2D Asymmetric Tensor Field Topology

Zhongzang Lin, Harry Yeh, Robert S. Laramee, and Eugene Zhang

Abstract

In this chapter we define the topology of 2D asymmetric tensor fields in terms of two graphs corresponding to the eigenvalue and eigenvector analysis for the tensor fields, respectively. Asymmetric tensor field topology can not only yield a concise representation of the field, but also provide a framework for spatial-temporal tracking of field features. Furthermore, inherent topological constraints in asymmetric tensor fields can be identified unambiguously through these graphs. We also describe efficient algorithms to compute the topology of a given 2D asymmetric tensor field. We demonstrate the utility of our graph representations for asymmetric tensor field topology with fluid simulation data sets.

1 Introduction

We define the topology of 2D asymmetric tensor fields, whose analysis can benefit a wide range of applications in solid and fluid mechanics [23, 24]. Topology-based analysis has achieved much success in the processing and visualization of scalar fields [1, 6], vector fields [2, 8, 14, 13], and symmetric second-order tensor fields [3,

Zhongzang Lin
Oregon State University, Corvallis, Oregon, e-mail: `lin@eecs.oregonstate.edu`

Harry Yeh
Oregon State University, Corvallis, Oregon, e-mail: `harry@engr.orst.edu`

Robert S. Laramee
Swansea University, Wales, UK e-mail: `r.s.laramee@swansea.ac.uk`

Eugene Zhang
Oregon State University, Corvallis, Oregon, e-mail: `zhange@eecs.oregonstate.edu`

16, 21]. Not only can topology provide a concise representation of the fields of interest in these applications, it also enables a framework for systematic multi-scale analysis and temporal feature tracking.

There has been relatively little work in asymmetric (second-order) tensor fields [24, 23]. Dodd [5] develops a method to represent the geometry of a symmetric tensor field in terms of its geodesics. The method can be used for a global realization of the tensor field, which allows the user to identify the presence of inconsistencies and singularities in the data. To the best of our knowledge, asymmetric tensor field topology has not been defined, even in the 2D case. Asymmetric tensor fields appear to have richer structures, in terms of both the number and the variety, than vector fields and symmetric tensor fields. We define the topology of a 2D asymmetric tensor field in terms of two topological graphs: *eigenvector graph* and *eigenvalue graph*, based on the eigen-analysis of the tensor field. The nodes of these graphs correspond to regions of tensor field features. The graphs not only describe adjacency relationships between the regions, but also dictate the topological constraints that need to be satisfied by any tensor field simplification procedures. We also provide algorithms to construct the graphs given an asymmetric tensor field.

The rest of the chapter is organized as follows: we review related work in Section 2, and provide the definition for eigenvalue and eigenvector graphs as well as an algorithm to extract them in Section 3. We demonstrate the usefulness of these graphs by applying them to some simulation data in Section 4. In Section 5, we conclude and discuss some possible future directions.

2 Related Work

Vector Field Topology Much work has been devoted to the extraction and visualization of vector field topology [11]. Helman and Hesselink define the topology for vector fields representing fluid flow and propose an analysis and visualization framework [8]. Scheuermann et al. [13] suggest a novel approach to detect higher-order singularities. Several algorithms have been proposed for extracting periodic orbits [19, 20, 2], an important constituent of vector field topology. Tricoche et al. [18] suggest a singularity tracking method for time-dependent vector fields.

To deal with noise in the data, various vector field simplification techniques have been proposed. First-order singularities are either merged into higher-order ones [15] or removed through systematic pair cancellation operations [14, 22]. The simplified topology is easier to perceive and understand while the most prominent structures of the original vector field are maintained.

Symmetric Tensor Field Topology The topology of symmetric tensor fields is also well studied. Delmarcelle and Hesselink [3] introduce hyperstreamlines for symmetric tensor fields. In addition, they visualize asymmetric tensor fields by using hyperstreamlines for symmetric tensor components while encoding non-symmetric components in an additional vector field.

Delmarcelle and Hesselink [4] and Hesselink et al. [9] define the topology for 2D and 3D symmetric tensor fields. Zheng et al. [25] point out that the degenerate features in 3D tensor fields form curves and propose a fast computation method for integrating degenerate lines in 3D symmetric tensor fields. Furthermore, a number of vector field simplification techniques have been adapted to tensor field simplification [16, 17, 21].

Asymmetric Tensor Field Analysis In contrast to symmetric tensor fields, there is relatively little work focusing on asymmetric tensor fields. However, tensor fields that appear in many engineering phenomena and problems are asymmetric in nature, such as the velocity gradient tensors of fluid flow and the deformation gradient tensors in solid medium. Zheng and Pang [24] define degeneracies of asymmetric tensor fields based on singular value decomposition of the tensor field. They also introduce the concept of dual-eigenvectors which allow directional information contained in the tensor field to be visualized even when real-valued eigenvectors do not exist. Zhang et al. [23] introduce the notions of eigenvalue manifold and eigenvector manifold, which are supported by tensor re-parameterizations with physical meaning and lead to effective visualization techniques. To the best of our knowledge, the topology of asymmetric tensor fields has not been defined nor systematically studied.

3 Asymmetric Tensor Field Topology

In this section, we define asymmetric tensor field topology in terms of two graphs based on eigen-analysis, i.e. the eigenvector graph and the eigenvalue graph.

3.1 Eigenvector Graph

A 2D asymmetric tensor has either two real eigenvalues (real domain) or a pair of complex conjugate eigenvalues (complex domain). Unlike a symmetric tensor, the eigenvectors corresponding to the two real eigenvalues are not orthogonal even in the real domain, and a separate treatment defining the dual-eigenvectors is needed in the complex domain. Zhang et al. [23] define the eigenvector manifold as a sphere illustrated in Figure 1. The equator of the sphere corresponds to symmetric tensors (pure anisotropic stretching), while the poles represent anti-symmetric tensors (pure rotations). The north and south 45° latitudes are boundaries between the real and complex domains, with the equator inside the real domain and the poles inside the complex domain. These arcs divide the sphere into four regions: $W_{r,n}$, $W_{r,s}$, $W_{c,n}$, and $W_{c,s}$. The subscripts r and c denote the real and complex domains, and n and s represent the northern and southern hemispheres, respectively. A real or complex region in the northern hemisphere has a counterclockwise rotational flow, while a region in the southern hemisphere has a clockwise rotational flow. A tensor field $T(\mathbf{p})$ introduces a continuous map τ_T from the domain of T to the eigenvector manifold, whose inverse $\beta_T = \tau_T^{-1}$ leads to a partition of the domain of T into a collection of four types of regions ($W_{c,n}$, $W_{r,n}$, $W_{r,s}$, and $W_{c,s}$). The boundaries of these regions correspond to the equator and north and south 45° latitudes. The pre-image of the poles are referred as the *degenerate points*.

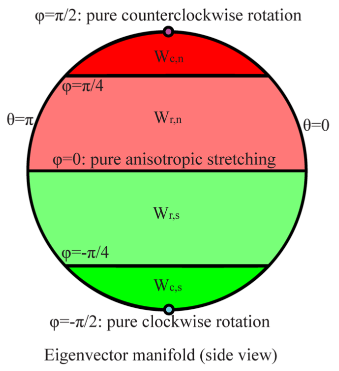
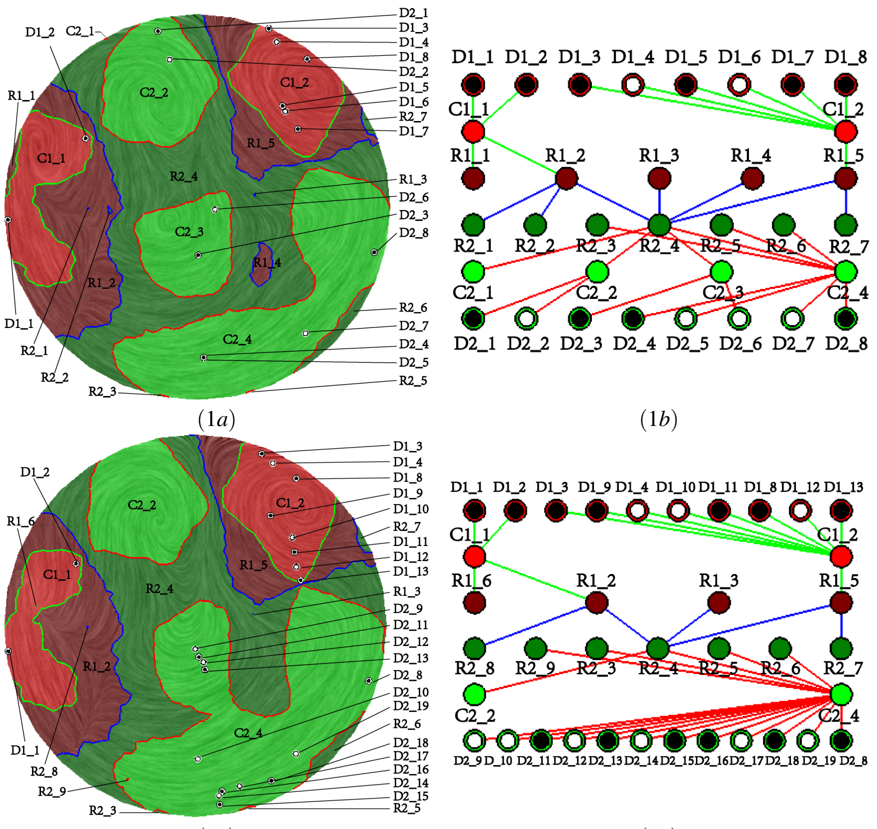

φ=π/2: pure counterclockwise rotation
$W_{c,n}$
φ=π/4
θ=π
$W_{r,n}$
θ=0
φ=0: pure anisotropic stretching
$W_{r,s}$
φ=-π/4
$W_{c,s}$
φ=-π/2: pure clockwise rotation
Eigenvector manifold (side view)

Fig. 1 Eigenvector manifold: the orientation of the rotational component is counterclockwise in the northern hemisphere and clockwise in the southern hemisphere. Each hemisphere is partitioned into real domains and complex domains. The equator represents pure symmetric tensors (irrotational flows), while the poles represent pure rotations [23].

We define the eigenvector graph of T such that each node in the graph corresponds to a (connected) region in the partition. In addition, every degenerate point is treated as a node. The edges in the eigenvector graph represent the adjacency relationship among the regions (including degenerate points). Due to the continuity of τ_T and β_T, the following adjacency relationships are *not possible*: (1) $W_{r,n}$ and $W_{c,s}$, (2) $W_{r,s}$ and $W_{c,n}$, and (3) $W_{c,n}$ and $W_{c,s}$. For velocity gradient tensors, (3) dictates that it is impossible to transition from a vortex core of counterclockwise rotation ($W_{c,n}$) into a vortex core of clockwise rotation ($W_{c,s}$), or vice versa, without going through the real domain, i.e. stretching-dominated region. Such constraints, when applied to a sequence of slices of a 3D data, have the potential of helping domain experts understand the evolution of flow features over time or in space as well as during continuous multi-scale analysis.

Figure 2 shows two slices from a diesel-engine simulation data set [10]. The four types of regions $W_{c,n}$, $W_{r,n}$, $W_{r,s}$, and $W_{c,s}$ are expressed using light red, dark red, dark green, and light green, respectively. The textures in the background depict the vector field.



(1a)
(1b)
(2a)
(2b)
D1_1 D1_2 D1_3 D1_4 D1_5 D1_6 D1_7 D1_8
C1_1 C1_2
R1_1 R1_2 R1_3 R1_4 R1_5
R2_1 R2_2 R2_3 R2_4 R2_5 R2_6 R2_7
C2_1 C2_2 C2_3 C2_4
D2_1 D2_2 D2_3 D2_4 D2_5 D2_6 D2_7 D2_8
D1_1 D1_2 D1_3 D1_9 D1_4 D1_10 D1_11 D1_8 D1_12 D1_13
C1_1 C1_2
R1_6 R1_2 R1_3 R1_5
R2_8 R2_9 R2_3 R2_4 R2_5 R2_6 R2_7
C2_2 C2_4
D2_9 D_10 D2_11 D2_12 D2_13 D2_14 D2_15 D2_16 D2_17 D2_18 D2_19 D2_8

Fig. 2 This figure illustrates the eigenvector graphs applied to two slices of a 3D engine simulation data set [12]. The slices are cut at the 63mm $(1a)$ and 73mm $(2a)$ from the top of the cylinder in the diesel engine simulation [10]. Note that the icons showing nearby degenerate points can overlap during rendering, such as $D2\_4$ and $D2\_5$ in $(1a)$. In addition, some regions are too small to be visible without zooming. For example, $R1\_3$ in $(2a)$ is a small $W_{r,n}$ region (dark red) surrounded by $R2\_4$, a large $W_{r,s}$ region (dark green).

3.2 Eigenvalue Graph

Zhang et al. [23] re-parameterize the set of 2×2 asymmetric tensors as follows:

$$\gamma_d \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \gamma_r \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} + \gamma_s \begin{pmatrix} \cos 2\theta & \sin 2\theta \\ \sin 2\theta & -\cos 2\theta \end{pmatrix} \tag{1}$$

where $\gamma_d = \frac{a+d}{2}$, $\gamma_r = \frac{c-b}{2}$, and $\gamma_s = \frac{\sqrt{(a-d)^2+(b+c)^2}}{2}$ are the strengths of isotropic scaling, rotations, and anisotropic stretching, respectively, and θ provides the directions of the stretching. Based on this parameterization, any non-zero asymmetric tensor can be mapped onto the *eigenvalue manifold* $\{v = (\gamma_d, \gamma_r, \gamma_s) \,|\, v \cdot v = 1$ and $\gamma_s \geq 0\}$ as shown in Figure 3. There are five special points in the eigenvalue manifold, corresponding to $v = (1,0,0)$ (positive isotropic scaling D^+), $(-1,0,0)$ (negative isotropic scaling D^-), $(0,1,0)$ (counterclockwise rotation R^+), $(0,-1,0)$ (clockwise rotation R^-), and $(0,0,1)$ (anisotropic stretching S). A tensor T is said to be dominated by one of the above five characteristics, if the image of T onto the eigenvalue manifold has a smaller spherical distance to the special point than any of the other special points. The inverse of this projection leads to a partition of the domain of the asymmetric tensor field, as each region in the partition is dominated by the same characteristic. The region dominated by a characteristic is expressed using colors: D^+ in yellow, D^- in blue, R^+ in red, R^- in green, and S in white (Figures 4).

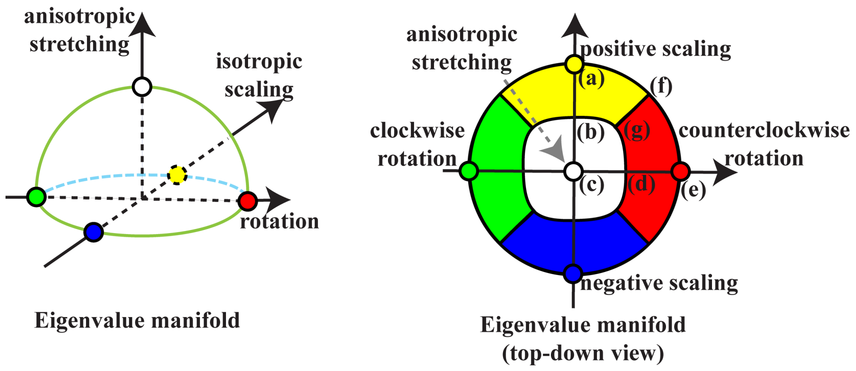


Fig. 3 Eigenvalue Manifold: there are five special points on the manifold, which are *positive* and *negative scaling*, *counterclockwise* and *clockwise rotation*, and *anisotropic stretching*. The Voronoi decomposition with respect to these five special points partitions the manifold into five cells where the flow is dominated by different characteristics [23].

We define the eigenvalue graph as follows: the nodes in the graph correspond to a region in the partition, while each edge encodes the adjacency relationship between a region pair. Notice that two regions may share more than one boundary segment. In this case, each boundary segment will be encoded into an edge in the eigenvalue graph. Figure 4 provides an example illustration using the same data shown in Figure 2. In Figure 4(1a), there are two segments between regions $R1\_1$ and S_5. Consequently, there are two edges between them in the graph. Notice that not every configuration pair is possible, such as a R^+ and R^- pair. In this case, any curve connecting the two regions must pass through some other type of region. This is a topological constraint that is specific to asymmetric tensor fields. Moreover, the eigenvalue graph can be considered as a two-dimensional cell complex [7], where each two-dimensional cell corresponds to a *junction point* shared by three regions (a tensor there satisfies $|\gamma_d| = |\gamma_r| = \gamma_s$). The cell complex must satisfy that its alternating sum equals the Euler characteristic of the underlying domain. This is yet another aspect of asymmetric tensor field topology.

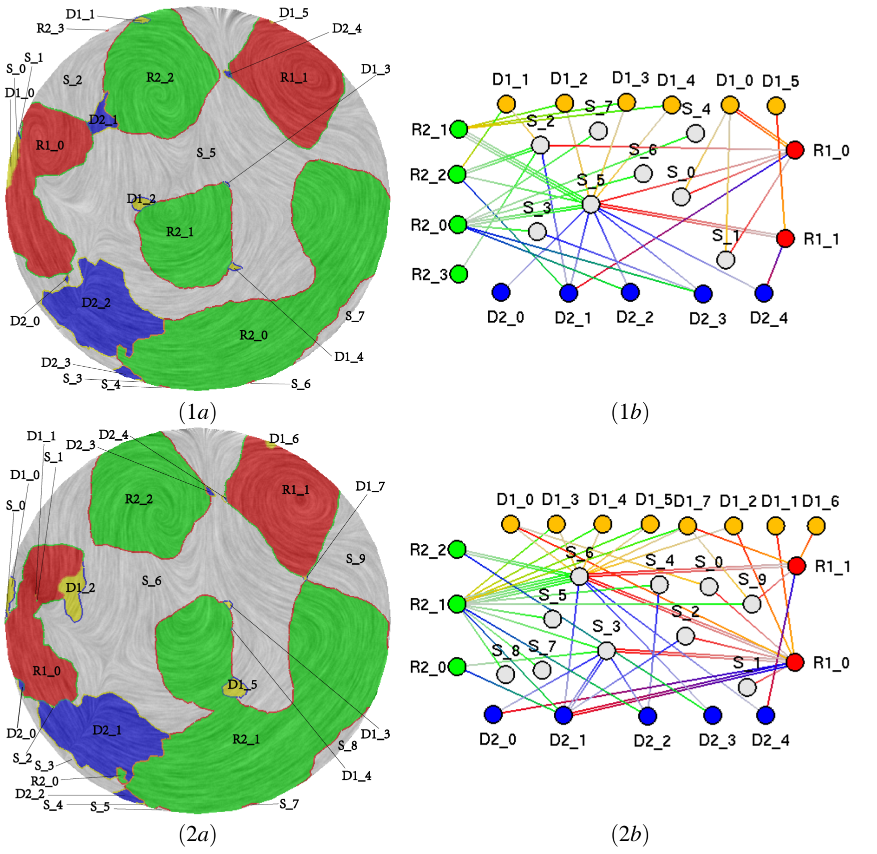


Fig. 4 This figure illustrates the eigenvalue graphs applied to slices of a 3D engine simulation data corresponding to those shown in Figure 2. Each segment of the boundary between a pair of regions is depicted as an edge in the graph between the corresponding two nodes. For instance, there are two edges connecting node $R1\_1$ and S_5 in (1b), which indicate that the boundary between region $R1\_1$ and S_5 in (1a) is divided into two segments.

3.3 Graph Construction Algorithms

We now describe our algorithm to compute the eigenvector and eigenvalue graphs. In our setting the underlying domain is represented by a triangular mesh. The tensor values are defined on the vertices of the mesh and propagated into edges and triangles through the interpolation scheme described in [14].

The most challenging aspect of the algorithms is to identify the nodes in the eigenvalue and eigenvector graphs. Notice that with the exception of degenerate points, a node in an eigenvector graph or eigenvalue graph is a connected component of the domain that has certain characteristics, e.g., $W_{r,n}$, $W_{r,s}$, $W_{c,n}$, $W_{c,s}$ for the eigenvector graph, and D^{+}, D^{-}, R^{+}, R^{-}, and S for the eigenvalue graph. We refer to part of the domain corresponding to such a node as a *region*. Notice that a region can intersect multiple triangles or be contained completely inside one triangle. We refer to the intersection of a region with a triangle to be a *subregion*. A subregion is represented by the set of its boundaries. We represent a region as a list of all subregions.

To construct the eigenvector graph, we first compute the intersection of each triangle with a particular type of region, i.e., $W_{r,n}$, $W_{r,s}$, $W_{c,n}$, or $W_{c,s}$. This leads to a partition of each triangle into subregions of different types. Next, we iteratively merge adjacency subregions in adjacent triangles that belong to the same type. This leads to the regions, i.e., the nodes in the eigenvector graph. As part of the second step, we also establish adjacency relationship between regions of different types, thus constructing the edges in the graph. Finally, we extract the degenerate points which are also nodes in the graph. We then generate edges that connect these nodes with their container nodes, which must be a complex region.

The most complicated part of the computation is the first step, i.e., computing the intersection of a triangle with a particular type of region. This requires the extraction of the region boundaries, which satisfy the conditions $\gamma_r = 0$ (equator), $\gamma_s = \gamma_r$ (north 45° latitude), and $\gamma_s = -\gamma_r$ (south 45° latitude). Given the interpolation scheme, $\gamma_r = 0$ corresponds to linear segments while $\gamma_s = |\gamma_r|$ correspond to quadratic curves. While all these types of boundaries can intersect the edges of a triangle, it is possible that an ellipse can be contained entirely inside a triangle. This last case corresponds to a $W_{c,n}$ or $W_{c,s}$ region strictly inside the triangle. Our algorithm is able to detect these cases as well. Figure 5 provides an illustration of this. Given a triangle (Figure 5(a)), we perform the following computation:

1. For each edge in the triangle, decide whether and where it intersects the boundary of a particular type of a region (Figure 5(b)).
2. Trace boundary curves within the triangle from the aforementioned intersection points on the edges of the triangle (Figure 5(c)).
3. Determine if any internal elliptical region exists, and locate its position (Figure 5(d)).
4. Create subregions and classify their types (Figure 5(e)).

After all subregions have been computed, graph nodes are created for the subregions and a merging process is conducted to consolidate adjacent subregions with the same type into regions (Figure 5 (f)).

Constructing the eigenvalue graph is similar (Figure 6). The only difference is that the boundaries of subregions can also intersect at junction points that are in the interior of a triangle. Consequently, we need to first locate all junction points (Figure 6(b)) before tracing the boundaries (Figure 6(c)).

Locate Intersection Points A triangle edge can be parameterized by parameter t where $0 \le t \le 1$. Denote the tensors at the two vertices of the edge as $T_i = \begin{pmatrix} a_i & b_i \\ c_i & d_i \end{pmatrix}$ where $i = 1, 2$, then the tensor can be linearly interpolated for points on the edge as

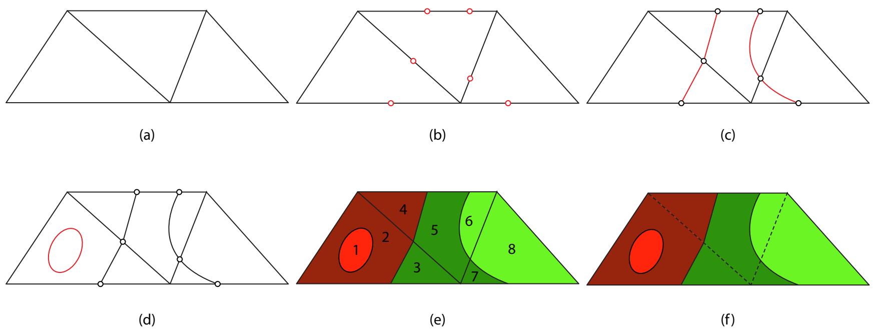


Fig. 5 Eigenvector graph construction: (a) given a tensor field defined on a triangular mesh, (b) each edge of the triangles is visited to locate possible intersections with region boundaries; (c) starting from the intersection points, the boundaries are traced within each triangle; (d) possible internal elliptical boundary is detected; (e) after subregions are created, their types are determined; (f) finally, a merging process is performed to consolidate adjacent subregions with same type into a single region.

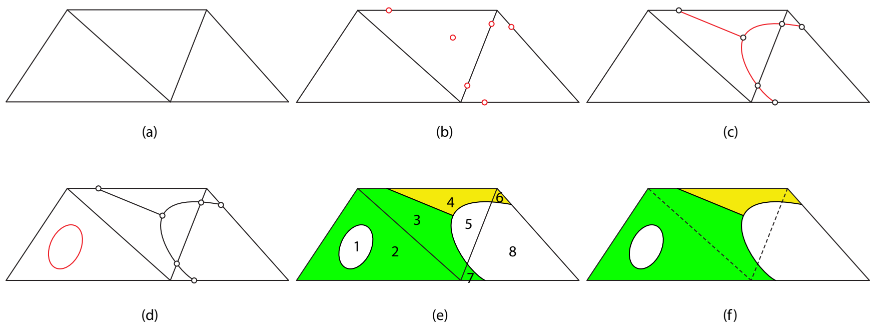


Fig. 6 Eigenvalue graph construction: (a) given a tensor field defined on a triangular mesh, (b) triangle edges are visited to locate possible intersections with region boundaries, possible junction points within the triangle are also detected; (c) the boundaries are then traced starting from the intersection points; (d) possible internal elliptical boundary is detected; (e) subregions are then created and their types are classified; (f) finally, adjacent subregions with same types are merged into a single region.

$$T_t = \begin{pmatrix} a_t & b_t \\ c_t & d_t \end{pmatrix} = \begin{pmatrix} (1-t)a_1 + ta_2 & (1-t)b_1 + tb_2 \\ (1-t)c_1 + tc_2 & (1-t)d_1 + td_2 \end{pmatrix}. \tag{2}$$

The points on the boundary between $W_{r,n}$ and $W_{r,s}$ regions satisfy $\gamma_r = 0$, i.e., $a_t + d_t = 0$, which is a linear equation. For points on the boundary between a real and complex domain pair satisfy $\gamma_s = |\gamma_r|$, which can be found by solving the following quadratic equation

$$g(t) = \gamma_s|_t^2 - \gamma_r|_t^2 = \frac{(a_t - d_t)^2}{4} + \frac{(b_t + c_t)^2}{4} - \frac{(c_t - b_t)^2}{4} = 0 \tag{3}$$

Note that not all solutions are real intersection points. We will only accept those appearing on the intended edge segment.

Locate Junction Points In the case of eigenvalue graph, there can be junction points located within the triangles, satisfying $|\gamma_d| = |\gamma_r| = \gamma_s$. We can find such points by identifying intersection points between two types of curves: the boundary between real and complex domains ($\gamma_s = |\gamma_r|$) and the boundary between scaling-dominant and rotation-dominant regions ($\gamma_d = \pm\gamma_r$). Recall that the former leads to a quadratic equation while the latter a linear one. There are a maximum of four solutions in every triangle, each of which corresponds to a junction point.

Trace Boundary Curves Starting from the intersection points and the junction points, we trace the boundaries in each triangle. As mentioned above, the boundaries are either piecewise linear or piecewise quadratic.

The components of the tensor, i.e., a, b, c, and d, inside a triangle can be linearly interpolated using local coordinates (x,y). For each boundary, we first determine a scalar function such that the boundary is one of the function's iso-lines. For instance, points on the boundary that separate real and complex regions satisfy $\gamma_s = |\gamma_r|$, i.e. $(a-d)^2 + (b+c)^2 - (c-b)^2 = 0$. We define the following scalar function

$$g(x,y) = (a(x,y) - d(x,y))^2 + (b(x,y) + c(x,y))^2 - (c(x,y) - b(x,y))^2. \quad (4)$$

The tracing direction is given by the gradient of this function from the intersection or junction point.

To determine the direction at a given point during tracing, we rotate the gradient vector of $g(x,y)$ by $\frac{\pi}{2}$. Directions for other types of boundaries are obtained similarly. Each boundary is guaranteed to end at an intersection point or a junction point (Figure 6 (c)).

Internal Elliptical Boundaries As previously mentioned, a region may be completely contained in a triangle. When this happens, the region boundary must be elliptical. Such an ellipse has no intersections with triangle edges or another region boundary except in degenerate cases, and cannot be identified by tracing from intersection points or junction points.

To detect such an ellipse, we first compute the coefficients in Equation 4 for the quadratic boundary and determine whether it is an ellipse or a hyperbola. If it is an ellipse, there can be four scenarios:

1. The ellipse intersects with the triangle;
2. The ellipse is entirely inside the triangle;
3. The ellipse is entirely outside of the triangle and has no intersections with the triangle;
4. The ellipse encloses the triangle.

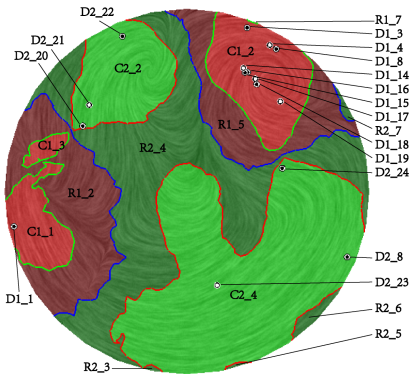
If the ellipse does not intersect the triangle, we evaluate the quadratic function from Equation 4 at the center of the ellipse and the vertices of the triangle. Based on the sign of these values as well as whether the center of the ellipse is inside the triangle, we can determine whether an internal ellipse exists, and if so create a subregion (Figure 5 (d)).

4 Results

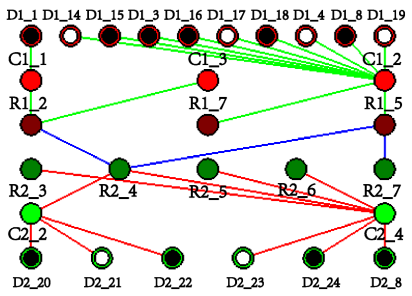
We apply our graphs to a diesel-engine simulation data set. The asymmetric tensor field we analyze is the velocity gradient tensor field. Figures 2 and 4 show the eigenvector and eigenvalue graphs, respectively, for two slices ($10mm$ apart). In both figures the textures in the background illustrate the input vector field.

Figure 7 uses another slice (83mm from the top of the engine cylinder) to compare the eigenvector graph and the eigenvalue graph with the entity connection graph (ECG) [2], a more general form of vector field topology than vector field skeleton. ECG highlights the connectivity among the fixed points via separatrices, which represents quite different topology from those of the tensor topology. We believe that vector field topology and tensor field topology can provide complemental information about the underlying flow.

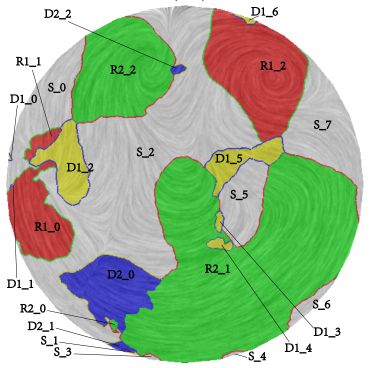
The eigenvector graphs in Figure 2 demonstrate how and where a pair of co-rotating vortices coalesce into a single vortex of the same rotation. This can be seen by examining multiple nodes in the complex domain (light green or light red) of the same hemisphere that are connected to a common node in the real domain (dark green or dark red) of the same hemisphere. A couple of vortex coalescence processes can be detected in the transition from (1) to (2): see $C2\_1$ and $C2\_2$, and $C2\_3$ and $C2\_4$ in Figure 2. On the other hand, a pair of counter-rotating vortices tends to maintain its form. A counter-rotating vortex pair can be identified by the edge connection from a node in the complex domain (light green or light red) to the corresponding node in the complex domain in the opposite side (light red or light green). The connection needs to go through two nodes of the different rotations in the real domain. It must be emphasized that the foregoing vortex behaviors are often discussed for line vortices in fluid mechanics. Our tensor field topology enables the users to analyze complex flows with basic understanding in fluid mechanics. The eigenvalue graphs shown in Figure 4 can provide systematic criteria for flow feature reduction, i.e. part of the multi-scale analysis. For example, an isolated node in the graph that is connected with only one edge may merge into the surrounding flow characteristic, such as S_1 and $D1\_1$ in Figure 4(2b).



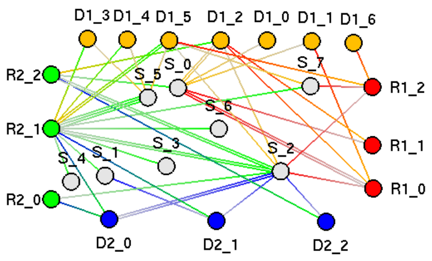
(1*a*)


D1_1 D1_14 D1_15 D1_3 D1_16 D1_17 D1_18 D1_4 D1_8 D1_19
C1_1
C1_3
C1_2
R1_2
R1_7
R1_5
R2_3
R2_4
R2_5
R2_6
R2_7
C2_2
C2_4
D2_20 D2_21 D2_22 D2_23 D2_24 D2_8

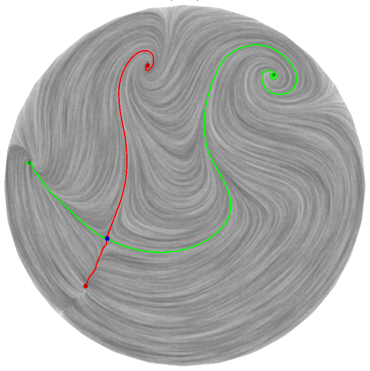
(1*b*)



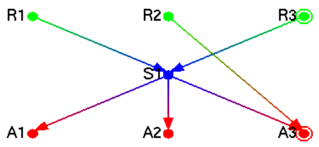
(2*a*)



(2*b*)



(3*a*)



(3*b*)

Fig. 7 This figure compares the eigenvector graph (1), eigenvalue graph (2), and vector field topology (3) of the slice at 83mm from the top of the cylinder in the diesel engine simulation. In (3b), *R* and *A* represent the repellers and attractors, and *S* indicates the saddles. Vector field topology and tensor field topology can provide mutually complemental information of the data.

5 Conclusion and Future Work

We have defined the topology of asymmetric tensor fields in terms of *eigenvector graphs* and *eigenvalue graphs*, and introduced algorithms to construct the graphs given a tensor field. We also point out several topological properties for asymmetric tensor field topology and their physical meanings in terms of fluid mechanics.

In future research, we plan to study how to combine the three types of graphs, i.e., eigenvalue graph, eigenvector graph, and entity connection graph (vector field topology), in order to provide stronger analysis of the vector fields. The quality of a graph is greatly effected by its layout. We plan to explore optimal graph layout algorithms for eigenvalue and eigenvector graphs. Compared to vector fields and symmetric tensor fields, the topological changes for time-dependent asymmetric tensor fields appear to be more complex and require more sophisticated techniques for feature tracking. In addition, it is less intuitive how to perform asymmetric tensor field simplification which is important to multi-scale analysis. We plan to address these challenges in our future research. Finally, it is our goal to extend this work to the topological analysis of 3D asymmetric tensor fields, a largely untouched topic in tensor field visualization.

Acknowledgements We wish to thank Guoning Chen and Qingqing Deng for their help in generating some of the images shown in this chapter. The constructive comments from the reviewers have made this work stronger. Hamish Carr did an excellent job in presenting this work during the TopoInVis 2011 workshop when none of the authors was able to attend it. The research was partially supported by NSF grants IIS-0546881 and CCF-0830808.

References

1. Bremer, P.T., Edelsbrunner, H., Hamann, B., Pascucci, V.: A topological hierarchy for functions on triangulated surfaces. IEEE Transactions on Visualization and Computer Graphics **10**(4), 385–396 (2004)
2. Chen, G., Mischaikow, K., Laramee, R.S., Pilarczyk, P., Zhang, E.: Vector Field Editing and Periodic Orbit Extraction Using Morse Decomposition. IEEE Transactions on Visualization and Computer Graphics **13**(4), 769–785 (2007)
3. Delmarcelle, T., Hesselink, L.: Visualizing Second-order Tensor Fields with Hyperstream Lines. IEEE Computer Graphics and Applications **13**(4), 25–33 (1993)
4. Delmarcelle, T., Hesselink, L.: The Topology of Symmetric, Second-Order Tensor Fields. In: Proceedings IEEE Visualization '94 (1994)
5. Dodd, R.K.: A New Approach to the Visualization of Tensor Fields. Graphical Models and Image Processing **60**(4), 286–303 (1998)
6. Edelsbrunner, H., Harer, J., Zomorodian, A.: Hierarchical morse complexes for piecewise linear 2-manifolds. In: SCG '01: Proceedings of the seventeenth annual symposium on Computational geometry, pp. 70–79. ACM, New York, NY, USA (2001). DOI http://doi.acm.org/10.1145/378583.378626
7. Hatcher, A.: Algebraic Topology. Cambridge University Press (2002)

8. Helman, J.L., Hesselink, L.: Representation and Display of Vector Field Topology in Fluid Flow Data Sets. IEEE Computer **22**(8), 27–36 (1989)
9. Hesselink, L., Levy, Y., Lavin, Y.: The Topology of Symmetric, Second-Order 3D Tensor Fields. IEEE Transactions on Visualization and Computer Graphics **3**(1), 1–11 (1997)
10. Laramee, R.S., Garth, C., Schneider, J., Hauser, H.: Texture-Advection on Stream Surfaces: A Novel Hybrid Visualization Applied to CFD Results. In: Data Visualization, The Joint Eurographics-IEEE VGTC Symposium on Visualization (EuroVis 2006), pp. 155–162 (2006)
11. Laramee, R.S., Hauser, H., Zhao, L., Post, F.H.: Topology-Based Flow Visualization: The State of the Art. In: The Topology-Based Methods in Visualization Workshop (TopoInVis 2005), Visualization and Mathematics, pp. 1–19 (2007)
12. Laramee, R.S., Weiskopf, D., Schneider, J., Hauser, H.: Investigating Swirl and Tumble Flow with a Comparison of Visualization Techniques. In: Proceedings IEEE Visualization 2004, pp. 51–58 (2004)
13. Scheuermann, G., Krüger, H., Menzel, M., Rockwood, A.P.: Visualizing nonlinear vector field topology. IEEE Transactions on Visualization and Computer Graphics **4**(2), 109–116 (1998)
14. Tricoche, X., Scheuermann, G.: Continuous Topology Simplification of Planar Vector Fields. In: Proceedings IEEE Visualization 2001, pp. 159–166 (2001)
15. Tricoche, X., Scheuermann, G., Hagen, H.: A Topology Simplification Method For 2D Vector Fields. In: Proceedings IEEE Visualization 2000 (2000)
16. Tricoche, X., Scheuermann, G., Hagen, H.: Scaling the Topology of Symmetric, Second-Order Planar Tensor Fields. In: Proceedings of NSF/DOE Lake Tahoe Workshop on Hierarchical Approximation and Geometrical Methods for Scientific Visualization (2001)
17. Tricoche, X., Scheuermann, G., Hagen, H.: Topology Simplification of Symmetric, Second-Order 2D Tensor Fields, Hierarchical and Geometrical Methods in Scientific Visualization. Springer (2003)
18. Tricoche, X., Wischgoll, T., Scheuermann, G., Hagen, H.: Topology Tracking for the Visualization of Time-Dependent Two-Dimensional Flows. Computers & Graphics **26**(2), 249–257 (2002)
19. Wischgoll, T., Scheuermann, G.: Detection and Visualization of Closed Streamlines in Planar Fields. IEEE Transactions on Visualization and Computer Graphics **7**(2) (2001)
20. Wischgoll, T., Scheuermann, G.: Locating Closed Streamlines in 3D Vector Fields. In: Proceedings of the Joint Eurographics - IEEE TCVG Symposium on Visualization (VisSym 02), pp. 227–280 (2002)
21. Zhang, E., Hays, J., Turk, G.: Interactive Tensor Field Design and Visualization on Surfaces. IEEE Transactions on Visualization and Computer Graphics **13**(1), 94–107 (2007)
22. Zhang, E., Mischaikow, K., Turk, G.: Vector Field Design on Surfaces. ACM Transactions on Graphics **25**(4), 1294–1326 (2006)
23. Zhang, E., Yeh, H., Lin, Z., Laramee, R.S.: Asymmetric Tensor Analysis for Flow Visualization. IEEE Transactions on Visualization and Computer Graphics **15**(1), 106–122 (2009)
24. Zheng, X., Pang, A.: 2D Asymmetric Tensor Analysis. IEEE Proceedings on Visualization pp. 3–10 (2005)
25. Zheng, X., Parlett, B., Pang, A.: Topological Lines in 3D Tensor Fields and Discriminant Hessian Factorization. IEEE Transactions on Visualization and Computer Graphics **11**(4), 395–407 (2005)