Robustness-Based Simplification of 2D Steady and Unsteady Vector Fields

Primoz Skraba, Bei Wang, *Member, IEEE,* Guoning Chen, *Member, IEEE,* and Paul Rosen, *Member, IEEE*

Abstract—Vector field simplification aims to reduce the complexity of the flow by removing features in order of their relevance and importance, to reveal prominent behavior and obtain a compact representation for interpretation. Most existing simplification techniques based on the topological skeleton successively remove pairs of critical points connected by separatrices, using distance or area-based relevance measures. These methods rely on the stable extraction of the topological skeleton, which can be difficult due to instability in numerical integration, especially when processing highly rotational flows. In this paper, we propose a novel simplification scheme derived from the recently introduced topological notion of robustness which enables the pruning of sets of critical points according to a quantitative measure of their stability, that is, the minimum amount of vector field perturbation required to remove them. This leads to a hierarchical simplification scheme that encodes flow magnitude in its perturbation metric. Our novel simplification algorithm is based on degree theory and has minimal boundary restrictions. Finally, we provide an implementation under the piecewise-linear setting and apply it to both synthetic and real-world datasets. We show local and complete hierarchical simplifications for steady as well as unsteady vector fields.

Index Terms—Flow visualization, Vector field simplification, Robustness, Computational topology

1 Introduction

Vector fields and their analysis are indispensable for many applications in science and engineering. With the increasing gap between the size and complexity of the vector field data from real-world applications and the limited bandwidth of our visual perception channel, it is more and more challenging for domain experts to interpret their data in detail or as a whole. This challenge is prominent in 2D turbulence flows, where features are everywhere and feature sizes differ by a few orders of magnitude. Vector field simplification aims at reducing the complexity of the flow by removing features in order of their relevance and importance, revealing prominent behavior, obtaining a compact representation for interpretation, and giving a consistent and multiscale view of the flow dynamics.

A considerable amount of research has been focused on vector field simplification based on the notion of a topological skeleton [1], [2]. A topological skeleton consists of critical points connected by special streamlines called *separatrices*, which provides a condensed representation of the flow by dividing the domain into regions of uniform flow behavior. However, existing simplification techniques rely on the stable extraction of the topological skeleton, which can be difficult due to instability in numerical integration, especially when processing highly rotational flows, e.g., Fig. 1. Furthermore, the distance and area-based relevance measures that are commonly used to determine the cancellation ordering of critical points typically rely on geometric proximities and do not consider the flow magnitude, an important physical property of the flow.

- *P. Skraba is with Jozef Stefan Institute, Slovenia. E-mail: primoz.skraba@ijs.si*
- *B. Wang and P. Rosen are with Scientific Computing and Imaging Institute, University of Utah. E-mails: {beiwang, prosen}@sci.utah.edu*
- *G. Chen is with University of Houston. E-mail: chengu@cs.uh.edu*

In this paper, we propose a new vector field simplification scheme derived from the recently introduced notion of *robustness*. Robustness, a notion related to *persistence* [3], [4], is used to represent the stability of critical points and assess their significance with respect to perturbations of the vector field. Intuitively, the robustness of a critical point is the minimum amount of perturbation, with respect to a metric encoding flow magnitude, that is required to cancel it within a local neighborhood. Our contributions are:

- We propose a new hierarchical simplification strategy based on robustness, which enables the pruning of sets of critical points according to a quantitative measure of their stability.

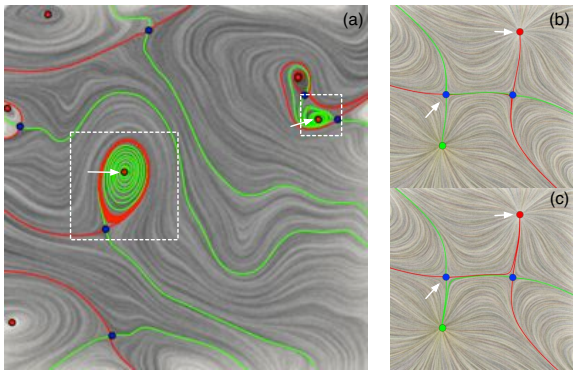

Fig. 1. Topological skeleton: Sinks (and saddle-sink separatrices) are red, sources (and saddle-source separatrices) are green, and saddles are blue. (a) A highly rotational flow field where the pointed critical points are close to Hopf-bifurcations. Numerical inaccuracies may accumulate during integration and separatrices may intersect or switch. (b)-(c) Instability of separatrices under a small perturbation: The upper right sink is not connected with the saddle on the left in (b), but is after a small perturbation in (c).

- An implementation in the piecewise-linear (PL) setting applied to a number of synthetic and real-world datasets included in the experimental results.
- We extend our simplification scheme from the steady vector field to the time-varying (i.e. unsteady) case, illustrating the utility and current limitations of our method due to the consistency issues of the time-varying case.

We *do not* intend to show that the robustness-based method is necessarily better than topological-skeleton-based or distance-based methods across all scenarios. In the typical situation involving pairs of critical points connected by separatrices, such methods offer comparable visual results. Rather, the robustness-based method provides a complementary perspective and can handle more general boundary situations; it is scalable and gives a novel, mathematically rigorous hierarchical simplification scheme. Our method finds, in the space of all vector fields, the one that is closest to the original vector field in the L_∞ norm (the maximum point-wise modification to the vector field) with a particular set of critical points removed. Our results are optimal in this norm, that is, there exists no simplification with a smaller perturbation. This paper includes and extend our earlier work [5] by providing a more complete exposition of the robustness-based simplification, including simplifications that explore the entire hierarchy and extensions to the time-varying setting.

2 Related Work

Vector field simplification can be classified into topology-based and non-topology-based techniques [6]. Non-topology-based techniques typically focus on Laplacian smoothing of the potential of a vector field [7], [8], [9]. Topology-based techniques modify the vector field topology explicitly by merging or canceling nearby critical points based on the notion of a topological skeleton [1], [2], [6], [10]. De Leeuw and Van Liere [11], [12] made use of a geometry-based relevance measure (e.g., with respect to distance or area proximity) to determine the pair of critical points to be cancelled. Tricoche et al. [13] focused on a piecewise analytic description for the simplified field, which was later extended to time-dependent 2D flows [14]. Theisel et al. [15] presented a topology-preserved compression and simplification of vector fields. Zhang et al. [6] introduced a framework for fixed point pair cancellation based on Conley index theory. Chen et al. [10] extended this idea to include periodic orbits and presented a more complete pairwise cancellation framework. Recently, Chen et al. [16], [17] introduced a multiscale hierarchy of the vector field topology based on the Morse Connection Graph (MCG) computed from Morse decomposition [17]. This work was extended to address piecewise constant vector fields by Szymczak el al. [18], [19]. Such representations could be used to simplify vector fields by iteratively merging pairs of Morse sets that are adjacent in the MCG. The order of the pairs for cancellation depends only on the geometric characteristics of the Morse sets, i.e., the pairs that lead to smaller merged Morse sets will be cancelled or merged first. Weinkauf et al. [20] introduced a topological simplification technique for 3D vector fields based on the extraction of higher-order critical points. The simplification is assisted by a derived auxiliary 2D vector field on a closed surface surrounding each higher-order critical point.

Simplification of time-dependent vector fields is typically performed slice-by-slice by considering the *bifurcations* that the critical points may involve [21]. Chen et al. [22] described a framework of simplifying a time-dependent vector field by systematically removing the detected bifurcations. In particular, only isolated bifurcations and bifurcations that are connected by a short life critical point can be removed. In addition, a spatio-temporal Laplacian smoothing algorithm was introduced to reduce the flow complexity within a given space-time sub-domain in a non-topology-based fashion. Simplifications have also been proposed in a combinatorial setting [23], [24]. Edelsbrunner et al. [3], [25] performed pair cancellation on scalar fields defined on surfaces by changing the values of the scalar function near the fixed point pair. This is equivalent to simplifying the gradient vector field of the scalar function. Finally, scale space techniques [4], [26] have also been proposed to assess the importance of a critical point for topology-based simplification.

Robustness is closely related to the notion of persistence [3]. While persistence has been used successfully for scalar field visualization and simplification of topological structures such as contour trees and Jacobi sets [27], [28], [29], [30], robustness, first introduced in [31], is specifically designed for vector-valued data [32], [33]. Recent work [34] assigns robustness to critical points in both stationary and time-varying settings and obtains a structural description of the vector field. Such a structural description implies the existence of a hierarchical simplification strategy based on robustness, which is the focus of this paper.

In general, topology-based simplification techniques pair the topological features for cancellation via the computation of separatrices, which can be numerically unstable [17]. In contrast, the proposed robustness-based method does not require the computation of topology, thus, is insensitive to the numerical errors. The simplification hierarchy obtained from topology-based methods is typically invariant to scaling (multiplying the vector field with a scalar field), whereas our technique is sensitive to the change of vector field magnitude as it directly corresponds to our perturbation metric (Sec. 3). The robustness-based method achieves comparable results to the topology-based simplification for typical scenarios and can handle more challenging cases in which the topology-based methods may fail (Sec. 5).

3 Background

We provide relevant background in degree theory and robustness by reviewing previous work [32], [34] with minimal algebraic definitions and illustrating the related concepts through an example (Fig. 2 adapted from [34]). We also provide introductory descriptions of isolating neighborhoods and Laplacian smoothing [6], [10].

Degrees. For a critical point x in 2D, its degree $\deg(x)$ equals its *(Poincaré) index*, that is, the number of field rotations while traveling along a closed curve centered at x counter-clockwise. Sources, sinks, centers, and saddles have indices $+1$, $+1$, $+1$ and -1, respectively. Furthermore, for a (path-)connected component C that encloses several critical points, its degree $\deg(C)$ is the sum of the respective degrees of those critical points [32]. For our robustness-based simplification strategy, we rely on a corollary of the Poincaré-Hopf theorem (which is also employed by topological-skeleton-based simplification, e.g., [21]), which states that if a connected component C in 2D has degree zero, then it is possible to replace the vector field inside C with a vector field free of critical points.

Merge tree. To analyze a continuous 2D vector field $f : \mathbb{R}^2 \to \mathbb{R}^2$, we define a corresponding scalar function (referred to as the flow

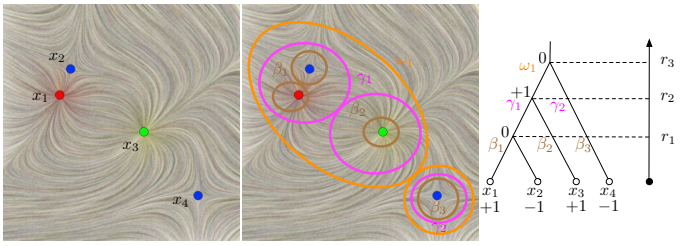

Fig. 2. Figure adapted from [34]. Suppose the vector field is continuous, where sinks are red, sources are green, and saddles are blue. From left to right: vector fields f, relations among components of $\mathbb{F}_r$, and augmented merge trees. f contains four critical points, a sink x_1, a source x_3, and two saddles x_2 and x_4. We use β, γ, ω, etc. to represent components of the sublevel sets.

magnitude function) $f_0 : \mathbb{R}^2 \to \mathbb{R}$ which assigns for each point the magnitude (Euclidean norm) of the corresponding vector, $f_0(x) = ||f(x)||_2$. We use $\mathbb{F}_r = f_0^{-1}(-\infty, r]$ to denote the *sublevel set* of f_0 for some $r \geq 0$. $\mathbb{F}_0$ is precisely the set of critical points of f. We assume that f is generic, which implies that the critical points of f are isolated.

Increasing r from 0, the space $\mathbb{F}_r$ evolves and we can construct a graph that tracks the (connected) components of $\mathbb{F}_r$ as they appear and merge. This is called a *merge tree* (or *join tree* as described in [35]). The root represents the entire domain of f_0 and the leaves represent the creation of a component at a critical point of f (a local minimum). An internal node represents the merging of two or more components. We further record an integer at each node, which is the degree of the corresponding component in the sublevel set, and refer to the result as an augmented merge tree. An initial computation of the degrees of critical points is sufficient to determine the degree of any component of any sublevel set by computing the sum of the degrees of the critical points lying in it [32]. An example is shown in Fig. 2. We ignore any components that appear after $r = 0$ as they have zero degrees and so do not correspond to any critical points of the vector field. The merge tree on the right shows how the components of the sublevel sets $\mathbb{F}_r$ evolve. At $r = 0$ there are four components that correspond to the four critical points, each with nonzero degree. At $r = r_1$, components that contain x_1 and x_2 merge into a single component β_1, which has zero degree. When $r = r_2$, components β_1 and β_2 merge into a single component γ_1 with degree $+1$, while β_3 grows into γ_2. Finally at $r = r_3$, the single component ω_1 has zero degree.

Static robustness and its properties. The *(static) robustness* of a critical point is the height of its lowest degree zero ancestor in the merge tree [34]. The static robustness quantifies the stability of a critical point with respect to perturbations of the vector fields through the following lemmas explicitly stated in [34].

We first define the concept of *perturbation*. Let $f, h : \mathbb{R}^2 \to \mathbb{R}^2$ be two continuous 2D vector fields. Define the distance between the two mappings as $\mathrm{d}(f,h) = \sup_{x \in \mathbb{R}^2} ||f(x) - h(x)||_2$. A continuous mapping h is an *r-perturbation* of f, if $\mathrm{d}(f,h) \leq r$.

Lemma 3.1 (Critical Point Cancellation [34]). Suppose a critical point x of f has robustness r. Let C be the connected component of $\mathbb{F}_{r+\delta}$ containing x, for an arbitrarily small $\delta > 0$. Then, there exists an $(r+\delta)$-perturbation h of f, such that $h^{-1}(0) \cap C = \emptyset$ and $h = f$ except possibly within the interior of C.

Lemma 3.2 (Degree & Critical Point Preservation [34]). Suppose a critical point x of f has robustness r. Let C be the connected component of $\mathbb{F}_{r-\delta}$ containing x, for some $0 < \delta < r$. For any ε-perturbation h of f where $\varepsilon \leq r - \delta$, the sum of the degrees of the critical points in $h^{-1}(0) \cap C$ is $\deg(C)$. If C contains only one critical point x, we have $\deg(h^{-1}(0) \cap C) = \deg(x)$. That is, x is preserved as there is no ε-perturbation that could cancel it.

Revisiting the example in Fig. 2, the robustness of the critical points x_1, x_2, x_3, and x_4 is r_1, r_1, r_3, and r_3, respectively. Since the robustness of x_3 is r_3, for any $\delta > 0$, we consider a component $C \subseteq \mathbb{F}_{r_3+\delta}$ that is slightly larger than ω_1 and contains x_3 (in fact, ω_1 contains all four critical points). Lemma 3.1 implies the existence of an $(r_3 + \delta)$-perturbation that cancels x_3 by locally modifying the component C. Conversely, a component $C' \subseteq \mathbb{F}_{r_3-\delta}$ where $r_2 < r_3 - \delta < r_3$, then has degree $+1$. Lemma 3.2 states that any $(r_3 - \delta)$-perturbation preserves the degree of C'.

Isolating neighborhood and Laplacian smoothing. Previously, topology-based simplification has focused on cancelling pairs of critical points that are connected by separatrices. Zhang et al. [6] and Chen et al. [10] proposed to compute an *isolating neighborhood* surrounding a pair of critical points, where a critical-point-free vector field can be found by solving a constrained optimization problem, referred to as a vector-valued *Laplacian smoothing* [6].

Based on Conley index theory, every boundary point of an isolating neighborhood can be classified as either an *entrance* or *exit* point. If an isolating region C in the domain contains multiple critical points and has a trivial Conley index, the flow inside C can be replaced with a new field free of critical points [6]. A typical situation for C to have a trivial Conley index is when its boundary ∂C consists of a single inflow and a single outflow component. However such an isolating neighborhood is not always straightforward to construct. The robustness-based method has no such a constraint.

4 Robustness-Based Simplification Algorithms

In robustness-based simplification, we first locate sets of critical points that share the lowest zero-degree ancestors in the merge tree and sort them based on their robustness values. For each set with a common robustness r, we compute the corresponding component of the sublevel set $C \subseteq \mathbb{F}_r$. Since by construction $\deg(C) = 0$, our strategy can simplify C, whereas the distance-based strategy requires an isolating neighborhood with trivial Conley index.

4.1 Preliminary

First we introduce the relevant constructions in a smooth setting, and then translate the corresponding language into the PL setting. Given a 2D vector field restricted to a degree-zero component C, $f : C \to \mathbb{R}^2$, we define the *image space* of C, $\mathrm{im}(C)$. For each point $p \in C$, we have a vector $v_p = f(p) \in \mathbb{R}^2$. $\mathrm{im}(C) \subset \mathbb{R}^2$ is constructed by mapping p to its vector coordinates v_p. The origin in $\mathrm{im}(C)$ corresponds to the critical points (0 vectors) in C. Since $C \subseteq \mathbb{F}_r$, it follows that $\forall p \in C$, $||v_p||_2 \leq r$, therefore $\mathrm{im}(C)$ is contained within a disc of radius r in $\mathbb{R}^2$. We denote the boundary of this disc by S. Now suppose the boundary of C, denoted as ∂C, is a simple closed curve[1]. Note that the above maps ∂C to S,

1. This is not needed, but it simplifies the algorithm and exposition.

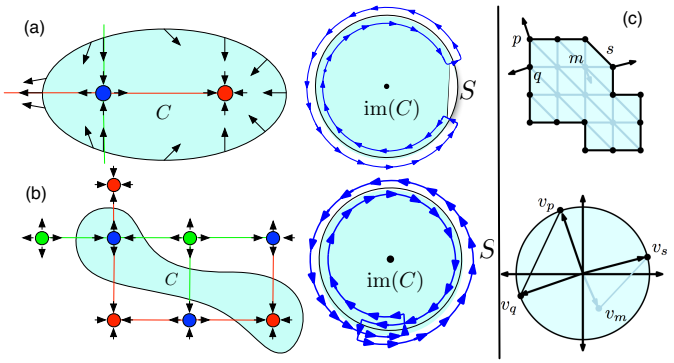

Fig. 3. (a)-(b): Illustrative examples for uncovered (a) and covered (b) boundaries of $\mathrm{im}\,(C)$. (c): A component and its image space with a few mappings highlighted in the PL setting.

obtaining the image, $\mathrm{im}\,(\partial C)$. We refer to the boundary of $\mathrm{im}\,(C)$ as *uncovered*, if $\mathrm{im}\,(\partial C) \subset S$, otherwise, as *covered*. Fig. 3(a)-(b) illustrate these concepts. Note that both examples have zero degree. In 3(a), the region C encloses a saddle-sink pair connected by a separatrix. By traversing counter-clockwise along ∂C and observing how its image $\mathrm{im}\,(\partial C)$ wraps around S, we see that the boundary of $\mathrm{im}\,(C)$ is uncovered. In 3(b), the region C encloses a saddle-sink pair not connected by separatrix and the boundary of $\mathrm{im}\,(C)$ is covered.

In the PL setting, the vector field f is restricted to a triangulation K of C, $f : K \to \mathbb{R}^2$, where the support of K, $|K| = C$. We construct the image of C by mapping each vertex $p \in K$ to its vector coordinates $v_p = f(p)$. Through linear interpolation, this construction also maps edges and triangles in K to edges and triangles in $\mathrm{im}\,(C)$ (Fig. 3(c)). The concept of *covered* and *uncovered* boundaries of $\mathrm{im}\,(C)$ can be defined similarly up to a small additive constant.

4.2 Algorithm Overview

Our simplification strategy consists of four operations:

- **Smoothing**(C): Perform Laplacian smoothing on C;
- **Cut**(C): Deform the vector field in its image space $\mathrm{im}\,(C)$ to remove critical points in C;
- **Unwrap**(C): Modify the vector field in its image space $\mathrm{im}\,(C)$ so part of its boundary is uncovered;
- **Restore**(C): Set the boundary to its original value.

Three cases are classified by the Conley index of C, denoted as $\mathrm{CH}_*(C)$. The operations to simplify each case are:

(a) If $\mathrm{CH}_*(C)$ is trivial, return $C_1 =$ **Smoothing**(C).

(b) If $\mathrm{CH}_*(C)$ is nontrivial and the boundary of $\mathrm{im}\,(C)$ is uncovered, then $C_1 =$ **Cut**(C), and return $C_2 =$ **Smoothing**(C_1).

(c) If $\mathrm{CH}_*(C)$ is nontrivial and the boundary of $\mathrm{im}\,(C)$ is covered, then $C_1 =$ **Unwrap**(C), $C_2 =$ **Cut**(C_1), $C_3 =$ **Restore**(C_2) and return $C_4 =$ **Smoothing**(C_3).

By construction, $deg(C) = 0$ in all three cases. Otherwise, there exists no simplification that can result in a vector field in C free of critical points.

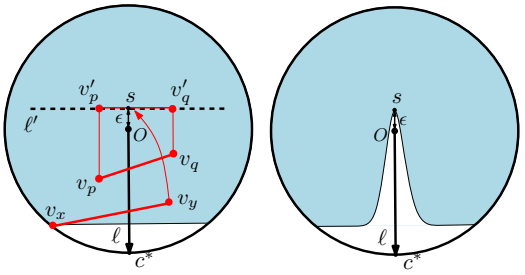

Fig. 5. **Cut** operation. Left: The projection of edges that intersect ℓ during the **Cut** operation. Right: After **Cut**, the light blue region represents $\mathrm{im}\,(C)$, which no longer contains (covers) the origin and so is critical point free.

4.3 Algorithm Details

We describe the **Cut** and **Unwrap** operations in detail and discuss the maximum amount of perturbation needed due to these operations. **Smoothing** is only used to achieve visually appealing results.

Cut operation. Suppose the boundary of $\mathrm{im}\,(C)$ is uncovered. The idea behind the **Cut** operation is to deform $\mathrm{im}\,(C)$ such that there is a small neighborhood surrounding the origin that is not covered by $\mathrm{im}\,(C)$. This corresponds to the situation where there is no critical point in C after the deformation. As shown in Fig. 5(left), we choose a point c^* on the uncovered part of the circle S (this point is referred to as the *cut point*) and define the line ℓ as the line segment beginning at the origin O and terminating at c^*. Define another line ℓ', which is orthogonal to ℓ and is ε away from the origin. The point $s \in \ell'$ is at a distance ε from the origin. Next, we find all the mesh edges $v_p v_q$ (corresponding to the edge pq in K) in the interior of $\mathrm{im}\,(C)$ that intersect with the line ℓ, and project their end points onto ℓ', forming the projected edge $v'_p v'_q$. In the original domain, the vectors at $p, q \in K$ are deformed from v_p and v_q to the vectors v'_p and v'_q, respectively. Third, we locate all the mesh edges $v_x v_y$ where $x \in \partial C$ (and so v_x is on the boundary of $\mathrm{im}\,(C)$ and v_y is in the interior). We move the point v_y to s so that the edge $v_x v_y$ no longer crosses ℓ and the boundary vector remains unchanged. Since the boundary of $\mathrm{im}\,(C)$ is uncovered, there is no edge that intersects ℓ whose end points are both located on the boundary of $\mathrm{im}\,(C)$ (i.e., whose corresponding points are both in ∂C). This operation creates an empty wedge around the origin (Fig. 5 right), which ensures that there are no critical points in C after the modification. By construction, the amount of perturbation is less than $r + \varepsilon$. When doing Laplacian smoothing, we keep the projected vertices (end points of the cut edges) fixed to ensure that the origin is not recovered.

The procedure to find a cut point c^* is shown in Fig. 4(a)-(b). In (a), by traversing counter-clockwise along ∂C and observing how its image $\mathrm{im}\,(\partial C)$ (blue curve) wraps around S, we define the angle θ of a point along S to be its *phase*. In (b), we showcase (in blue) the corresponding phase plot,a function $h : \partial C \to \theta$ where $\theta \in [-\pi, \pi]$. Traversing ∂C again, we can use *phase-unwrapping* to compute a continuous function $\varphi : \partial C \to \phi$ for $\phi \in \mathbb{R}$ (shown in red) using the following equation

$$\varphi(i) = \lfloor \theta(i) - \varphi(i-1) + 1/2 \rfloor + \theta(i).$$

This is a standard discretization of phase-unwrapping commonly used in signal processing. Since the boundary of $\mathrm{im}\,(C)$ is uncovered, it follows that $\max_{\partial C}(\varphi) - \min_{\partial C}(\varphi) < 2\pi$. We set the

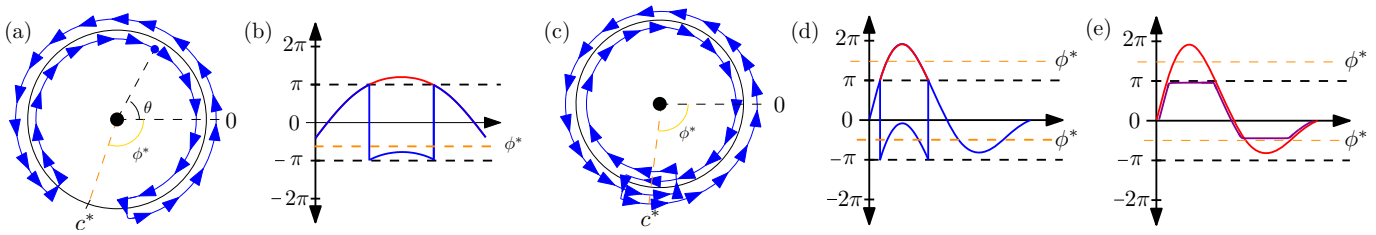

Fig. 4. (a)-(b) Locating a cut point for the **Cut** operation: (a) Track the angle (a.k.a. phase) of a point in $\mathrm{im}(\partial C)$ along S as we move along ∂C counter-clockwise. (b) The corresponding phase plot (a.k.a. angle-valued function) is shown in blue. The result of phase-unwrapping is shown in red. (c)-(e) Locating an unwrap point in **Unwrap** operation: (c) Track the angle of a point in $\mathrm{im}(\partial C)$ along S as we move along ∂C counter-clockwise. (d) The corresponding angle-valued function (shown in blue), the result of phase-unwrapping (shown in red), and the optimal unwrap point c^* corresponding to phase ϕ^*. (e) The modified boundary of $\mathrm{im}(C)$ (shown in purple), which becomes uncovered.

cutting angle ϕ^* as the mid-point of the uncovered part and the corresponding cut point is

$$\phi^* = \frac{1}{2}\left(\max_{\partial C}(\varphi) + \min_{\partial C}(\varphi)\right) + \pi,\ c^* = (r\cos\phi^*, r\sin\phi^*),$$

where r is the robustness parameter of the sublevel set (and the radius of the disk S, where $\mathrm{im}(C) \subset S$). By using the phase parameter θ, we do not need to worry about PL effects when computing c^*.

Unwrap operation. If the boundary of $\mathrm{im}(C)$ is covered, we must first **Unwrap** the boundary before we perform the **Cut** procedure. The **Unwrap** operation is divided into the steps illustrated in Fig. 4(c)-(e). Similarly to the *cut point*, we determine the optimal *unwrap point*. As before, we traverse ∂C and compute a phase plot $h : \partial C \to \theta$, unwrapping the phase to obtain a continuous function $\varphi : \partial C \to \phi$ (Fig. 4(c)-(d)). The unwrapping point ϕ^*, is

$$\phi^* = \frac{1}{2}\left(\max_{\partial C}(\phi) + \min_{\partial C}(\phi) + 2n\pi\right),$$

where n is the smallest integer such that $|\min(\theta) + 2n\pi - \max(\theta)| < \pi$, and $c^* = (r\cos\phi^*, r\sin\phi^*)$. To **Unwrap** the boundary, let $X \in \partial C$ be the set of points on the boundary such that $\phi(X) > \phi^* - \delta$, and $Y \in \partial C$ be the set of points that $\phi(Y) < \phi^* + \delta - 2n\pi$. As illustrated in Fig. 4(e) , to **Unwrap** we set

$$v_x = \begin{pmatrix} r\cos(\phi(x)) \\ r\sin(\phi(x)) \end{pmatrix} \quad x \in X, \qquad v_y = \begin{pmatrix} r\cos(\phi(y)) \\ r\sin(\phi(y)) \end{pmatrix} \quad y \in Y,$$

where r is the magnitude of the vectors on the boundary (e.g., the sublevel set parameter). The final **Restore** step replaces the vectors on the boundary of the simplified region with their original values (i.e. the values before the **Unwrap** step). As in case (b), the deformation is bounded by $r + \varepsilon$ (since the internal nodes move less than $r + \varepsilon$ and boundary nodes have their original values).

Bounded perturbations. By construction, the algorithm comes with theoritical guarantees on the amount of perturbation introduced to the vector field. We formally state the following theorem whose proof is deferred to the supplementary material.

Theorem 4.1. Let x be a critical point of robustness r, with the corresponding component $C(x)$. Let f and $\hat{f}$ denote the vector fields before and after simplification respectively using **Unwrap** and **Cut** operations. Then $||f(p) - \hat{f}(p)||_\infty \leq r + \varepsilon$ for all $p \in C(x)$ and $f(p) = \hat{f}(p)$ for $p \notin C(x)$.

4.4 Synthetic Examples

We illustrate our simplification strategy on three PL synthetic examples, highlighting the different cases.

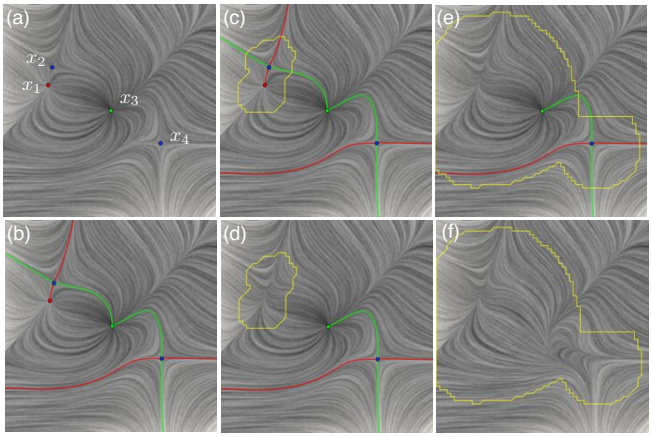

Fig. 6. SyntheticA. (a) The original vector field: sinks are red, sources are green and saddles are blue. (b) The topological skeleton: saddle-sink separatrices are red and saddle-source separatrices are green. (c)-(d) 1st level simplification: before (c) and after (d) **Smoothing**. (e)-(f) 2nd level simplification: before (e) and after (f) **Smoothing**.

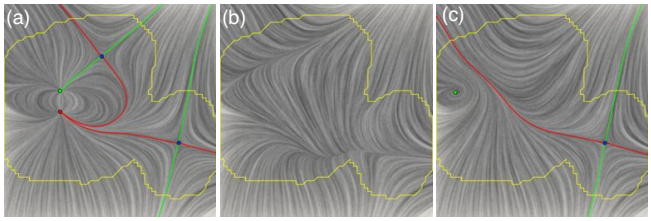

Fig. 7. SyntheticB. (a) the original vector field with its topological skeleton. (a)-(b): Single level simplification before (a) and after (b) by **Cut** and **Smoothing**. (c) Only applying **Smoothing** does not make the region a critical point free field.

SyntheticA (Fig. 6) corresponds to the example in Fig. 2. It involves pairs of critical points connected by separatrices. At r_1, we have a component that contains critical points x_1 and x_2 and at r_3 we have a component that contains all four critical points x_1 to x_4. The simplification hierarchy involves two steps ranked by robustness values: first x_1 and x_2 are simplified, and then x_3 and x_4. Since both components (marked by yellow boundary) have a trivial Conley index, this corresponds to case (a), where only **Smoothing** operations are needed. SyntheticB (Fig. 7) contains four critical points interconnected by separatrices, which could be simplified in a single level using a robustness-based strategy. Since the component of interest has a nontrivial Conley index, directly applying Laplacian smoothing fails (as shown in Fig. 7(c)). The

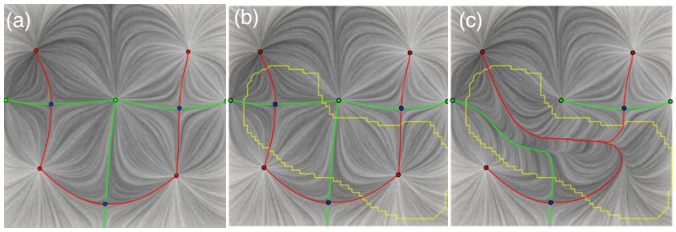

Fig. 8. SyntheticC. (a) the original vector field with topological skeleton. (b)-(c) Before (b) and after (c) simplification by combining **Unwrap**, **Cut** and **Smoothing**.

component's boundary is uncovered, so we apply case (b) of our simplification by combining **Cut** with **Smoothing**.

SyntheticC (Fig. 8) corresponds to case (c) of our algorithm. This is an atypical case involving a pair of critical points not directly connected by a separatrix. In this case, the component of interest C has nontrivial Conley index, and the boundary of its image is covered. The robustness-based strategy cancels the critical point pair without any issue by combining **Unwrap**, **Cut** and **Smoothing** operations. We further focus on this example by illustrating the image space of C, $\mathrm{im}(C)$, during various steps of simplification in Fig. 9. In Fig. 9(a), the entire boundary and disk are covered. However, from the left phase plot in Fig. 10, we can see that the degree is 0. Once the optimal unwrapping point is computed, we perform the **Unwrap** operation, giving the right phase plot in Fig. 10 and the image space in Fig. 9(b), leaving the boundary S uncovered. The effect of the **Cut** operation in image space is shown in Fig. 9(c), creating a void surrounding the origin. Lastly, in Fig. 9(d), the boundary is restored for the final output.

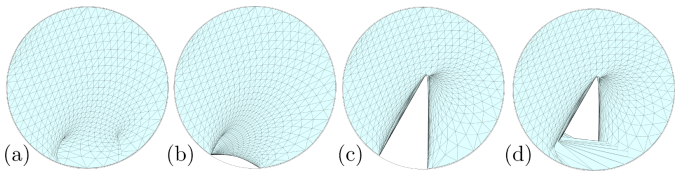

Fig. 9. SyntheticC. The image space is shown through the different steps: (a) original, (b) after **Unwrap**, (c) after **Cut**, and (d) final output after **Restore**.

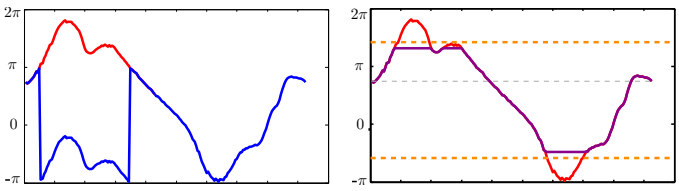

Fig. 10. SyntheticC. Left: The phase plot, original version (blue), and the phase-unwrapped version (red). Right: The phase plot with optimal unwrap point (orange) and the modified phase plot with boundary uncovered (purple).

5 Results

We demonstrate our robustness-based simplification strategy on a number of real-world datasets. When possible, we compare our method with distance-based simplification.

The first real-world dataset we explore is the top layer of a 3D simulation of global oceanic eddies [36] for 350 days of the year 2002. The 2D time-varying vector field has resolution 3600×2400. We extract tiles representing the flow in the central Atlantic Ocean (60×60) and construct standard triangulation on the point samples. We select multiple time slices from this data: OceanA contains slices #21217 and #21311; OceanB and OceanC correspond to slices #20904 and #20821, respectively; OceanD includes a time-varying sequence of slices from #20710 to #20715; OceanE includes slice #21232. We also extract a tile representing the flow in the south Atlantic Ocean (100×100): OceanF contains slice #138 (under different indexing scheme). We refer to the entire datasets as CentralAtlantic and SouthAtlantic.

Our second real-world dataset is a 2D time-varying vector field simulation of homogeneous charge compression ignition (HCCI) engine combustion [37] represented as a 640×640 regular grid with a periodic boundary. The data consists of 299 time-steps at intervals of 10^{-5} seconds. We select slices #173 and #152 from this data, referred to as CombustionA and CombustionB respectively. We refer to the entire dataset as the Combustion dataset.

We apply vector field simplification to OceanA-OceanF, CombustionA and CombustionB. In particular, we use datasets OceanE, OceanF, CombustionB to demonstrate complete hierarchical simplification. Finally, we illustrate an extension to the method by applying simplification to time-varying vector field on datasets CentralAtlantic, SouthAtlantic and Combustion.

5.1 Topologically Equivalent Scenarios

In many scenarios, our approach produces topologically equivalent and visually comparable results to the distance-based approach, such as for the OceanB dataset (Fig. 11(a)). The critical point pairs of interest are highlighted by the black dashed boxes in the top row left. The critical points are colored by their robustness values (red—low, blue—high). The upper right pair is more robust than the lower middle pair and is further apart. The simplification results generated by distance-based and the robustness-based approaches are shown in the second and third rows, respectively. The approximated isolating neighborhoods are highlighted by the white boxes (middle row), whereas the sublevel sets the yellow enclosure (bottom row). From the comparison, we observe that, both the distance and robustness metrics generate the same pairs of critical points and the simplification orderings determined by these two metrics agree. A subtle difference in the resulting vector fields is visible due to differences in the local regions determined by the two metrics and algorithms for modifications.

OceanA dataset (Fig. 12 (a)-(b)) shows a more complex scenario where the region encloses more than two critical points. The vector fields in this example are from slices #21217 and #21311. Each of these two clusters (highlighted by the black dashed boxes in Fig. 12(a)-(b)) consists of four critical points that are close in distance and have small identical robustness values. The robustness metric groups them as one cluster automatically and computes a region based on their sublevel set. The bottom row of Fig. 12(a)-(b) provides the simplification results using the algorithm introduced in Section 4. Although the distance-based method cannot group these four critical points in one simplification, for comparison we compute an isolating neighborhood that encloses them and apply Laplacian smoothing. Both methods return similar results. Nevertheless, the robustness-based method can handle regions with more complex boundary configurations.

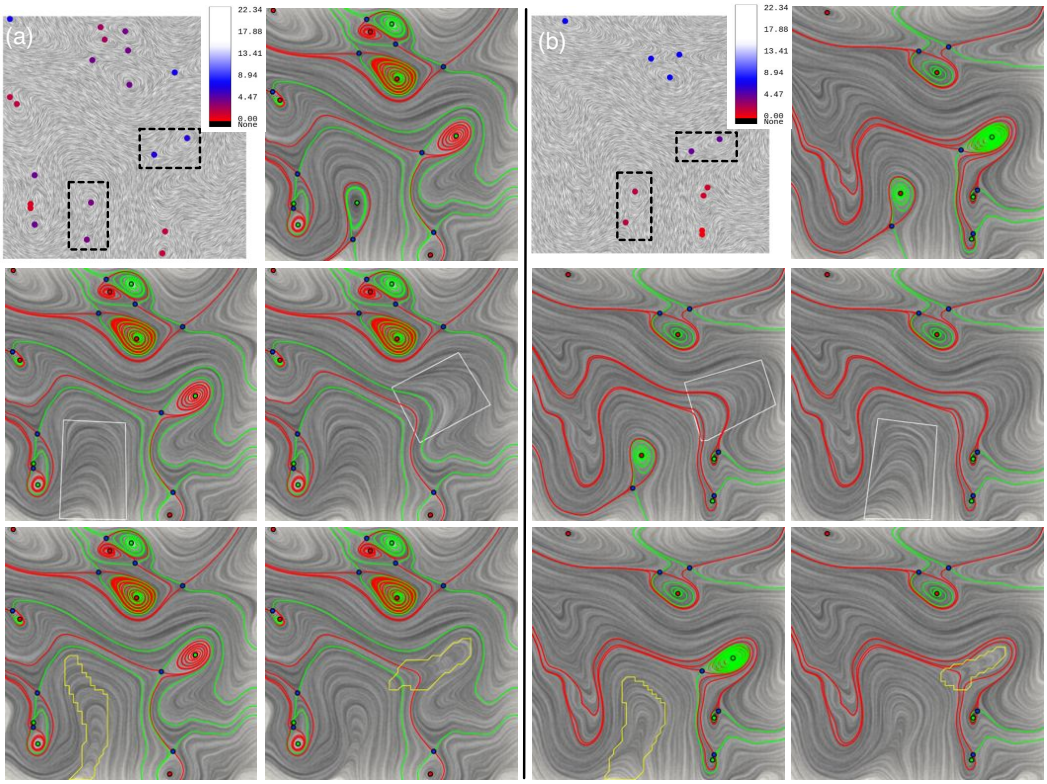

Fig. 11. (a) The OceanB dataset. (b) The OceanC dataset. For each subfigure: Top Row: Left – shows robustness values with the region of interest highlighted (robustness values are colored from red to white, where red means low and white means high robustness); Right – shows the vector field marked by critical point types along with separatrices. Middle Row: the two-step hierarchical simplification based on distance. Bottom Row: the two-step hierarchical simplification based on robustness.

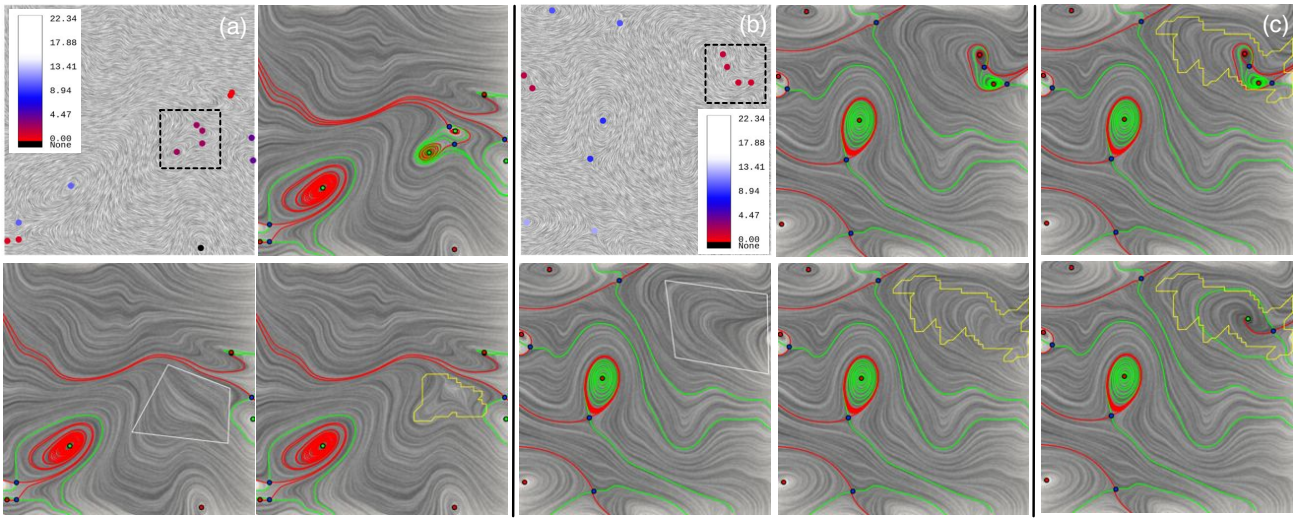

Fig. 12. The OceanA dataset: (a) #20311; (b)-(c) #21217. (a)-(b) For each subfigure, Top Row: Left – shows robustness values with region of interest highlighted; Right – shows the vector field marked by critical point types and its topological skeleton. Bottom Row: results after distance-based (left) and robustness-based simplifications (right). (c) A region (yellow boundary) with a nontrivial Conley index and uncovered boundary (top), where smoothing does not remove its critical points (bottom).

5.2 Inconsistent Hierarchical Scenarios

We also identified a number of scenarios where the distance-based and robustness-based methods disagree. One example is the OceanC dataset (Fig. 11(b)). Here, two pairs of critical points are studied (highlighted in the top row of Fig. 11(b)). Even though the pairing of these four critical points is consistent with both metrics, their actual simplification orderings are different. The distance-based method cancels the pair in the middle-right of the domain first, while the robustness-based method cancels the lower-middle pair first. Fig. 13 provides another example that shows the discrepancy of the two approaches in determining the simplification ordering of critical point pairs in the time-varying setting. In this example, we look at consecutive time steps from the OceanD dataset. Fig. 13(a) highlights the critical points of

interest. The pairings of these four critical points again agree with each other using both topological-skeleton and robustness metrics. We perform a per-slice simplification using the two approaches. The results are shown in the second (distance-based) and third (robustness-based) rows in (b)-(c), respectively. From the results, we see that the cancellation orderings change over time using the distance-based metric. This is due to an increased distance between the two critical points near the upper-right corner, resulting in a change of the simplification order. On the other hand, the robustness for these two pairs is stable. In this example, the robustness-based simplification returns a consistent outcome (i.e., pairing and simplification hierarchies) over the two time steps.

5.3 Challenging Scenarios

There are a number of cases where the topological-skeleton-based metric combined with the Laplacian smoothing technique is incapable of simplifying the given vector field. For example, for the SyntheticB dataset shown in Fig. 7, it is impossible to find an isolating neighborhood with a trivial Conley index that encloses all the critical points due to the boundary condition. Therefore, even though the obtained local region is guaranteed to be zero degree, Laplacian smoothing fails to solve for a critical point free field. On the other hand, the simplification algorithm introduced in Section 4 successfully simplifies the field. A similar situation occurs in Fig. 12(c) (OceanA slice #21217). In this example, we try to apply Laplace smoothing in the local region computed based on robustness (top). The boundary configuration of this region is rather complex and does not satisfy a trivial Conley index. The Laplacian smoothing based on this boundary configuration fails (bottom), but the proposed simplification method succeeds. These two examples showcase the utility of the proposed algorithm in solving a critical point free field within any given regions with zero degree. This relieves the requirement of the trivial Conley index whose corresponding isolating neighborhood is sometimes difficult to obtain.

Fig. 8 shows a nontypical case that involves the cancellation of a pair of critical points not directly connected by separatrix. It is impossible for the topological-skeleton-based method to compute an isolating neighborhood that encloses two critical points (but not the others) not connected by separatrix [10]. Nonetheless, the robustness metric derives a local region that encloses only these two critical points with total degree equal to zero under a certain configuration of the flow magnitude. Hence, these two critical points may be cancelled. Whereas this may rarely occur in the real-world data, it illustrates the flexibility and generality of the proposed method. In practice, a simpler but similar situation may occur.

In the slice of the combustion data (Fig. 14), the simplification results and hierarchies of the distance-based metric (first row), and the robustness-based metric (second row) do not agree. The corresponding vector field is a high resolution incompressible flow. The topology-based simplification requires the extraction of topology and the subsequent isolating neighborhood, which can be difficult due to instability in numerical integration and computational cost with large-scale dataset. In contrast, the robustness-based method does not require the computation of topology, and its computation is fast and parallelizable, making it more practical for large datasets.

5.4 Complete Hierarchy

In this experiment, we show complete hierarchical simplifications, for OceanE, OceanF and CombustionB datasets. We start with the OceanE dataset (Fig. 15). The merge tree contains 10 zero-degree internal nodes (that correspond to 10 unique robustness values), therefore, there are 10 levels in the merge tree that we simplify where the simplification ordering is determined by robustness values of the critical points. We use Li to denote the i-th level of simplification, where level L0 corresponds to the original data without any simplification and L10 is the highest level canceling all critical points. At each level of simplification, we highlight the sublevel sets (i.e., the colored regions) that will be simplified; and after simplification, we highlight the simplified regions with colored boundaries. There are several pairs of critical points whose corresponding sublevel sets are nested within one and another. Noticeably in (e), there is a maximum of three levels of nesting at the lower half of the domain. The most interesting observation is that the heat map visualizations of vector field magnitude function before and after simplification ((f) & (g)) exhibit extremely similar patterns. This indicates that the robustness based simplification introduces only a small amount of perturbation to simplify this vector field, which largely preserves the vector field magnitude over the space. Fig. 16 shows a second example with the OceanF dataset. There are 13 levels of simplification (e.g., 13 pairs of critical points to be simplified) based on the merge tree structure. We show simplification up to L12 where the pair with the largest robustness remains intact. Here the vector field magnitude function displays almost no changes between level 1 and level 11, however, it exhibits a large change in the upper region between level 11 and 12 ((b) & (c)). This is because the pair of critical points to be simplified at L12 has a larger robustness value compared to previous ones (as indicated by a large sublevel set that encloses more than 60% of the domain). Therefore, the required amount of perturbation to eliminate this pair is relatively large, causing the observed discrepancy between heat maps in (b) and (c). Finally, the smoothing process makes the simplified region visually smooth.

Fig. 17 demonstrates the example of the CombustionB dataset. There are 11 levels of simplification based on the merge tree. We performs 10 levels of simplification, leaving a pair of critical points with the largest robustness. This dataset has a higher resolution mesh compared to the above two datasets, thus, the extracted sublevel sets possess smooth boundaries. Similar to the example of OceanE, after 10 levels of simplification, the vector field magnitude is preserved at large (see the comparison between (f) and (g)).

6 Time-Varying Vector Fields

A natural extension is to apply robustness-based simplification to 2D time-varying data. Assume a 2D time-varying vector field is represented by a discrete number of time slices. Robustness has been used to give provable results on tracking critical points [38] across time slices as well as highlighting interesting changes to the topology [34]. The definitions extend naturally to the time-varying case. We index the vector field by time t, $f_t : \mathbb{R}^2 \to \mathbb{R}^2$, and assume that it is c-Lipschitz in space and time (i.e., it varies slowly). We further assume that critical points can be tracked under certain conditions. In practice, we employ region overlap heuristic [39] to track critical points, which works relatively well for our testing datasets.

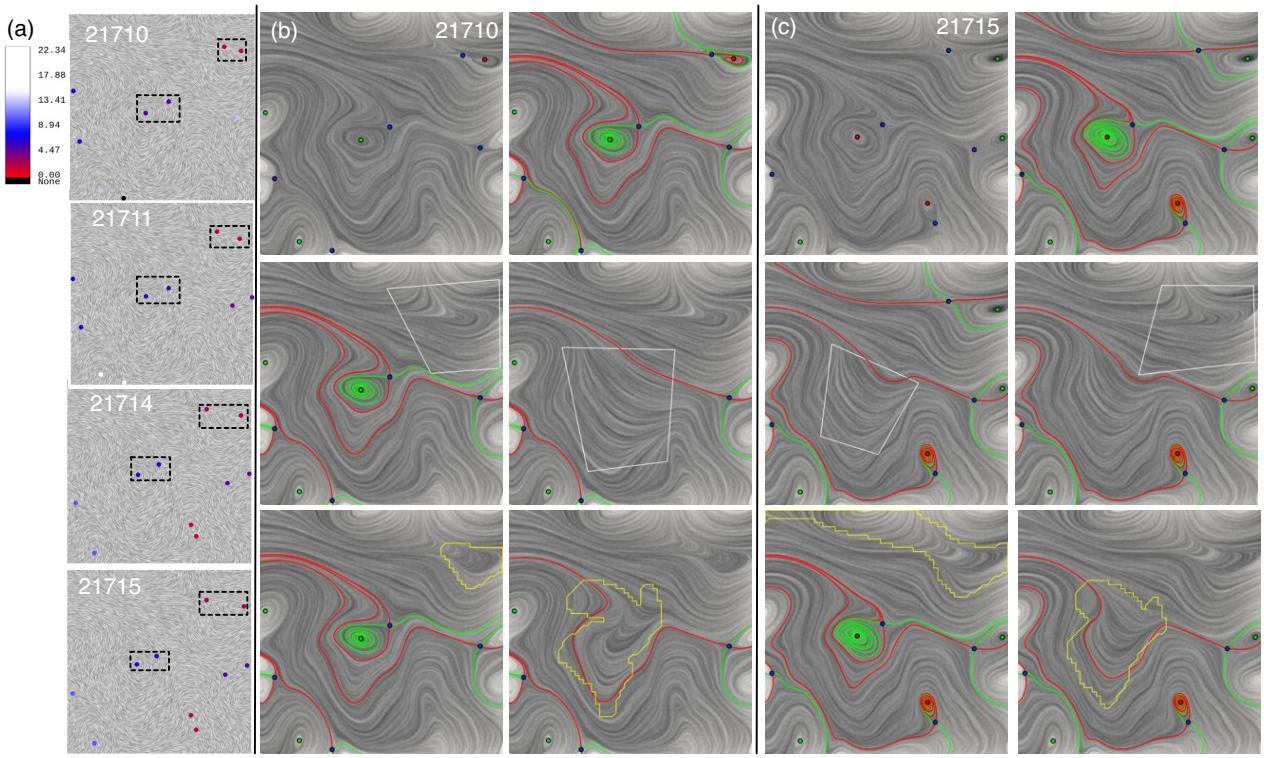

Fig. 13. The OceanD dataset. (a) A sampled time series with pairs of critical points highlighted, where white numbers indicate time stamps. (b) #21710. (c) #21715. For each subfigure (b)-(c), Top Row: The original vector field (left) and with the separatrices (right). Middle Row: The simplification ordering for the distance-based strategy. Bottom Row: The simplification ordering for the robustness-based strategy. Orderings for distance and robustness-based methods are consistent in (b) and different in (c).

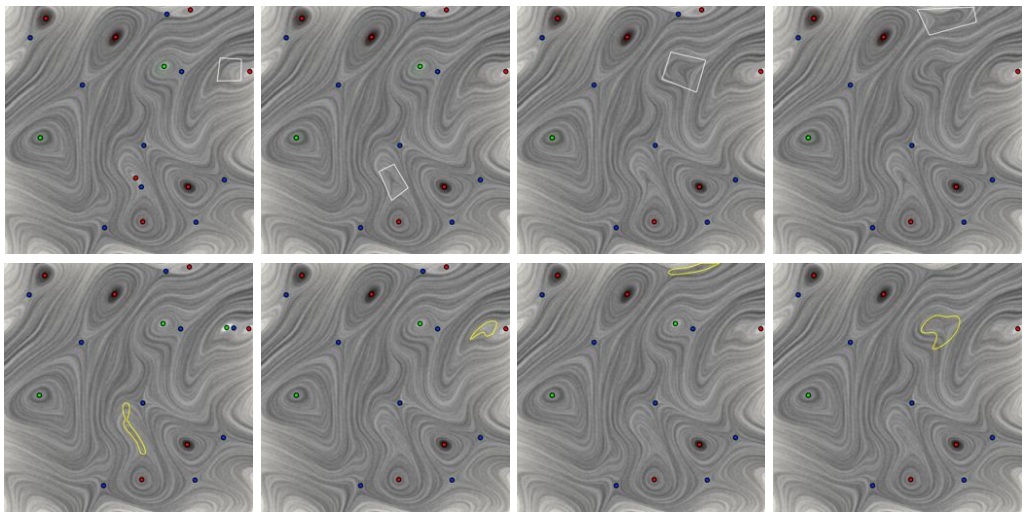

Fig. 14. The CombustionA dataset. The bottom-up hierarchical simplifications (Top) from the distance-based strategy and (Bottom) from the robustness-based strategy.

For time-varying vector fields, we would like to obtain consistent simplification based on robustness. By consistency, we mean that the simplified results do not introduce new critical points nor create discontinuities in the vector field. A main challenge for obtaining a consistent simplification across time is that the per-slice robustness measure is not necessarily stable over time. This instability is characterized by the so-called *pairing switches*. Recall in Section 4, we identify a set of critical points that share the lowest zero-degree ancestors in the merge tree and apply simplification algorithm on such a set. The critical points belong to the same set are considered *paired* with each other. In other words, they are *(pairing) partners*. In the time-varying setting, a given critical point may change its partner(s), and this change is referred to as a *pairing switch*. For example, see the two merge trees in Fig. 18(a)-(b) that correspond to two adjacent time slices. At time i, critical points x_1 and x_2 are paired at r_1 and x_3 and x_4 are paired at r_2. At time $i+1$, x_2 and x_3 become paired at r'_1, while x_1 and x_4 are paired at r'_3. This constitutes a pairing switch. This type of event can lead to inconsistencies during the simplification if we apply the simplification on the

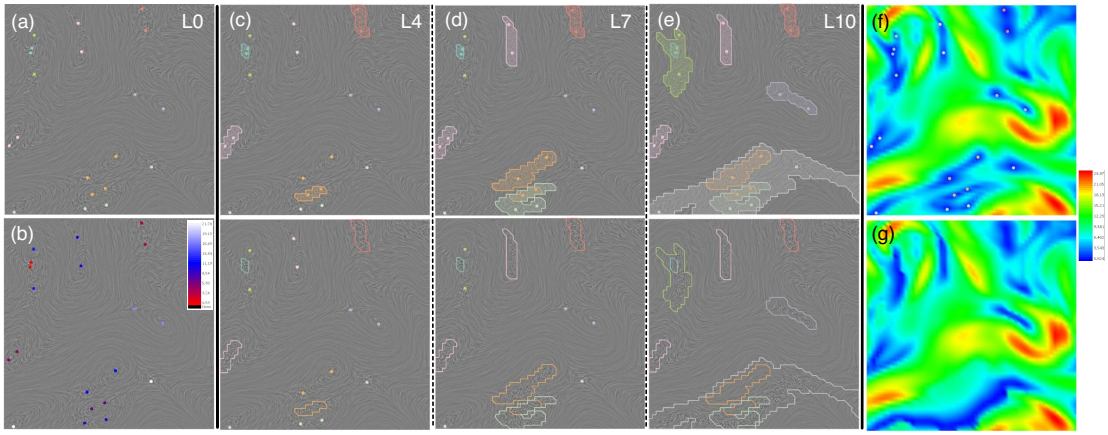

Fig. 15. Hierarchical simplification of OceanE. (a)-(b): Vector field before simplification where critical points are colored by robustness pairing (a) and robustness values (b). (c)-(e): Different levels of simplification, levels 4, 7 and 10, which correspond to robustness values 2.74, 4.06 and 7.48, respectively; vector fields before (top row) and after (bottom row) simplification. (f)-(g) Vector field magnitude functions before (f) and after (g) simplification; the function values are colored by heat map (where blue means low and red means high values).

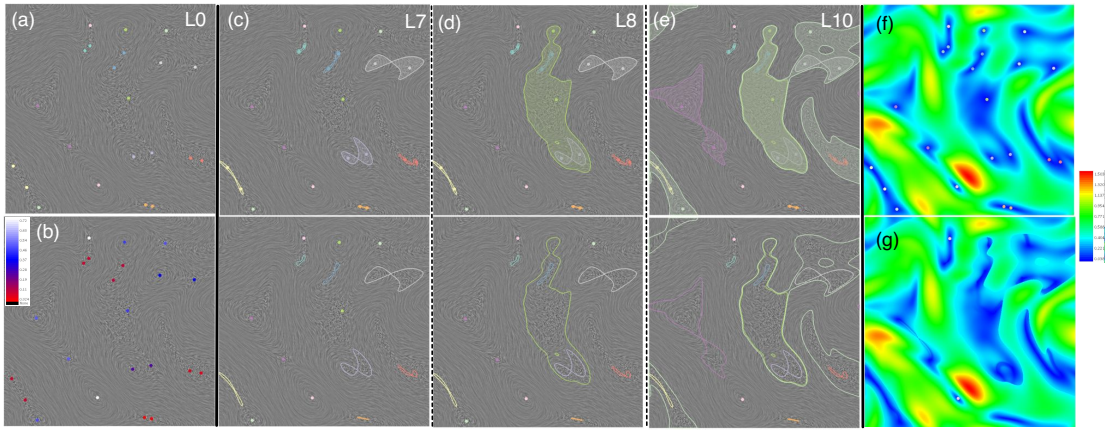

Fig. 17. Hierarchical simplification of CombustionB. (a)-(b): Vector field before simplification where critical points are colored by robustness pairing (a) and robustness values (b). (c)-(e): Different levels of simplification, levels 7, 8 and 10, with robustness values 0.29, 0.35 and 0.39, respectively; vector fields before (top row) and after (bottom row) simplification. (f)-(g) Vector field magnitude functions before (f) and after (g) simplification.

individual time slices independently. It is an interesting question to understand what the pairing switches indicate in terms of the structure of the time-varying vector field. Our interpretation is that the robustness simplification after all is a global process, and that pairing switches may indicate discrete, local, structural changes in the vector fields (that complement the conventional structural changes, e.g., bifurcations).

To ensure that we can maintain consistency across the entire dataset (i.e., global consistency), we consider the task of simplifying a specific critical point trajectory, that is, a PL path a critical point takes when it moves through time. A set of trajectories are then considered *strict pairing partners* if their associated critical points are pairing partners across all time slices, and exhibit no pairing switches. A simple solution to guarantee a globally consistent simplification is to simplify the trajectories which are strict pairing partners.

To do so, we assign a robustness value to a trajectory by considering the maximum robustness value of the critical point over its entire trajectory. If $x(t)$ is a critical point at time slice t and $r(x(t))$ returns its robustness at time t, the robustness value over its trajectory is $R(x) = \max_t r(x(t))$. This is the minimum amount of perturbation required to simplify the entire trajectory. If we take a threshold smaller than R, there exists at least one time slice where the critical point cannot be simplified. However, we cannot simplify the sublevel sets corresponding to the value $R(x)$ in all the time slices. For example, in Fig. 18(b), suppose that the maximum robustness value of x_1 is its robustness value at time $i+1$, that is, $R(x_1) = r_3'$. Then at time $i+2$, assume $r_3'' > r_3' > r_2''$, the corresponding sublevel set at the value $R(x_1)$ has degree -1, implying that it cannot be simplified. Therefore, simplification should be performed according to the robustness value of the critical point in each time slice, e.g., x_1 would be simplified at time $i+2$ within a sublevel set at a value in the range $[r_1'', r_2'']$.

We now simplify a group of strict pairing partners on a per-slice basis. At each time slice, the sublevel set to be simplified is based on a minimum (i.e., local) robustness value that encloses all the targeted points. The operations **Cut**, **Unwrap** and **Restore** may be applied on each time slice independently and this will result in a consistent vector field over time. The only problematic operation is **Smoothing**. While this is not strictly needed, it does make the resulting vector field look smoother. If the smoothing is done on a per time slice basis (referred to as the *spatial smoothing*), the result may not be coherent over time. We therefore consider the PL tubes surrounding the critical points, which can be

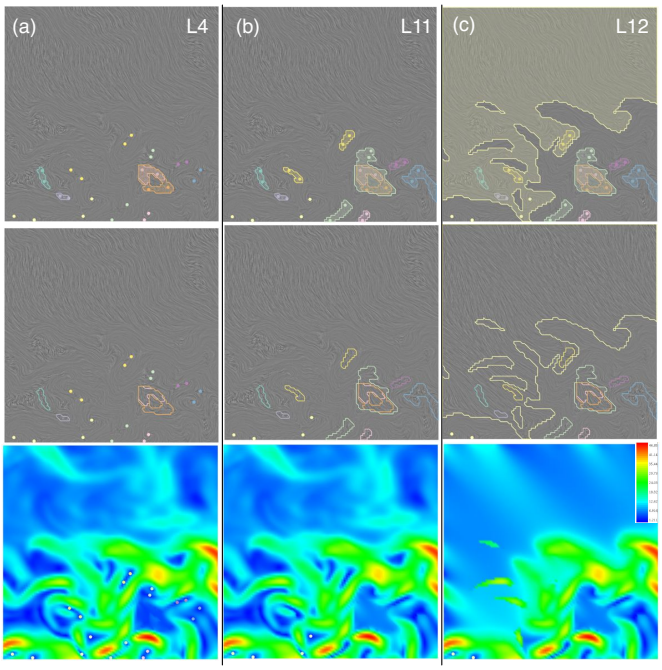

Fig. 16. Hierarchical simplification of OceanF. Top and middle rows: vector fields before (top) and after (middle) simplification. Bottom: vector field magnitude functions colored by heat map, where blue means low and red means high values. (a)-(c) Different levels of simplification: levels 4, 11 and 12, with corresponding robustness values 3.79, 12.72 and 14.11, respectively.

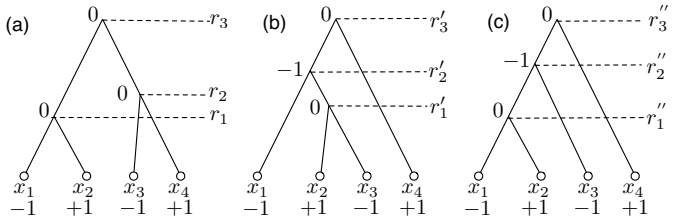

Fig. 18. Examples of two pairing switches between adjacent time slices. (a)-(c) $t = i$, $t = i+1$ and $t = i+2$ respectively.

considered as an envelop surrounding the trajectories (according to the local robustness values). In *spatial-temporal smoothing* (as first introduced in [22]), we fix the boundary of this tube (and the cut points/edges) and perform Laplacian smoothing within it.

We apply such a simplification algorithm to several time-varying datasets, namely, CentralAtlantic, SouthAtlantic and Combustion (Fig. 20). Each one shows our simplification result in the space-time domain, where the line segments correspond to the trajectories of the critical points over time. Each trajectory is colored in (b) based on their pairing partnerships. A pairing switch is indicated by the change of color along the trajectory. We see that a handful of trajectories that exhibit no pairing switches in (c) could be simplified. Most of them are in the neighborhoods of bifurcations. We also observe some special scenarios (Fig. 19) where sublevel sets that overlap with one and another across time are simplified by our algorithm. See the supplementary video for the animations of these three time-dependent vector fields before and after simplification.

Using this first-order approach, we can guarantee the consistency of our simplified results. However, to further improve the algorithm to simplify more trajectories, while still maintaining

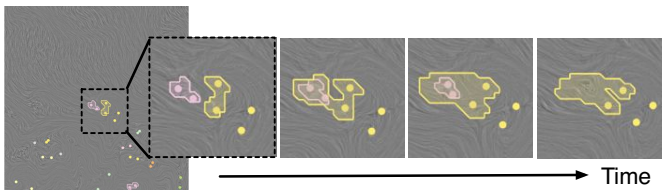

Fig. 19. Simplified critical points trajectories that intersect with one another. Time-varying simplification of SouthAtlantic.

global consistency, we relax the previous definition of strict pairing partners, and define a set of trajectories to be *pairing partners* if their associated critical points are paired among themselves. That is, we partition all the trajectories into groups of trajectories so that pairing switches occur only among trajectories within the same group. A breadth-first search would suffice to construct these groups. For CentralAtlantic, SouthAtlantic and Combustion datasets, we obtain 48, 76 and 10 groups of pairing partners, respectively. For Combustion, 9 out of 10 groups contain two trajectories each, which are identical to the strict pairing partners obtained by the first-order approach (Fig. 20 bottom row (c)); while the remaining 42 trajectories form a single group. The largest groups for CentralAtlantic and SouthAtlantic contain 115 and 161 trajectories, respectively. We illustrate pairing partner groups for CentralAtlantic and SouthAtlantic datasets in Fig. 21, e.g., the purple group highlighted in Fig. 21(b) involves a total of 8 critical points (see the supplementary material for results involving this group). Groups with a single trajectory (i.e., trajectories with no pairing partners) and groups with the largest number of trajectories are ignored (colored in grey).

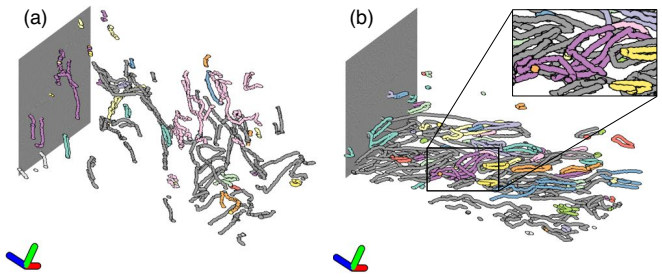

Fig. 21. Pairing partners colored by their group memberships for (a) CentralAtlantic and (b) SouthAtlantic datasets.

There are two challenges when applying simplifications to time-varying vector fields. First, over a large time scale, algorithms based on (strictly) pairing partners may not generate sufficient number of trajectories to be simplified. Second, to guarantee global consistency, we may need to simultaneously simplify a single group based on pairing partners, or multiple groups containing intersecting/mixing trajectories (e.g., in scenarios similar to Fig. 19). It is possible (in the worse case scenarios) that the target of simplification contains a large collection (or almost all) of the trajectories. In these cases, consistent simplifications become less informative. In these situations, we argue that our algorithms would be most useful by localizing the simplification in time and space. By focusing our analysis on a small, user-defined time interval, there are fewer pairing switches and thus more simplification targets (Fig. 22). Alternatively, we could focus

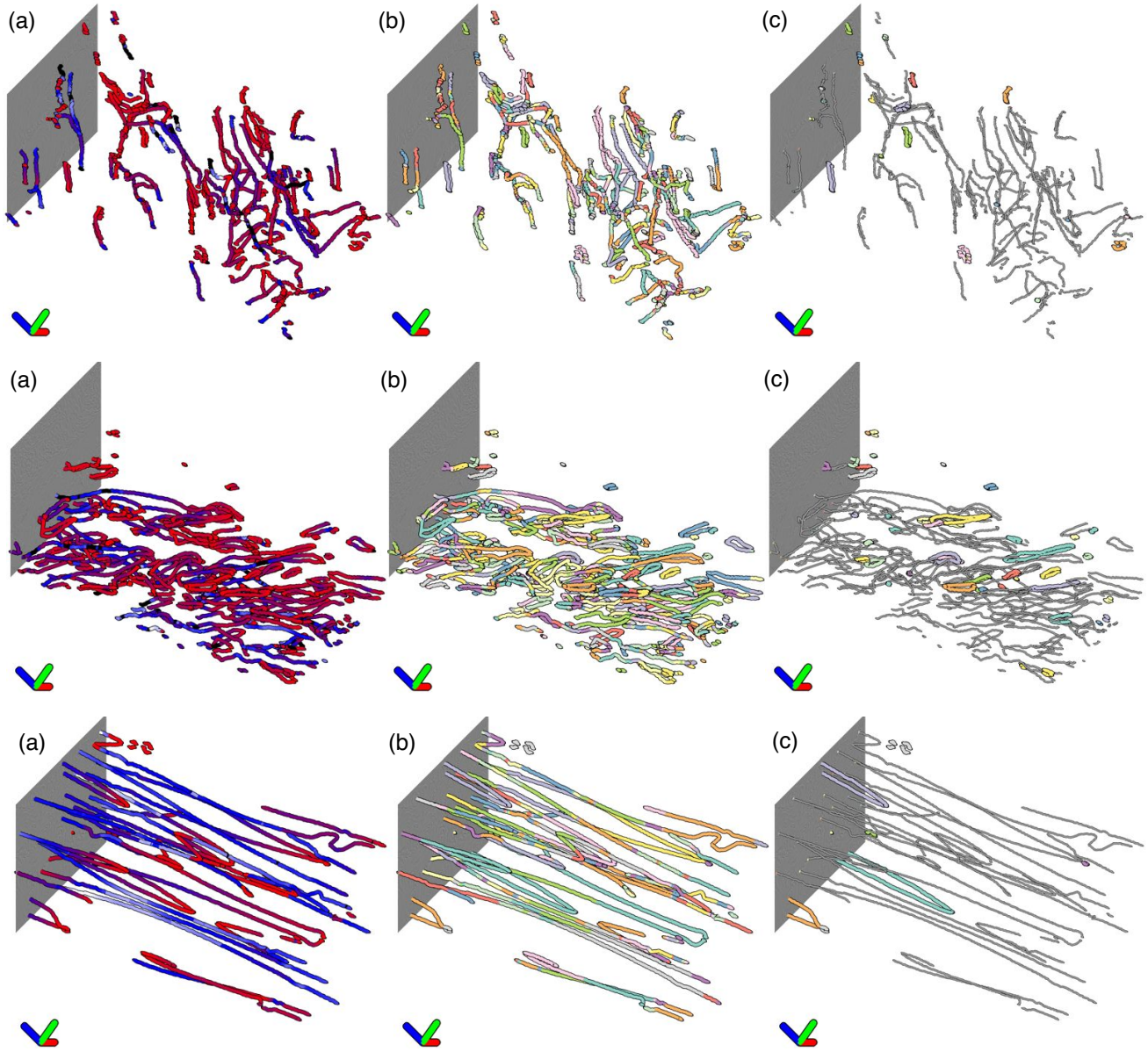

Fig. 20. Time-varying simplification of CentralAtlantic (top row), SouthAtlantic (middle row) and Combustion (bottom row). (a) Trajectories colored by per-slice robustness values. (b) Trajectories colored by pairing before the simplification. (c) Trajectories (i.e., strict pairing partners) that are simplified are highlighted with color.

our analysis on a sub-domain of the vector field or on a handful of selected groups/trajectories, where hierarchical simplification might no longer be maintained.

7 Discussions

We have presented a new and complementary simplification framework that does not depend on the topological skeleton but incorporates topological information through *robustness*. Rather than considering the geometric proximity of critical points, we consider the minimum perturbation required to remove critical points. Our algorithm comes with theoretical guarantees on the amount of perturbation we introduce. The motivation for Laplacian smoothing is to produce more visually appealing results. However, to the best of our knowledge, no nontrivial bounds exist on the amount of perturbation introduced by such a smoothing. In practice, smoothing only marginally increases the amount of perturbation. For the detailed perturbation measurements and performance numbers, see the supplementary material.

Scalability and generality: Our method should scale to very large datasets. The robustness computation and the simplification steps (e.g., **Cut** and **Unwrap**) run in linear time in the size of the mesh. For example, for a region of 21k vertices and 64k edges, **Cut** required 2 seconds in MATLAB and 0.03 seconds in C++. The simplification procedure requires only that the degree of the boundary be zero and so applies to a wide range of cases. It can deal with highly rotational data (e.g., centers) as well as cases where critical points are not connected by separatrices.

Other metrics: We use robustness and the L_∞ norm, but using other metrics such as the L_2-norm, which incorporates both the

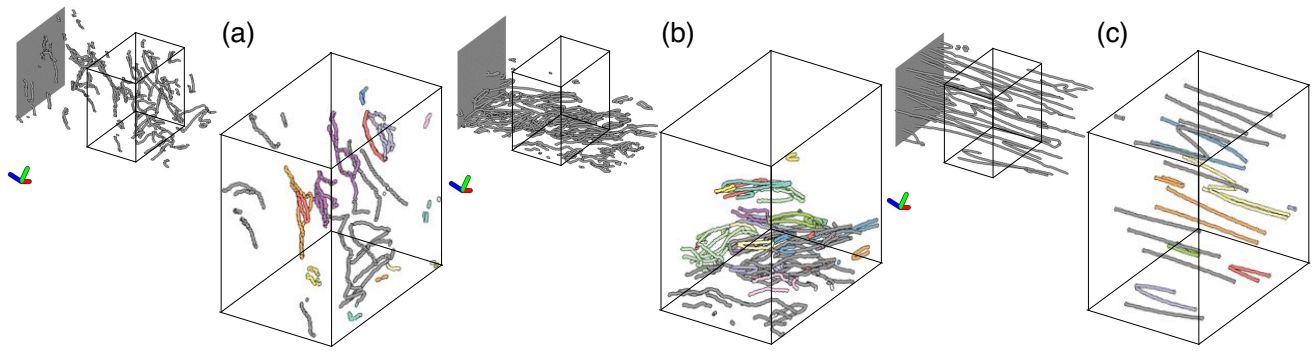

Fig. 22. Time-varying simplification with user-specified time intervals. (a) CentralAtlantic, interval $140-264$; (b) SouthAtlantic, interval $140-264$; (c) Combustion, interval $100-175$.

magnitude of the vectors and the area to capture a quantity closer to the energy of a perturbation, would be interesting. The algorithm only requires components have zero degree. While any metric could be used to construct a hierarchy, it is an open question to find degree-zero regions under different metrics.

3D vector fields: Many of the techniques in this framework extend to 3D vector fields. The robustness computation works for vector fields in any dimension [32], and certain operations (such as cutting and smoothing) can be easily extended. For example, the **Cut** operation involves projection to a plane rather than a line. One obstacle for 3D simplification is performing the Unwrap operation on a 2D sphere. The technical and theoretical obstacles will be addressed thoroughly in a follow-up work.

ACKNOWLEDGMENTS

We thank Jackie Chen for the combustion dataset and Mathew Maltude from LANL and the BER Office of Science UV-CDAT team for the ocean datasets. PR was supported by DOE NETL and KAUST award KUS-C1-016-04. PS was supported by TOPOSYS (FP7-ICT-318493). GC was supported by NSF IIS-1352722. BW was supported by INL 00115847 DE-AC0705ID14517 and DOE NETL.

REFERENCES

[1] J. Helman and L. Hesselink, "Representation and display of vector field topology in fluid flow data sets," *IEEE Computer*, vol. 22, no. 8, pp. 27–36, 1989.
[2] R. Laramee, H. Hauser, L. Zhao, and F. Post, "Topology based flow visualization: state of the art," *Topo. Meth. in Vis.*, pp. 1–19, 2007.
[3] H. Edelsbrunner, D. Letscher, and A. Zomorodian, "Topological persistence and simplification," *DCG*, vol. 28, pp. 511–533, 2002.
[4] J. Reininghaus, N. Kotava, D. Guenther, J. Kasten, H. Hagen, and I. Hotz, "A scale space based persistence measure for critical points in 2d scalar fields," *IEEE TVCG*, vol. 17, no. 12, pp. 2045–2052, 2011.
[5] P. Skraba, B. Wang, G. Chen, and P. Rosen, "2D vector field simplification based on robustness," *IEEE PacificVis*, 2014.
[6] E. Zhang, K. Mischaikow, and G. Turk, "Vector field design on surfaces," *ACM ToG*, vol. 25, pp. 1294–1326, 2006.
[7] K. Polthier and E. Preuß, "Identifying vector fields singularities using a discrete hodge decomposition," *Vis. and Math. III*, pp. 112–134, 2003.
[8] Y. Tong, S. Lombeyda, A. Hirani, and M. Desbrun, "Discrete multiscale vector field decomposition," *ACM ToG*, vol. 22, pp. 445–452, 2003.
[9] R. Westermann, C. Johnson, and T. Ertl, "A level-set method for flow visualization," *IEEE Vis*, pp. 147–154, 2000.
[10] G. Chen, K. Mischaikow, R. Laramee, P. Pilarczyk, and E. Zhang, "Vector field editing and periodic orbit extraction using Morse decomposition," *IEEE TVCG*, vol. 13, no. 4, pp. 769–785, 2007.
[11] W. de Leeuw and R. van Liere, "Collapsing flow topology using area metrics," in *IEEE Vis*, 1999, pp. 349–354.
[12] ——, "Multi-level topology for flow visualization," *Comp. & Graph.*, vol. 24, no. 3, pp. 325–331, 2000.
[13] X. Tricoche, G. Scheuermann, and H. Hagen, "A topology simplification method for 2D vector fields," in *IEEE Vis*, 2000, pp. 359–366.
[14] ——, "Topological visualization of time-dependent 2D vector fields," *EuroVis*, pp. 117–126, 2001.
[15] H. Theisel, C. Rössl, and H.-P. Seidel, "Combining topological simplification and topology preserving compression for 2D vector fields," in *Pacific Graphics*, 2003, pp. 419–423.
[16] G. Chen, Q. Deng, A. Szymczak, R. Laramee, and E. Zhang, "Morse set classification and hierarchical refinement using Conley index," *IEEE TVCG*, vol. 18, no. 5, pp. 767–782, 2012.
[17] G. Chen, K. Mischaikow, R. Laramee, and E. Zhang, "Efficient Morse decompositions of vector fields," *IEEE TVCG*, vol. 14, no. 4, pp. 848–862, 2008.
[18] A. Szymczak and E. Zhang, "Robust morse decompositions of piecewise constant vector fields," *IEEE TVCG*, vol. 18, no. 6, pp. 938–951, 2012.
[19] L. Sipeki and A. Szymczak, "Simplification of Morse decompositions using morse set mergers," *TopoInVis*, pp. 39–53, 2014.
[20] T. Weinkauf, H. Theisel, K. Shi, H.-C. Hege, and H.-P. Seidel, "Extracting higher order critical points and topological simplification of 3d vector fields," in *IEEE Vis*, 2005, pp. 559–566.
[21] X. Tricoche, G. Scheuermann, and H. Hagen, "Continuous topology simplification of planar vector fields," *IEEE Vis*, pp. 159–166, 2001.
[22] G. Chen, V. Kwatra, L.-Y. Wei, C. D. Hansen, and E. Zhang, "Design of 2d time-varying vector fields," *TVCG*, vol. 18, no. 10, pp. 1717–1730, 2012.
[23] J. Reininghaus, J. Kasten, T. Weinkauf, and I. Hotz, "Efficient computation of combinatorial feature flow fields," *IEEE TVCG*, 2011.
[24] J. Reininghaus, C. Lowen, and I. Hotz, "Fast combinatorial vector field topology," *IEEE TVCG*, vol. 17, no. 10, pp. 1433–1443, 2011.
[25] H. Edelsbrunner, J. Harer, and A. Zomorodian, "Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds," *DCG*, vol. 30, pp. 87–107, 2003.
[26] T. Klein and T. Ertl, "Scale-space tracking of critical points in 3D vector fields," *Topo. Meth. in Vis.*, pp. 35–49, 2007.
[27] H. Edelsbrunner and J. Harer, "Jacobi sets of multiple Morse functions," in *Found. of Comp. Math.*, 2004, pp. 37–57.
[28] S. N and V. Natarajan, "Simplification of jacobi sets," *TopoInVis*, pp. 91–102, 2011.
[29] H. Bhatia, B. Wang, G. Norgard, V. Pascucci, and P.-T. Bremer, "Local, smooth, and consistent jacobi set simplification," *Computational Geometry Theory and Applications*, vol. Accepted, 2014.
[30] G. H. Weber, S. E. Dillard, H. Carr, V. Pascucci, and B. Hamann, "Topology-controlled volume rendering," *TVCG*, vol. 13, no. 2, pp. 330–341, 2007.
[31] H. Edelsbrunner, D. Morozov, and A. Patel, "The stability of the apparent contour of an orientable 2-manifold," *TopoInVis*, pp. 27–41, 2010.
[32] F. Chazal, A. Patel, and P. Skraba, "Computing well diagrams for vector fields on R^n," *Applied Math. Letters*, vol. 25, no. 11, pp. 1725–1728, 2012.
[33] H. Edelsbrunner, D. Morozov, and A. Patel, "Quantifying transversality by measuring the robustness of intersections," *Found. of Comp. Math.*, vol. 11, pp. 345–361, 2011.

[34] B. Wang, P. Rosen, P. Skraba, H. Bhatia, and V. Pascucci, "Visualizing robustness of critical points for 2D time-varying vector fields," *CGF*, vol. 32, no. 3, pp. 221–230, 2013.
[35] H. Carr, J. Snoeyink, and U. Axen, "Computing contour trees in all dimensions," *ACM-SIAM SODA*, pp. 918–926, 2000.
[36] M. Maltrud, F. Bryan, and S. Peacock, "Boundary impulse response functions in a century-long eddying global ocean simulation," *Environ. Fluid Mech.*, vol. 10, pp. 275–295, 2010.
[37] E. Hawkes, R. Sankaran, P. Pébay, and J. Chen, "Direct numerical simulation of ignition front propagation in a constant volume with temperature inhomogeneities," *Combust. Flame*, vol. 145, pp. 145–159, 2006.
[38] P. Skraba and B. Wang, "Interpreting feature tracking through the lens of robustness," *TopoInVis*, pp. 19–37, 2014.
[39] F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch, "The state of the art in flow visualisation: Feature extraction and tracking," *CGF*, vol. 22, pp. 775–792, 2003.

Primoz Skraba received his Ph.D. in Electrical Engineering from Stanford University in 2009. He is currently a Senior Researcher at the Jozef Stefan Institute in Slovenia. His main research interests are applications of topology to computer science including data analysis, machine learning, sensor networks, and visualization.

Bei Wang Bei Wang received his Ph.D. in Computer Science from Duke University in 2010. She is currently a Research Scientist at the Scientific Computing and Imaging Institute, University of Utah. Her main research interests are computational topology, computational geometry, scientific data analysis and visualization. She is also interested in computational biology and bioinformatics, machine learning and data mining.

Guoning Chen is an Assistant Professor at the Department of Computer Science at the University of Houston. He earned a PhD degree in Computer Science from Oregon State University in 2009. His research interests include visualization, data analytics, computational topology, geometric modeling, and geometry processing, and physically-based simulation. He is a member of ACM and IEEE.

Paul Rosen is a Research Assistant Professor at the University of Utah with appointments in the Scientific Computing and Imaging (SCI) Institute and the School of Computing. Dr. Rosen received his PhD from the Computer Science Department of Purdue University.Since joining the University of Utah in 2010, Dr. Rosen's research has included topics in scientific visualization, such as vector field and uncertainty visualization, and information visualization.