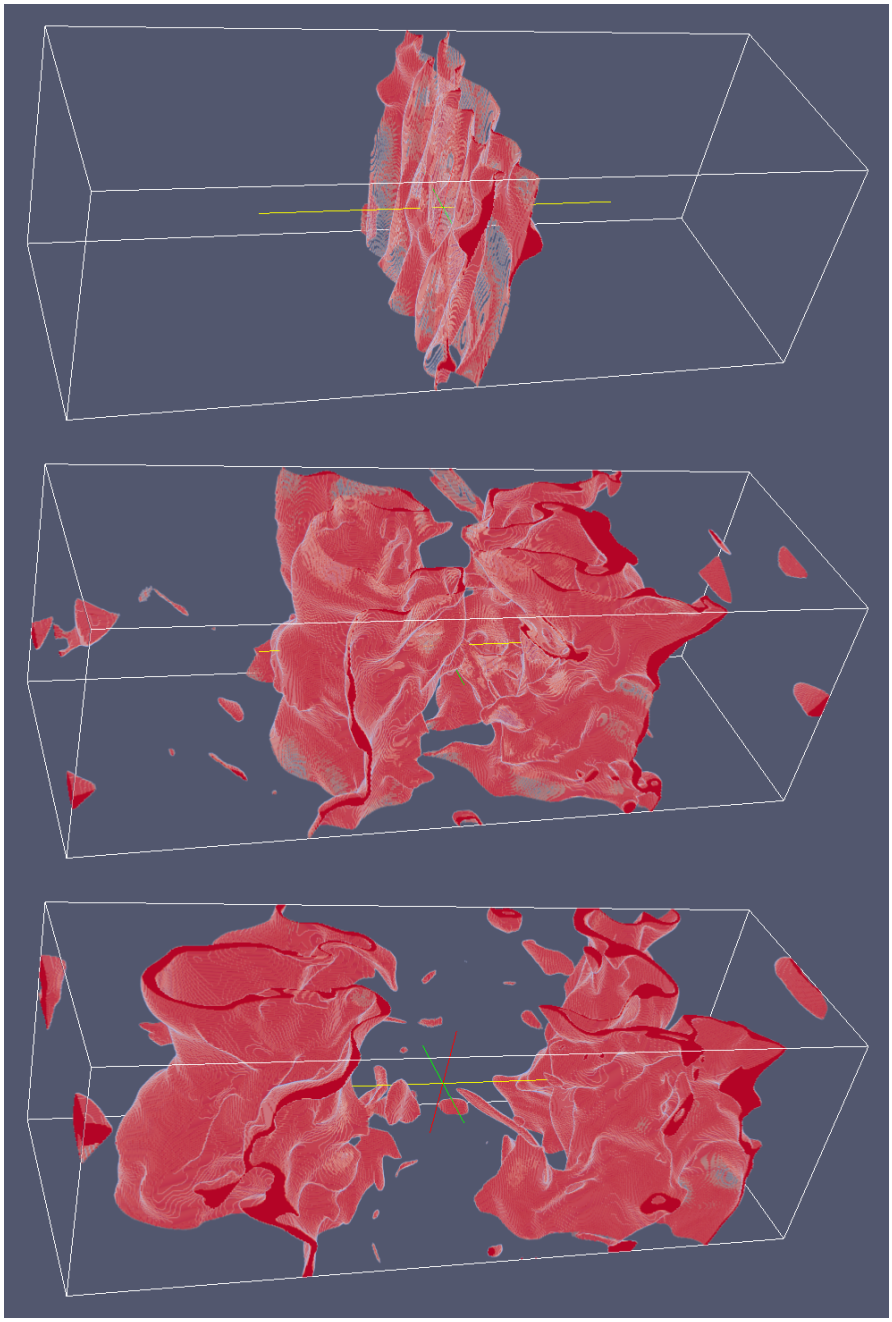


Fig. 1.3 Volume rendering of the region with $Ma \approx 1$. In this rendering, all points with Mach number within the 0.05 threshold of one have been replaced with 1 and the rest with 0; the resulting two-valued volume is volume-rendered in ParaView. From top to bottom; timesteps 70, 325, and 600.

The goal for the second dataset was to see if the distinct tensor field regions have a clear relationship with regions of the flow suitable for the location of a shock surface. Suitable conditions involve the transistion from Mach number greater than 1 to Mach number less than 1, *e.g.* the isosurface depicted in Fig. 1.3. Figure 1.5 shows the star-plot glyphs corresponding to the cluster centroids for the four-cluster segmentation, at timesteps 70, 75 and 80. Note that the different signatures of the cluster centroids make possible the consistent labeling and thus tracking of clusters over time.

Figure 1.4 compares strain tensor clustering with the magnitude of the Mach number. The cluster analysis enabled the experts to identify the regions of the flow potentially suitable for the location of a shock surface. More detailed analysis is still required to detect shocks, such as the adherence to tabulated thermodynamic properties across a shock wave. However, using the clustering results, the domain experts were able to significantly limit this further analysis to only the regions of interest identified through the clustering.

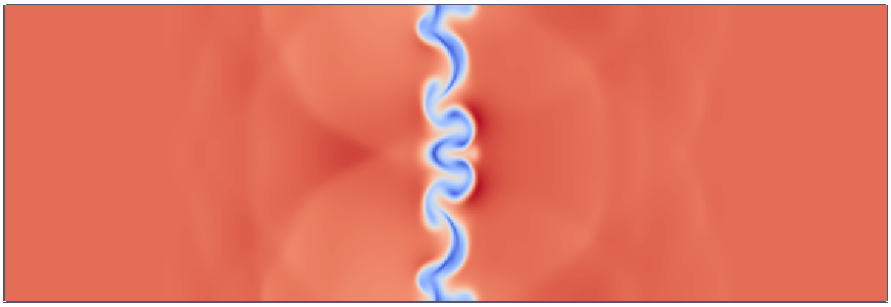
1.5 Discussion and Conclusion

As previously discussed, the goal of this project was to examine the potential of using cluster analysis on tensor field data generated by turbulent combustion simulations. First, we were interested in finding out whether an unsupervised approach can detect structures in the data, and whether these structures correlate with the regions of interest. Second, we wanted to see whether tensor field clustering and rendering could give researchers insights into the structure of the flow through a volume. The answer to both questions is affirmative.

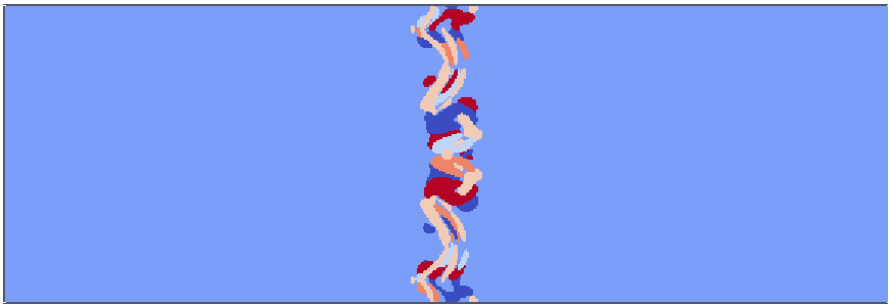
In summary, we found that a machine learning inspired approach, though computationally intensive, can extract and track regions of interest in large, dense tensor fields. Our K-means approach yielded interesting results: the clusters correlate well with the regions of interest. Thus, as an *in situ* technique, the approach has potential for compression. While the clustering itself may miss potential artifacts, the clustering approach is also a foundation for automated anomaly detection, and thus a base for further *in situ* benefits.

We found that the performance of machine learning algorithms is a major issue, and note that such algorithms need to be first adapted for large scale data. In our approach we adapted K-means for large data by using a pre-clustering step; this pre-clustering step was performed via sub-sampling. In an *in situ* setting, the pre-clustering could be coupled with the more intelligent data partitioning that is mandatory when distributing the computational simulation load across multiple processors. The run-time cost of the unsupervised machine learning approach should also decrease when distributed across multiple processors.

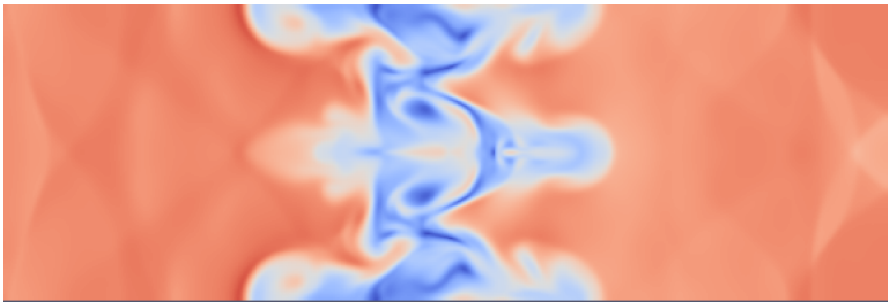
In our implementation and preliminary experiments, the clustering technique was run offline, not in an authentic *in situ* setting. We note however, that the approach was designed as an *in situ* technique, and could be deployed as such in a compu-



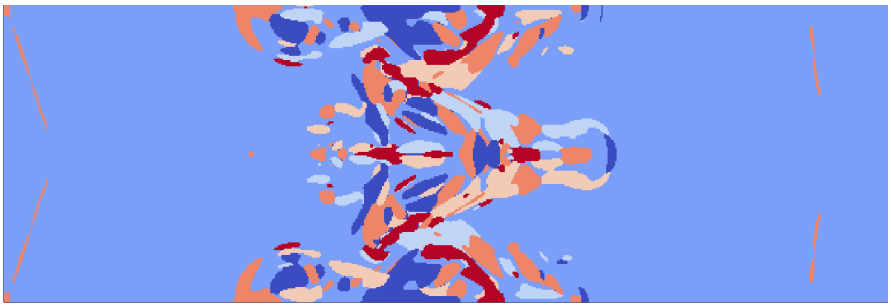
(a) Timestep 70: Mach number



(b) Timestep 70: 6-group strain tensor clustering

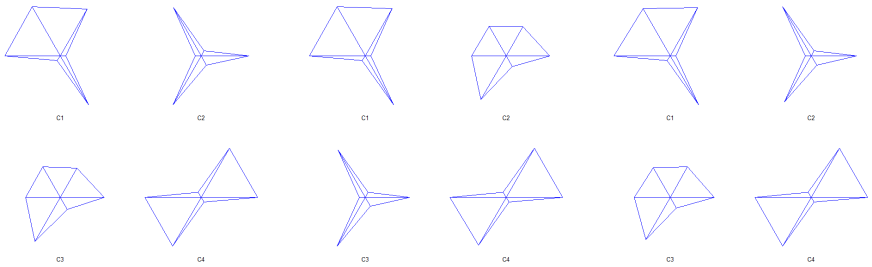


(c) Timestep 325: Mach number



(d) Timestep 325: 6-group strain tensor clustering

Fig. 1.4 Two snapshots from the shocklet dataset, same YZ slice: renderings of Mach number (a and c), and the corresponding 6-group strain tensor K-means results (b and d). The six clusters are encoded with three shades of blue plus three shades of orange-red. The clustering captures regions of interest for Mach number in both cases. The cluster analysis enables domain experts to identify the regions of the flow potentially suitable for the location of a shock surface. Cluster centroid similarity can be further used to ensure consistent cluster labeling across multiple timesteps.



(a) timestep 70 (b) timestep 75 (c) timestep 80

Fig. 1.5 Star-plot glyphs corresponding to the cluster centroids for a four-cluster segmentation of the shock datasets. From left to right: cluster centroids at timesteps 70, 75, and 80. Note that the different signatures of the cluster centroids make possible the consistent labeling and thus tracking of clusters over time.

tational setting, with the added benefit of data partitioning for pre-clustering. The only current "manual" component of the approach is the selection of K, the numbers of clusters. The K-selection step can be automated, however, as indicated in the Fig. 1.1 shape analysis.

In our approach, we have used a simple feature vector representation for the tensor field. Similarly, we have used a simple Euclidean distance metric to compute distances and means over the tensor field. We obtained good correlation between the cluster-based regions of interest and flow features. Nevertheless, defining more meaningful feature vectors and distance metrics for tensor similarity that have improved semantic meaning are important directions of future research.

Visualization of the tensor fields associated with turbulent combustion simulations is particularly challenging. Difficulties arise from the sheer scale and density of the data, but also from the small range of values these tensors take. Our previous attempts at visualizing these types of fields using glyphs or streamlets have had partial success–these tensors have very small, very similar eigenvalues [9]. Furthermore, many existing tensor representations do not have an intuitive equivalent in combustion turbulent flow; to combustion researchers, tensors do not have direction or shape. As the domain experts put it, "the tensor itself is very useful for computation, and pretty complete... but its individual components are not so useful to understand what is going on." This observation makes feature extraction and tracking through an unsupervised clustering approach particularly useful.

In conclusion, we have introduced an approach for the segmentation, visualization and potential tracking of regions of interest in large scale tensor field datasets generated by computational turbulent combustion simulations. The approach is novel in that it integrates machine learning with visualization–interactive volume rendering and starplots–to extract, cluster, and track regions of interest in the tensor field. Our evaluation on two rich combustion datasets shows this approach can assist in the visual analysis of the combustion tensor field.

Acknowledgments

This work was supported by NSF CBET-1250171 and NSF CAREER IIS-0952720.

References

1. J. D. J. Anderson. Modern compressible flow : with historical perspective, 1982.
2. J. J. Caban, A. Joshi, and P. Rheingans. Texture-based feature tracking for effective time-varying data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1472–1479, 2007.
3. N. Elmqvist, J. Stasko, and P. Tsigas. Datameadow: A visual canvas for analysis of large-scale multivariate data. In *VAST IEEE Symposium on Visual Analytics Science and Technology 2007, Proceedings*, pages 187–194, 2007.
4. A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry, 1990.
5. G. Ji, H.-W. Shen, and R. Wenger. Volume tracking using higher dimensional isosurfacing. In *Proceedings of the 14th IEEE Visualization 2003*, pages 209–216, Oct 2003.
6. A. Klippel, F. Hardisty, R. Li, and C. Weaver. Colour-enhanced star plot glyphs: Can salient shape characteristics be overcome? *Cartographica: The International Journal for Geographic Information and Geovisualization*, 44(3):217–231, 2009.
7. D. Lovely and R. Haimesy. Shock detection from computational fluid dynamics results. *Proceedings of the 14th AIAA Computational Fluid Dynamics Conference*, 1:M2, 1999.
8. K.-L. Ma, J. V. Rosendale, and W. Vermeer. 3d shock wave visualization on unstructured grids. *Volume Visualization and Graphics, IEEE Symposium on*, 0:87–104, 1996.
9. A. Maries, M. Haque, S. Yilmaz, M. Nik, and G. Marai. Interactive exploration of stress tensors used in computational turbulent combustion. In D. Laidlaw and A. Villanova, editors, *New Developments in the Visualization and Processing of Tensor Fields*, pages 137–156. Springer-Verlag, 2012.
10. A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '00, pages 169–178, New York, NY, USA, 2000. ACM.
11. F. Meyer and P. Bouthemy. Region-based tracking using affine motion models in long image sequences. *CVGIP: Image Understanding*, 60(2):119–140, 1994.
12. C. Muelder and K.-L. Ma. Interactive feature extraction and tracking by utilizing region coherency. In *Visualization Symposium, 2009. PacificVis '09. IEEE Pacific*, pages 17–24, April 2009.
13. S. Ozer, J. Wei, D. Silver, K.-L. Ma, and P. Martin. Group dynamics in scientific visualization. In *Large Data Analysis and Visualization (LDAV), 2012 IEEE Symposium on*, pages 97–104, Oct 2012.
14. H.-G. Pagendarm and B. Seitz. An algorithm for detection and visualization of discontinuities in scientific data fields applied to flow data with shock waves. *Scientific Visualization: Advanced Software Techniques*, pages 161–177, 1993.
15. F. H. Post, B. Vrolijk, H. Hauser, R. S. Larameeand, and H. Doleisch. The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum*, 22(4):775–792, 2003.
16. R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *Computer*, 27(7):20–27, 1994.
17. D. Silver and X. Wang. Volume tracking. In *Proceedings of Seventh Annual IEEE Visualization '96*, pages 157–164, 1996.
18. D. Silver and X. Wang. Tracking and visualizing turbulent 3d features. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):129–141, 1997.
19. S. M. Smith and J. M. Brady. Asset-2: real-time motion segmentation and shape tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):814–820, 1995.
20. F.-Y. Tzeng and K.-L. Ma. Intelligent feature extraction and tracking for visualizing large-scale 4d flow simulations. In *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference*, pages 6–6, Nov 2005.
21. E. J. Wegman. Hyperdimensional data analysis using parallel coordinates. *Journal of the American Statistical Association*, 85(411):664–675, 1990.