Mathematics and Visualization

Hamish Carr · Christoph Garth
Tino Weinkauf *Editors*

Topological Methods in Data Analysis and Visualization IV

Theory, Algorithms, and Applications

Springer

Mathematics and Visualization

Hamish Carr • Christoph Garth • Tino Weinkauf
Editors

Topological Methods in Data Analysis and Visualization IV

Theory, Algorithms, and Applications

Springer

Editors
Hamish Carr
University of Leeds
Leeds, United Kingdom

Christoph Garth
Department of Computer Science
Technical University of Kaiserslautern
Kaiserslautern, Germany

Tino Weinkauf
School of Computer Science
and Communication
KTH Royal Institute of Technology
Stockholm, Sweden

ISSN 1612-3786 ISSN 2197-666X (electronic)
Mathematics and Visualization
ISBN 978-3-319-44682-0 ISBN 978-3-319-44684-4 (eBook)
DOI 10.1007/978-3-319-44684-4

Library of Congress Control Number: 2011944972

Mathematics Subject Classification (2010): 00A66, 57-04, 54-04

Cover illustration from Morse-Smale Analysis of Ion Diffusion in Ab Initio Battery Materials Simulations by A. Gyulassy, A. Knoll, K. Chun Lau, B. Wang, P.-T. Bremer, M. E. Papka, L. A. Curtiss and V. Pascucci. By courtesy of the authors.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Since Helman and Hesselink's landmark paper on vector topology in 1989, topological analysis has formed an increasingly important part of scientific visualization. This is not only because it opens up novel forms of understanding but also because as our data has increased past terascale, machine analysis necessarily substitutes for laborious human inspection of visualizations. More and more, one can argue that data analysis precedes rather than succeeds visualization and that topological analysis is one of the key approaches given its strong mathematical underpinnings, precise answers and verifiable outcomes.

From its early starts in vector field topology, topological visualization has expanded to embrace analysis of scalar fields in the form of contour trees, Reeb graphs and Morse-Smale complexes, analysis of abstract graphs and high-dimensional data and, most recently, analysis of multivariate fields through Jacobi sets, Reeb spaces and the joint contour net, linking with the mathematical field of fibre topology in the process.

Topological visualization is, however, not concerned only with the topological computation per se. One of the strongest features of the community is its focus on the full range of theoretical understanding, algorithmic advances and application work, all of which are represented in this volume.

Starting in 2005, biennial workshops have been held on topological visualization in Budmerice (2005), Grimma (2007), Snowbird (2009), Zürich (2011), Davis (2013) and Annweiler (2015), where informal discussions supplement formal presentations and knit the community together. Notably, these workshops have consistently resulted in quality publications under the Springer imprint which form a significant part of the working knowledge in the area.

In the most recent workshop (2015), at Kurhaus Trifels in Annweiler, Germany, bivariate analysis, Reeb spaces and fibre topology increased in importance, anchored by keynotes from Professor Osamu Saeki (Kyushu), one of the leaders in fibre topology, and Professor Kathrin Padberg-Gehle (Lüneburg), who works on computational methods for nonlinear dynamical systems.

Of the 23 papers presented at TopoInVis 2015, 20 passed a second-round review process for this volume. In addition, Professor Saeki contributed a survey of the relevant fibre topology to this volume for the benefit of the community, which we expect to shape approaches to data visualization in future years, and a further paper was contributed directly to this volume.

We have grouped this paper in Part I with the two most closely related papers. Of these, one deals with multi-modal analysis in a particular application domain (atmospheric impacts of volcanic eruptions). The other deals with joint contour nets (a quantized approximation of fibre topology) and their relation to analysis based on Pareto set analysis.

We have then collected papers relating to high-dimensional data in Part II. Here, the first paper applies scalar field topology to optimization problems, based on the common description of optimization as a search landscape. In contrast, the second paper discusses algorithms for computing and visualizing merge trees (one of the principal forms of scalar analysis) in high-dimensional data. These are grouped with a paper that considers the relative quality of different measures applied to reduce the dimensionality of the data.

Part III then collects papers that use scalar topology in relatively low-dimensional spaces (i.e. three-dimensional space). Here, the first paper compares similarity between scalar fields, using histograms as summaries of geometric information to supplement the underlying topological analysis. The second paper is more applied in nature, as it addresses a practical domain problem—how to track diffusion of ions into a battery material, using Morse-Smale analysis, to identify the potential diffusion channels. Lastly, the third paper addresses the inverse problem of (re-)constructing a scalar field from a known Morse-Smale complex.

Where Part III deals with scalar fields, Part IV considers vector and tensor fields. Here, while the broad strokes of the analysis are well-understood, actual computation of topological invariants has a number of practical problems. At the heart of these is the tension between formal mathematical expression of continuous models and practical numerical computation. The papers in this part therefore primarily address issues of discontinuity and degeneracy in the analysis process.

Of these, the first paper deals with issues at the boundary of flow fields through computation of escape maps, while the second computes similarity measures between nearby integral curves to detect regions of shared behaviour. A third paper extends existing ideas for decomposition of vector fields, in order to underpin a future generation of algorithmic approaches, while a fourth paper extends existing mathematical analysis of tensor fields as a preliminary to developing new techniques.

Part V then considers a theme common to many of the newest approaches—indirect detection of topological features to avoid the numerical problems of early methods. Here, the goal is to detect coherent structures in a variety of contexts and use them as the basis of the visualization. The best known techniques for this use finite time Lyapunov exponents (FTLEs), and three of these papers extend these techniques, while the fourth considers related computations.

In the first paper on FTLEs, they are used to detect regions of topological change as a scalar field, which is then subjected to a second round of topological analysis to detect ridge features. The second paper builds on the observation that not all topological boundaries are equally important and maps secondary evaluations to these boundaries to aid in interpretation. The third paper considers an orthogonal but crucial issue—the effect of approximation on this form of analysis. Lastly, the remaining paper considers alternate measures of topological importance, replacing FTLEs with stochastic computations based on transfer operators.

The last part, Part VI, is devoted to papers that are more explicitly about algorithms or software engineering. Here, the first paper is about the selection of thresholds for topological analysis, while the second considers the software instruction necessary for practical deployment of topological techniques. A third paper looks at improving the computation of merge trees in a distributed setting, while the last paper considers knotted graphs, an area of topology not previously represented in the TopoInVis community.

We note that these areas have followed a common pattern in development—initially, there were only one or two papers published on flow topology, but over time, they expanded and triggered the development of the TopoInVis workshop. Later, new techniques were introduced, in particular the detection of coherent structures using finite-time Lyapunov exponents, and we now see this separating as a related but different topic.

Equally, the first TopoInVis workshop did not involve a significant amount of scalar topology, but this area has increased over time and is represented primarily in Parts I and II, since Reeb and Morse analyses are sufficiently well-developed to justify two distinct areas. This growth then triggered developments in analysis of high-dimensional data: hence Part V.

It is therefore encouraging to see the development of fibre topology and multivariate analysis as an emerging theme in topological visualization, as it shows that the pattern continues. Equally, we have started to see work submitted on the peculiar software engineering challenges of topology, and we expect this theme to develop further in future.

As with any workshop, however, the measure of quality is not the breadth of the papers, nor the number of people attending, but whether new ground is being broken. Here, the pattern of development is clear, and we confidently look forward to additional themes emerging in future TopoInVis workshops.

We would like to thank all of the participants in TopoInVis 2015, as well as Springer, for their continued support and look forward to further developments in all of these areas.

Leeds, UK Hamish Carr
Kaiserslautern, Germany Christoph Garth
Stockholm, Sweden Tino Weinkauf

Contents

Part I
Topology-Based Analysis of Multi-Variate Data Sets

Theory of Singular Fibers and Reeb Spaces for Visualization

Osamu Saeki

Abstract This is a survey article on singularity theory of differentiable maps with applications to visualization of scientific data in mind. Special emphasis is put on Morse theory on manifolds with boundary, singular fibers of multi-fields, their Reeb spaces, and their topological transitions.

2000 Mathematics Subject Classification (MSC): Primary 57R45; Secondary 57R35, 58K15, 68U05, 65D18

1 Introduction

This is a survey article on singularity theory of differentiable maps, which focuses on their singular fibers and Reeb spaces. The author will try to explain those materials which may help researchers in the visualization community to use them for their own purposes. Therefore, for many of the statements, theorems, etc., their rigorous proofs are not given in this article, and instead the author will try to give appropriate references. Furthermore, some of the results might have little importance from a mathematical point of view, basically because they are classical and/or well known to mathematicians in general. However, the author will try to include them as long as they can play important roles in visualization of scientific data. Surprisingly, many of the important results that will be explored in this article are quite new in singularity theory. Some of the problems treated in this article are being investigated in singularity theory as central issues. This means that problems in the visualization community may lead to interesting problems or sometimes essential solutions in singularity theory.

The contents of the article are as follows. In Sect. 2, we review the theory of generic scalar fields, namely, Morse functions. As this is widely known to the

O. Saeki (✉)
Institute of Mathematics for Industry, Kyushu University, Motooka 744, Nishi-ku, Fukuoka 819-0395, Japan
e-mail: saeki@imi.kyushu-u.ac.jp

visualization community in general, we will try to cover issues which might not be so popular. Special attention is paid to Morse functions on manifolds with boundary, since in many situations, data sets are given on a bounded domain in Euclidean spaces which have boundary. We also describe the deformation of Morse functions, from the view point of the simplification of the Reeb graph.

In Sect. 3, we review the singularity theory for generic multi-fields, which are called stable maps in singularity theory. We will see that Morse theory can be extended naturally to some dimension pairs, but not to all cases, at least theoretically.

In Sect. 4, the theory of singular fibers of differentiable maps is explained. Mathematically, a fiber is a map around a given pre-image and it contains the information of nearby pre-images. This is why it is important for grasping the topological transitions of pre-images, which is essential in visualization.

In Sect. 5, we explain the concept of Reeb space of a given multi-field. This is the straightforward generalization of Reeb graph for a scalar field. We will see that several structure theorems are already known in singularity theory. In fact, once you have a classification of singular fibers, a structure theorem then follows. Some topological transitions of Reeb spaces are also presented with the simplification of Reeb spaces in mind.

In Sect. 6, we will give several open problems related to singularity theory and visualization of singular fibers. We will also explain how the visualization techniques can be useful in singularity theory itself. We will give several examples of ongoing projects in this direction as well. We end this paper by summarizing the impact of such singularity theoretical results and techniques on computational topology and visualization.

Throughout this paper, all manifolds and maps between them are differentiable of class C^∞ unless otherwise indicated. A manifold is *closed* if it is compact and has no boundary. The symbol D^k denotes the unit disk in $\mathbb{R}^k$.

2 Morse Functions

2.1 Functions on Manifolds Without Boundary

Let us first consider scalar functions on manifolds without boundary. Let N be a closed manifold of dimension n, $n \geq 1$, and consider a smooth scalar function $f : N \to \mathbb{R}$. For a point p, let $df_p : T_pN \to \mathbb{R}$ denote the *differential* of f at p, where T_pN denotes the tangent space of N at p. This is a linear map defined as follows. For a tangent vector $v \in T_pN$, the value $df_p(v)$ is the derivative of f in the direction of v at p. We say that p is a *critical point* of f if df_p is the zero map. This is equivalent to the following: for local coordinates $(x_1, x_2, \ldots, x_n)$ of N around p, we have

$$\frac{\partial f}{\partial x_1}(p) = \frac{\partial f}{\partial x_2}(p) = \cdots = \frac{\partial f}{\partial x_n}(p) = 0.$$

For example, if p attains a local minimum or maximum of f, then it is a critical point, since the partial derivatives of f at such a point necessarily vanish.

Definition 2.1 For a critical point $p \in N$ of f, let us take local coordinates $(x_1, x_2, \dots, x_n)$ around p. Then the $n \times n$ symmetric matrix

$$H_f(p) = \left(\frac{\partial^2 f}{\partial x_i \partial x_j}(p) \right)_{1 \le i,j \le n}$$

is called the *Hessian* of f at p. We say that the critical point p is *non-degenerate* if $\det H_f(p) \neq 0$. It is not difficult to see that this is independent of a choice of local coordinates.

The following lemma is fundamental (for example, see [24, 25]).

Theorem 2.2 (Morse Lemma) *If $p \in N$ is a non-degenerate critical point of f, then there exist local coordinates $(x_1, x_2, \dots, x_n)$ of N around p such that f is locally expressed as*

$$f(x_1, x_2, \dots, x_n) = \pm x_1^2 \pm x_2^2 \pm \cdots \pm x_n^2 + c, \tag{1}$$

where $c = f(p)$ is a constant.

In the above theorem, the number of negative signs appearing in (1) is called the *index* of the critical point p. The Morse Lemma implies that *the differential topological behavior of f around a non-degenerate critical point is completely determined by its index.*

Definition 2.3 A smooth function $f : N \to \mathbb{R}$ is a *Morse function* if its critical points are all non-degenerate, and their f-values are all distinct.

As a corollary to the Morse Lemma, we see that a non-degenerate critical point is isolated in the set of all critical points. In particular, if N is closed, then a Morse function has only finitely many critical points.

Theorem 2.4 *Any smooth function $f : N \to \mathbb{R}$ can be approximated arbitrarily well, including partial derivatives, by a Morse function.*

Thus, in many situations, we may assume, at least theoretically, that a given function is a Morse function. Even if it is not, we can approximate it by a Morse function by perturbing it arbitrarily slightly. For a computational technique for perturbation which is very useful in practical situations, called the *simulation of simplicity*, the reader is referred to [10].

Definition 2.5 Let $f : N \to \mathbb{R}$ be a smooth function. For a real number $r \in \mathbb{R}$, we say that it is a *critical value* of f if there exists a critical point $p \in N$ of f such that $r = f(p)$; otherwise, a *regular value*.

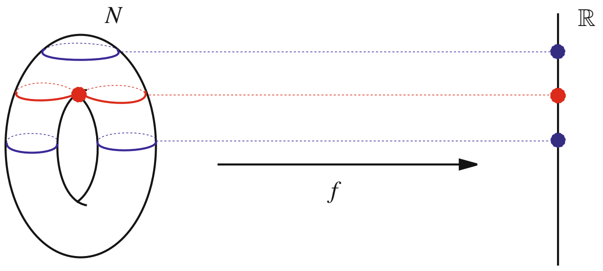


Fig. 1 Example of level sets: one can observe a topological change of level sets as the value in $\mathbb{R}$ passes through a critical value

By the implicit function theorem, we see easily that for a regular value r, the pre-image $f^{-1}(r)$ is a smooth submanifold of dimension $n-1$ of N, as long as it is non-empty (for example, see [26]). Based on this observation, we introduce the following notion.

Definition 2.6 For a real number $r \in \mathbb{R}$, the set

$$f^{-1}(r) = \{p \in N \,|\, f(p) = r\}$$

is called a *level set* of f.

An example of level sets is depicted in Fig. 1.

The following theorem implies that a topological transition of level sets never occurs around a regular value.

Theorem 2.7 *Let $f : N \to \mathbb{R}$ be a smooth function on a closed manifold N. Suppose that the closed interval $[a, b] \subset f(N) \subset \mathbb{R}$ contains no critical value of f. Then, there exists a diffeomorphism $h : f^{-1}([a, b]) \to f^{-1}(a) \times [a, b]$ such that $h(x) = (x, a)$ for all $x \in f^{-1}(a)$, and that the following diagram is commutative:*

$$\begin{tikzcd}
f^{-1}([a,b]) \arrow[rr, "h"] \arrow[dr, "f"'] & & f^{-1}(a) \times [a,b] \arrow[dl, "p_2"] \\
& {[a,b],} &
\end{tikzcd}$$

where p_2 is the projection to the second factor.

Furthermore, the topological transitions of level sets of a Morse function can be described by using the Morse Lemma. If p is a non-degenerate critical point of index λ, then the transition of level sets is locally described around p by

$$-x_1^2 - x_2^2 - \cdots - x_\lambda^2 + x_{\lambda+1}^2 + x_{\lambda+2}^2 + \cdots + x_n^2 = \varepsilon$$

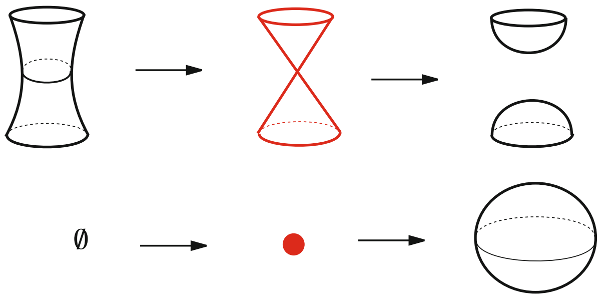


Fig. 2 Example of local level set changes for dimension 3: the *top half* corresponds to the case of a critical point of index 1 or 2, while the *bottom half* corresponds to the case of index 0 or 3

for $\varepsilon \in \mathbb{R}$ with $|\varepsilon|$ sufficiently small. The other parts which sit outside of a neighborhood of p do not change topologically in a sense similar to that in Theorem 2.7. Examples for the case $n = 3$ are depicted in Fig. 2.

Definition 2.8 For a real number $r \in \mathbb{R}$, we call the set

$$N_r = \{p \in N \,|\, f(p) \le r\}$$

a *sub-level set* of f.

Note that if r is a regular value, then the corresponding sub-level set is a smooth manifold whose boundary is the level set.

Suppose that a Morse function f is given. Starting from a real number r_0 strictly less than the minimum of f, let us consider the topological transition of the sub-level sets N_r. Then, according to Theorem 2.7, its topological transition occurs near a real number r only if r is a critical value of f. Furthermore, if p is a critical point with value r, then by using the Morse Lemma, we can show that $N_{r+\varepsilon}$, with $\varepsilon > 0$ sufficiently small, is obtained by attaching a λ-handle to $N_{r-\varepsilon}$, where λ is the index of the critical point p. A λ-*handle* is an n-dimensional disk of the form $D^\lambda \times D^{n-\lambda}$ attached to $\partial N_{r-\varepsilon}$ along $\partial D^\lambda \times D^{n-\lambda}$. In this way, we get a so-called *handle decomposition* of the manifold N [24].

On the other hand, if we look at the transitions of the homotopy types of the sub-level sets, then we get a decomposition of N as a CW complex [25].

For more details about handles, the reader is referred also to [12, Chap. 6]. An application of handle decompositions for morphing 3D shapes has been explored in [35].

Let us now define the Reeb graph of a Morse function.

Definition 2.9 Let $f : N \to \mathbb{R}$ be a Morse function on a closed manifold. Then, each level set of f has finitely many connected components. Contracting each such component to a point, we get a space R_f. More precisely, two points $x, x' \in N$ are equivalent if they lie in the same component of a level set. This is an equivalence relation, and the *quotient space* of N with respect to this equivalence relation is

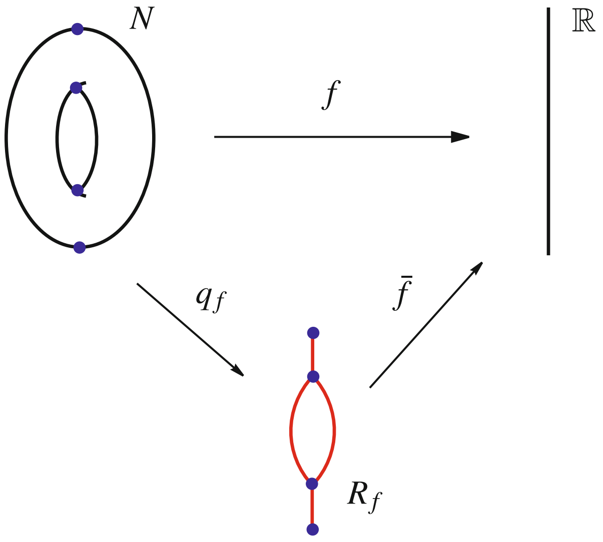


Fig. 3 Reeb graph of the height function f on the torus: the original function is decomposed into the composition of the quotient map q_f and the function $\bar{f}$ defined on the Reeb graph. The vertices of R_f are the q_f-images of the critical points of f

denoted by R_f, which is endowed with the quotient topology of N. Let $q_f : N \to R_f$ denote the quotient map. By definition, we have a natural map $\bar{f} : R_f \to \mathbb{R}$ such that $f = \bar{f} \circ q_f$. Such a decomposition of f is sometimes called a *Stein factorization* [20].

Please note that, by definition, the quotient space R_f is a topological space. In fact, it is not difficult to show, with the help of Theorem 2.7, that R_f is a 1-dimensional cell complex, or a graph: for each critical point p, its image $q_f(p)$ is a vertex of R_f. For this reason, the space R_f is very often called the *Reeb graph* [28]. An example of a Reeb graph is depicted in Fig. 3. We warn the reader that a vertex of R_f may have degree two if the manifold is non-orientable or has dimension greater than or equal to three.

2.2 *Functions on Manifolds with Boundary*

So far, we have considered Morse functions on manifolds without boundary. However, in many practical situations, scalar functions are defined on a manifold with non-empty boundary, such as a region in $\mathbb{R}^n$ with smooth boundary. In this subsection, we review the theory of Morse functions on manifolds with boundary. For details, the reader is referred to [14, Sect. 3] or [2].

Let N be a compact manifold with boundary and $f : N \to \mathbb{R}$ a smooth function. There are two types of critical points. One is a usual critical point, i.e., a point $p \in N$

such that the differential $df_p : T_pN \to \mathbb{R}$ vanishes. We call such a point p a (usual) *critical point* of f. Note that such a point p may lie in the interior as well as the boundary of N in general. The other is a critical point of the restricted function $f_\partial = f|_{\partial N} : \partial N \to \mathbb{R}$. Such a point is called a *boundary critical point* of f. Note that a critical point of the second type necessarily lies on the boundary ∂N. Note also that a usual critical point on ∂N is a boundary critical point, while the converse is not true in general.

The following lemma is well-known.

Theorem 2.10 (Morse Lemma along Boundary) *Let $f : N \to \mathbb{R}$ be a smooth function defined on a manifold with boundary. If $p \in \partial N$ is not a usual critical point of f, but is a non-degenerate critical point of f_∂, then there exist local coordinates $(x_1, x_2, \dots, x_n)$ of N around p such that*

(1) *$\{x_n \geq 0\}$ corresponds to N, and $\{x_n = 0\}$ corresponds to ∂N,*
(2) *f is locally expressed as*

$$f(x_1, x_2, \dots, x_n) = \pm x_1^2 \pm x_2^2 \pm \cdots \pm x_{n-1}^2 \pm x_n + c, \tag{2}$$

where $c = f(p)$ is a constant.

Examples for the case $n = 2$ are depicted in Fig. 4.

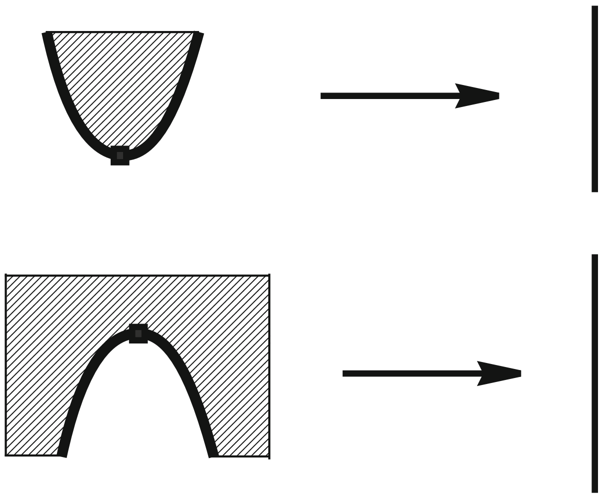


Fig. 4 Boundary critical points of functions defined on surfaces with boundary: the *top half* represents $x_1^2 + x_2$, while the *bottom half* represents $-x_1^2 + x_2$

Definition 2.11 A smooth function $f : N \to \mathbb{R}$ defined on a manifold with boundary is a *Morse function* if

(1) it does not have a usual critical point on the boundary,
(2) every critical point of f in $\mathrm{Int}\, N$ is non-degenerate,
(3) every critical point of $f_\partial = f|_{\partial N}$ is non-degenerate, and
(4) the critical values of f and f_∂ are all distinct.

It is easy to verify that the critical points and the boundary critical points of a Morse function f are all isolated. As we are assuming that N is compact, f has only finitely many critical and boundary critical points.

Then, the same approximation theorem as Theorem 2.4 holds also for functions on compact manifolds with boundary.

A real number r is a *critical value* of f if there exists a critical point or a boundary critical point p of f such that $r = f(p)$. Otherwise, it is called a *regular value*. Note that if r is a regular value, then the level set $f^{-1}(r)$ is a smooth submanifold of N of dimension $n - 1$ with boundary [26]. In this case, we can also show that $f^{-1}(r)$ intersects ∂N transversely along the boundary and that $\partial(f^{-1}(r)) = f^{-1}(r) \cap \partial N$.

Then, the same product theorem as Theorem 2.7 for functions on compact manifolds with boundary also holds.

Such a theorem implies that a topological transition of level sets or sub-level sets occurs only around critical values. Note that these values may be the critical values of the function $f_\partial = f|_{\partial N}$. Therefore, in practical applications, one needs to look for boundary critical points, or sometimes one may need to distinguish usual critical values from boundary critical values. This should be carefully treated, since a topological transition merely implies that the relevant value is either a critical value or a boundary critical value, and in some cases, a boundary critical value may have no importance.

Remark 2.12 As in the case where N has no boundary, if we pass through a usual critical value of a Morse function, then the topology of the sub-level set necessarily changes. On the other hand, if we pass through the value of a boundary critical point, then the topology of the sub-level set changes if and only if the gradient vector of f at the boundary critical point points "inward" [14]. See Fig. 5.

With the help of the product theorem (cf. Theorem 2.7), we can show that the space R_f of a Morse function f defined on a compact manifold with boundary is again a 1-dimensional cell complex, or a graph, which is called the *Reeb graph* of f. The vertices are the q_f-images of critical points and boundary critical points. The structure of such Reeb graphs for scalar functions defined on compact surfaces with boundary is studied in [5, 33].

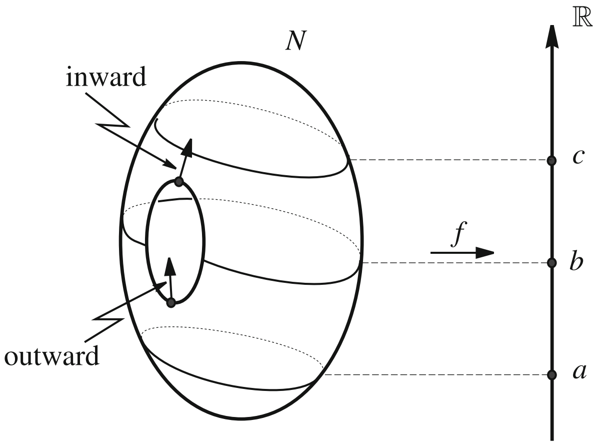


Fig. 5 Topological transition of sub-level sets may not occur even if we pass through a boundary critical value. In this example, $f^{-1}((-\infty, a])$ is diffeomorphic to $f^{-1}((-\infty, b])$, while $f^{-1}((-\infty, b])$ is not diffeomorphic to $f^{-1}((-\infty, c])$

2.3 Deformations of Morse Functions

The following deformations of Morse functions are well known: birth-death of a pair of (usual) critical points, and birth-death of a pair of boundary critical points. We also have a birth-death of a usual critical point near the boundary.

The first one refers to a 1-parameter family $f_t : N \to \mathbb{R}$, $t \in I$, of smooth functions, where $I = (-\varepsilon, \varepsilon)$ and $\varepsilon > 0$ is very small, with the following properties:

1. f_t is a Morse function for $t \neq 0$,
2. there exists a coordinate neighborhood U in N such that f_t does not depend on t on a neighborhood of $N \setminus U$,
3. on a smaller open set V with $\overline{V} \subset U$, f_t is given by

$$f_t(x_1, x_2, \ldots, x_n) = x_1^3 \pm tx_1 \pm x_2^2 \pm \cdots \pm x_n^2$$

with respect to some local coordinates $(x_1, x_2, \ldots, x_n)$.

Mathematically, this can be regarded as a path in the space of functions $C^\infty(N, \mathbb{R})$ which crosses a "codimension 1 non-Morse stratum" transversely at one point. Note that this transition creates a pair of critical points of adjacent indices, or eliminates such a pair.

A birth-death of a pair of boundary critical points is described in a similar fashion. This is locally described by

$$f_t(x_1, x_2, \ldots, x_n) = x_1^3 \pm tx_1 \pm x_2^2 \pm \cdots \pm x_{n-1}^2 \pm x_n,$$

where $\{x_n \geq 0\}$ corresponds to N and $\{x_n = 0\}$ corresponds to ∂N.

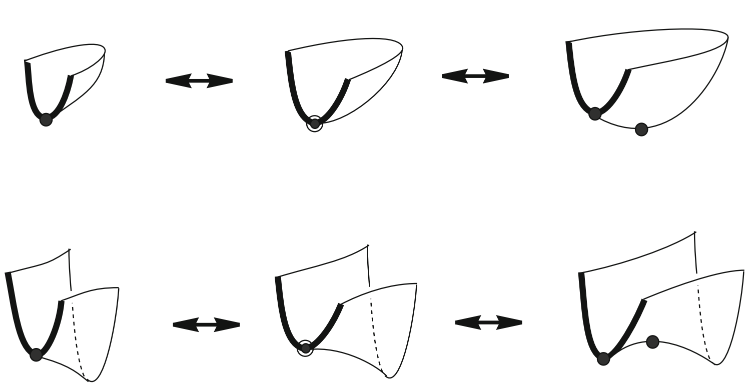


Fig. 6 Birth-death of a usual critical point near the boundary: in the middle figures, the encircled *dots* are usual critical points and are, at the same time, boundary critical points. The two functions on the *left* have only a boundary critical point, while the two on the *right* have both a boundary critical point and a usual critical point in the interior

A birth-death of a usual critical point near the boundary is locally described by

$$f_t(x_1, x_2, \dots, x_n) = \pm x_1^2 \pm x_2^2 \pm \cdots \pm x_{n-1}^2 + tx_n \pm x_n^2.$$

For $n = 2$, this deformation is locally depicted in Fig. 6.

We have other types of generic transitions: crossings of critical values. These deformations do not create nor eliminate critical points, but they interchange the values of two (usual or boundary) critical points.

In the course of the above deformations, a topological transition of Reeb graphs may occur as follows if the manifold has no boundary (see also [7]).

Theorem 2.13 *If a topological transition of the Reeb graphs for a generic deformation of Morse functions on a closed manifold of dimension $n \geq 2$ occurs, then it is one of the transitions locally described in Fig. 7 (or their upside down versions).*

Note that in Fig. 7, (1) and (2) correspond to birth-death of a pair of critical points, where in (1), one of the critical points is a local minimum or a local maximum. The other three correspond to crossings of critical values. Note also that the indices of the relevant critical points have certain restrictions. For example, for (1) and (2), their indices are adjacent, while for (4), they must be 1 and $n - 1$.

The reader should be careful, since not all transitions are realizable. For example, the transition (2) from the right to left is always possible if $n \geq 3$. More precisely, given a Morse function whose Reeb graph has a subgraph as in the right hand side of (2), then one can construct a generic 1-parameter deformation of the given function

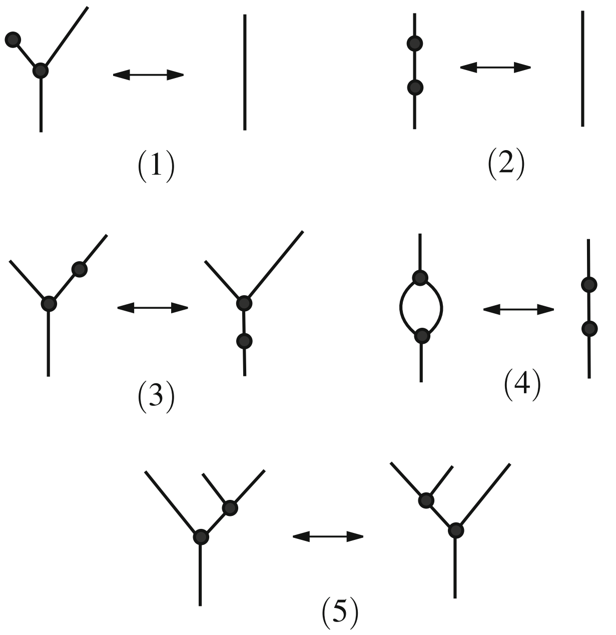


Fig. 7 Possible transitions of Reeb graphs for generic 1-parameter deformations of Morse functions on closed manifolds near a given parameter: these are local descriptions, and the part lying outside of these graphs does not change during the deformation

to another Morse function whose Reeb graph is locally of the form as in the left hand side of (2). However, for the resulting graph, the transition (4) from the right to left cannot be applied. This is because after the birth of a pair of critical points, they are so involved with each other that their values cannot be interchanged to make a crossing of critical values.

We can also prove that the transition of Fig. 7 (1) is always possible. More precisely, if the Reeb graph of a Morse function contains one of the two graphs in the figure as a subgraph, then we can deform the given Morse function by passing through a birth-death exactly once so that the resulting Morse function has the Reeb graph obtained from the original one by replacing the subgraph with the other graph in the figure. This gives a theoretical justification for simplification of a Reeb graph for visualization purposes that uses the topological transition described in Fig. 7 (1).

The above theorem can be proved by using a result on local structures of Reeb spaces of generic maps into the plane [19]. (For the definition of a Reeb space, refer to Sect. 5.1.)

3 Stable Maps

3.1 Notion of Stable Maps

Let N be a smooth manifold and $f : N \to \mathbb{R}^m$ a *multi-field*: in other words, we have a set of m scalar functions $f_i : N \to \mathbb{R}$, $i = 1, 2, \ldots, m$, such that $f = (f_1, f_2, \ldots, f_m)$. In this section, we survey the singularity theory which studies such a set of m smooth functions at the same time, and not individually.

For a straightforward generalization of the theory of Morse functions to the case of smooth maps into $\mathbb{R}^m$, we need a theorem similar to the Morse Lemma. However, in singularity theory of differentiable maps, it is known to be very difficult in general. Therefore, we adopt the following definition.

Definition 3.1 Denote by $C^\infty(N, \mathbb{R}^m)$ the set of C^∞ maps $N \to \mathbb{R}^m$ equipped with the Whitney C^∞ topology (for details, see [13]). A smooth map $f : N \to \mathbb{R}^m$ is called a C^∞ *stable map*, or a *stable map* for short, if there exists a neighborhood $U(f) \subset C^\infty(N, \mathbb{R}^m)$ of f such that every map $g \in U(f)$ is C^∞ equivalent to f [13], where two maps f and $g \in C^\infty(N, \mathbb{R}^m)$ are C^∞ *equivalent* if there exist diffeomorphisms $\Psi : N \to N$ and $\psi : \mathbb{R}^m \to \mathbb{R}^m$ such that $f \circ \Psi = \psi \circ g$. This means that even if one perturbs a stable map slightly, we end up with a map which behaves exactly the same as the original map up to diffeomorphisms of the domain and the range. This is the origin of the terminology "stable".

We can also define the notion of a C^0 *stable map* by replacing diffeomorphisms by homeomorphisms in the above definition.

Let $S^\infty(N, \mathbb{R}^m)$ (or $S^0(N, \mathbb{R}^m)$) be the subspace of $C^\infty(N, \mathbb{R}^m)$ that consists of all C^∞ (resp. C^0) stable maps. By definition, both of them form open subsets of $C^\infty(N, \mathbb{R}^m)$. Mather [22] showed that if N is compact, then $S^\infty(N, \mathbb{R}^m)$ is dense in $C^\infty(N, \mathbb{R}^m)$ if and only if the dimension pair (n, m) lies in the so-called *nice range*. In other words, if (n, m) is in the nice range, then every smooth map $N \to \mathbb{R}^m$ can be approximated arbitrarily well by a C^∞ stable map. On the other hand, Mather [23] showed that if N is compact, $S^0(N, \mathbb{R}^m)$ is always dense in $C^\infty(N, \mathbb{R}^m)$. Hence, every map can always be approximated by a C^0 stable map.

For example, for $(n, m) = (8, 6)$, $S^\infty(N, \mathbb{R}^m)$ is never dense in $C^\infty(N, \mathbb{R}^m)$. Remarkably, for the 4-dimensional complex projective space $\mathbb{C}P^4$ viewed as a real 8-dimensional manifold, there exists no C^∞ stable map $\mathbb{C}P^4 \to \mathbb{R}^6$ [1]. In fact, there are uncountably many equivalence classes of "generic" singularities for these dimensions, and a theorem like the Morse Lemma cannot be expected.

On the other hand, it is known that every dimension pair (n, m) with $m \leq 5$ or $m \geq 2n - 1$ is in the nice range (see [13, 22]).

In the following, for a smooth map $f : N \to \mathbb{R}^m$, we denote by $S(f)$ the set of points $p \in N$ such that the differential $df_p : T_pN \to T_{f(p)}\mathbb{R}^m$ has rank strictly less than $\min\{n, m\}$, where the *differential* is the linear map associated with the $m \times n$ Jacobian matrix of f at p with respect to some local coordinates around p and $f(p)$. The set $S(f)$ is called the *singular point set* or the *Jacobi set* of f, and a point in $S(f)$ is called a *singular point* of f. The singular point set $S(f)$ is often denoted by $J(f)$ in the visualization community (see [6]). Note that when $n \geq m$, the notation $J(f)$ encodes the behavior of multiple Morse functions in the sense that it collects the points where the gradient of the m Morse functions are linearly dependent.

3.2 Characterization of Stable Maps for Specific Cases

Although theorems like the Morse Lemma do not exist in general for multi-fields, for some specific dimension pairs (n, m), we do have such theorems. In the following, if we say that a map is stable, then it means that it is C^∞ stable.

In the function case, the following is known.

Theorem 3.2 *Let N be a closed n-dimensional manifold, $n \geq 1$. Then a smooth function $f : N \to \mathbb{R}$ is stable if and only if it is a Morse function. In particular, the dimension pair $(n, 1)$ is always in the nice range.*

The above theorem means that the notion of a stable map generalizes the notion of a Morse function in a reasonable sense.

Let us introduce the following notion.

Definition 3.3 Let $f_i : N_i \to \mathbb{R}^m$, $i = 0, 1$, be smooth maps with $\dim N_0 = \dim N_1 = n$. For singular points $p_i \in N_i$ of f_i, $i = 0, 1$, we define that they have the same *singularity type* if for some open neighborhoods U_i of p_i and V_i of $f_i(p_i)$ and diffeomorphisms $\Psi : U_0 \to U_1$ and $\psi : V_0 \to V_1$ with $\Psi(p_0) = p_1$ and $\psi(f_0(p_0)) = f_1(p_1)$ such that the following diagram is commutative:

$$\begin{tikzcd}
U_0 \arrow[r, "\Psi"] \arrow[d, "f_0"'] & U_1 \arrow[d, "f_1"] \\
V_0 \arrow[r, "\psi"] & V_1.
\end{tikzcd}$$

Note that the Morse Lemma claims that a non-degenerate critical point of a function has the same singularity type as the critical point of a quadratic function $\pm x_1^2 \pm x_2^2 \pm \cdots \pm x_n^2$.

Let us now consider the case $m = 2$. Let $f : N \to \mathbb{R}^2$ be a smooth map of a closed n-dimensional manifold, $n \geq 2$.

Definition 3.4 A point $p \in S(f)$ is a *fold point* if f can be expressed by

$$f(x_1, x_2, \ldots, x_n) = (x_1, \pm x_2^2 \pm x_3^2 \pm \cdots \pm x_n^2)$$

with respect to appropriate local coordinates around p and $f(p)$. In other words, p has the same singularity type as the above polynomial map. Similarly, a point $p \in S(f)$ is a *cusp point* if it has the same singularity type as the map

$$(x_1, x_2, \ldots, x_n) \mapsto (x_1, \pm x_2^3 + x_1x_2 \pm x_3^2 \pm \cdots \pm x_n^2).$$

We denote by $F(f)$ the set of fold points of f, and by $C(f)$ the set of cusp points. It is easy to verify that $F(f)$ is a smooth 1-dimensional submanifold of N, while $C(f)$ is a discrete set of points.

For the case of $n = 2$, see Fig. 8. See also [9].

Then the following characterization of stable maps is known.

Theorem 3.5 (Whitney [38]) *A smooth map $f : N \to \mathbb{R}^2$ is stable if and only if the following conditions are satisfied.*

(1) *Every singular point is either a fold point or a cusp point.*
(2) *The restriction $f|_{F(f)} : F(f) \to \mathbb{R}^2$ is an immersion (i.e. a non-singular curve) with normal crossings: i.e. for every point $q \in \mathbb{R}^2$, the pre-image $(f|_{F(f)})^{-1}(q)$ consists of at most two points, and if it consists of two points, then the images of the differentials at the two points are linearly independent in $T_q\mathbb{R}^2$.*
(3) $f(F(f)) \cap f(C(f)) = \emptyset$.
(4) *The restriction $f|_{C(f)}$ is injective.*

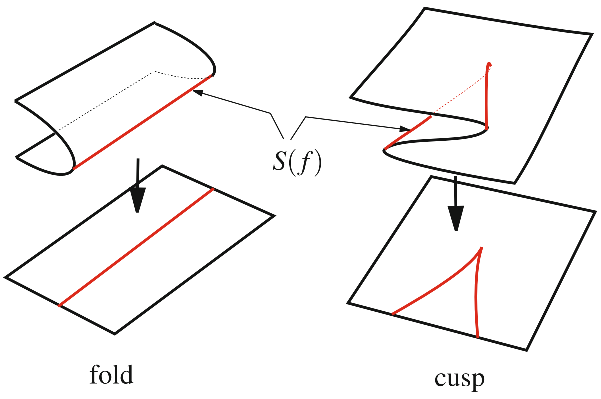


Fig. 8 Fold and cusp points for the case $n = 2$: these are the singularities that can appear for stable maps of surfaces into $\mathbb{R}^2$

Let us now consider the case $m = 3$. Let $f : N \to \mathbb{R}^3$ be a smooth map of a closed n-dimensional manifold, $n \geq 3$.

Definition 3.6 A point $p \in S(f)$ is a *fold point* if it has the same singularity type as the map

$$(x_1, x_2, \dots, x_n) \mapsto (x_1, x_2, \pm x_3^2 \pm x_4^2 \pm \cdots \pm x_n^2).$$

A point $p \in S(f)$ is a *cusp point* if it has the same singularity type as

$$(x_1, x_2, \dots, x_n) \mapsto (x_1, x_2, \pm x_3^3 + x_2 x_3 \pm x_4^2 \pm \cdots \pm x_n^2).$$

A point $p \in S(f)$ is a *swallowtail point* if it has the same singularity type as

$$(x_1, x_2, \dots, x_n) \mapsto (x_1, x_2, \pm x_3^4 + x_1 x_3^2 + x_2 x_3 \pm x_4^2 \pm \cdots \pm x_n^2).$$

We denote by $F(f)$ the set of fold points of f, by $C(f)$ the set of cusp points, and by $ST(f)$ the set of swallowtail points. It is easy to verify that $F(f)$ and $C(f)$ are smooth 2- and 1-dimensional submanifolds of N, respectively, while $ST(f)$ is a discrete set of points.

Theorem 3.7 *A smooth map $f : N \to \mathbb{R}^3$ of a closed n-dimensional manifold N, $n \geq 3$, is stable if and only if the following conditions are satisfied.*

(i) *Every singular point is either a fold point, a cusp point, or a swallowtail point.*
(ii) *The singular point set $S(f)$ is a smooth 2-dimensional submanifold of N under the above condition. Then, for every $r \in f(S(f))$, $f^{-1}(r) \cap S(f)$ consists of at most three points and the map $f|_{S(f)}$ around $f^{-1}(r) \cap S(f)$ is equivalent to one of the six maps whose images are as described in Fig. 9: (1), (2) and (4) correspond to 1, 2 or 3 fold sheets, respectively, (3) corresponds to a cusp point, (5) represents a transverse crossing of a cuspidal edge as in (3) and a fold sheet, and (6) corresponds to a swallowtail point.*

4 Singular Fibers

4.1 Concept

Let $f : N \to \mathbb{R}^m$ be a smooth map of a closed n-dimensional manifold, $n \geq m \geq 1$. In this section, we mainly consider the case with $m \geq 2$, i.e. the case of a multi-field.

Definition 4.1 For a point $r \in \mathbb{R}^m$, the set $f^{-1}(r) = \{p \in N \,|\, f(p) = r\}$ is called the *fiber* of f over r. In particular, when $m = 1$, this notion coincides with that of a level set.

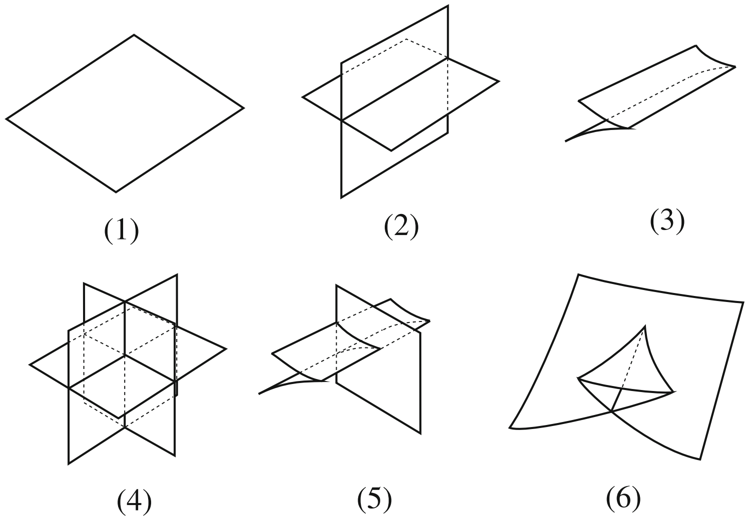


Fig. 9 Possible local configurations of the image of $f|_{S(f)}$ in $\mathbb{R}^3$ for a stable map $f : N \to \mathbb{R}^3$ of a closed n-dimensional manifold N, $n \geq 3$

In fact, in singularity theory, we use the terminology "fiber" in such a way that it contains more information than just the pre-image as follows [29].

Definition 4.2 Let $f_i : N_i \to \mathbb{R}^m$ be smooth maps of n-dimensional manifolds, $i = 0, 1$. For $r_i \in \mathbb{R}^m$, we say that the fibers over r_0 and r_1 of f_0 and f_1, respectively, are *equivalent* if for some open neighborhoods U_i of r_i in $\mathbb{R}^m$, there exist diffeomorphisms $\Phi : (f_0)^{-1}(U_0) \to (f_1)^{-1}(U_1)$ and $\varphi : U_0 \to U_1$ with $\varphi(r_0) = r_1$ which make the following diagram commutative:

$$\begin{tikzcd}
((f_0)^{-1}(U_0), (f_0)^{-1}(r_0)) \arrow[r, "\Phi"] \arrow[d, "f_0"'] & ((f_1)^{-1}(U_1), (f_1)^{-1}(r_1)) \arrow[d, "f_1"] \\
(U_0, r_0) \arrow[r, "\varphi"] & (U_1, r_1).
\end{tikzcd}$$

When $r \in \mathbb{R}^m$ is a regular value of f, we call $f^{-1}(r)$ a *regular fiber*; otherwise, a *singular fiber*.

For example, the following is well known [29].

Theorem 4.3 *Let $f : N \to \mathbb{R}$ be a Morse function on a closed surface N. Then the fiber over each critical value in $\mathbb{R}$ is equivalent to one of the three types of fibers as depicted in Fig.* 10.

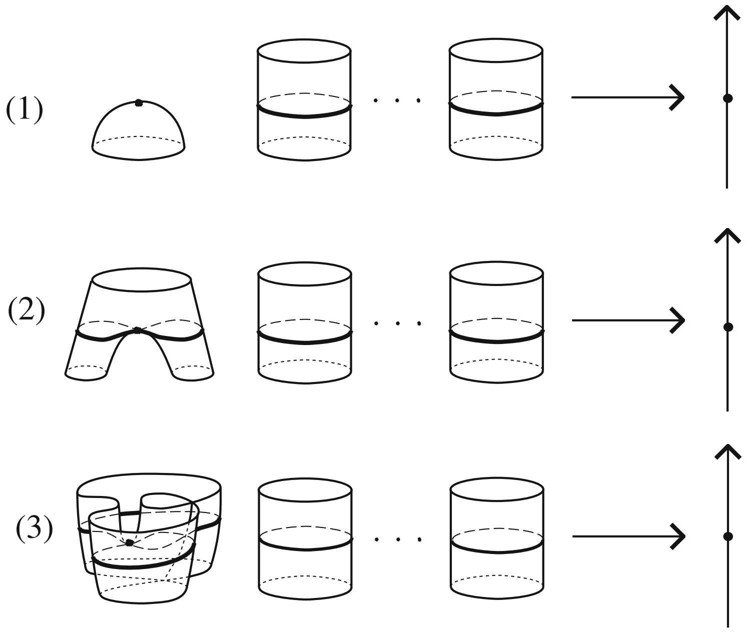


Fig. 10 List of equivalence classes of singular fibers for Morse functions on closed surfaces. For each horizontal arrow, the *left* hand side depicts the neighborhood of the pre-image of a critical value, and the arrow represents the map as a height function. Thus, for example, the function in (1) has exactly one local extremal point as a critical point, but the corresponding pre-image may not be connected and may have several circle components consisting of regular points whose neighborhoods are diffeomorphic to a cylinder. The component of the neighborhood containing a critical point is diffeomorphic to a disk for (1), a 2-sphere with three disks removed for (2), and a Möbius band with a disk removed for (3). In particular, singular fibers of type (3) never occur if the domain surface is orientable

4.2 Ehresmann Fibration Theorem

In the following, a map is *proper* if the pre-image of a compact set is always compact.

Theorem 4.4 (Ehresmann Fibration Theorem [11]) *Let $f : N \to \operatorname{Int} D^m$ be a proper submersion of an n-dimensional manifold N (possibly with boundary) into the interior of the m-dimensional disk with $n \geq m$ such that $f|_{\partial N} : \partial N \to \operatorname{Int} D^m$ is also a submersion if $\partial N \neq \emptyset$. Then for the center 0 of $\operatorname{Int} D^m$, the pre-image $f^{-1}(0)$ is a compact $(n-m)$-dimensional manifold with boundary, and for an arbitrary diffeomorphism $h : f^{-1}(0) \to F$ onto a manifold F, there exists a diffeomorphism*

$\tilde{h} : N \to F \times \operatorname{Int} D^m$ such that the diagram

$$\begin{tikzcd}
N \arrow[rr, "\tilde{h}"] \arrow[dr, "f"'] & & F \times \operatorname{Int} D^m \arrow[dl, "p_2"] \\
& \operatorname{Int} D^m &
\end{tikzcd}$$

is commutative and that $\tilde{h}|_{f^{-1}(0)} = h : f^{-1}(0) \to F \times \{0\}$, where $p_2 : F \times \operatorname{Int} D^m \to \operatorname{Int} D^m$ is the projection to the second factor.

Note that Theorem 2.7 can be regarded as a corollary to the above theorem.

The above Ehresmann Fibration Theorem implies that the equivalence class of a regular fiber is completely determined by the diffeomorphism type of the pre-image submanifold.

Example 4.5 Let us consider the case with $n = 3$ and $m = 2$. If f is a submersion, then its central fiber $f^{-1}(0)$ is a compact 1-dimensional manifold possibly with boundary. Thus, in general, it is a disjoint union of circles and arcs. Suppose $f^{-1}(0)$ is an arc. Then, the Ehresmann Fibration Theorem implies that all fibers of f are arcs. Moreover, the map f is equivalent to the projection $[0, 1] \times \operatorname{Int} D^2 \to \operatorname{Int} D^2$. See Fig. 11.

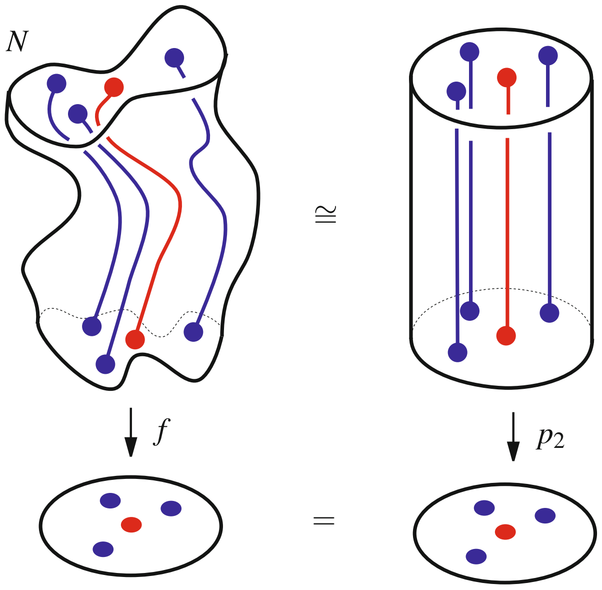


Fig. 11 Example of a submersion of a compact 3-dimensional manifold into $\operatorname{Int} D^2$ with an arc central fiber. All the fibers are arcs, and they are situated in N as a "trivial family" parameterized by $\operatorname{Int} D^2$

Definition 4.6 A smooth map as in Theorem 4.4, or in other words, a map which is equivalent to the projection $p_2 : F \times \operatorname{Int} D^m \to \operatorname{Int} D^m$ is called a *trivial bundle* or a *trivial F-bundle*. For example, a map as in Example 4.5 is a trivial arc bundle.

Remark 4.7 Here is a very important remark for the visualization purpose. Let $f : N \to \mathbb{R}^m$ be a smooth map of a compact n-dimensional manifold, $n \geq m \geq 1$. Then, $\Sigma(f) = f(S(f)) \cup f(S(f|_{\partial N}))$ divides the range $\mathbb{R}^m$ into some regions. The Ehresmann Fibration Theorem implies that over each of these regions, the behavior of f is constant. This is a very important observation from the visualization viewpoint:

Topological transitions of fibers occur only along the Jacobi set image $\Sigma(f)$. Over each point of $\Sigma(f)$ lies a singular fiber of f. That is why singular fibers are important in grasping the topological features of a given multi-field.

4.3 Classification Results

In this section, let us review some known classification results for singular fibers of stable maps.

Definition 4.8 Let $f : N \to \mathbb{R}^m$ be a proper smooth map of a manifold of dimension n with $n \geq m \geq 1$. For a positive integer ℓ, we call the map

$$f \times \mathrm{id}_{\mathbb{R}^\ell} : N \times \mathbb{R}^\ell \to \mathbb{R}^m \times \mathbb{R}^\ell$$

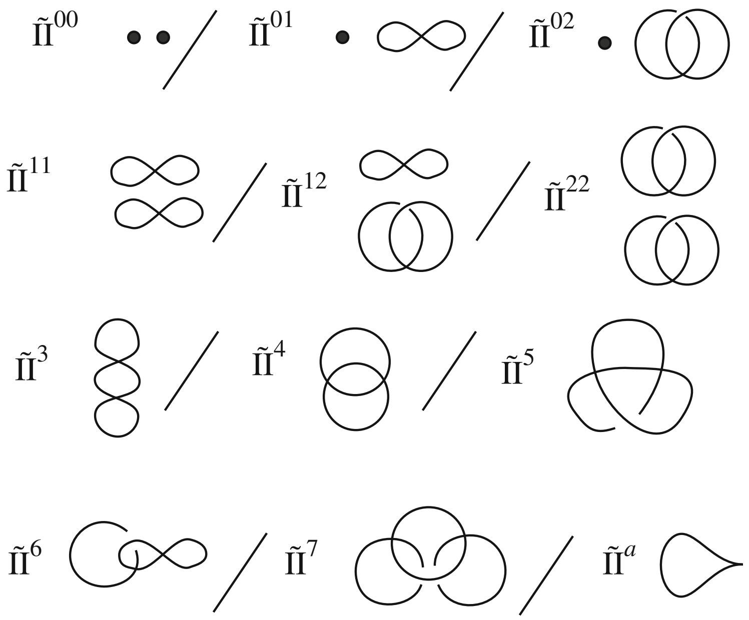
the *ℓ-th suspension* of f, where $\mathrm{id}_{\mathbb{R}^\ell}$ is the identity map of $\mathbb{R}^\ell$. When $\ell = 1$, we sometimes call it the *suspension* of f. Furthermore, to the fiber of f over a point $r \in \mathbb{R}^m$, we can associate the fiber of $f \times \mathrm{id}_{\mathbb{R}^\ell}$ over $r \times \{0\}$. We say that the latter fiber is obtained from the original fiber by the *ℓ-th suspension*.

The following list of singular fibers is obtained in [29].

Theorem 4.9 *Let $f : N \to \mathbb{R}^2$ be a stable map of a closed 3-dimensional manifold N. Then, every singular fiber of f is equivalent to the suspension of a fiber as appearing in Fig. 10, or a disjoint union of one of the fibers as in Fig. 12 and a finite number of copies of a fiber of the trivial circle bundle.*

In Fig. 12, “II” means that they have codimension 2, which refers to the codimension of the set of points in $\mathbb{R}^2$ whose corresponding fibers are equivalent to the relevant one. The tilde symbol means that non-orientable domain is also considered. Each digit 0–7 corresponds to a connected fiber and if a superscript consists of two digits, then it means that it has two components corresponding to the two digits.

Note that a part of the above list is mentioned in the introduction of [20].



$\tilde{\mathrm{II}}^{00}$
$\tilde{\mathrm{II}}^{01}$
$\tilde{\mathrm{II}}^{02}$
$\tilde{\mathrm{II}}^{11}$
$\tilde{\mathrm{II}}^{12}$
$\tilde{\mathrm{II}}^{22}$
$\tilde{\mathrm{II}}^{3}$
$\tilde{\mathrm{II}}^{4}$
$\tilde{\mathrm{II}}^{5}$
$\tilde{\mathrm{II}}^{6}$
$\tilde{\mathrm{II}}^{7}$
$\tilde{\mathrm{II}}^{a}$

Fig. 12 List of singular fibers of stable maps of closed 3-dimensional manifolds into $\mathbb{R}^2$. Each figure symbolically depicts the relevant pre-image in the domain 3-manifold: however, it also represents the map into $\mathbb{R}^2$ defined on a neighborhood of the pre-image. For example, the domain X of the map corresponding to $\tilde{\mathrm{II}}^{00}$ is the disjoint union $(D_1^2\times[-1,1])\cup(D_2^2\times[-1,1])$, and the relevant map $X \to [-1,1]\times[-1,1] \subset \mathbb{R}^2$ is given by $(x,y,z)\mapsto(-x^2-y^2,z)$ for $(x,y,z)\in D_1^2\times[-1,1]$, and by $(x,y,z)\mapsto(z,-x^2-y^2)$ for $(x,y,z)\in D_2^2\times[-1,1]$

Example 4.10 (1) Let us consider the singular fiber $\tilde{\mathrm{II}}^4$. First recall that by Theorem 3.5, the singular value set $f(S(f))$ is a curve possibly with double points and cuspidal points. A fiber of type $\tilde{\mathrm{II}}^4$ corresponds to a double point, say $r \in \mathbb{R}^2$. Thus, $f^{-1}(r)$ contains two fold points. Furthermore, they belong to the same component. If one of them is a definite fold point, then it forms a whole connected component of $f^{-1}(r)$, which is a contradiction. Therefore, both of them must be indefinite fold points. Therefore, near each of them, f is locally given by $(x_1,x_2,x_3)\mapsto(x_1,x_2^2-x_3^2)$. Hence, the pre-image $f^{-1}(r)$ is locally given by the equation $x_1=0$ and $(x_2+x_3)(x_2-x_3)=0$, which consists of two line segments intersecting at the singular point. Since two fold points lie on the same connected component, these two sets of "crossings" must be connected by curves. In our case of $\tilde{\mathrm{II}}^4$-type fiber, the result is as depicted in Fig. 12.

Let us consider the connected component K of $f^{-1}(U)$ containing the singular fiber, where U is a small disk neighborhood of r. Fig. 13 depicts the behavior of the map $f|_K : K \to U$. It depicts U together with the singular value set $f(S(f))$ in red, which locally divides the range U into four quadrants. Over the crossing of $f(S(f))$, which is nothing but the point r, we have a fiber of type $\tilde{\mathrm{II}}^4$. Furthermore, over each

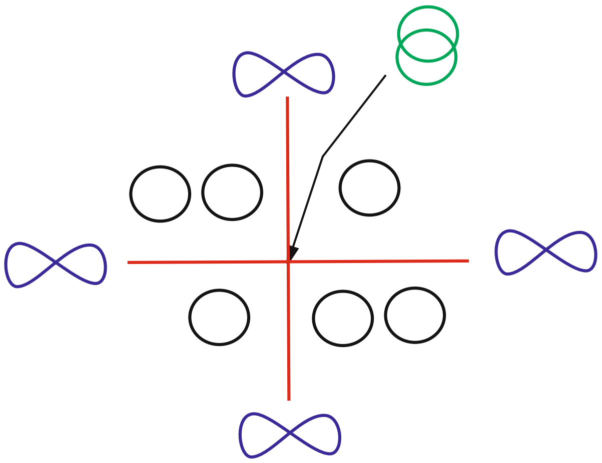


Fig. 13 Fiber over each part of $f(S(f))$ and $\mathbb{R}^2 \setminus f(S(f))$: the central fiber is of type $\tilde{\mathrm{II}}^4$

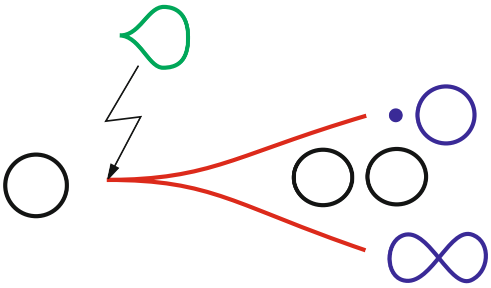


Fig. 14 Fiber over each part of $f(S(f))$ and $\mathbb{R}^2 \setminus f(S(f))$: the central fiber is of type $\tilde{\mathrm{II}}^a$

point of $f(S(f)) \setminus \{r\}$, we have the "figure eight fiber", which is the suspension of the fiber as in Fig. 10 (2). Finally, over each point of the four neighboring quadrants, we have the regular fiber consisting of one or two circles.

This means that if we know that a given map has a singular fiber of type $\tilde{\mathrm{II}}^4$, then we also know its neighboring fibers as well. In other words, topological transitions of fibers are encoded in each member of the classification list.

(2) The neighboring fibers around a fiber of type $\tilde{\mathrm{II}}^a$ are as depicted in Fig. 14.

As to maps of manifolds of dimension 4 to $\mathbb{R}^3$, we have the following [29].

Theorem 4.11 *Let $f : N \to \mathbb{R}^3$ be a stable map of a closed orientable 4-dimensional manifold N. Then, every singular fiber of f is equivalent to the disjoint union of one of the fibers as in Fig. 15 and a finite number of copies of a fiber of the trivial circle bundle.*

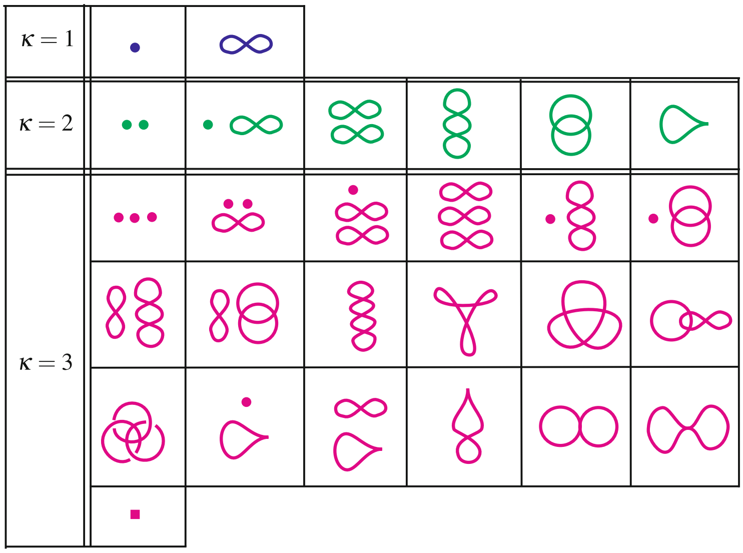


Fig. 15 List of singular fibers of stable maps of closed orientable 4-dimensional manifolds into $\mathbb{R}^3$

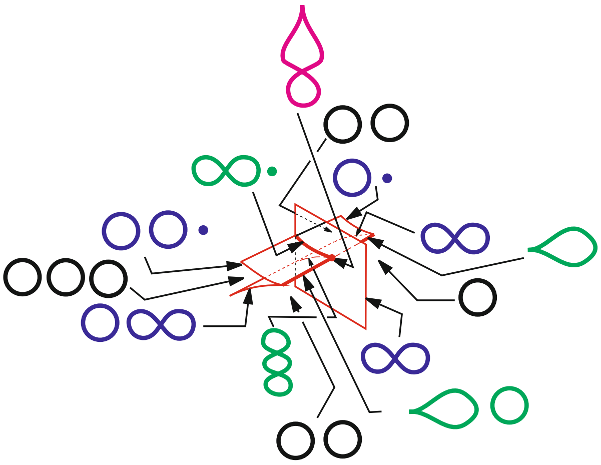


Fig. 16 Example of neighboring fibers of a specific singular fiber of codimension three for a stable map $f : N \to \mathbb{R}^3$ of a closed 4-dimensional manifold N

In Fig. 15, κ denotes the codimension of the set of points in $\mathbb{R}^3$ whose corresponding fibers are equivalent to the relevant one.

An example of the neighboring fibers of a singular fiber of codimension three is depicted in Fig. 16.

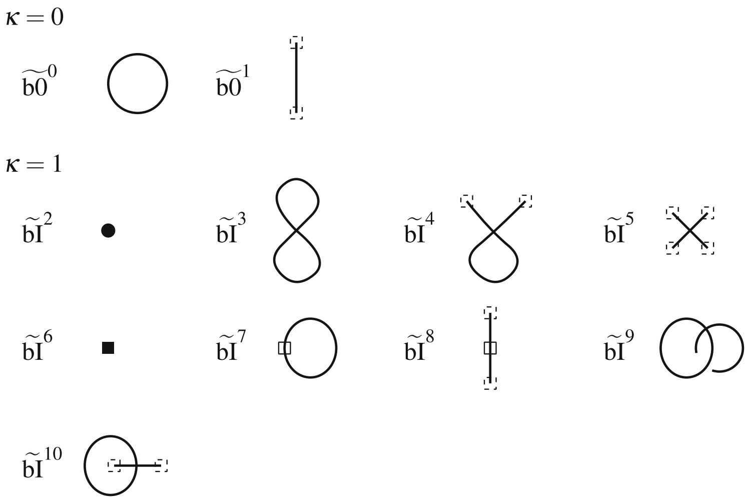

$\kappa = 0$
$\widetilde{\mathrm{b}0}^0$
$\widetilde{\mathrm{b}0}^1$
$\kappa = 1$
$\widetilde{\mathrm{bI}}^2$
$\widetilde{\mathrm{bI}}^3$
$\widetilde{\mathrm{bI}}^4$
$\widetilde{\mathrm{bI}}^5$
$\widetilde{\mathrm{bI}}^6$
$\widetilde{\mathrm{bI}}^7$
$\widetilde{\mathrm{bI}}^8$
$\widetilde{\mathrm{bI}}^9$
$\widetilde{\mathrm{bI}}^{10}$

Fig. 17 List of fibers for stable maps of compact 3-dimensional manifolds with boundary into $\mathbb{R}^2$ (1): squares correspond to boundary; *dotted* ones represent transverse intersections with boundary, while *thick open* ones represent tangencies. *Black solid square* represents the suspension of the first fiber depicted in Fig. 4

For maps of 3-dimensional manifolds with boundary into $\mathbb{R}^2$, we have the following [32].

Theorem 4.12 *Let $f : N \to \mathbb{R}^2$ be a stable map of a compact 3-dimensional manifold N with boundary. Then, every fiber of f is C^∞ equivalent to the disjoint union of one of the fibers in the following list and a finite number of copies of a fiber of the trivial circle or arc bundle:*

(1) *fibers as depicted in Fig. 17, i.e.* $\widetilde{\mathrm{b}0}^0$, $\widetilde{\mathrm{b}0}^1$, *and* $\widetilde{\mathrm{bI}}^\mu$ *with* $2 \le \mu \le 10$,
(2) *fibers* $\widetilde{\mathrm{bII}}^{\mu,\nu}$ *with* $2 \le \mu \le \nu \le 10$, *where* $\widetilde{\mathrm{bII}}^{\mu,\nu}$ *means the disjoint union of* $\widetilde{\mathrm{bI}}^\mu$ *and* $\widetilde{\mathrm{bI}}^\nu$,
(3) *fibers as depicted in Fig. 18, i.e.* $\widetilde{\mathrm{bII}}^\mu$ *with* $11 \le \mu \le 39$, $\widetilde{\mathrm{bII}}^a$, $\widetilde{\mathrm{bII}}^b$, $\widetilde{\mathrm{bII}}^c$, $\widetilde{\mathrm{bII}}^d$, $\widetilde{\mathrm{bII}}^e$ *and* $\widetilde{\mathrm{bII}}^f$.

Our convention for the symbols follows that for Fig. 12, except that we put "b" for indicating manifolds with boundary for the domains.

An example of the neighboring fibers of a singular fiber of type $\widetilde{\mathrm{bII}}^{24}$ is depicted in Fig. 19.

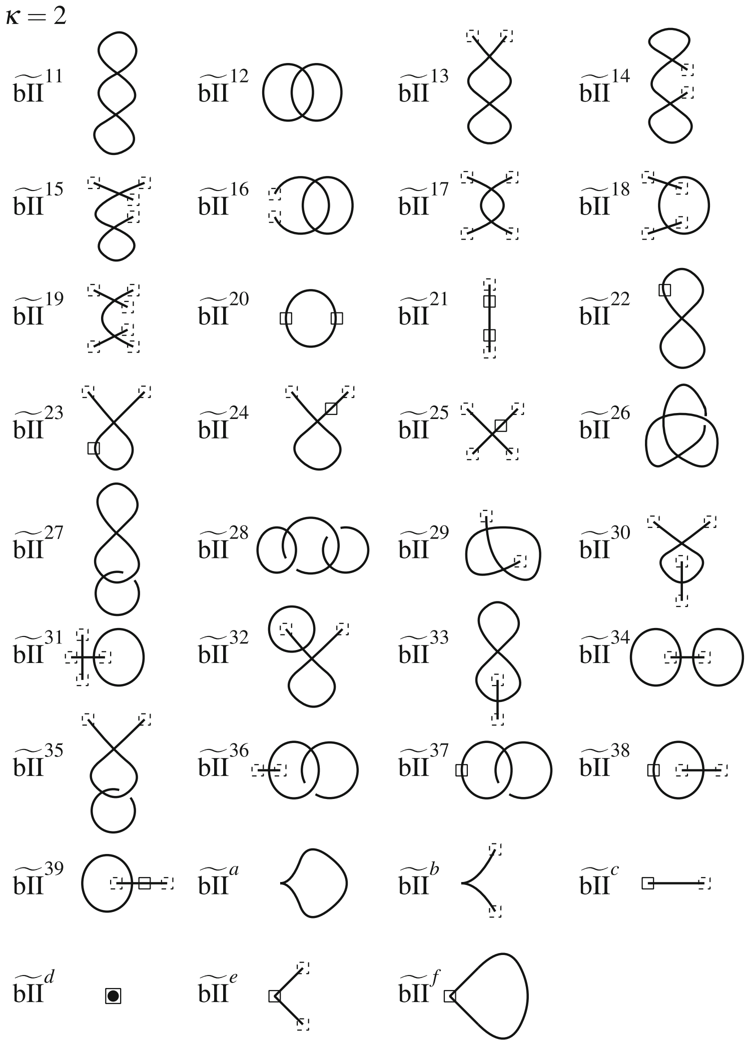

$\kappa = 2$
$\widetilde{\text{bII}}^{11}$
$\widetilde{\text{bII}}^{12}$
$\widetilde{\text{bII}}^{13}$
$\widetilde{\text{bII}}^{14}$
$\widetilde{\text{bII}}^{15}$
$\widetilde{\text{bII}}^{16}$
$\widetilde{\text{bII}}^{17}$
$\widetilde{\text{bII}}^{18}$
$\widetilde{\text{bII}}^{19}$
$\widetilde{\text{bII}}^{20}$
$\widetilde{\text{bII}}^{21}$
$\widetilde{\text{bII}}^{22}$
$\widetilde{\text{bII}}^{23}$
$\widetilde{\text{bII}}^{24}$
$\widetilde{\text{bII}}^{25}$
$\widetilde{\text{bII}}^{26}$
$\widetilde{\text{bII}}^{27}$
$\widetilde{\text{bII}}^{28}$
$\widetilde{\text{bII}}^{29}$
$\widetilde{\text{bII}}^{30}$
$\widetilde{\text{bII}}^{31}$
$\widetilde{\text{bII}}^{32}$
$\widetilde{\text{bII}}^{33}$
$\widetilde{\text{bII}}^{34}$
$\widetilde{\text{bII}}^{35}$
$\widetilde{\text{bII}}^{36}$
$\widetilde{\text{bII}}^{37}$
$\widetilde{\text{bII}}^{38}$
$\widetilde{\text{bII}}^{39}$
$\widetilde{\text{bII}}^{a}$
$\widetilde{\text{bII}}^{b}$
$\widetilde{\text{bII}}^{c}$
$\widetilde{\text{bII}}^{d}$
$\widetilde{\text{bII}}^{e}$
$\widetilde{\text{bII}}^{f}$

Fig. 18 List of fibers for stable maps of compact 3-dimensional manifolds with boundary into $\mathbb{R}^2$ (3): the singular fiber $\widetilde{\text{bII}}^d$ corresponds to the deformation as described in the *top half* of Fig. 6 in the sense of Sect. 5.1, while $\widetilde{\text{bII}}^e$ and $\widetilde{\text{bII}}^f$ correspond to the *bottom half* of Fig. 6.

Translating these concepts to computational representations and associated algorithms is an area of ongoing research. In particular, recent work on the joint contour net [3, 4, 31, 34] approximates Reeb spaces (see Sect. 5.1) based on a quantization of the function.

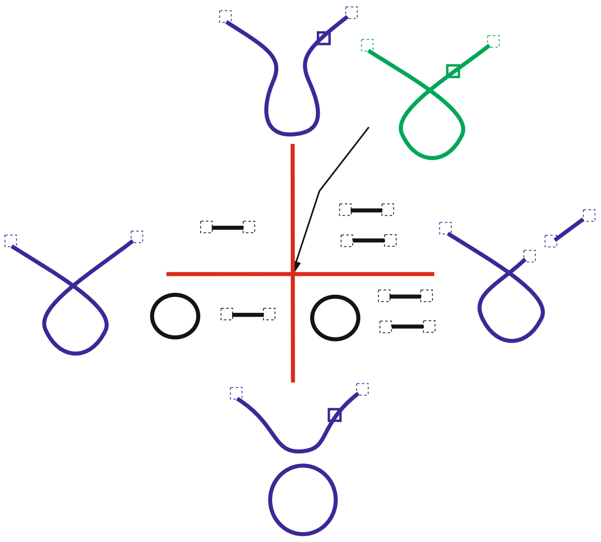


Fig. 19 Neighboring fibers of the singular fiber of type $\widetilde{\mathrm{bII}}^{24}$. The points on the *vertical red line* correspond to a singular fiber tangent to the boundary. By crossing that line, the fiber breaks at the part marked by the *thick square* or two of the components are connected

5 Reeb Space

5.1 Notion of Reeb Space

Let $f : N \to \mathbb{R}^m$ be a smooth map of a compact n-dimensional manifold, $n \geq m$. By contracting each connected component of a fiber to a point, we get the space R_f, which is called the *Reeb space* of f (see also [8]). As in Definition 2.9, we can also define the *quotient map* $q_f : N \to R_f$ and the *Reeb map* $\bar{f} : R_f \to \mathbb{R}^m$ so that f is decomposed as $f = \bar{f} \circ q_f$, which is called the *Stein factorization* of f.

By Hiratuka and Saeki [16], it is known that if f is a stable map, then its Reeb space is triangulable. Furthermore, for some specific dimension pairs (n, m), the local structures of Reeb spaces have been determined (see, for example, [15, 19, 32]). In fact, once you have a classification of singular fibers, the local structures of Reeb spaces can easily be obtained by examining the connected components of the nearby fibers for each singular fiber in the classification list.

Let $f_t : N \to \mathbb{R}^m$, $t \in I$, be a 1-parameter deformation of smooth maps. Then, it gives rise to the new map $F : N \times I \to \mathbb{R}^m \times I$, defined by $F(x, t) = (f_t(x), t)$. It is known that $\{f_t\}$ is a generic 1-parameter family if and only if F is a stable map. Therefore, if we can describe the local structures of the Reeb space of F, then we can describe the transitions of Reeb spaces of f_t, as t varies. Theorem 2.13 is such an example.

Let $f : N \to \mathbb{R}^m$ be a stable map of a compact n-dimensional manifold and $q_f : N \to R_f$ the quotient map onto the Reeb space. The following is well known.

Proposition 5.1 *The homomorphism $q_{f*} : \pi_1(N) \to \pi_1(R_f)$ induced by the quotient map on the fundamental groups is surjective. In particular, if N is simply connected, then so is the Reeb space R_f. Furthermore, the homomorphism $q_{f*} : H_1(N;\mathbb{Z}) \to H_1(R_f;\mathbb{Z})$ induced on the* 1*st homology groups is also surjective.*

However, $q_{f*} : H_2(N;\mathbb{Z}) \to H_2(R_f;\mathbb{Z})$ may not be surjective. In fact, we can construct a stable map $f : D^3 \to \mathbb{R}^2$ such that $H_2(R_f;\mathbb{Z}) \neq 0$, where D^3 is the 3-dimensional disk.

5.2 *Deformation of Reeb Spaces*

Suppose that a 1-parameter deformation of smooth maps $f_t : N \to \mathbb{R}^m$, $t \in I$, of a compact manifold N is given, where I is an interval. If the family $\{f_t\}$ is generic enough, then we have a discrete set $B \subset I$ of parameters such that every f_t for $t \notin B$ is stable. In fact, if t_0 and t_1 lie in the same component of $I \setminus B$, then we can show that f_{t_0} and f_{t_1} are C^∞ equivalent in the sense of Definition 3.1. The values in B are called *bifurcation parameters*. The behaviors of f_t near bifurcation parameters for the scalar function case has been treated in Sect. 2.3.

Mata-Lorenzo [21] studied generic 1-parameter families of smooth maps of a closed 3-dimensional manifold into $\mathbb{R}^2$, and gave a list of possible (local) topological transitions of the Reeb spaces near the bifurcation parameters. There are, in fact, 22 types of local topological transitions of Reeb spaces. Four of them are depicted in Fig. 20.

Here, we need to note that not every transition in the list is always realizable by a generic 1-parameter deformation of maps into $\mathbb{R}^2$ [27] as in the remarks given just after Theorem 2.13. Takao [36, 37] has studied which of the Mata-Lorenzo transitions can always be realized by such a 1-parameter deformation of maps into $\mathbb{R}^2$, and has shown that the lip move and the swallowtail moves as depicted in Fig. 20 can always be realized.

In fact, Takao [36] shows a stronger result as follows. If the map from the Reeb space into $\mathbb{R}^2$ satisfies certain reasonable conditions, then the relevant Mata-Lorenzo transitions can be realized by isotopies of the two component Morse functions. Here, an isotopy is a 1-parameter family of Morse functions that are generated by 1-parameter families of diffeomorphisms of the domain and the range. Thus, without changing the topological behaviors of the individual Morse functions, one can perform certain Mata-Lorenzo transitions.

This result guarantees, for example, that Reeb space simplifications based on Fig. 20 is realized by slightly changing the component functions. This is a good result from a practical viewpoint when one considers visualization of scientific data and its simplifications.

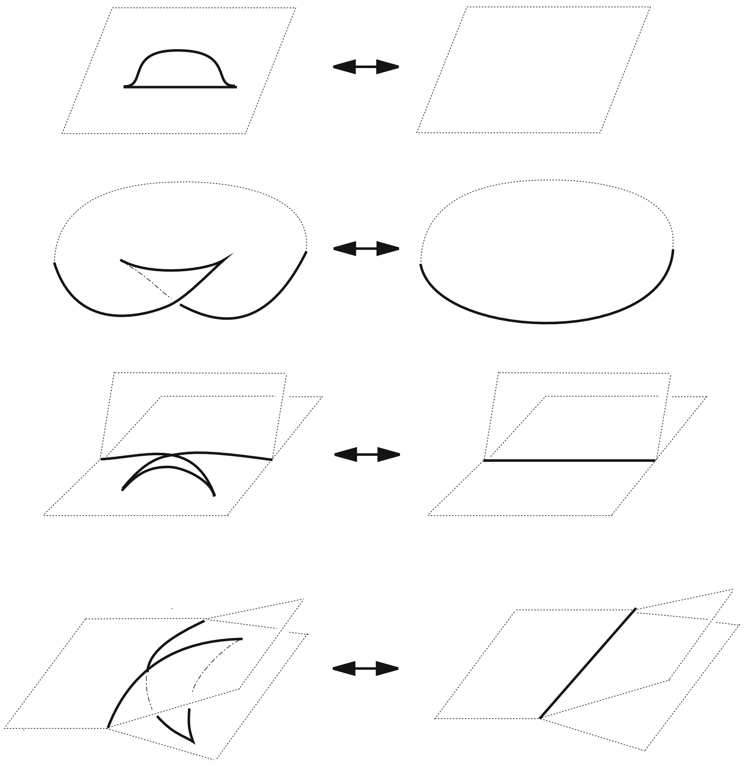


Fig. 20 Several transitions of Reeb spaces for a generic 1-parameter family of smooth maps of a closed 3-dimensional manifold into $\mathbb{R}^2$. The *top* one is called a *lip move* and each of the other three is called a *swallowtail move*

6 Future Problems

Our technique based on the theory of singular fibers of stable maps of compact 3-dimensional manifolds into $\mathbb{R}^2$ works very well for visualizing analytic multi-fields, as has been seen in [30, 31]. This is very promising from a mathematician's viewpoint, because many important analytic maps are waiting for us to analyze their structures visually.

On the other hand, our technique should be improved for visualizing general scientific data. It works relatively well for simulation data, but sometimes we have serious problems with noise or sparsity of real data.

6.1 Impact on Mathematics

The visualization techniques as mentioned above have substantial applications to Mathematics as has been explored in [34].

A supporting example has been exhibited in [31, Fig. 17]. It supports a theoretical result on the number of cusps that appear in a stable perturbation of a non-generic map. The original map is degenerate: however, after a perturbation, two or more cusps appear. This had been predicted by a theorem [17] in singularity theory, and it was visually verified.

We have another challenging problem for which our techniques might help a lot. Let $\mathbb{R}_+$ denote the set of strictly positive real numbers. For $a, b \in \mathbb{R}_+$, set

$$f_{a,b}(z, w) = z^3 + w^2 + a\bar{z} + b\bar{w}, \quad (z, w) \in \mathbb{C}^2.$$

How does the family $\{f_{a,b}\}$ bifurcate if $(a, b) \in \mathbb{R}^2_+$ varies? The following is known [18]. The parameter space $\mathbb{R}^2_+$ is divided into two regions A and B. The left two figures in Fig. 21 show the Jacobi set images of $f_{a,b} : \mathbb{C}^2 = \mathbb{R}^4 \to \mathbb{R}^2 = \mathbb{C}$ for $(a, b) \in A$ and $(a, b) \in B$, respectively.

The author would be very happy if we can visualize the singular fibers for $f_{a,b}$.

6.2 Impact on Computational Topology and Visualization

As has been pointed out, understanding fiber topology and Reeb spaces is essential for visualizing a given set of multi-field data. Like the Morse theory and the associated Reeb graphs for visualization of scalar functions, the theory of singular fibers of differentiable maps and the associated Reeb spaces furnishes theoretical back grounds for visualization of multi-fields. In particular, it gives classification results

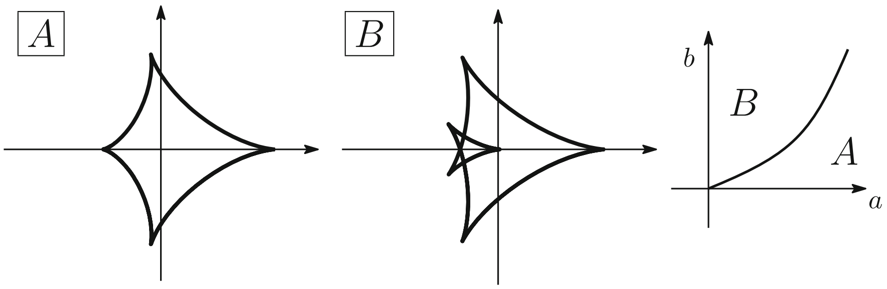


Fig. 21 Jacobi set images for $f_{a,b}$ with $(a, b) \in A$ and B

of local singularities and singular fibers, which are expected to help the recognition problem of singularities and singular fibers from algorithmic viewpoints. This would lead to potentially new solutions of problems in computational topology and visualization.

Acknowledgements The author would like to express his sincere gratitude to Hamish Carr, Christoph Garth and Tino Weinkauf for organizing the TopoInVis 2015, and for inviting him for a keynote lecture. Special thanks are to Hamish Carr for his essential comments for an earlier version, which drastically improved the presentation of the article. He would also like to thank all the participants for exciting discussions after the talk. The author also would like to thank Shigeo Takahashi, Daisuke Sakurai, Masayuki Kawashima, Kazuto Takao and Amit Chattopadhyay for stimulating discussions and for posing interesting questions. Finally, the author would like to thank the anonymous reviewer for various comments which improved the presentation of the paper.

The author has been supported in part by JSPS KAKENHI Grant Numbers JP23244008, JP23654028, JP25540041, JP15K13438.

References

1. Ando, Y.: Elimination of certain Thom-Boardman singularities of order two. J. Math. Soc. Jpn. **34**, 241–267 (1982)
2. Borodzik, M., Némethi, A., Ranicki, A.: Morse theory for manifolds with boundary. Algebraic & Geom. Topol. **16**(2), 971–1023 (2016)
3. Carr, H., Duke, D.J.: Joint contour nets: computation and properties. In: PacificVis, pp. 161–168. IEEE (2013)
4. Carr, H., Duke, D.J.: Joint contour nets. IEEE Trans. Vis. Comput. Graph. **20**(8), 1100–1113 (2014)
5. Cole-McLaughlin, K., Edelsbrunner, H., Harer, J., Natarajan, V., Pascucci, V.: Loops in Reeb graphs of 2-manifolds. Discret. Comput. Geom. **32**(2), 231–244 (2004)
6. Edelsbrunner, H., Harer, J.: Jacobi sets of multiple Morse functions. In: Cucker, F., DeVore, R., Olver, P., Sueli, E. (eds.) Foundations of Computational Mathematics, Minneapolis 2002, pp. 37–57. Cambridge University Press, Cambridge (2004)
7. Edelsbrunner, H., Harer, J., Mascarenhas, A., Pascucci, V., Snoeyink, J.: Time-varying Reeb graphs for continuous space-time data. Comput. Geom. **41**(3), 149–166 (2008)
8. Edelsbrunner, H., Harer, J., Patel, A.K.: Reeb spaces of piecewise linear mappings. In: Symposium on Computational Geometry, pp. 242–250. ACM (2008)
9. Edelsbrunner, H., Morozov, D., Patel, A.: The stability of the apparent contour of an orientable 2-manifold. In: Pascucci, V., Tricoche, X., Hagen, H., Tierny, J. (eds.) Topological Methods in Data Analysis and Visualization: Theory, Algorithms, and Applications, pp. 27–41. Springer, Berlin/Heidelberg (2011)
10. Edelsbrunner, H., Mücke, E.P.: Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Trans. Graph. **9**(1), 66–104 (1990)
11. Ehresmann, C.: Sur l'espaces fibrés différentiables. C. R. Acad. Sci. Paris **224**, 1611–1612 (1947)
12. Fomenko, A.T., Kunii, T.L.: Topological Modeling for Visualization. Springer, Berlin (1997)
13. Golubitsky, M., Guillemin, V.: Stable Mappings and Their Singularities. In: Graduate Texts in Mathematics, vol. 14. Springer, Berlin (1973)
14. Hamm, H.A., Tráng, L.D.: Un théorème de Zariski du type de Lefschetz. Ann. Sci. l'École Norm. Supér. **6**, 317–355 (1973)
15. Hiratuka, J.T.: A fatorização de Stein e o número de singularidades de aplicações estáveis. Ph.D. Thesis, Instituto de Matemática e Estatística, University of São Paulo (2001)

16. Hiratuka, J.T., Saeki, O.: Triangulating Stein factorizations of generic maps and Euler characteristic formulas. RIMS Kôkyûroku Bessatsu **B38**, 61–89 (2013)
17. Ikegami, K., Saeki, O.: Cobordism of Morse maps and its application to map germs. Math. Proc. Camb. Philos. Soc. **147**, 235–254 (2009)
18. Inaba, K., Ishikawa, M., Kawashima, M., Nguyen, T.T.: On linear deformations of Brieskorn singularities of two variables into generic maps. Tohoku Math. J. **69**(1) (2017). arXiv:1412.0310v3 [math.GT]
19. Kobayashi, M., Saeki, O.: Simplifying stable mappings into the plane from a global viewpoint. Trans. Am. Math. Soc. **348**, 2607–2636 (1996)
20. Levine, H.: Classifying Immersions into $\mathbb{R}^4$ over Stable Maps of 3-manifolds into $\mathbb{R}^2$. Lecture Notes in Mathematics, vol. 1157. Springer, Berlin (1985)
21. Mata-Lorenzo, L.: Polyhedrons and pi-stable homotopies from 3-manifolds into the plane. Bol. Soc. Brasil. Mat. (N.S.) **20**, 61–85 (1989)
22. Mather, J.N.: Stability of C^∞ mappings, VI: the nice dimensions. In: Proceedings of Liverpool Singularities-Symposium I (1969/70). Lecture Notes in Mathematics, vol. 192, pp. 207–253. Springer, Berlin (1971)
23. Mather, J.N.: Stratifications and mappings. In: Dynamical systems (Proceedings of a Symposium Held at the University of Bahia, Salvador, 1971), pp. 195–232. Academic Press, New York (1973)
24. Matsumoto, Y.: An Introduction to Morse Theory, translated from the 1997 Japanese original by K. Hudson and M. Saito. Translations of Mathematical Monographs, Iwanami Series in Modern Mathematics, vol. 208. American Mathematical Society, Providence (2002)
25. Milnor, J.: Morse Theory. Based on lecture notes by M. Spivak and R. Wells. Annals of Mathematics Studies, vol. 51. Princeton University Press, Princeton (1963)
26. Milnor, J.: Topology from the Differentiable Viewpoint. Based on Notes by David W. Weaver. The University Press of Virginia, Charlottesville (1965)
27. Motta, W., Porto, Jr. P., Saeki, O.: Stable maps of 3-manifolds into the plane and their quotient spaces. Proc. Lond. Math. Soc. **71**(3), 158–174 (1995)
28. Reeb, G.: Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique. C. R. Acad. Sci. Paris **222**, 847–849 (1946)
29. Saeki, O.: Topology of Singular Fibers of Differentiable Maps. Lecture Notes in Mathematics, vol. 1854. Springer, Berlin (2004)
30. Saeki, O., Takahashi, S.: Visual data mining based on differential topology: A survey. Pac. J. Math. Ind. **6**, 4 (2014)
31. Saeki, O., Takahashi, S., Sakurai, D., Wu, H.Y., Kikuchi, K., Carr, H., Duke, D., Yamamoto, T.: Visualizing multivariate data using singularity theory. In: Wakayama, M., Anderssen, S.R., Cheng, J., Fukumoto, Y., McKibbin, R., Polthier, K., Takagi, T., Toh, K.C. (eds.) The Impact of Applications on Mathematics: Proceedings of the Forum of Mathematics for Industry 2013, pp. 51–65. Springer Japan, Tokyo (2014)
32. Saeki, O., Yamamoto, T.: Singular fibers of stable maps of 3-manifolds with boundary into surfaces and their applications. Algebraic Geom. Topol. **16**, 1379–1402 (2016)
33. Saeki, O., Yamamoto, T.: Cobordism group of Morse functions on surfaces with boundary. In: Nabarro, A.C., Nuno-Ballesteros, J., Sinha, R.O., Ruas, M.A.S (eds.) Real and Complex Singularities, São Carlos, 2014. Contemporary Mathematics, vol. 675, pp. 279–297 (2016)
34. Sakurai, D.: Extracting and visualizing singular fibers for the analysis of multivariate data. Ph.D. Thesis, University of Tokyo (2015)
35. Takahashi, S., Kokojima, Y., Ohbuchi, R.: Explicit control of topological transitions in morphing shapes of 3D meshes. In: Pacific Conference on Computer Graphics and Applications, pp. 70–79. IEEE Computer Society (2001)

36. Takao, K.: Lips and swallow-tails of singularities of product maps. J. Singularities **10**, 286–295 (2014)
37. Takao, K.: Local moves of the Stein factorization of the product map of two functions on a 3-manifold (2015). Preprint
38. Whitney, H.: On singularities of mappings of Euclidean spaces. I. Mappings of the plane into the plane. Ann. Math. **62**(3), 374–410 (1955)

Topology-Based Analysis for Multimodal Atmospheric Data of Volcano Eruptions

Alexander Kuhn, Wito Engelke, Markus Flatken, Hans-Christian Hege, and Ingrid Hotz

Abstract Many scientific applications deal with data from a multitude of different sources, e.g., measurements, imaging and simulations. Each source provides an additional perspective on the phenomenon of interest, but also comes with specific limitations, e.g. regarding accuracy, spatial and temporal availability. Effectively combining and analyzing such *multimodal* and *partially incomplete* data of limited accuracy in an *integrated* way is challenging. In this work, we outline an approach for an integrated analysis and visualization of the atmospheric impact of volcano eruptions. The data sets comprise observation and imaging data from satellites as well as results from numerical particle simulations. To analyze the clouds from the volcano eruption in the spatiotemporal domain we apply topological methods. We show that topology-related extremal structures of the data support clustering and comparison. We further discuss the robustness of those methods with respect to different properties of the data and different parameter setups. Finally we outline open challenges for the effective integrated visualization using topological methods.

1 Introduction

The analysis of atmospheric gas and particle traces, such as of SO_2 or ash particles, is of fundamental importance for a better understanding of global atmospheric processes. Specifically volcano eruption events can produce massive amounts of such 'tracers' over a short time period. These substances can have severe global impact and may trigger complex atmospheric interactions. To better understand

A. Kuhn (✉) • H.-C. Hege
Zuse-Institute Berlin (ZIB), Takustr. 7, 14195 Berlin, Germany
e-mail: Kuhn@zib.de; hege@zib.de

W. Engelke • M. Flatken
Deutsches Zentrum für Luft- und Raumfahrt (DLR), Lilienthalplatz 7, 38108 Braunschweig, Germany
e-mail: Wito.Engelke@dlr.de; markus.flatken@dlr.de

I. Hotz
Media and Information Technology, Linköping University, 58183 Linköping, Sweden
e-mail: ingrid-hotz@liu.se

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_2

those processes an increasing number of modalities (including measurements and simulations) is utilized. One major challenge is to extract and combine the essential information, which is spread over various, mostly (w.r.t. space and time) sparse data sources. This requires a careful integration of information from each modality, considering its quality and limitations. For instance, sparse but particularly reliable measurement data can be used to calibrate the simulations producing denser data, while simulation data provide a means to interpolate measured data and to fill spatial and temporal gaps. The increased amount of information available today provides more complete representations of the physical phenomena, but increases the complexity of the analysis; hence a main objective are effective methods for data filtering, information reduction and abstraction. In this work we address those challenges by applying topology-based methods. Using the example of trace gas clouds we show that topological data analysis can serve as an effective tool to address essential analysis tasks. We focus on the extraction of extremal graphs as means for feature-oriented data reduction and as basis for visual comparison of the data from different sources. Spatio-temporal clustering in the space-time domain is used for a visual representation of the evolution of aforementioned atmospheric events. Hence, our main contributions are:

- Integration of spatially and/or temporally sparse data into a space-time domain
- Topological analysis and visualization of features in the common domain
- Topology-based spatio-temporal segmentation of features in the space-time domain, and fused visualization thereof.

The specific application considered is the analysis of pollutant clouds that emerged from volcanic eruptions. We are using data provided for the 2014 IEEE Visualization Contest [18]. This work is a follow-up of a contest contribution [11] that focuses on topological aspects and deepens them.

2 Related Work

Climate research is a data-intensive field. Depending on the application the data comes from various sources, e.g. large-scale simulations or observations from satellites. Accordingly, understanding and analyzing the data plays an important role. While visualization is used on an everyday basis, it is mostly limited to simple methods such as diagrams, statistical plots, color plots in a geographical context. More advanced visualization tools are often unknown due to their limited dissemination [29]. A particularly important topic is cloud evolution, which requires identification and tracking of clouds. Many methods have been proposed for this purpose, but they are often very complicated and only suitable for a specific application. For example Kober et al. follow a multi-scale approach based on image pixel displacement to track thunderstorm clouds from satellite data [21]. Extraction of contours and isosurfaces as well as tracking their change over time is part of multiple other approaches. Gambheer et al. track clouds by considering the overlap of sub-level sets across time steps [13]. Isosurface cloud-tracking in VR environments was introduced by Griffith et al. [15]. An example for cumulus cloud tracking is based on region growing followed by a space-time segmentation [19].

Many of the ideas used in these papers fit well into the framework of *topological* feature identification and tracking. A perspective that seems not yet to be present in meteorology and climate research. For instance, topological methods can be very powerful regarding abstraction, simplification and comparison of features such as clouds or trace gas emissions clouds. This has been shown in a recent paper, where cloud systems are tracked by employing topological segmentation for the identification of clouds and an optical flow method for sub-scale motion tracking [7]. Topological data analysis plays an increasing role in the field of visualization with many different applications. The full topological information of a scalar field is given by its Morse decomposition or topological graph. There are different strategies available to extract the topological graph from sampled scalar fields. The most common approaches for its computation go back either to Edelsbrunner [9] (using a piecewise linear interpolation) or to Forman [12] (who proposed a discrete Morse theory). Nowadays, efficient algorithms exist implementing these approaches [16, 17, 27, 28]. In many applications it is not necessary to compute the complete topological graph. A reduced structure keeping track only of changes in the number of components is the contour tree [3]. A structure that frequently represents features of interest is the extremal graph, which focuses on maxima and ridge lines [4]. Overviews on topological approaches in the context of vector fields are presented by Laramee et al. [23] and Pobitzer et al. [25]. Also tracking of topological structures over time has been considered. The proposed methods can roughly be categorized according to the geometric and topological criteria they are using to establish correspondences between features at successive time points. Geometric methods considering overlap of regions have been used for the tracking of burning cells [1, 2]. An example for topological tracking is mapping of critical points across time steps defining a feature flow field [26]. Other approaches are based on Jacobi sets [8].

3 Description of the Application and Data Sources

The aim of this work is to analyze the atmospheric impact of volcanic eruptions covered by several measurement modalities and simulations during the time from 30.05.2011 until 07.09.2011. In the focus of the presented analysis are three dominant volcanic events that have been recorded during this period: The eruptions of the volcanos Puyehue-Cordón Caulle (Ranco Province, Chile), Nabro (Red Sea Region, Eritrea), and Grimsvötn (South-East Iceland). The eruptions can be characterized by considering atmospheric tracers, like ash particles and SO_2 gas concentrations. The tracers form ‘plumes’, whose spatio-temporal evolution can be observed in all modalities. An overview of the temporal availability of the modalities is shown in Fig. 1.

All data sources are inherently time-dependent and have a common geographic reference grid (longitude, latitude, and optionally height). The presented methods

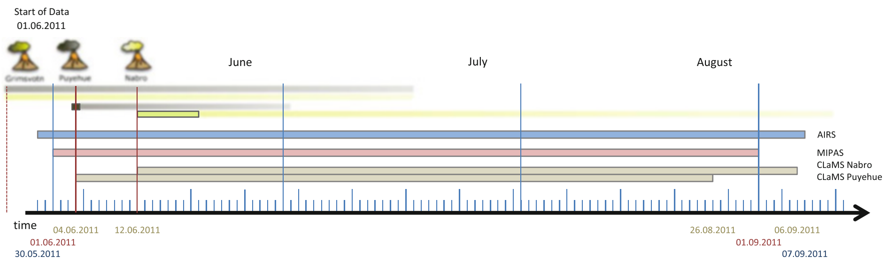


Fig. 1 Overview over the temporal availability of the various data: The three eruption events and the time span of the respective measurements or simulations are displayed. The *yellow* and *gray* bars indicate the 'lifetime' of the ejected sulfur, respectively ash particles

address the following questions that have been raised by the domain scientists [18]:

1. What are suitable methods to combine and relate the given modalities into a common reference space (integrated domain)? See Sect. 4.1.
2. How to reduce the amount of data to focus on the phenomena of interest (i.e., events related to volcano eruptions)? See Sect. 4.2.
3. How to integrate sparse data sources, derive compact feature-oriented visualizations, and how to associate this information to specific events? See Sect. 5.

3.1 Input Data Sources

The first type of data was obtained using the Michelson Interferometer for Passive Atmospheric Sounding (MIPAS) [14]. MIPAS measurements provide vertical sampling profiles at altitudes between 5–70 km, with approximately 14 orbits per day. The measuring technique is highly sensitive towards aerosol tracers and offers a good vertical resolution. The data is stored as single trajectory, containing the sampling points (longitude, latitude, altitude, and time) and values for different events (clear sky, ice detection, ash, and sulfate aerosol detection) [18]. Note that each event is represented as a binary value that indicates whether a predefined threshold has been exceeded. All sampling points ($\sim$ 1.3 Million points at 48 MB, starting from 01.06.2011 until 01.09.2011) of the MIPAS data set are shown in Fig. 2a.

The second data source are simulated trajectories using the Chemical Lagrangian Model of the Stratosphere (CLaMS) developed at the Institute for Energy and Climate Research, RWTH Aachen Univ. [22, 24]. This is an hierarchical model to describe the global chemical transport and contains a selected subset of pre-integrated trajectories, seeded at MIPAS detections. These are sulfur detections on the northern hemisphere for Grimsvötn and Nabro ($\sim$ 55.000 trajectories at 1 GB) and MIPAS ash detections on the southern hemisphere for Puyehue-Cordón Caulle ($\sim$ 5.800 trajectories at 62 MB). The input boundary conditions for the simulations are based on ERA interim data [18]. CLaMS trajectories are characterized by

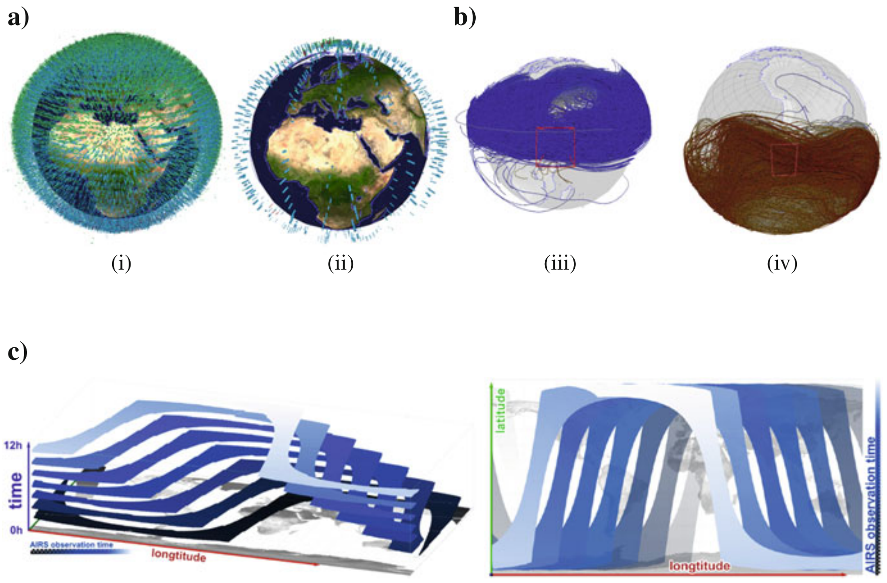


Fig. 2 Overview of the available data sets: (**a**) MIPAS satellite measurements, (**b**) CLaMS simulated trajectories, and (**c**) AIRS satellite measurements.

an high spatial and temporal resolution and contain additional information about physical scalars (e.g., pressure, temperature, potential vorticity). However, since numerical simulation can only approximate the real world phenomenon, the reliability of the trajectories decreases with their temporal distance to the seeding point. The trajectories are shown in Fig. 2b.

The third data source are measurements from the Atmospheric Infrared Sounder (AIRS), acquired by the NASA Aqua satellite. It measures thermal emissions in the atmosphere [20]. The satellite scans horizontal cross-sections of the atmosphere at very high resolutions and performs 14.5 orbits per day. Individual scanning samples are organized on a high-resolution quad-strip, which has been cut into 200 segments describing 12 h of measurement with ∼1.4 Million quads at 95 MB each (in total 19 GB). Every segment provides almost global coverage (i.e., temporal delay produces gaps between neighboring strips) and provides index information about SO_2 and ash concentrations [18]. Note that the index summarizes atmospheric information of a vertical column, hence height information is lost during acquisition. The original data acquired at a 12 h interval is shown at Fig. 2c.

4 Analysis of the Common Reference Domain

To combine and analyze the different data sources we proceed as follows: The basis is a projection of all individual data sets into one *common space-time domain*; second, *extremal structures* for both major measurement sources are extracted for visual comparison; finally, a *space-time segmentation* method is applied for a *combined visualization* integrating standard methods as isosurfaces, trajectory rendering for a detailed visual analysis.

4.1 Construction of the Common Reference Domain

In a first step, we sample the given data sets (Sect. 3.1) into a common reference domain. It is constructed as discrete regularly-sampled 4D domain with the dimensions longitude, latitude, altitude, and time. For visualization purposes, we mainly refer to the 3D subspace consisting of longitude, latitude, and time. Thereby, the measurement data builds the core of the new data set. Each source adds specific information to this domain according to the characteristic of the respective modality, e.g., SO_2 or ash concentrations, number of detection events. To increase the spatial and temporal coverage we interpolate the data on basis of the simulated CLaMS trajectories. In detail we introduce two filtering procedures:

1. **Gaussian filtering of the raw data:** The goal of this step is to assign a small volume to the point measurements and to the one-dimensional trajectories. This is achieved using a Gaussian convolution kernel, which adds an *isotropic footprint* with decreasing intensity to the samples in space *and/or* time. The size of the kernel is chosen very small in the order of a couple of gird cells. After this filtering step the field is still sparse and does not cover the entire domain.
2. **Spatio-temporal interpolation using trajectories:** The CLaMS particle simulation provides the necessary information for a realistic interpolation of the sparsely filled domain. It is assumed that the detected particles roughly follow these trajectories. This leads to a convolution of the measurements along the CLaMS trajectories with a *non-linear* and *anisotropic* spatio-temporal footprint. Technically, we apply an advection of the measurements along the trajectories. The length of the chosen trajectory segment τ and the decay of the signal along the trajectory are parameters of the method. In the following we assume a linear decay to zero within an interval $[+\tau, -\tau]$. For small values of τ the accuracy of the trajectories is high enough to obtain a reliable approximation of the particle distribution.

If not mentioned differently, we use a sampling resolution of $720 \times 360 \times 800$ cells for the reference sub-domain. Detailed views for sub-spaces of higher resolution or dimension may be extracted and analyzed using the same methodology. In the following the data specific projections are described in more detail.

MIPAS Data Processing The MIPAS samples are point samples that only carry a flag indicating a detection. The first step to integrate the MIPAS data is to count the number of detections per grid cell. Next, we apply at first the isotropic Gaussian filter and then the anisotropic interpolation filter. For the Figures a kernel size of 5 cells and $\sigma = 0.4$ is used. A temporal reasonable range of τ is in the order of 48 h (see [22, 24] for details, the impact of different setups are illustrated in [11]). This result of the MIPAS integration is a scalar density field approximating the spatial and temporal distribution of the particles.

AIRS Data Processing AIRS data has a high spatial but low temporal resolution due to the orbiting of the measuring satellite. There is no altitude information attached to these measurements. The temporal gaps between two measurements are in the order of 12 h. A Gaussian filter in temporal direction provides a simple but not very accurate solution to bridge the gaps. A more realistic result is obtained by interpolating the data using the CLaMS data as described above.

CLaMS Data Processing For the particle simulation trajectories are seeded at relevant MIPAS detections during the Puyehue-Cordón Caulle and Nabro eruptions. The detections are pre-filtered by domain experts before the simulation [18]. Thus, the density of the trajectories depends on the number of MIPAS samples and they are only available at irregularly distributed locations. Its reliability is decreasing with the simulation time. Therefor we use the CLaMS data mainly as basis for the interpolation of the satellite observation-data via advection.

To speed up the advection procedure along the trajectories, we generate an auxiliary time-dependent vector field from the CLaMS trajectories. To construct this field, we compute a weighted average of the velocity information within each cell. The velocity is given as the tangent directions of the trajectories passing through that cell. The weight accounts for the decreasing reliability along the trajectories. Here we use a linear decay along the trajectory. Note that the resulting field is not an approximation of the meteorological wind field used for the simulation, since the simulation also accounts for chemical reactions in the atmosphere [22, 24].

4.2 Topological Analysis in the Common Reference Domain

The integration of all input data into the common space-time domain yields fields that characterize the spatio-temporal distribution of ash and sulfur. In our framework, topological techniques are applied to analyze and visualize these fields. The benefit of topological analysis is a high level of abstraction which is compactly represented by explicit geometrical structures. These are graphs, segmented volumes, or surfaces characterizing the underlying field. The applied methods provide also access to topological simplification and filtering. The topological features can be used to link and compare the given modalities against each other, e.g., reference ash against SO_2, or advected MIPAS against AIRS. Further they can be associated to observable phenomena, like specific volcano eruptions, at an abstract feature-based level. In the following we explore possibilities to characterize physical

```
Data: Regular scalar field $s(\mathbf{x})$
Result: Segmentation by hierarchical labels, topological graph
1) Init union-find data structure $UF(G_j)$,
where each group $G_j$ has an label $j$ and stores a set of grid nodes
2) Sort all sample points, based on its value into sorted list $L$
(ascending: maxima, descending: minima features)
while $L$ contains grid nodes $g_i$ with values larger than threshold $h_{min}$ do
    classify current grid node $g_i$:
    if $g_i$ is isolated extrema then
        add new group to $UF$
    else if $g_i$ is adjacent to a single group $G_j$ then
        add $g_i$ to $G_j$
    else if $g_i$ is adjacent to multiple groups (saddle point) then
        process node $g_i$: construct graph edge (see Sect. 4.2) or
        merge two groups (see "Topology-Based Space-Time Segmentation" in Sect. 4.2)
end
```

Algorithm 1: Approximated topological feature algorithm

features, as eruption plumes, using *extremal structures* in the respective field. For an algorithmic outline to derive *approximate* topological structures similar to topological spines [4, 6] and segmentations of a scalar field $s(\mathbf{x})$, see Algorithm 1.

Extremum Graph Extraction An approximate extrema graph [4] is used to describe the spatial structure of the particle distribution for every time slice. Extrema graphs represent the target phenomena, in our case ash and SO_2 plumes, as spatially embedded graphs. They connect saddle points with extremal points (see Algorithm 1) according to the approximate Morse-Smale complex of the function s. Note that in this step a persistence based filtering is not necessary since the construction of the common reference domain already comprises a smoothing and down-sampling step. Instead we define a minimum threshold h_{min} filtering out parts of the graph below this value. The graph is used in three ways: Visualization—the graph describes the spatial connectivity of extremal events in each time slice. It visually captures advection patterns that are eminent in temporal snapshots in AIRS or advected MIPAS data (see Fig. 6). It also serves as basis for visual comparison of the fields from the different modalities as illustrated in Fig. 3. Filtering—the graphs are further used as input for filtering, e.g. by considering only trajectories in the vicinity of those structures. This allows to focus on specific topological events. It further supports feature-based filtering with respect to size or the location of the plume. Space-time segmentation—the extremal graph lives by construction in single time slices. To obtain a connected structure in space-time the graphs are further used as input for spatio-temporal clustering, see below.

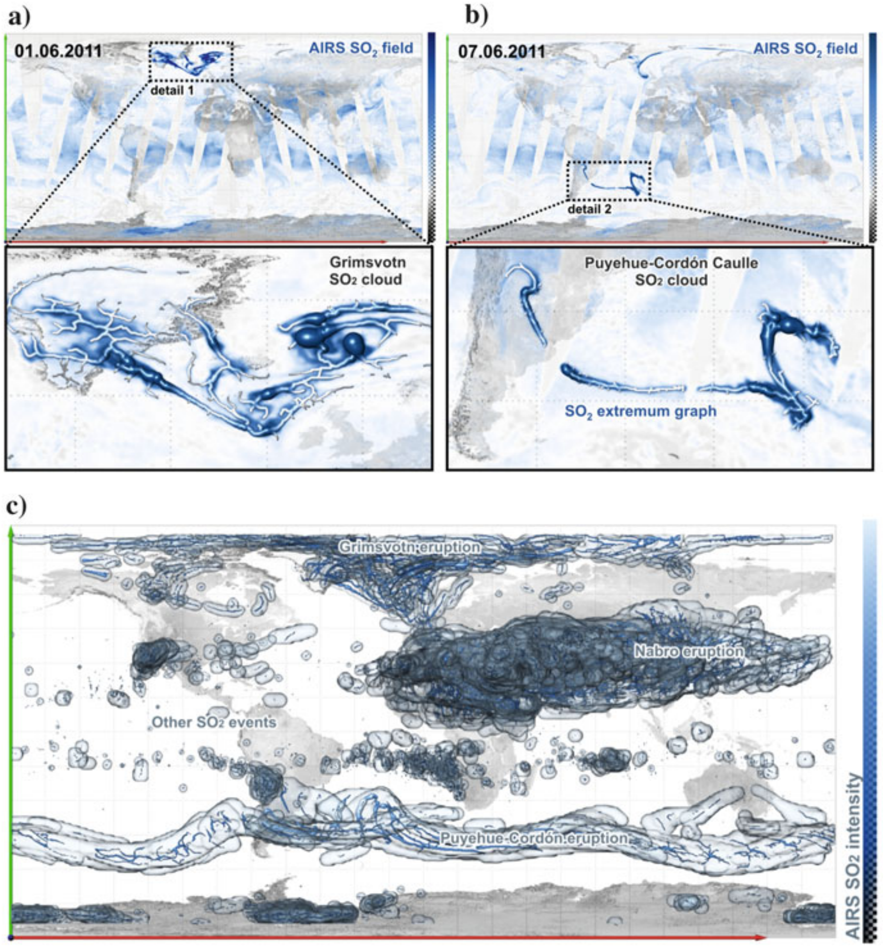


Fig. 3 Combining AIRS and CLaMS data: AIRS and CLaMS data describe the spatio-temporal structure of occurring SO_2 events and characterize the evolution of the SO_2 plumes: The SO_2 cloud created by the Puyehue-Cordón Caulle eruption moves rapidly eastwards, following the major jet streams on the southern hemisphere. In contrast, the Nabro event remains 'trapped' in a larger vortex structure and is distributed across central Asia, while the Grimsvötn SO_2 cloud circulates towards the north pole. In combination, the space-time view of multiple modalities conveys location and time of strongest value concentrations and their distribution during the plume development. (**a**) AIRS SO_2 & graph 1.6.2011. (**b**) AIRS SO_2 & graph 7.6.2011. (**c**) All SO_2 graphs with offset surfaces over the observation time (space-time see Fig. 4).

Topology-Based Space-Time Segmentation To capture the temporal evolution of extremal structures we use a topological method to segment regions within the common reference domain. This segmentation results in labels for extremal events that can be associated to individual physical phenomena (see Fig. 4b). Two adjacent

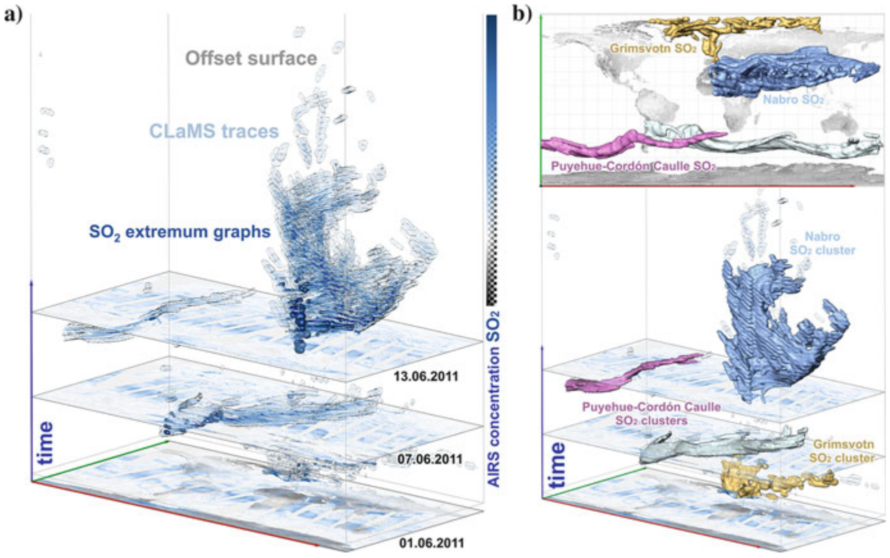


Fig. 4 Combining AIRS and CLaMS data: The space-time segmentation of AIRS SO_2 graph fields conveys location and time of largest features and their distribution during the plume development. The segmentation determines cluster volumes (filtered such that only volumes larger than 10.000 cells remain) that can be used to classify individual events. (**a**) CLaMS segments & surfaces (slices in Fig. 3). (**b**) Topology-based segmentation

groups are merged, if the difference of their maximum values is lower than a user-defined persistence threshold p_{min}. Technically this corresponds to a watershed algorithm [5] with a persistence-based hierarchical merging step [10]. We can obtain one connected cluster for dominant events choosing a maximum persistence value for $p_{min} = max(s(\mathbf{x}))$. The results of this segmentation are shown in Fig. 4.

4.3 Topology-Based Analysis and Visualization

The extraction of topological features, such as graphs or segmentation surfaces ("Topology-Based Space-Time Segmentation" in Sect. 4.2), allows characterizing and grouping extremal features of the underlying scalar fields. Extremum graphs capture the spatial structure, e.g., longitudinal and latitudinal extents, as well as spatial connectivity of extremal regions. To enhance the information content of the graph, we map the local values of the scalar field to its thickness and color (see Fig. 5a). The picked color schemes emphasize different data sources and modalities in combined visualizations (i.e., AIRS SO_2 in blue, AIRS ash in red/yellow, MIPAS in green/blue scheme). Extremal structures can further be emphasized by visualizing corresponding iso-surfaces in the underlying advection fields (see Sect. 3.1).

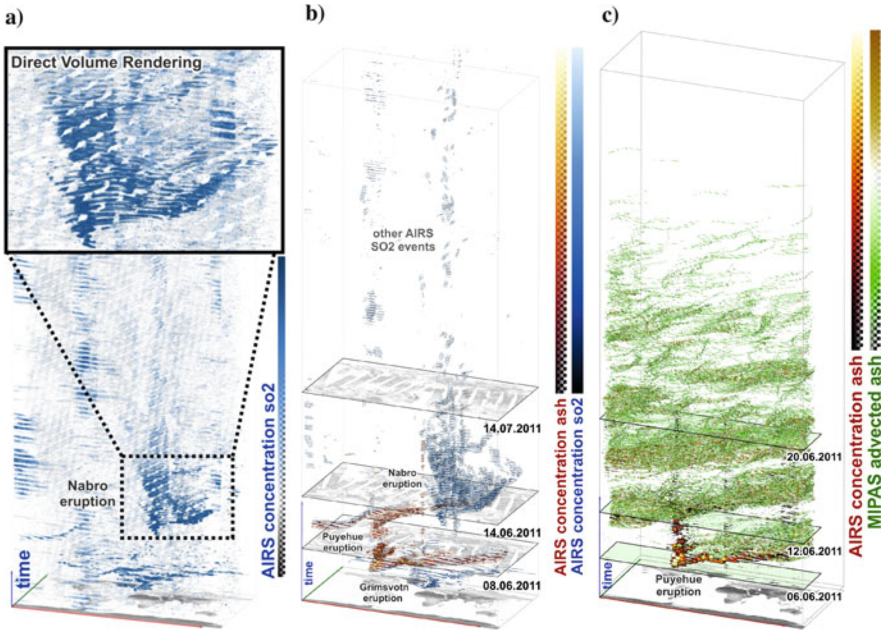


Fig. 5 **Space-time AIRS, CLaMS, MIPAS:** AIRS data is used to analyze SO_2 and ash concentration during the eruption events in space and time. In subfigures (**b**) and (**c**) the extremal graphs are displayed for each time slice. They are colored and sized by AIRS field values to emphasize strong production events. Transparent surfaces enclose the extremal graphs connecting them in temporal direction. The extend of these surfaces in time direction (z-axis) characterizes the life time of a plum. In (**c**) the circulations of the tracers in the southern hemisphere can be observed in terms of visual stripes. (**a**) AIRS SO_2 rendering. (**b**) AIRS SO_2, ash graphs. Details see Fig. 4. (**c**) AIRS ash, MIPAS graphs. Details see Fig. 6

5 Results

We use the topology-based methods from Sect. 4.2 to visualize characteristics of the data in the common reference domain. Figure 5 shows a visualization of the integrated data in a space-time frame. The combination of filtering, interpolation and topology allows to extract a set of distinct extremal objects that can be associated to the major events of interest. The respective parameter setup allows to derive different levels of detail. Thus, not only the three dominant volcano eruptions but also a set of small-scale events are captured, which exhibit a much lower particle density.

To get a better impression of the movement of the particles we further augment this visualization with CLaMS trajectory segments, illustrated in Fig. 5b. The display of events associated to specific eruptions allows to compare ash concentrations based on AIRS satellite and MIPAS advected measurements. Mapping additional scalar information to the geometry of the topological graphs emphasize regions

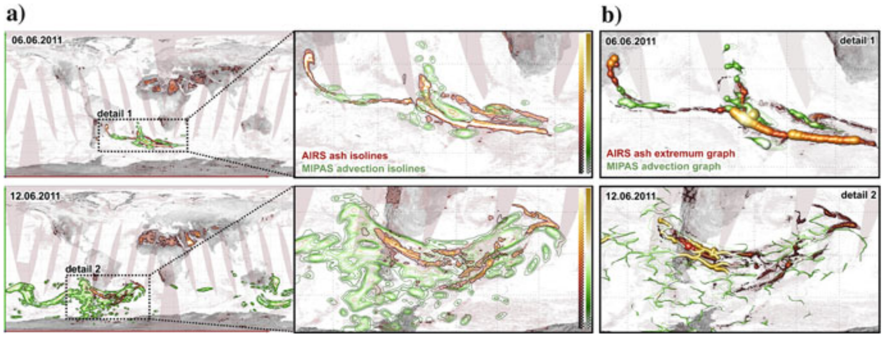


Fig. 6 **Combining AIRS, MIPAS and CLaMS:** Comparison of the AIRS ash data and the advected MIPAS ash field for the Puyehue-Cordon Caulle eruption. (**a**) shows the reference fields for both modalities (AIRS, advected MIPAS). The detail views reveal the strengths and weaknesses of both information sources: AIRS captures fine spatial advection patterns, but suffers a limited detection accuracy. MIPAS advection fields are more sensitive, but do not convey information of the strength of the detection and sometimes miss samples across thin cloud structures. (**a**) AIRS ash & MIPAS advection field for 6.6. and 12.6.2011. (**b**) AIRS, MIPAS graphs

of specifically high scalar values. This is a clear advantage compared to slice-based color plots or direct volume visualizations. The spatial connectivity of the extremum graphs serves as a suitable description of the spatial structures of the corresponding plumes. Figure 5c shows a visualization of ash events form AIRS (red) and advected MIPAS (green) data. The higher sensitivity of the MIPAS data in space-time, compared to AIRS ash detection is clearly visible.

Detailed visualizations of single time slices, see Fig. 6, illustrate the different characteristics of both measurement modalities. While AIRS data captures very detailed transport structures of the ash cloud, the MIPAS-CLaMS combination results in a blurred reconstruction and misses thin cloud structures. The concentration derived from the binary MIPAS signal is only an approximation of the true values. Due to its higher sensitivity, advected MIPAS better illustrates the spread of the ash cloud during the time of observation over the southern earth hemisphere.

Figure 3 illustrates the benefits of the high spatial resolutions of the AIRS measurements. Depending on the filtering scales, it is possible to detect also small-scale volcanic eruptions. An example is the Lokon-Epung event in Indonesia around July 2011 and multiple SO_2 and ash emissions associated to the Etna Volcano in Sicily. These smaller events are often not covered by the MIPAS data or are represented only by a very small number of detections. There are also multiple additional SO_2 events that are typically associated to mining or shipping trails (Fig. 7).

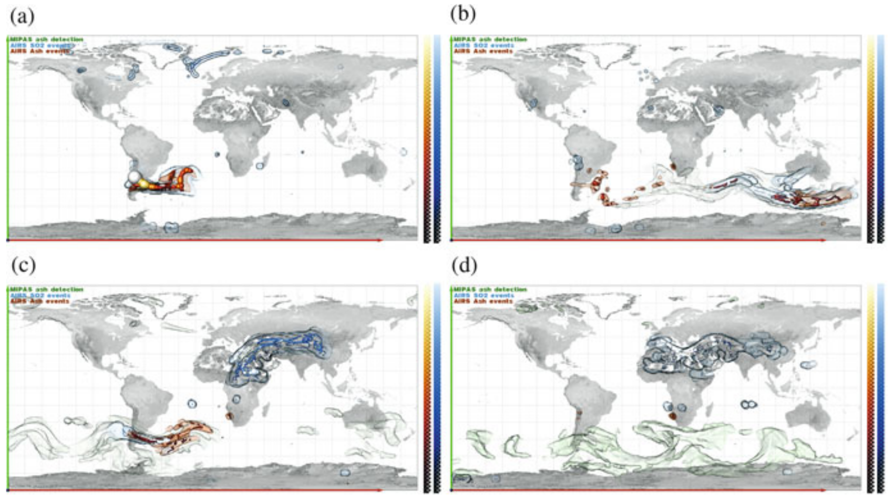


Fig. 7 **Integrated Visualization:** This image shows snapshots of an integrated visualization of AIRS SO_2 (*blue*) and AIRS ash (*red*) with its corresponding graphs (scaled and colored by AIRS value) and MIPAS advection surfaces (*green* transparent surfaces). Each image captures the situation of one day and highlights the different characteristics of the Puyehue-Cordon Caulle and the Nabro eruptions, especially with respect to ash production (animation in accompanying video). (**a**) AIRS & MIPAS for 6.6.2011. (**b**) AIRS & MIPAS for 9.6.2011. (**c**) AIRS & MIPAS for 12.6.2011. (**d**) AIRS & MIPAS for 25.6.2011

5.1 Limitations

Topological analysis tools have been shown to be valuable for reduction of complex input data to a smaller and visually assessable set of extremal structures. However, transferring sparsely sampled data from different modalities into a common reference domain still requires multiple filtering and interpolation steps (see Sect. 4.1). These use implicit assumptions, heuristic parametrizations and approximations of the real world phenomena, which are not covered by the data. Critical applications, as flight route planning after volcanic eruptions, require a more profound assessment of the impact of those techniques w.r.t. uncertainty and sensitivity in the final results. We assume that by using higher-order interpolation methods (e.g., for CLaMS and AIRS data) and samplings of the integrated domain, approximations can get more accurate, while some fundamental sources of uncertainty will remain (e.g., reliability of the CLaMS simulation, AIRS sensitivity).

6 Conclusion

In summary, we have described a basic procedure to construct a common data domain and integrated data from a variety of input modalities, including measurements (MIPAS, AIRS) and simulations (CLaMS), to analyze volcanic eruptions. We have showed that topological processing provides suitable tools to characterize extremal features of such integrated data. The availability of explicit feature descriptors (i.e., topological graphs, segmentations) allows for improved filtering, compact visualization, feature-based comparison and visual analysis. The presence of topological features allows utilizing a broad set of tools known from the topology-related literature (e.g., persistence-based filtering, topology-driven rendering, high-dimensional visualization). As topological methods have been shown to scale in settings with large amounts of data (e.g., [6, 17]) and in high-dimensional settings (e.g., [3]), similar analysis procedures should be able to cover future high-resolution input modalities. For future work we plan to analyze the impact of filtering methods in further detail. Of interest are also feature-based statistics (e.g., life-time and size distributions) and automated quantitative analytics of the data (automated event detection and classification).

Acknowledgements We would like to thank Patrick Jöckel from Inst. of Atmospheric Physics—DLR for his support and explanations of the data. This work was funded by the German Federal Ministry of Education and Research under grant number 01LK1213A and by the European Union Framework Programme FP7 under grant agreement number 607177.

References

1. Bremer, P.T., Weber, G., Pascucci, V., Day, M., Bell, J.: Analyzing and tracking burning structures in lean premixed hydrogen flames. IEEE Trans. Vis. Comput. Graph. **16**(2), 248–260 (2010)
2. Bremer, P.T., Weber, G., Tierny, J., Pascucci, V., Day, M.S., Bell, J.B.: Interactive exploration and analysis of large-scale simulations using topology-based data segmentation. IEEE Trans. Vis. Comput. Graph. **17**(9), 1307–1324 (2011)
3. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. **24**(2), 75–94 (2003)
4. Correa, C., Lindstrom, P., Bremer, P.T.: Topological spines: a structure-preserving visual representation of scalar fields. IEEE Trans. Vis. Comput. Graph. **17**(12), 1842–1851 (2011)
5. Couprie, M., Bertrand, G.: Topological gray-scale watershed transformation. In: Proceedings of SPIE. Vision Geometry VI, vol. 3168, pp. 136–146 (1997)
6. Doraiswamy, H., Natarajan, V.: Efficient algorithms for computing Reeb graphs. Comput. Geom. **42**(6–7), 606–616 (2009)
7. Doraiswamy, H., Natarajan, V., Nanjundiah, R.S.: An exploration framework to identify and track movement of cloud systems. IEEE Trans. Vis. Comput. Graph. **19**(12), 2896–2905 (2013)
8. Edelsbrunner, H., Harer, J.: Jacobi sets of multiple Morse functions. In: Cucker, F., DeVore, R., Olver, P., Sueli, E. (eds.) Foundations of Computational Mathematics, Minneapolis 2002, pp. 37–57. Cambridge University Press, Cambridge (2004)

9. Edelsbrunner, H., Harer, J., Zomorodian, A.: Hierarchical Morse complexes for piecewise linear 2-manifolds. In: Proceedings of 17th Symposium on Computational Geometry, pp. 70–79 (2001)
10. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. Discret. Comput. Geom. **28**(4), 511–533 (2002)
11. Engelke, W., Kuhn, A., Flatken, M., Chen, F., Hege, H.C., Gerndt, A., Hotz, I.: Atmospheric impact of volcano eruptions (SciVis Contest 2014). In: Proceedings of IEEE SciVis, p. 10 (2014)
12. Forman, R.: A user's guide to discrete Morse theory. In: Proceedings of 2001 International Conference on Formal Power Series and Algebraic Combinatorics. Advances in Applied Mathematics (2001)
13. Gambheer, A.V., Bhat, S.: Life cycle characteristics of deep cloud systems over the Indian region using INSAT-1B pixel data. Mon. Weather Rev. **128**, 4071–4083 (2000)
14. Griessbach, S., Hoffmann, L., Spang, R., Riese, M.: Volcanic ash detection with infrared limb sounding: MIPAS observations and radiative transfer simulations. Atmos. Meas. Tech. **7**(5), 1487–1507 (2014)
15. Griffith, E.J., Post, F.H., Koutek, M., Heus, T., Jonker, H.J.: Feature tracking in VR for cumulus cloud life-cycle studies. In: Proceedings 11th Eurographics conference on Virtual Environments, pp. 121–128. Eurographics Association (2005)
16. Günther, D., Reininghaus, J., Wagner, H., Hotz, I.: Efficient computation of 3D Morse-Smale complexes and persistent homology using discrete Morse theory. Vis. Comput. **28**(10), 959–969 (2012)
17. Gyulassy, A., Natarajan, V., Pascucci, V., Hamann, B.: Efficient computation of Morse-Smale complexes for three-dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. **13**(6), 1440–1447 (2007)
18. Hentschel, B.: 2014 IEEE SciVis Contest (2014). http://www.viscontest.rwth-aachen.de/
19. Heus, T., Seifert, A.: Automated tracking of shallow cumulus clouds in large domain, long duration large eddy simulations. Geosci. Model Dev. **6**(4), 1261–1273 (2013)
20. Hoffmann, L., Griessbach, S., Meyer, C.I.: Volcanic emissions from AIRS observations: detection methods, case study, and statistical analysis. In: Proceedings of SPIE. Remote Sensing of Clouds and the Atmosphere XIX; and Optics in Atmospheric Propagation and Adaptive Systems XVII, vol. 9242, p. 924214 (2014)
21. Kober, K., Tafferner, A.: Tracking and nowcasting of convective cells using remote sensing data from radar and satellite. Meteorol. Z. **1**(18), 075–084 (2009)
22. Konopka, P., Grooß, J.U., Günther, G., Ploeger, F., Pommrich, R., Müller, R., Livesey, N.: Annual cycle of ozone at and above the tropical tropopause: observations versus simulations with the chemical lagrangian model of the stratosphere (CLaMS). Atmos. Chem. Phys. **10**(1), 121–132 (2010)
23. Laramee, R.S., Hauser, H., Zhao, L., Post, F.H.: Topology-based flow visualization, the state of the art. In: Hauser, H., Hagen, H., Theisel, H. (eds.) Topology-based methods in visualization, Mathematics and Visualization, pp. 1–19. Springer, Berlin/Heidelberg (2007)
24. McKenna, D.S., Konopka, P., Grooß, J.U., Günther, G., Müller, R., Spang, R., Offermann, D., Orsolini, Y.: A new chemical lagrangian model of the stratosphere (CLaMS) 1. formulation of advection and mixing. J. Geophys. Res. Atmosp. **107**(D16), ACH–15 (2002)
25. Pobitzer, A., Peikert, R., Fuchs, R., Schindler, B., Kuhn, A., Theisel, H., Matkovic, K., Hauser, H.: On the way towards topology-based visualization of unsteady flow. In: Hauser, H., Reinhard, E. (eds.) Eurographics 2010 - State of the Art Reports, pp. 137–154. The Eurographics Association (2010)
26. Reininghaus, J., Kasten, J., Weinkauf, T., Hotz, I.: Efficient computation of combinatorial feature flow fields. IEEE Trans. Vis. Comput. Graph. **18**(9), 1563–1573 (2012)

27. Robins, V., Wood, P., Sheppard, A.: Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1646–1658 (2011)
28. Shivashankar, N., Natarajan, V.: Parallel computation of 3D Morse-Smale complexes. Comput. Graph. Forum **31**(3), 965–974 (2012)
29. Tominski, C., Donges, J., Nocke, T.: Information visualization in climate research. In: 2011 15th International Conference on Information Visualisation, pp. 298–305 (2011)

A Comparison of Joint Contour Nets and Pareto Sets

Lars Huettenberger, Christian Heine, and Christoph Garth

Abstract For the scientific visualization and analysis of univariate (scalar) fields several topological approaches like contour trees and Reeb graphs were studied and compared to each other some time ago. In recent years, some of those approaches were generalized to multivariate fields. Among others, data structures like the *joint contour net* (JCN) and the *Pareto set* were introduced and improved in subsequent work. However, both methods utilized individual data sets as test cases for their proof-of-concept sections and partially lacked a complete comparison to other multivariate approaches. Hence, to better understand the relationship between those two data structures and to gain insights into general multivariate topology, we present a deeper comparison of JCNs and Pareto sets in which we integrate data sets applied in the original JCN and Pareto set papers.

1 Introduction

In recent years, several ideas towards multivariate topology were presented to the visualization community. Their common goal is to visualize high-dimensional and multivariate data sets like ensemble data using a topological point of view. These approaches are often based on the extension of topological methods for univariate data like Reeb graphs, contour trees, or Morse-Smale complexes. The visualizations use those ideas to identify interesting features and/or facilitate abstracted views on the data without loss of important information. They thereby support users in their exploration of the data.

While for Reeb graphs, contour trees, and other univariate scalar field methods their (dis)advantages and their relations to each other are well understood, such research is not yet done thoroughly for most of the multivariate methods.

L. Huettenberger (✉) • C. Garth
University of Kaiserslautern, Kaiserslautern, Germany
e-mail: l_huette@cs.uni-kl.de; garth@cs.uni-kl.de

C. Heine
ETH Zurich, Zurich, Switzerland
e-mail: cheine@inf.ethz.ch

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_3

For this paper, we therefore take a closer look at two recently proposed approaches: the joint contour nets (JCN) and Pareto sets. We chose these methods since both aim to identify and visualize interesting features in multivariate scalar data sets given with arbitrary dimensionality in both domain and range space. Furthermore, they are both relatively novel and have matching requirements on the input data. Hence, if a data set is suited for a JCN approach then the Pareto set approach can also be applied and vice versa.

Both theoretical definitions require scalar functions f on a continuous d-manifold M, while for practical usage, the methods assume simplicial complexes where the function values $f = (f_1, \ldots, f_n)$, n being the number of underlying functions, are only given at vertices and expanded to the whole domain through barycentric interpolation. Note, that both methods have no restriction with respect to the number of underlying functions or dimensionality, though focus on two- or three-manifolds for visualization purposes.

We specified a subset of nodes in the JCN and prove a close relation between these and the Pareto set. In detail, our theoretical considerations in Sect. 6 suggest equality of both sets in the limit of fine nets and simplicial complexes, respectively. This allows a combination of the algorithms to calculate those structures and the respectively inner relations between connected components in the Pareto set and the mentioned subset of JCN nodes.

Both structures are based on different approaches, yet yield similar, in the limit even equal, results. This promotes their impact on the understanding of multivariate topology.

2 Previous Work

In the following sections, we present the idea and definition of JCNs and Pareto sets as well as some related work to each method, such that the theoretical considerations in Sect. 6 and the visualization examples in Sect. 7 are easier to follow. Both methods are topological approaches to the analysis of multivariate scalar data. Those concepts are delimited on one side by non-topological methods like multiple linked views [6] or approaches that use, for example, correlation [20], multidimensional transfer functions [16], or field coherence [17] to define a similarity measure between the underlying fields. However, those usually do not provide a single view to show common behavior and become confusing for large numbers of functions.

The second delimitation is given by univariate methods which usually cannot be extended ad hoc to multivariate fields. Such methods are, for example, Morse theory [12], Reeb graphs [8], and contour trees [3]. As mentioned this area is already well researched. For example, it includes for contour trees parallel algorithms [19], simplification [4], and relations to other concepts like Reeb graphs [9].

The third delimitation is towards multivariate approaches for other data types like vectors. One concept for this kind of data sets are Morse decomposition [7, 22].

Other data types like tensors are also possible, though less studied and beyond the context and scope of this work.

Within the field of topological approaches to multivariate scalar data, other work exists like largest contours and Jacobi sets. Largest contours was proposed by Schneider et al. [21] as a comparison of the underlying functions based on normalized spatial overlap of the largest contours, i.e. sets of maximal contours containing only one critical point each. Edelsbrunner and Harer [10] introduced Jacobi sets which work with the intersection and restriction of level sets, see Sect. 3, to multivariate data analysis.

3 Joint Contour Nets

The JCN was introduced by Carr and Duke [2] as an extension of contour trees to multivariate data. The contour tree [3] is a graph structure based on contours, i.e. connected components in the isosurface $f^{-1}(c) = \{x \in M \mid f(x) = c\}$ for some isovalue $c \in f(M) \subseteq \mathbb{R}$. Each contour corresponds to a point on an edge in the contour tree while locations where contours appear, disappear, or merge/split when c is only changed by a small amount ϵ correspond to nodes. In univariate topology these locations are denoted as maxima, minima, and saddle points, respectively.

Carr and Duke also compared their approach to Reeb spaces [11], where $f^{-1}(c)$, called a *fiber*, is calculated for the high-dimensional isovalue $c \in \mathbb{R}^n$. The connected components are again compressed to points, though in a high-dimensional structure in $\mathbb{R}^n$. Edelsbrunner et al. [11] gave a mathematical algorithm to compute the Reeb space which only worked for four underlying functions. Further work on a related notion, persistence, by Carlson et al. [1] produced results for a higher number of functions at the cost of higher running time. However, even without run time restrictions, note that $f^{-1}(c)$ will be reduced to a point (dimension zero) or become empty if $d \leq n$ such that the Reeb space is equivalent to the original manifold $M \subseteq \mathbb{R}^d$.

Hence, Carr and Duke used a method without loss in dimensionality and defined a *joint level set* (JCN) for $c \in f(M) \subseteq \mathbb{R}^n$:

$$f^{-1}(c) = \{x \in M \mid round(f(x)) = c\}$$

The rounding function can be, for example, the floor function, $f : \mathbb{R} \mapsto \mathbb{N}$ with $round(x) = \lfloor x \rfloor$. Hence, the joint level sets of f are equivalent to the fibers of the discretized function $round(f)$. The connected components in the joint level set are defined as *slabs*. Figure 1 illustrates the creation of slabs in a simple one-dimensional example with two functions.

Carr and Duke transformed the slabs into a graph structure in which each slab corresponds to a node and adjacent slabs result in an edge between their nodes.

Note how the size of the slabs depends on the *round*-function, such that the coarseness of that function allows an user to adjust the level of detail of the JCN.

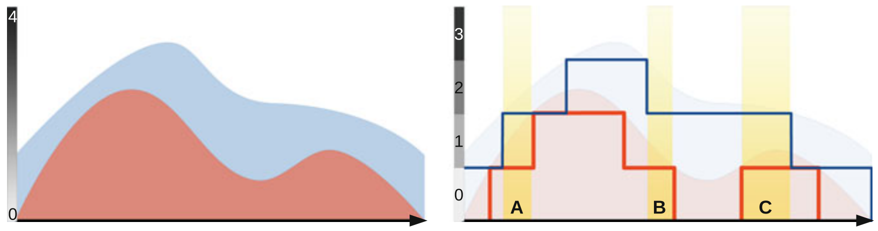


Fig. 1 The *left image* shows two functions f and g *colored red* and *blue*, respectively, with an image interval from zero to four. The *right image* overlays the functions with their discretized versions *round*(f) and *round*(g) colored similarly using the floor function. The *yellow colored regions* A, B, and C indicate the slabs for the joint level set for $c = (1, 2)$

The changes in the JCN based on this coarseness parameter is another option to analyze the data not given in the Pareto set concept we present in the next section.

For an alternative approach to JCN, note that $round^{-1}(c)$ is a continuous n-dimensional interval in $\mathbb{R}^d$ with a lower bound l_i and an upper bound u_i in each of the n dimension. Each slab containing x, with $f(x) = c$, can also be defined by a set of isosurfaces $f_i^{-1}(l_i)$ and $f_i^{-1}(u_i)$ for each $1 \leq i \leq n$. In Fig. 1 the slabs A, B, and C are bounded by the fibers with $f(x) = 1$ or $f(x) = 2$, and $g(x) = 2$ or $g(x) = 3$.

Note that the contour for $l_i \in \mathbb{R}$ is equivalent to the set of fibers for all $c \in \mathbb{R}^n$ with $c_i = l_i$ such that this set builds a *fiber surface* [5]. Hence, the domain M can either be separated into slabs by a *round*-function or by a set equivalent to contours or fiber surfaces, respectively.

Increasing the number of used fiber surfaces to infinity without having the same contour twice or, analogously, an infinitely fine *round*-function will result in a structure equivalent to the Reeb space.

4 Pareto Sets

In contrast to JCN, in which a slab can contain several vertices and depends on the *round*-function or the set of used isosurfaces, the Pareto set is calculated for every simplex separately. To decide if a point x is *Pareto extremal*, the ascending set

$$H^+_{U(x)}(x) = \{y \in U(x) \mid f_i(x) < f_i(y) \forall_i\}$$

and the descending set

$$H^-_{U(x)}(x) = \{y \in U(x) \mid f_i(x) > f_i(y) \forall_i\}$$

are calculated for a sufficiently small open neighborhood $U(x)$ around x. If either of these sets is empty, the point x is Pareto extremal. Assuming that the function values are only given at the vertices and barycentric interpolated otherwise, it is sufficient

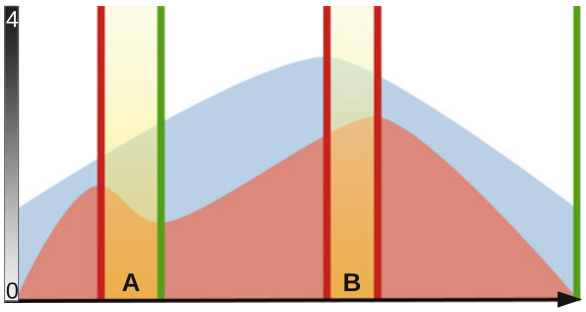


Fig. 2 For two functions f and g, colored *red* and *blue*, respectively, the regions A and B indicate the location of the Pareto extrema. Pareto optima that have both empty ascending and descending sets are implied by *yellow color*. Pareto minima and Pareto maxima either have an empty descending or ascending set, respectively, and their positions in are shown in the image by *green* and *red strips*, again respectively

to calculate Pareto extremity for one point inside a simplex to determine the status of all points inside that simplex.

The Pareto set, the set of all Pareto extrema, was defined by Huettenberger et al. [14] on piecewise linear scalar fields. The authors also introduced the definition of Pareto maxima, Pareto minima, and Pareto optima, special cases of the Pareto extrema. However, for this paper, the general definition of Pareto extrema is sufficient. Figure 2 illustrates the Pareto set in a simple one-dimensional example.

Note that both, the Pareto set and the JCN, are invariant to some transformations of the function values. For Pareto sets those transformations have to be continuous, monotonically increasing while for JCNs linear transformations are sufficient. However, then also the *round*-function and thus the slab width has to be transformed.

5 Runtime Complexity Comparison

Huettenberger et al. used marching triangle and marching tetrahedron [14] algorithms for $d = 2$ and $d = 3$, respectively, to calculate the ascending and descending set inside the simplices of highest order, i.e. for $d = 2$ inside triangles and for $d = 3$ inside tetrahedrons with a worst-case running time of $O(d^n)$.

For points inside simplices of lower order, the sets are calculated for all adjacent high-order simplices separately such that the total worst-case running time is $O(d^n \cdot N \cdot d!)$ with N the number of high-order simplices.

However, the question wether the ascending or descending set is empty or not, can be seen as a linear inequation system where each underlying function provides one linear inequation [13]. Solvers for those systems are known to run in $O(n^{3.5}L^2)$ and better, with L the number of bits in the input [15]. Such an improvement therefore suggests a running time of $O(n^{3.5} \cdot N \cdot d!)$ and is mentioned as future work by Huettenberger et al. but not yet published.

The time to calculate the JCN was bounded from above by Carr and Duke by $O(nN_e + N_e\alpha(N_e))$ with $N_e = O((2n+d)kN)$, with parameters n, N, and d as above, the upper bound of slabs k, and α the slow-growing inverse Ackermann function.

Note that the number of slabs k depends on the *round*-function and can be limited by $k < \Pi_{i=1}^{n} c_i$ with c_i the number of used isosurfaces for function f_i. Hence, under the assumption that an approach using a linear inequation solver is provided in future work for Pareto sets, the major difference in running time between JCNs and Pareto sets depends on the ratio of $d! < d^d$ to $k < c_{max}^d$, $c_{max} = \max c_i$ and thus of d to c_{max}.

6 Theoretical Comparison

As mentioned in the introductory paper of Pareto sets [14], Pareto extremality of a point depends on the orientation of the underlying functions, i.e. whether $f_i(x)$ is considered or $-f_i(x)$ instead.

The JCN however is not effected by such a consideration. To facilitate comparison, we introduce the structure of a *directed joint contour net* (dJCN) which uses directed edges. If two slabs are adjacent, the edge between the corresponding nodes is directed towards the node with the higher rounded function values. In case this classification is ambiguous because one underlying function f_i has a higher rounded function value and another f_j has a lower one the edge is excluded from the dJCN.

The extension from a JCN to a dJCN is straightforward to implement and does not influence worst-case running time stated in the previous section. Hence, while a comparison of JCN and Pareto extrema is rather unfruitful, the inexpensive consideration of orientation results in some direct relation between the dJCN and the Pareto set. Furthermore this relation can then be directly translated into relations between JCNs and Pareto sets since the relation between dJCNs and JCNs, namely the difference between directed and undirected edges, is straightforward.

The usage of directed edges allows the definition of *critical slabs* in a dJCN. Slabs are called critical if their corresponding nodes have only incoming or only outgoing edges. It is possible to distinguish maximal and minimal critical slabs similar to Pareto maxima and Pareto minima, also introduced by Huettenberger et al. [14]. However, this is beyond the scope of this paper and not necessary for the remaining work. Furthermore, note that neither the complete definition of Pareto sets given by Huettenberger et al. [14] nor the definition of critical nodes for the dJCN supports an equivalent to saddle points.

Hence, we focus an critical slabs, and, in the remaining section, show the relation between those and Pareto extrema under the assumption of sufficiently small slabs.

In this context, sufficiently small means that $round(f(x)) \approx f(x)$, especially neither the ascending nor descending set of x become empty due to rounding inside the slab. The latter case is possible if the slab contains maxima or minima with persistence smaller than the rounded range in the slab S, namely $[\min_{x \in S} f(x), \max_{x \in S} f(x)]$.

Note that in the limit, when $round(f(x)) = f(x)$ such that each point x is a slab itself, the neighborhood $U(x)$ consists of all adjacent slabs of x. If the edge between x and an adjacent slab y is in the dJCN, the point y can clearly be placed in either $H^+_{U(x)}(x)$ or $H^-_{U(x)}(x)$ depending on the edge direction. This translation from ascending set and descending set to edge directions in the limit straightforwardly implies a translation from the set of Pareto extrema to the set of critical slabs. Hence both sets are equal in the limit.

For practical reasons, though, instead of the limit only $round(f(x)) \approx f(x)$ can be reached. However, we claim that already a sufficiently fine *round*-function is enough such that the critical slabs are a good approximation of the Pareto set.

To back up this assumption, we investigate in two directions. First, we consider critical and non-critical slabs and study if the unrounded function f has Pareto extrema in the same location, or not. Contrariwise in the second direction, we consider Pareto extrema and regular points and study if the slabs that contain these points are critical or not.

To simplify the following theoretical considerations, for the remainder of this paper we assume no two fiber surfaces used during the construction of the JCN are locally equal. A property that otherwise can be achieved by only slight perturbation of the data. This implies that the rounded function values of two adjacent slabs only differ in one of the underlying functions while all other values are the same. Hence, during transformation from JCN to dJCN no edges are removed.

Direction 1 The status or criticality of a slab is based on the number of in- and outgoing edges of the corresponding node in the dJCN. Thus, each slab can be placed in one of four groups.

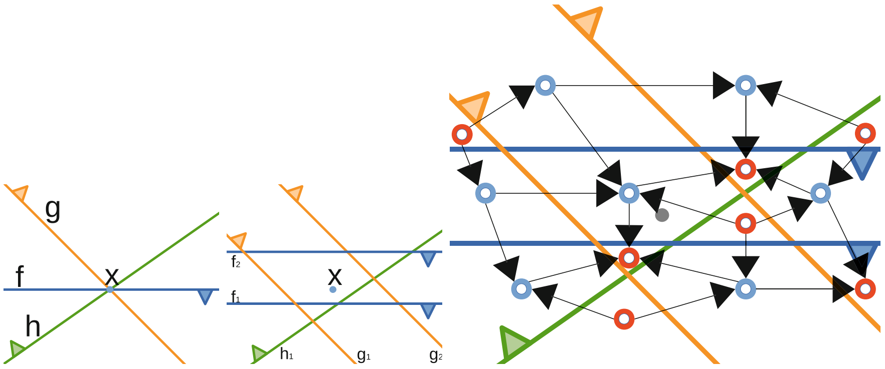
Case 1.1 Consider a critical slab whose corresponding node, without loss of generality, only has incoming edges. Hence, all adjacent slabs contain only values with equal or lower function values. Starting from any border point of the critical slab, moving in ascending direction, this path cannot leave that slab. Furthermore, since the slab is a finite space and an ascending path cannot run in circles, the path has to reach an end at a Pareto maximum inside that slab.

Ascending direction requires that at least one unrounded function value increases which implies that the functions are not allowed to be all constant inside the slab.

Case 1.2 The same holds if the corresponding node only has outgoing edges, though then the critical slabs contains a point with an empty descending set.

Note that the absence of incoming edges or outgoing edges does not imply a statement regarding the descending set or ascending set, respectively. Hence, a critical slab implies only the existence of a Pareto extremum and not if those are Pareto minimal or Pareto maximal.

Case 1.3 For the third, special case assume that a node has neither incoming nor outgoing edges in the dJCN. However, the JCN and therefore the dJCN only has isolated nodes, if the functions f_i are relatively flat or the *round*-function is very coarse and the JCN only consists of one node. If the functions are flat, every point is Pareto optimal and otherwise the *round*-function needs to be refined.



(a) (b) (c)

Fig. 3 Given three linear functions f, g, and h and a Pareto extremal point x, (**a**) shows the isosurfaces for each function based on the function value at x—the *central blue point* at the isosurfaces' intersection—while *triangles* at these isosurfaces point towards the ascending direction of the functions. Since f, g, and h are linear, all isosurfaces based on the same function are parallel to each other. (**b**) shows the same three functions and central point x as (**a**) though with a different set of isosurfaces. The labeling at these *lines* indicate the functions on which the isosurfaces are based. Using this set of isosurfaces, slabs, and the corresponding dJCN are created and presented as an overlay in (**c**). Nodes are printed as circles inside their corresponding slabs and *red circles* mark critical slabs. Note that the slab containing x is not critical however, since at least two of its bordering isosurfaces are based on the same underlying function

Case 1.4 Conversely, consider a non-critical slab that corresponds to a node with both incoming and outgoing edges, as Fig. 3 illustrates. Note how this does not imply the non-existence of Pareto extrema inside that slab. While all our following theoretical examples suggest that if a slab contains a Pareto extremum, at least one slab nearby is critical; we were not able to prove this so far.

Direction 2 For the opposite direction, from Pareto extrema to critical slabs, we consider different cases based on the ascending and descending set. Note that the ascending set, $\{y \in U(x) \mid f_i(x) < f_i(y) \forall_i\}$ for some point $x \in M$, if non-empty, is bordered by isosurfaces $\{y \in M \mid f_i(x) = f_i(y)\}$ for some $1 \leq i \leq n$ and $U(x)$. The same holds for the descending set.

Case 2.1 Assume a Pareto extremum $x \in M$ with a non-empty ascending set. Since the functions are linear inside highest-order simplices due to the barycentric interpolation, ascending and descending sets are symmetric for points inside these simplices. Hence, either also the descending set of x is non-empty, a contradiction of the definition of Pareto extrema, or x lies inside a proper face of a d-simplex σ.

Furthermore, let $x' \in \sigma$ with $x' \in H^+_\sigma(x)$ and $|x - x'| \leq \epsilon$ for some small value ϵ such that $f_i(x) \leq f_i(x')$ for all $1 \leq i \leq n$. Note that based on the symmetry of ascending and descending sets this implies that $x \in H^-_\sigma(x')$. Also let the ascending set in σ of x be bordered by fiber surfaces based on some fixed function values $f_i(x)$.

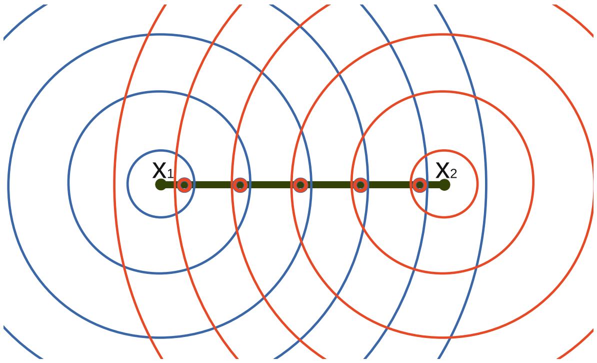


Fig. 4 The image shows a section of two sets of isosurfaces colored in *red* and *blue*, respectively, based on two distance functions with x_1 and x_2, respectively, as centers. The visible section also shows the Pareto extrema colored in *green* and critical slabs based on those isosurfaces marked with a *red circle*

The symmetry again implies that the descending set in σ of x' is bordered by the same functions though based on the values at x'.

Hence for a slab based on the fiber surfaces for fixed function values at $f_i(x')$ we can conclude the following statement: first that x is inside this slab, second, if ϵ is small enough, the slab is only bordered by those fiber surfaces, and third, for each of these borders the function values increase towards the slab. Hence, the slab containing x is critical. Note however, this does not hold if the slab is also banded by other fiber surfaces, especially those with different fixed values $f_i(x)$ but for the functions f_i. Figure 4 shows such a counterexample.

Case 2.2 A Pareto extremal point with a non-empty descending set is analogous to the first case again based on the symmetry of the ascending and descending sets.

Case 2.3 Hence, excluding the two previous cases, a point is Pareto extremal if $H^+_{U(x)}(x) = H^-_{U(x)}(x) = \emptyset$. Following the definitions of H^+ and H^-, this directly implies that for every point $y \in U(x)$ there exist two functions f_i and f_j such that $f_i(x) < f_i(y)$ and $f_j(x) > f_j(y)$.

For every function f_i we choose a point $y \in U(x)$ with $f_i(x) < f_i(y)$ to construct a isosurface $\{z \in U(x) \mid f_i(z) = f_i(y)\}$. Note that these isosurfaces enclose x and thus creating a slab. Otherwise a point x' would exist with $f_i(x) \leq f_i(x')$ for all $1 \leq i \leq n$ in contradiction to our assumption that both ascending and descending sets are empty. An example of such a slab and its corresponding isosurfaces based on the function from Fig. 3 is presented in Fig. 5.

Since $round(f_i(x)) < round(f_i(y))$ for any y in an adjacent slab and f_i the function on which the bordering isosurface is based on, the created slab is critical.

Again, a counterexample is provided in Fig. 3 showing that a Pareto extremum does not enforce the existence of a critical slab at the same position. Especially if the slab containing x has at least two bordering fiber surfaces with fixed function

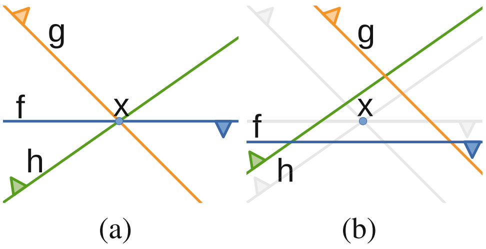


Fig. 5 Given the same functions as in Fig. 3, reprinted in (**a**), the second image (**b**) uses a different set of isosurfaces. Using these isosurfaces, the created slab which contains the central point x is critical. To see this, note that the *blue triangles* which point towards the ascending directions of the functions also indicate the edge directions in the corresponding dJCN

values $f_i(u)$ and $f_i(v)$, respectively, based on the same underlying function f_i and with $f_i(u) < f_i(x) < f_i(v)$.

Case 2.4 For the last case, namely a point $x \in M$ that is not Pareto extremal, let $i_0, \ldots, i_k$ be the indices of functions such that the fiber surfaces with fixed values in these functions border the ascending set of x. Note that a slab created through the isosurfaces $\{y \in M \mid f_{i_j}(y) = f_{i_j}(x) - \epsilon\}$ and $\{y \in M \mid f_{i_j}(y) = f_{i_j}(x) + \epsilon\}$ for every $0 \leq j \leq k$ is obviously not critical since every pair of those isosurfaces provide incoming and outgoing edges for the corresponding node in the dJCN. This is under the assumption that ϵ is very small such that all functions can be considered linear.

Assume that some additional isosurface $\{y \in M \mid f_{i'}(y) = c\}$ splits the slab such that the created slab containing x becomes critical. Without loss of generality, let the critical slab only have outgoing edges, which implies $c > f_{i'}(x)$. Since we assumed ϵ to be sufficiently small such that $|c - f_{i'}(x)|$ is also sufficiently small, $f_{i'}$ can be considered linear close to x. Hence, $\{y \in M \mid f_{i'}(y) = f_{i'}(x)\}$ runs parallel to the additional isosurface at least in a small neighborhood around x.

Therefore, for every point y in this neighborhood it holds that $f_i(x) \leq f_i(y)$ for either $i = i'$ or some $i = i_j$. This implies x to be Pareto extremal in contradiction to the original assumption.

Summary Given that the underlying functions are linear in a close neighborhood around x, for every Pareto extremum we find a set of fiber surfaces and thereby a suitable *round*-function such that the slab containing the extremum is critical. Analogously for points that are not Pareto extremal, non-critical slabs can be created.

Note that we assumed small slabs but also indicated that splitting slabs with additional fiber surfaces to create smaller ones usually results in non-critical slabs. Hence, reducing the slab size by introducing new fiber surfaces to receive a better correlation between critical slabs and the Pareto set is not trivial. Furthermore, the existence of suitable *round*-functions for each Pareto extremum does not imply the existence of a single *round*-function under which the set of all Pareto extrema is equivalent to the set of all critical slabs.

A main result is that the location of a critical slab implies a Pareto extremum inside that slab. However, these consideration do not provide a statement about the approximation error, in other words the volume ratio between the critical slab and the Pareto set. In future work and in illustrations in the next section we are going to investigate that under a finer *round*-function this error will be reduced.

7 Visual Comparison

We applied dJCN and Pareto set calculation on the LAMPS data set [2] set and some artificial data sets to identify further relationship between those concepts.

LAMPS stands for "Limited Area Mesoscale Prediction System" an atmospheric simulation, is distributed with the VIS5D system, and provides a set of 10 scalar fields, including wind, temperature, pressure, and specific humidity. We focus on the east/west component (U) and the north/south component (V) of wind.

Note that as an additional step we smoothed the LAMPS data with a Gaussian filter [18] before calculating the Pareto set. Thereby we removed smaller noise-based features, which is automatically done by the *round*-function in the dJCN method.

The following figures present the dJCN and the Pareto set for this and other, artificial, data sets. In the right images, Pareto extrema are colored red, green, or yellow, depending on whether the ascending, descending, or both sets are empty. In the left images, the nodes of the dJCNs corresponding to critical slabs are colored red and blue, depending on the existence of only incoming or only outgoing edges. Otherwise, the nodes are either colored green or, to avoid obscuring, have been reduced to points.

Figure 6 shows the dJCN and Pareto set for a synthetic example based on two functions. At each point, the first is based on the distance to the center of the domain

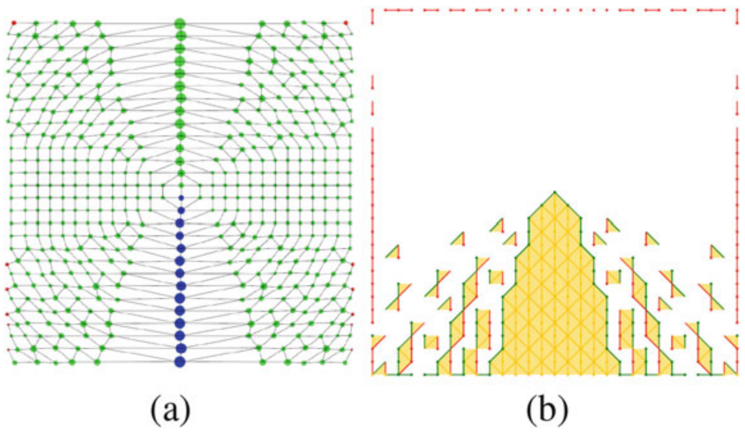


Fig. 6 dJCN and Pareto set for a synthetic example based on a high and a distance function

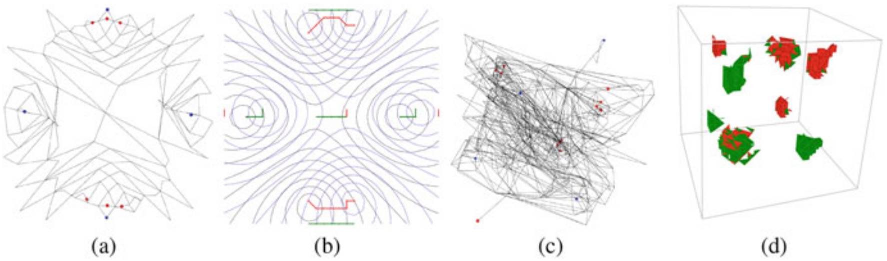


Fig. 7 dJCNs(**a, c**) and Pareto sets (**b, d**) for two synthetic examples based on a set of Gaussian functions in 2D and 3D, respectively. (**b**) Shows isosurfaces in *blue* and *gray* for the 2*D* functions

and the second is based on the value of the *y*-direction. Note how the pyramid-shaped component of Pareto extrema in the lower middle corresponds to the row of critical nodes at the same location in the dJCN image. This figure also shows some issues the Pareto set calculation has with degenerated functions, especially locations where adjacent vertices have the same function values. Such functions together with the triangulation of the data result in faulty Pareto extrema left and right to the mentioned pyramid-shape and the domain border. Note that this does not seem to be an issue for the calculation of the dJCN.

To consider further synthetic data sets, we used those presented in the paper for Pareto sets [14]. Therefore, for each underlying function a set of fixed points in $\mathbb{R}^2$ for Fig. 7a, b and in $\mathbb{R}^3$ for Fig. 7d, c is given and the Gaussian function value is calculated as the sum of the exponentiated and weighted distance maps for these fixed points. For our examples, between the underlying functions only the location of the fixed points is moved by some degree.

As with the previous images, note the correspondence between most of the connected components of Pareto extrema with the groups of colored nodes in the dJCN. However, there are also Pareto extrema which cannot be linked to a critical slab, for example the central component in Fig. 7a and the larger two components in Fig. 7d bordered by both red and green colored triangles. Those components correspond to the location of saddle points in terms of a univariate topology. Therefore and since the slab width is too large as the isosurfaces in Fig. 7b show, the corresponding nodes in the dJCN have incoming as well as outgoing edges. Hence, Fig. 7a does not contain critical slabs in the image center, even though Fig. 7a clearly shows Pareto extrema at this position.

The last example is based on the LAMPS data set introduced in the beginning of this section. Figure 8 presents the Pareto set and the dJCN, for which each node is positioned in the barycenter of their corresponding slab. The figure shows how both visualizations have problems with 3*D* data. Interactivity, i.e. rotation, zoom, etc., with the dJCN and the Pareto set helps to identify corresponding components of Pareto extrema and groups of critical nodes. While we highlighted some of these pairs through yellow circles, a complete identification of all links has to be computer-based and is part of future work.

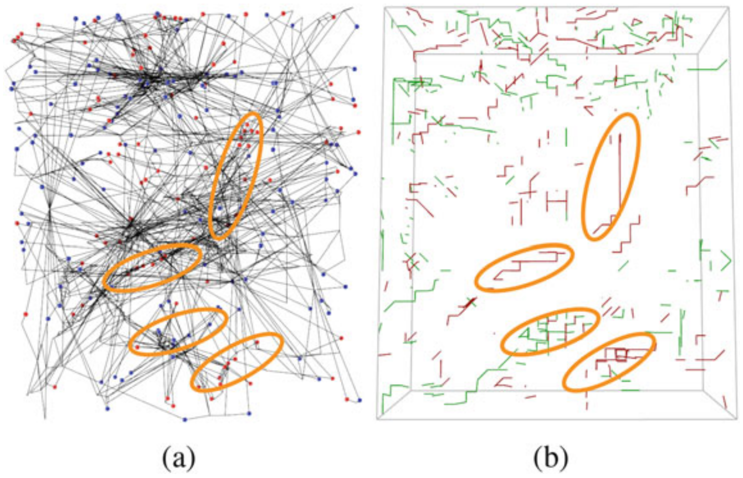


Fig. 8 dJCN and Pareto set for the multivariate function based on the U and V component of the LAMPS data set. The grid has over 21,000 vertices which results in 95,200 tetrahedrons. For the dJCN the slab width is set to 5 for each underlying function. The calculation of 895 nodes and 1667 edges took around 25 s. As suspected, the Pareto set calculation used a larger amount of time, around 2 min. The *encircled areas* highlight recognizable similarities between the critical slabs of the dJCN and the Pareto set

Note that some critical nodes in Fig. 8a do not seem to correspond to any Pareto extremum in Fig. 8b. However, also note that we made Pareto extrema that lie on the frontal domain border completely transparent to avoid occlusions.

8 Conclusion

As a final remark, consider that points are Pareto extremal if the underlying functions disagree so to say in terms of ascending and descending direction and are regular if there is an agreement. This agreement is similar to the question of how close the isosurfaces are to be parallel which is closely related to the size of the slabs based on these isosurfaces. While our artificial data suggest such a relationship, we were not able to see some evidence for this assumption between the structures in the LAMPS dataset seen in the dJCN and the Pareto extrema.

In conclusion, our theory suggest that for a sufficiently fine *round*-function, critical slabs in dJCNs and Pareto sets are sufficiently equivalent and even completely the same in the limit. Though, latter would result in unacceptable running time to calculate the dJCN. However, our examples show that also a relatively coarse *round*-function provides a well enough approximation of the Pareto set and is computed much faster than the Pareto set itself in the current published form. The dJCN thereby provides the location of critical slabs and how those are connected, though just an approximation of the real data.

The location of the critical slabs can then be used to calculate the Pareto set. First, Pareto extrema are only identified in those critical slabs and then the calculation is expanded to adjacent simplices until no more Pareto extrema are found.

This provides an exact shape of the connected components in the Pareto sets, regions where the underlying functions are conflicting, while the dJCN provides an approximated map on how those components are interconnected to each other in the region where the underlying functions do not conflict with respect to ascending and descending directions.

Furthermore, this map, in particular the paths between critical slabs in the dJCN following the directed edges, is a discretization of ascending paths. The latter are defined by Huettenberger et al. [14] as continuous paths between Pareto extrema in which the function values between subsequent points on the path either stay the same or increase. Note that these paths are calculated separately in the Pareto set paper which results in even further running time while an approximation is already given in the dJCN almost for free.

Hence, under the examples and theory given in the previous sections, we conclude that dJCNs are a good approximation of Pareto sets and the connections inside these sets and can be used to speed up the calculation of Pareto sets itself. While the Pareto set and its inner connections are close to the data but take long to compute, the dJCN calculation is only an approximation but runs fast and is just a small extension to the JCN calculation which provides further options, like the coarseness of the *round*-function to analyze the data. How exactly the coarseness influences the approximation error is a task for future work.

Finally note that our work can be translated easily to results regarding the Reeb space because that structure is also approximated by the JCN [2] and equal in case of an infinitively fine *round*-function. Thus, all our considerations can be done also for a directed form of the Reeb space and, combined with prior work [13], indicates close relations between Jacobi set, Reeb space, JCNs, and Pareto set.

References

1. Carlsson, G.E., Singh, G., Zomorodian, A.: Computing multidimensional persistence. J. Comput. Geom. **1**(1), 72–100 (2010)
2. Carr, H., Duke, D.J.: Joint contour nets. IEEE Trans. Vis. Comput. Graph. **20**(8), 1100–1113 (2014)
3. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. **24**(2), 75–94 (2003)
4. Carr, H., Snoeyink, J., van de Panne, M.: Flexible isosurfaces: Simplifying and displaying scalar topology using the contour tree. Comput. Geom. **43**(1), 42–58 (2010)
5. Carr, H., Geng, Z., Tierny, J., Chattopadhyay, A., Knoll, A.: Fiber surfaces: Generalizing isosurfaces to bivariate data. Comput. Graph. Forum **34**(3), 241–250 (2015)
6. Chen, C.H., Härdle, W.K., Unwin, A. (eds.): Handbook of Data Visualization. Handbooks of Computational Statistics. Springer, Berlin (2008)
7. Chen, G., Mischaikow, K., Laramee, R.S., Zhang, E.: Efficient Morse decompositions of vector fields. IEEE Trans. Vis. Comput. Graph. **14**(4), 848–862 (2008)

8. Doraiswamy, H., Natarajan, V.: Efficient algorithms for computing Reeb graphs. Comput. Geom. **42**(6-7), 606–616 (2009)
9. Doraiswamy, H., Natarajan, V.: Computing Reeb graphs as a union of contour trees. IEEE Trans. Vis. Comput. Graph. **19**(2), 249–262 (2013)
10. Edelsbrunner, H., Harer, J.: Jacobi sets. In: Cucker, F., DeVore, R., Olver, P., Süli, E. (eds.) Foundations of Computational Mathematics: Minneapolis, 2002, pp. 37–57. Cambridge University Press, Cambridge (2004)
11. Edelsbrunner, H., Harer, J., Patel, A.K.: Reeb spaces of piecewise linear mappings. In: Proceedings of the Twenty-fourth Annual Symposium on Computational Geometry, SCG '08, pp. 242–250. ACM, New York (2008)
12. Forman, R.: Morse theory for cell complexes. Adv. Math. **134**(1), 90–145 (1998)
13. Huettenberger, L., Garth, C.: A comparison of Pareto sets and Jacobi sets. In: Bennett, J., Vivodtzev, F., Pascucci, V. (eds.) Topological and Statistical Methods for Complex Data: Tackling Large-Scale, High-Dimensional, and Multivariate Data Spaces, pp. 125–141. Springer, Berlin (2015)
14. Huettenberger, L., Heine, C., Carr, H., Scheuermann, G., Garth, C.: Towards multifield scalar topology based on Pareto optimality. Comput. Graph. Forum **32**(3), 341–350 (2013)
15. Karmarkar, N.: A new polynomial-time algorithm for linear programming. In: Proceeding of STOC, pp. 302–311. ACM, New York (1984)
16. Kniss, J., Kindlmann, G.L., Hansen, C.D.: Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In: Proceeding of IEEE Visualization, pp. 255–262. IEEE Computer Society, New York (2001)
17. Nagaraj, S., Natarajan, V., Nanjundiah, R.S.: A gradient-based comparison measure for visual analysis of multifield data. Comput. Graph. Forum **30**(3), 1101–1110 (2011)
18. Nixon, M.S., Aguado, A.S.: Feature Extraction and Image Processing, 2nd edn. Academic Press, New York (2008)
19. Pascucci, V., Cole-McLaughlin, K.: Parallel computation of the topology of level sets. Algorithmica **38**(1), 249–268 (2003)
20. Sauber, N., Theisel, H., Seidel, H.: Multifield-graphs: an approach to visualizing correlations in multifield scalar data. IEEE Trans. Vis. Comput. Graph. **12**(5), 917–924 (2006)
21. Schneider, D., Heine, C., Carr, H., Scheuermann, G.: Interactive comparison of multifield scalar data based on largest contours. Comput. Aided Geom. Des. **30**(6), 521–528 (2013)
22. Szymczak, A., Zhang, E.: Robust Morse decompositions of piecewise constant vector fields. IEEE Trans. Vis. Comput. Graph. **18**(6), 938–951 (2012)

Part II
Topological Techniques for High-Dimensional Data

Visualizing Topological Properties of the Search Landscape of Combinatorial Optimization Problems

Sebastian Volke, Dirk Zeckzer, Martin Middendorf, and Gerik Scheuermann

Abstract Discrete combinatorial optimization problems such as the Traveling Salesman Problem have various applications in science and in everyday life. The complexity of the problem and the effectiveness of search algorithms depend not only on the problem itself but also on the search operator in use. Therefore, investigating search operators and the search landscapes they give rise to is an important field of research. However, a full topological analysis of the landscapes is impossible due to their exponentially growing size. We propose a visualization system that gives a visual intuition about topological properties of the search landscape. We obtain representative samples of the search landscape and its optima by random sampling and by computing the related optima using local search. The distribution and the correlation of this data within the search landscape is visualized with a combination of one and two dimensional visualizations. Using the TSP as an example we illustrate how the visualization supports the understanding and comparison of search landscapes and their complexity.

1 Introduction

Optimization is an important field of research with many applications in economy, engineering, natural sciences, and operations research. A frequent class of problems are *discrete, combinatorial* optimization problems, where an optimal solution has to be found from a finite set of solutions. Typically, the solution space grows exponentially in the problem size. Therefore, exhaustive search by complete enumeration is only possible for very small problem instances. In practice, optimal solutions can only be approximated using heuristic search algorithms.

One class of these algorithms are local search algorithms [18] that construct a graph—the *search landscape*—on top of the set of solutions (cf. Sect. 2) and search optimal solutions by traversing it. Thereby, the connectivity in the graph

S. Volke (✉) • D. Zeckzer • M. Middendorf • G. Scheuermann
Leipzig University, Institut für Informatik, Leipzig, Germany
e-mail: volke@informatik.uni-leipzig.de; zeckzer@informatik.uni-leipzig.de; middendorf@informatik.uni-leipzig.de; scheuermann@informatik.uni-leipzig.de

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_4

is determined by a search operator. The topological complexity of the search landscape, e.g., the number of local minima or the average length of search paths, depends on the interaction between the search operator and the optimization function. Optimization is much easier in search landscapes with simpler topology. Thus, an optimization problem can be solved by finding a search operator that induces a preferably simple search landscape. The analysis of search landscapes is crucial in identifying well-performing search operators. However, to our knowledge, few effective and generally applicable visualization approaches exist that support this task.

In this paper, we build upon a method for analyzing search landscapes that is based on steepest descent walks in the landscape [40]. The method does not depend on a specific problem definition and thus is widely applicable. Our main contribution is the introduction of a visualization system that facilitates the fast and easy interpretation of the analysis results. We confirm the correctness of the method by reproducing known results about the Traveling Salesman Problem. Thereby, we also showcase the effectiveness of both, the method and the visualization system.

2 Background

Discrete combinatorial optimization addresses the selection of the best solution out of a finite set of solutions X of a problem instance. The notion of one solution being "better" than another is usually modeled by means of a cost function $f : X \to \mathbb{R}$ where $x \in X$ is better than $y \in X$ if $f(x) < f(y)$. Thus, one searches for a solution that minimizes f.

In this work, we are particularly interested in permutation problems, i.e., optimization problems where the set of solutions is a set of permutations of size n or a subset thereof. In the following, n is referred to as the *problem size*. Many practically relevant permutation problems are NP-hard [10], a famous example being the *Symmetric Traveling Salesman Problem* (STSP), which is defined as follows: given n locations and for each two locations $1 \leq i,j \leq n$ a distance $d_{ij} = d_{ji}$, find a shortest route that visits each location exactly once and returns to the initial location. This is equivalent to finding a permutation π of the locations that minimizes $f_{TSP}(\pi) = \sum_{i=1}^{n-1} d_{\pi(i),\pi(i+1)} + d_{\pi(n),\pi(1)}$.

A major problem with permutation problems in general and the STSP in particular is the factorial growth of the set of solutions with respect to the problem size. This makes a complete enumeration of all possible solutions computationally infeasible even for relatively small values of n. Therefore, different search strategies have been developed for efficiently finding optimal or near optimal permutations.

Local search methods rely on the notion of a neighborhood among the solutions. We define a neighborhood as a mapping $N : X \to 2^X$ that associates every solution with a set of neighbor solutions. The set of solutions together with the neighborhood define a neighborhood graph $G_{X,N} = (X, E_N)$. The set of directed edges is defined as $E_N = \{(x, y) | x \in X \wedge y \in N(x)\}$ and reflects the neighborhood function. This type of graph was introduced, e.g., in computational biology already in 1932 [42] and is often called a *landscape*. Following this terminology, we refer to the neighborhood graph as the *search landscape* of the optimization problem for the given neighborhood. Local searches can then be viewed as locally improving (with respect to f) walks in the search landscape, starting from some initial solutions.

Local search algorithms can be characterized with respect to the strategy that they use for traversing the search landscape (e.g., steepest descent or random walks) as well as with respect to the neighborhood that they use. The latter can be described by means of a search operator. Following Schiavinotto et al. [30], we define a search operator Δ as a collection of operator functions $\delta : X \to X$ such that $y \in N(x) \iff \exists \delta \in \Delta$ with $\delta(x) = y$. Then, the edge set of the landscape can also be defined as $E_N = E_\Delta = \{(x, y) | x \in X \wedge \exists \delta \in \Delta : y = \delta(x)\}$. Thus, we can define the search landscape through the search operator as $G_{X,\Delta} = G_{X,N} = (X, E_\Delta = E_N)$. In this paper, we restrict ourselves to operators that result in a connected and symmetric (i.e., for each two nodes i, j holds $(i, j) \in E_\Delta \iff (j, i) \in E_\Delta$) search landscapes.

The topological structure of f on the search landscape $G_{X,\Delta}$ depends both on the cost function f as well as on the search operator Δ. This includes the number of local minima, the number of paths to these minima, and the lengths of these paths. Therefore, the topology of the search landscape reflects how well the search operator matches the optimization problem. In particular, we are interested in shortest paths between solutions in the search landscape. Due to the construction of $G_{X,\Delta}$, every edge in a path between solutions x and y corresponds to one application of an operator function from Δ. Therefore, the graph theoretic distance between x and y, i.e., the length of the shortest path between x and y, is equal to the minimal number of applications of operator functions from Δ that is needed to transform x into y. We denote this distance with $d_\Delta(x, y)$ for $x, y \in X$. For many operators, there exist efficient algorithms to compute the distance between two solutions or at least a good approximation for this distance (cf. [30]).

The following two operators are of particular interest and will serve as examples throughout the paper. The *2-opt operator* Δ_{2opt}, also known as 2-edge exchange operator, is a well established search operator for TSP [4, 19]. Basically, it disentangles a route by eliminating edge crossings, i.e., by flipping the order of some consecutive cities in the route (see Fig. 1a for an illustration). Formally, the application of this operator corresponds to the reversal of a subsequence of a (circular) permutation. The corresponding distance is called reversal distance. Unfortunately, the computation of this distance—also called the *sorting by reversal* problem—is NP-hard. However, the *bond distance* provides a good approximation for it: the number of edges is counted that are not in common between the two tours. The bond distance differs from the reversal distance at most by a factor of 2 as proven by Boese [2].

The *interchange operator* $\Delta_X = \{\delta_X^{ij} | 1 \leq i < j \leq n\}$ is the set of transpositions, i.e., $\delta_X^{ij}(\pi) = (\pi_1 \dots \pi_{i-1} \pi_j \pi_{i+1} \dots \pi_{j-1} \pi_i \pi_{j+1} \dots \pi_n)$ for each permutation $\pi = (\pi_1 \dots \pi_n)$. As illustrated in Fig. 1b, this corresponds to a swap of two permutation

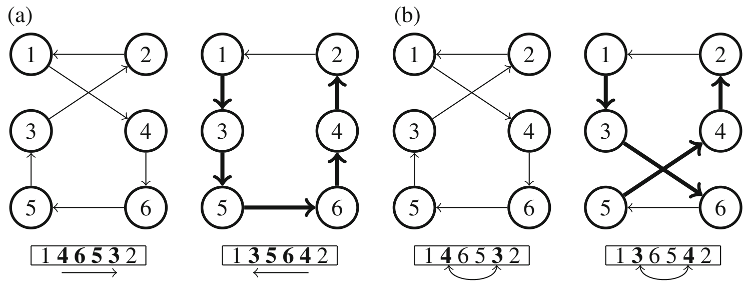


Fig. 1 Illustration of the functioning of the 2-opt operator Δ_{2opt} (*left*) and the interchange operator Δ_X (*right*) on the TSP problem. The operator function characterized by the indices 5 and 8 is applied to the permutation that corresponds to the shown round-trip. The result of the application is shown on the right for both operators, whereas the *arrows* indicate how the elements of the permutation are shifted around. (**a**) 2-opt operator. (**b**) Interchange operator

elements. The distance between two permutations with respect to Δ_X can be determined by $d_{\Delta_X}(\pi, \pi') = n - c(\pi^{-1} \circ \pi')$; see Schiavinotto and Stützle [30] for details. Thereby, $c(\pi)$ is the number of cycles of the permutation π, $\circ$ is the composition of permutations, i.e., $(\pi_1 \circ \pi_2)(i) = \pi_1(\pi_2(i))$, and π^{-1} is the inverse permutation, i.e., $\pi \circ \pi^{-1} = \pi^{-1} \circ \pi = (1 \ldots n)$.

Both search operators have $|\Delta| \in O(n^2)$. The number of solutions being $|X| \sim n!$, this results in very dense search landscapes with $|E_\Delta| \in O(n^2 \cdot n!)$.

3 Related Work

Concerning the visualization of landscapes in general, several papers have been published—starting with the original paper by Wright [42]—that use projections to create images resembling geological landscapes. Among them the work by McCandlish [25] is closely related to our approach. He uses an approximation of the number of evolutionary generations between individuals to obtain a transition or distance matrix. Afterward, he performs an Eigen-decomposition of this matrix in order to project the individuals into 2D space while preserving their evolutionary distance. Instead, we use a force-directed layout described in Sect. 5 together with additional plots. Furthermore, since our area of application is different, we use a different analysis method for acquiring the data.

Another direction of research uses the barrier tree [7] for visualizing the search landscape. Hallam and Prügel-Bennett [15] study the search landscape of SAT problems. Volke et al. [38, 39] apply the barrier tree to RNA secondary structure prediction (among others) and also integrate search results of metaheuristics into the visualization. This work has also been extended for a more in-depth analysis of search operators by Bin et al. [1]. However, all of these approaches are limited to small problem instances or problems where all solutions below a certain threshold can be enumerated efficiently. Our proposed approach does not result in a rigorous representation of the topology of the search landscape, but is applicable to problem instances that can not be exhaustively searched. Instead of a tree representation of the search landscape, we derive visualizations that allow to draw conclusions about the shape of the search landscape from a small subset of it.

There is also a couple of visualization approaches for the Traveling Salesman Problem in particular. TSPAntViz [36] is a simulation and visualization environment for analyzing the performance of metaheuristics on TSP problems. Different to our work, the focus is on the metaheuristic and on the visualization of individual solutions as opposed to a visualization of the search landscape. The same author also proposed algorithms for TSP problems on cuboids [35] and on spheres [37]. Both of them incorporate visualizations of individual solutions, i.e., routes plotted on cuboids and spheres, but no discussion or visualization of the search landscape. Halim et al. [14] propose a visualization system called “Viz” that facilitates the analysis of local search behavior. They use a force-directed layout to create a 2D embedding of important landmarks from local search runs based on the Hamming distance function between the landmarks. Then, search runs are plotted as trajectories in this depiction. Again, the focus is on the analysis of the search algorithm, not on the search landscape itself. Furthermore, the fixed distance function introduces a gap between the perceived search space and the actual behavior of the search strategy. In contrast, we use distance functions that match the search operator in order to visualize the search landscape as seen by the search strategy.

There is also much previous work about the analysis of search landscapes, e.g., [9, 11, 18, 27, 31] and others. To our knowledge, these papers mostly rely on statistical methods and do not include effective visualizations apart from simple function plots.

4 Topological Analysis of the Search Landscape

In the following, we discuss the topological analysis method of search landscapes [40] that we developed to support research in the field of swarm intelligence and optimization. The goal is to ascertain data about the search landscape that enables multiple analysis approaches known from literature. Furthermore, the method should not be specific to a certain problem definition, problem instance, or search operator. Thus, the comparison of search landscapes of different problem instances, different search operators, and even different problems should be enabled.

The basic idea is to reduce the amount of data by considering walks within the search landscape and to analyze their properties instead of analyzing the whole search landscape directly. Therefore, we obtain a set of initial solutions, called *samples*, by generating random permutations using Knuth shuffles [22]. For every such sample we have the permutation itself as well as the cost value given by the cost function of the problem instance. Additionally, we compute the steepest descent search path, i.e., we track the sample to its associated local minimum. For every path we compute certain characterizing measures:

- the length of the search path, i.e., the number of solutions in the path
- the cost difference between the initial solution and the local minimum
- the length of a *shortest* path between the initial solution and the local minimum, i.e., the distance between these two solutions

The last measure can be easily computed by means of the distance function that accompanies the search operator. These measures are often used for correlation analysis of the search landscape (cf. Chap. 5 in [18]). The measures for every path can be associated with the samples, as there is exactly one such path for every sample.

Furthermore, we derive additional sets of solutions from the search paths. First, we have the set of found local minima, which is a subset of the local minima within the search landscape. Note, that multiple steepest descent paths may end at the same local minimum. Therefore, the number of found local minima is less or equal to the number of samples. The number of distinct local minima can be used as a measure of the complexity of the problem instance and is related to the difficulty of the problem (e.g., [5, 6, 11]). Furthermore, given a number of independent group optima, it is possible to statistically estimate the cost value of the global optimum [12, 26].

The set of all solutions that lead to the same local minimum via steepest descent, are called the basin of the corresponding minimum. To some extent, information about the basins of a problem instance can be estimated from the search paths. The size of a basin can be estimated by using the sum of the lengths of all paths that lead to the corresponding local minimum. The height of the basin can be estimated from the maximum of the cost differences among the solutions on the paths. The "appeal" of the basin, i.e., the likelihood that a random solution belongs to a basin, can be estimated from the size of the basin or from the number of paths that lead to the corresponding local minimum.

Beside the set of local minima, we also derive the set of all median solutions of the generated paths. The main purpose thereof is to provide additional data for characterizing the basins and the search paths. Thereby, the median solutions are used as representatives of their paths to keep to total amount data points small. This avoids visual clutter, leads to a clear visualization (see Sect. 5), and permits interactive exploration.

To obtain information about the connectivity and adjacency within the search landscape, we compute the distance with respect to the search operator between any two solutions from all three sets (i.e., the sample set and the sets of the corresponding local minimum solutions and the corresponding median solutions). This results in six groups of distances. This approach is inspired by Fonlupt et al. [9] who have drawn several conclusions about the shape of the TSP landscape from an analysis of the mutual distances between local minima. Based on this, they introduced a new algorithmic approach—combining local search with a genetic algorithm—that is tailored toward the TSP problem. However, they have considered only distances between local minima and also used only a simple means of visualization. We generalize their approach by computing distances between different types of solutions, broadening the scope of the investigation, and accompany the analysis with a more appropriate, well-suited visualization (see Sect. 5).

4.1 Alternative Approaches

Topological methods such as the barrier tree [7] require the set of all solutions below a certain cost threshold to be available. Therefore, they cannot be applied to search landscapes of even medium sized problems because of their complexity (see Sect. 2). Furthermore, no branch-and-bound techniques (as used in [15]) that would allow for an efficient reduction of the search space are available for permutation problems in general.

There are basically two approaches in literature for dealing with large landscapes. The first one tries to acquire information about the search landscape by obtaining a representative subset of the solutions, i.e., by sampling. The other approach considers walks in the search landscape and obtains information about the whole landscape from properties of these walks [9, 31]. Both approaches are combined in our method described above.

Multiple strategies are possible for sampling the search landscape. One extreme is a full enumeration of all possible solutions. It reveals most information about the search landscape but is computationally infeasible for all but very small problem instances. Random sampling is the other extreme revealing least information. It has been shown, that much more representative samplings can be obtained when guiding and filtering the random sampling within a Monte Carlo method [21]. In particular, the obtained results contain near optimal solutions and allow reasoning about the overall distribution of costs within the landscape. However, assessing the distribution of costs is not the primary concern of this paper. Therefore, the application of a random sampling is sufficient here. Knuth shuffles are often used to generate initial solutions for search heuristics. In our tests, we found that Knuth shuffles usually generate solutions in a small range of very high costs. The combination with local search paths guarantee that local minima are included in our sampling and major topological properties of the landscape are reflected. Our sampling leads to long search paths which reveal more information about the basins.

Most analyses that exploit walks within the search landscape, apply random walks, e.g., Stadler and Schnabl [31]. While random walks also descent monotonically within the search landscape—the costs of solutions on the path are smaller than the costs of their predecessors—and they also end up in local minima, they are ambiguous: there are many possible random walks starting from every solution in the search landscape. As a consequence, every solution can be associated with multiple local minima using random walks. This makes topological notions such as basins much harder to define (e.g., [32]). On the other hand, there is exactly one steepest descent walk for every solution in the search space, so that each solution is associated with exactly one local minimum. Furthermore, there is a close connection between the basins defined by steepest descent walks and the barrier tree [7]. Therefore, we preferred steepest descent walks here.

5 Topological Visualization of the Search Landscape

From the analysis of the search landscape (see previous section), we obtain a multivariate dataset consisting of three groups of data points: the samples, the local minima, and specific intermediate solutions on the search paths. Every data point has a permutation and a cost value associated. For the set of samples we have the following additional attributes:

- the associated local minimum
- the path length to the local minimum
- the cost difference to the associated local minimum
- the distance within the search landscape to the associated local minimum

The attributes of the set of local minima are:

- the number of samples that lead to the specific minimum
- the maximal known cost difference along a path that leads to the minimum
- the number of solutions on known paths to the minimum

Additionally, the distance between any two data points is known.

The visualization of the data should facilitate two important tasks: correlation analysis (cf. [18]) and analysis of the shape of the search landscape (cf. [9]). Thus, the requirements on the visualizations are that they support these tasks effectively and that they allow to see the essential results of the analysis at a glance. Since the data is multivariate, we select and configure known visualization techniques for multivariate data visualization.

Correlation analysis, the first task, examines the correlation between different distances within the search landscape. Thereby, the cost difference between solutions and their associated minima, the length of the search paths from solutions to local minima, as well as the distance between solutions and their associated minima with respect to the search operator are compared. In particular, we are interested whether there is a proportional or anti-proportional correlation between any two of these three measures. We use scatterplots (see, e.g., [13]) to visualize relationships between these variates, since they allow to visually identify many different types of correlation [16].

The analysis of the *shape of the search landscape* is more complex and requires domain specific knowledge. Domain experts use information like the data generated (see Sect. 4) and interpret it in the light of their own experience and previous knowledge about the optimization problem (cf. the procedure described by Fonlupt et al. [9] and Fleurent and Glover [8]). First of all, the relative number of minima within the search landscape is considered. Being a single number, it can be represented as text.

Second, the distribution of cost values among the found solutions is analyzed. The distribution of the cost values is visualized by computing histograms for the three groups of data points (samples, minima, and intermediate solutions) and using stacked bar charts [41] for their presentation. This allows to analyze the cost distribution of each individual group of data points at the same time as the global cost distribution.

Third, the distances between the found solutions are investigated. We use the same approach as for the visualization of the distribution of distances. In this case, however, we have six different distributions, resulting in six bar charts, that are stacked above each other: the distributions of distances within each group (3) and the distributions of distances between any two of these groups (3). This allows investigating the distance distribution between pairs of groups in the context of the intra-group distances.

For both histogram visualizations coloring is used for distinguishing between the different groups of data points. Thereby, all data points of one group are assigned the same color. The colors of the groups are determined using a qualitative color scheme from color brewer [17]. This ensures that the groups are easily distinguishable. In the case of the distance distribution additional colors are assigned for the distances between different groups.

A topological visualization is proposed that supports the investigation of the shape of the search landscape. Therefore, we exploit that the distances between the solutions carry topological information. We use them to reconstruct a depiction of the search landscape that preserves the topological properties as good as possible. In particular, we map the solutions onto the 2D plane such that the Euclidean distances between the points in the plane approximate the distances within the search landscape. The visualization then allows to differentiate, e.g., between crater-like landscapes and landscapes that resemble large plains with many small cavities.

The mapping is done using a metric Multidimensional Scaling [3] technique. In this case, we rely on a variant of the Shepard-Kruskal algorithm which simulates a spring force model. All data points are placed at random initial positions in the 2D plane. Every data point exerts a force on every other data point. Let d_{ij} be the distance between i and j in the search landscape (according to the search operator), and be x_i the position of i in the plane. Then, the force on i is

$$F(i) = \alpha \sum_{j \neq i} \frac{\mathbf{x_j} - \mathbf{x_i}}{\| \mathbf{x_j} - \mathbf{x_i} \|} \cdot \left(d_{ij} - \| \mathbf{x_j} - \mathbf{x_i} \| \right)$$

In this case, we use $\alpha = 1/\#points$. In every iteration, the force on every data point is computed and the data points are moved accordingly ($x'_i = x_i + F(i)$). This is repeated until an equilibrium or a maximal number of iterations is reached.

5.1 Design Alternatives

Multiple visualization techniques have been proposed to facilitate the detection of correlations. Harrison et al. [16] investigate many different such techniques and evaluate their effectiveness for this task. It shows that scatterplots are the most effective technique, even superior to parallel coordinates, when different types of correlation can be present within the data.

The cost and distance distributions are multimodal and thus cannot be captured in individual statistical quantities. This excludes box plots [34] that show only those. Histograms in stacked bar charts, on the other hand, allow for a more fine-grained visualization and depict the shape of the distribution.

Multiple techniques are available for the task of projecting the data points onto the 2D plane while preserving their relative distances as good as possible. Principal Components Analysis [20] would require a spatial embedding of the original high-dimensional dataset. Because we are not aware of any useful spatial embedding of the set of permutations, we cannot apply PCA here. Projection techniques have been proposed that facilitate eigenvalue decomposition of the distance matrix, e.g., [25]. The main problem in this case is the bad scalability of these techniques [33]. Self-Organizing Maps [23] could be used, however, they require setting several parameters and the interpretation of the final layout is not straight forward.

Clustering techniques are not adequate for the problem presented in this paper. First of all, in our case, the points are already assigned to different categories: sample points, minima, and path medians. Further, as can be seen from the histograms, a distance based clustering will not adequately reflect the structure of the data (cf. Fig. 2).

Using the proposed layout algorithm has multiple advantages: data points with small distances to each other are clustered visually, the relative placement of different groups of data points represents their actual distances, and thus the relation

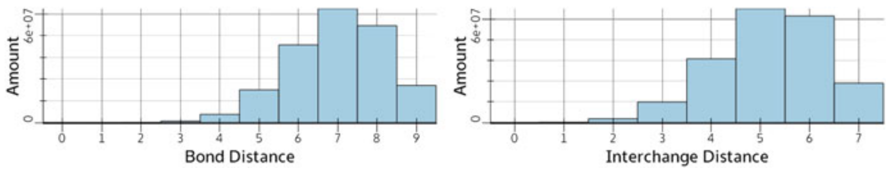


Fig. 2 Distribution of the distances within the whole set of permutations of length 9. The bond distance is used on the *left side* and the interchange distance on the *right side*. Both distance metrics lead to a similar distribution with most pairs of permutations having a very large distance (almost the maximal distance) and few such pairs having a distance of less than half of the maximal distance. This distribution is the reason, why both clustering of the solutions as well as projection of the solutions into low-dimensional spaces are difficult

between distinct clusters is kept, and data points that can not be clustered do not break the layout, but are presented as outliers.

6 Confirmation of the Method: Topological Analysis of the Search Landscape of TSP

To confirm the method presented above, we perform a topological analysis of the search landscape of TSP. Thereby, we show the effectiveness of the method in reproducing known results mainly from Fonlupt et al. [9]. Furthermore, we describe two additional findings during our experiments.

Fonlupt et al. [9] concluded from their analysis of the TSP landscape using the 2-opt operator, that it there is a “massif central” around the global optima. This is in accordance with the previous findings by Stadler and Schnabl [31]. We applied our method to all problems in the TSPlib [29] up to a problem size of 1048 to confirm these results (see Fig. 3). In all problem instances, we found mutual distances between the minima of $1/3$ to $1/2$ of the problem size. This can easily be seen from the histograms in Fig. 3. On the other hand, the samples that we randomly generated, usually had an almost maximal distance to every other solution. These results are in accordance with Fonlupt’s findings and suggest the same interpretation of a “massif central”. The force-directed layout (top row in Fig. 3) shows these findings in a very intuitive way: the local minima are clustered in the center of the depiction. The intermediate solutions are located on a ring around the local minima and the samples with their large distance to all other solutions are located on the outer ring around the local minima and the intermediate solutions. This suggests a crater-like structure and thus is an appropriate visualization of the common understanding of the TSP landscape among domain experts. Furthermore, Fonlupt et al. [9] found

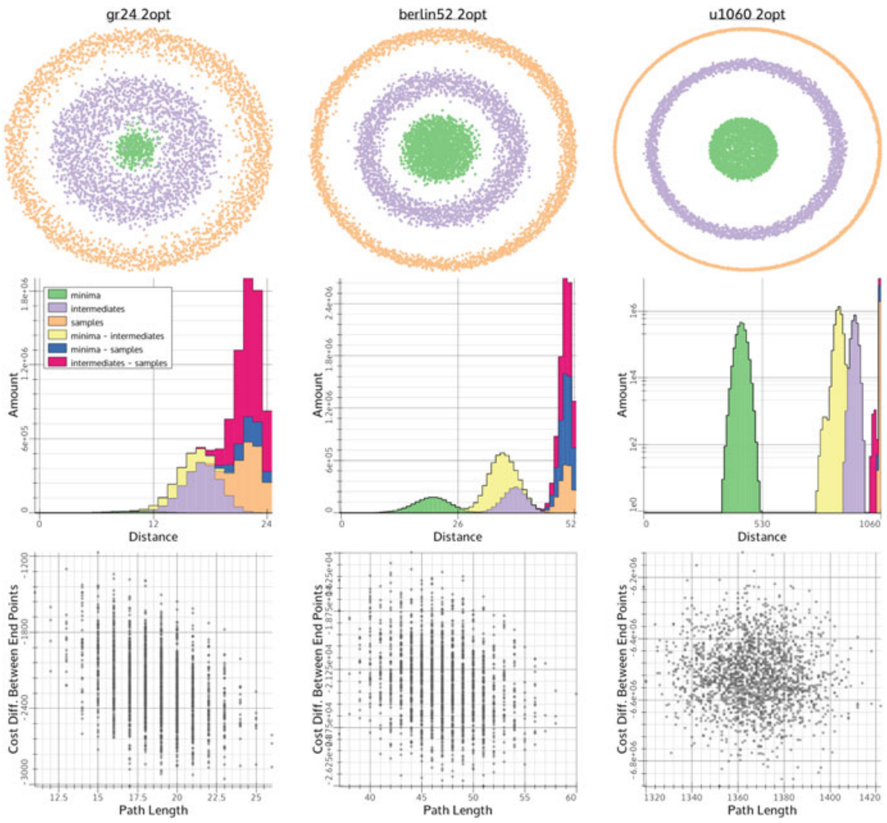


Fig. 3 2-opt search landscapes for three TSP instances from the TSPlib. The number in the problem names indicates the problem size. Notably, the distances between the local minima are very small. Also, the minima are well separated from both, the samples and even the intermediate solutions, in the distance histograms. This results in the crater-like layout of the search landscape (*top pictures*), showing the cluster of local minima with close proximity. The scatterplots in the *bottom row* show no correlation between path lengths and cost differences

no correlation between the cost value of both, the initial solutions and the local minimum, and the length of related local search paths. This can also be seen from the scatterplot in the bottom row of Fig. 3.

While evaluating these results, we also noticed a difference between small problem instances and larger ones. Small problem instances (up to a size of 12 to 15 cities) are sometimes used to test search algorithms because they are fully enumerable and easy to reason about. This has been criticized because small and large problem instance have severely different characteristics so that investigations based on small test instances can be misleading (cf. Sect. 5.1 in [28]). We found a clearly visible shift in the search landscapes of TSP instances somewhere between

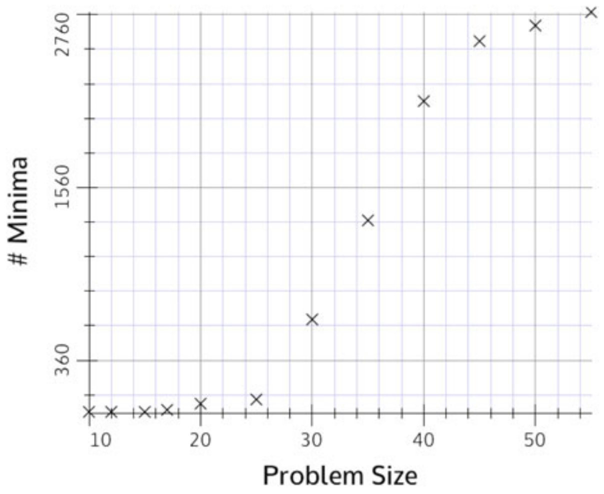


Fig. 4 Analysis of the complexity of TSP problems with increasing size. The total number of distinct local minima found by 3000 search paths for various problem sizes is shown. For distance distributions and search landscape layouts of chosen examples from these problems see Fig. 5

20 and 50 cities. Figures 4 and 5 show the results from analyzing randomly generated TSP instances of increasing size. Thereby, we found that for small problem sizes (up to 20 cities) there is a very small number of local minima. However, somewhere between 20 and 50 cities, the number of local minima increases until almost every search path reaches a different local minimum (cf. Fig. 4). Another effect is visible in the distribution of the distances (cf. Fig. 5). In problems of size 25 and above the distances between the local minima are clearly separated from the distances between the samples within the bar chart. That means that the local minima are located closer to each other than to the random samples. This is also clearly visible in the layouts of the search landscapes in Fig. 5. For the small problem of size 12, local minima and random samples are intermingled. While the layout of the 25-city problem already shows an accumulation of local minima in the center of the layout, the minima are still mixed with the random samples. For the larger problem of size 30, a clear separation between local minima and random samples is shown. From these observations we conclude, that TSP problems indeed change their complexity and become harder when exceeding a size of about 50 cities.

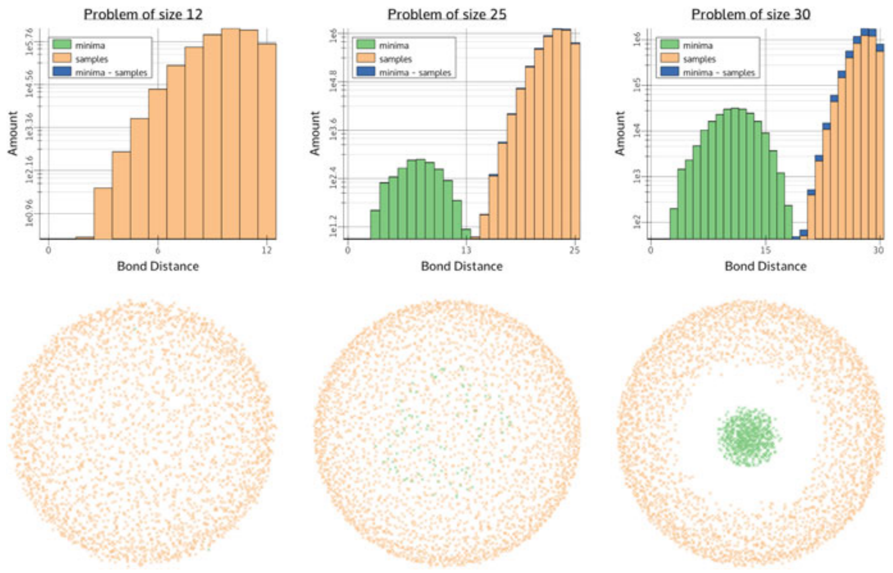


Fig. 5 Randomly generated TSP problems with increasing size, analyzed using 3000 search paths. The *top row* shows the distance distributions in the search landscape. Distances between local minima are clearly separated in the bar chart from distances between samples when the problem size increases (see also Fig. 3). Distances between local minima are not visible in the left histogram (problem size 12), because the number of minima is orders of magnitude less than the number of random samples. The *bottom images* show the layout of the search landscape for various problem sizes. Up from around 25 cities, a separation between minima and samples becomes visually apparent in the layout

Further, we compared the search landscapes of different search operators. In particular, we applied the interchange operator to the TSP problem. Previous research showed, that steepest descent using the 2-opt operator is more efficient than using the interchange operator for TSP [24, 31]. Figure 6 shows a juxtaposition of both search landscapes of the Bier127 problem from the TSPlib. This is only one example, but we also found the same behavior in the other instances from the TSPlib. When using the interchange operator, the search paths are longer on average (as can be seen from the histograms on the right column of Fig. 6) and the found local minima are not located near each other. Instead, there are almost maximal distances both between the samples as well as between the local minima. This results in a much different landscape layout, revealing almost no structure. Our intuition is that the missing structure makes TSP harder when using the interchange operator. A potential reason for this might be the interaction between the search operator and the cost function: the interchange operator changes four inter city connections in the route while 2opt only changes two and thus potentially causes smaller changes in the route length.

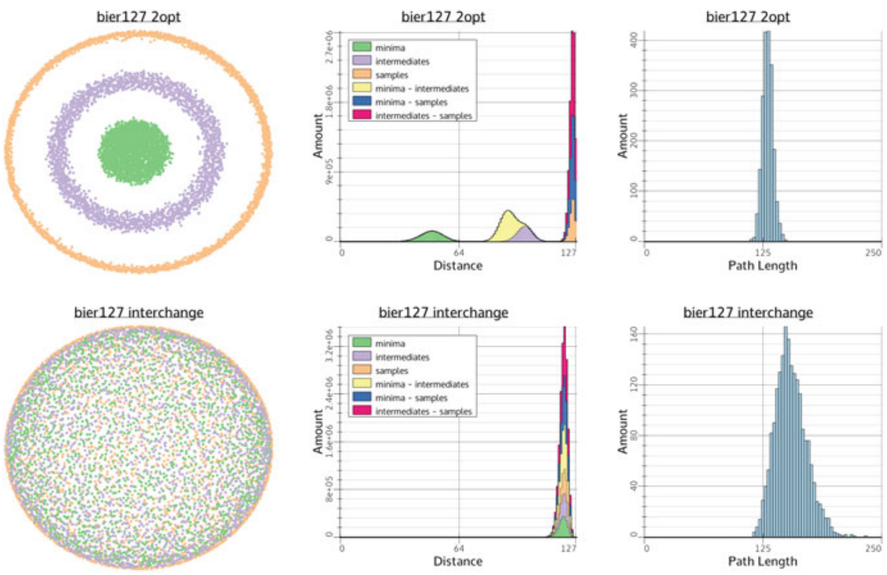


Fig. 6 Juxtaposition of the layouts and the distance distributions of the search landscape of the Bier127 problem for the 2-opt operator (*top row*) and the interchange operator (*bottom row*). On the *right*, the distribution of path lengths are shown. The search landscape from the interchange operator reveals much less structure, which hints that the interchange operator performs worse on the TSP

7 Conclusions and Future Work

We introduced a visualization system for the analysis of search landscapes of discrete optimization problems. The underlying analysis method is described in more depth in a companion paper [40]. It uses steepest descent paths from random solutions to reveal topological properties of the search landscape. The presented system involves visualization of statistical data as well as a topological visualization, and supports the interpretation of the search landscape by the domain expert. In particular, different topological shapes, e.g., crater-like or moon surface-like shapes, are revealed in an intuitive manner. We applied the method to many instances of the Traveling Salesman Problem to reproduce and confirm previous findings. Thereby, the effectiveness of the analysis method and the visualization were demonstrated.

We mainly focused on the Traveling Salesman Problem in this paper, which is well understood and is suitable as a benchmark. However, the method itself is not specific to it. It might be interesting to apply the method to different discrete optimization problems, not necessarily limited to permutation problems. We expect that new directions of research for these problems can be identified, as well as indications of how our visualization can be adapted to problem specific analysis tasks.

Acknowledgements This work was supported by the European Social Fund (ESF App. No. 100098251) and by the German Federal Ministry of Education and Research (BMBF) within the project Competence Center of Scalable Data Services and Solutions (ScaDS) Dresden/Leipzig (BMBF grant 01IS14014B).

References

1. Bin, S., Volke, S., Scheuermann, G., Middendorf, M.: Comparing the optimization behaviour of heuristics with topology based visualization. In: Dediu, A.H., Lozano, M., Martín-Vide, C. (eds.) Theory and Practice of Natural Computing. Lecture Notes in Computer Science, vol. 8890, pp. 47–58. Springer International Publishing, Berlin (2014)
2. Boese, K.D.: Models for iterative global optimization. Ph.D. thesis, University of California at Los Angeles, Los Angeles, CA (1996). UMI Order No. GAX96-14468
3. Cox, T., Cox, M.: Multidimensional Scaling. CRC Press, Boca Raton (2010)
4. Croes, G.A.: A method for solving Traveling Salesman problems. Oper. Res. **6**, 791–812 (1958)
5. Eremeev, A., Reeves, C.: Non-parametric estimation of properties of combinatorial landscapes. In: Cagnoni, S., Gottlieb, J., Hart, E., Middendorf, M., Raidl, G. (eds.) Applications of Evolutionary Computing. Lecture Notes in Computer Science, vol. 2279, pp. 31–40. Springer, Berlin/Heidelberg (2002)
6. Eremeev, A.V., Reeves, C.R.: On confidence intervals for the number of local optima. In: Proceedings of the 2003 International Conference on Applications of Evolutionary Computing, EvoWorkshops'03, pp. 224–235. Springer, Berlin/Heidelberg (2003)
7. Flamm, C., Hofacker, I.L., Stadler, P.F., Wolfinger, M.T.: Barrier trees of degenerate landscapes. Z. Phys. Chem. **216**, 1–19 (2002)
8. Fleurent, C., Glover, F.: Improved constructive multistart strategies for the quadratic assignment problem using adaptive memory. INFORMS J. Comput. **11**(2), 198–204 (1999)
9. Fonlupt, C., Robilliard, D., Preux, P., Talbi, E.G.: Fitness landscapes and performance of meta-heuristics. In: Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, pp. 257–268. Kluwer Academic Publishers, Boston (1999)
10. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman, San Francisco (1979)
11. Garnier, J., Kallel, L.: How to detect all maxima of a function. In: Theoretical Aspects of Evolutionary Computing, pp. 343–370. Springer, London (2001)
12. Golden, B.L., Alt, F.B.: Interval estimation of a global optimum for large combinatorial problems. Naval Res. Log. Q. **26**, 69–77 (1979)
13. Grinstein, G., Trutschl, M., Cvek, U.: High-dimensional visualizations. In: Proceedings of the VII Data Mining Conference KDD Workshop 2001, pp. 7–19. Association for Computing Machinery Press, New York/San Francisco, CA (2001)
14. Halim, S., Yap, R.H.C., Lau, H.C.: Viz: a visual analysis suite for explaining local search behavior. In: User Interface Software and Technology, pp. 57–66 (2006)
15. Hallam, J., Prügel-Bennett, A.: Large barrier trees for studying search. IEEE Trans. Evol. Comput. **9**(4), 385–397 (2005)
16. Harrison, L., Yang, F., Franconeri, S., Chang, R.: Ranking visualizations of correlation using weber's law. IEEE Trans. Vis. Comput. Graph. **20**(12), 1943–1952 (2014)
17. Harrower, M., Brewer, C.A.: Colorbrewer.org: an online tool for selecting colour schemes for maps. Cartogr. J. **40**(1), 27–37 (2003)
18. Hoos, H.H., Stützle, T.: Stochastic Local Search: Foundations and Applications. Elsevier, Amsterdam (2004)

19. Johnson, D.S.: Local optimization and the Traveling Salesman problem. In: Paterson, M.S. (ed.) Automata, Languages and Programming. Lecture Notes in Computer Science, vol. 443, pp. 446–461. Springer, Berlin/Heidelberg (1990)
20. Jolliffe, I.T.: Principle Component Analysis, 2nd edn. Springer, New York (2002)
21. Khor, S.: Search space analysis with Wang-Landau sampling and slow adaptive walks. CoRR, abs/1112.5980 (2011)
22. Knuth, D.E.: The Art of Computer Programming. Seminumerical Algorithms, vol. 2, 3rd edn. Addison-Wesley Longman Publishing Co., Boston, MA (1997)
23. Kohonen, T.: Self-organizing Maps. Springer, Berlin (1995)
24. Lin, S., Kernighan, B.W.: An effective heuristic algorithm for the Travelling-Salesman problem. Oper. Res. **21**, 498–516 (1973)
25. McCandlish, D.M.: Visualizing fitness landscapes. Evolution **65**(6), 1544–1558 (2011)
26. Ovacik, I.M., Rajagopalan, S., Uzsoy, R.: Integrating interval estimates of global optima and local search methods for combinatorial optimization problems. J. Heuristics **6**, 481–500 (2000)
27. Preux, P., Robilliard, D., Fonlupt, C., Karp, R.M., Steele, J.M.: Fitness landscapes of combinatorial problems and the performance of search algorithms (1997)
28. Rardin, R., Uzsoy, R.: Experimental evaluation of heuristic optimization algorithms: a tutorial. J. Heuristics **7**(3), 261–304 (2001)
29. Reinelt, G.: TSPLIB—a Traveling Salesman problem library. ORSA J. Comput. **3**(4), 376–384 (1991)
30. Schiavinotto, T., Stützle, T.: A review of metrics on permutations for search landscape analysis. Comput. Oper. Res. **34**(10), 3143–3153 (2007)
31. Stadler, P.F., Schnabl, W.: The landscape of the traveling salesman problem. Phys. Lett. A **161**(4), 337–344 (1992)
32. Stadler, B., Stadler, P.: Combinatorial vector fields and the valley structure of fitness landscapes. J. Math. Biol. **61**(6), 877–898 (2010)
33. Sun, L., Ceran, B., Ye, J.: A scalable two-stage approach for a class of dimensionality reduction techniques. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 313–322. Association for Computing Machinery, New York (2010)
34. Tukey, J.: Exploratory Data Analysis. Addison-Wesley Series in Behavioral Sciences. Addison-Wesley Publishing Company, Reading, MA (1977)
35. Uğur, A.: Path planning on a cuboid using genetic algorithms. Inf. Sci. **178**(16), 3275–3287 (2008). Including Special Issue: Recent advances in granular computing Fifth Internation Conference on Machine Learning and Cybernetics
36. Uğur, A., Aydin, D.: An interactive simulation and analysis software for solving TSP using Ant Colony Optimization algorithms. Adv. Eng. Softw. **40**(5), 341–349 (2009)
37. Uğur, A., Korukoğlu, S., Çalişkan, A., Cinsdikici, M., Alp, A.: Genetic algorithm based solution for TSP on a sphere. Math. Comput. Appl. **14**(3), 219–228 (2009)
38. Volke, S., Middendorf, M., Hlawitschka, M., Kasten, J., Zeckzer, D., Scheuermann, G.: dPSO-Vis: topology-based visualization of discrete particle swarm optimization. Comput. Graph. Forum **32**(3), 351–360 (2013)
39. Volke, S., Bin, S., Zeckzer, D., Middendorf, M., Scheuermann, G.: Visual analysis of discrete particle swarm optimization using fitness landscapes. In: Richter, H., Engelbrecht, A. (eds.) Recent Advances in the Theory and Application of Fitness Landscapes. Emergence, Complexity and Computation, vol. 6, pp. 487–507. Springer, Berlin/Heidelberg (2014)
40. Volke, S., Zeckzer, D., Scheuermann, G., Middendorf, M.: A visual method for analysis and comparison of search landscapes. In: Proceedings of the 2015 on Genetic and Evolutionary Computation Conference, pp. 497–504. Association for Computing Machinery, New York (2015)
41. Wilkinson, L.: The Grammar of Graphics. Springer, New York (1999)
42. Wright, S.: The roles of mutation, inbreeding, crossbreeding and selection in evolution. In: Proceedings of the Sixth Congress on Genetics, pp. 356–366 (1932)

Computing and Visualizing Time-Varying Merge Trees for High-Dimensional Data

Patrick Oesterling, Christian Heine, Gunther H. Weber, Dmitriy Morozov, and Gerik Scheuermann

Abstract We introduce a new method that identifies and tracks features in arbitrary dimensions using the merge tree—a structure for identifying topological features based on thresholding in scalar fields. This method analyzes the evolution of features of the function by tracking changes in the merge tree and relates features by matching subtrees between consecutive time steps. Using the time-varying merge tree, we present a structural visualization of the changing function that illustrates both features and their temporal evolution. We demonstrate the utility of our approach by applying it to temporal cluster analysis of high-dimensional point clouds.

1 Introduction

With increasing size and dimensionality, time-varying data has become difficult to visualize and analyze. One solution to this challenge is to detect features, i.e., salient data subsets, at each point in time and track their evolution to obtain more compact and less cluttered visualizations. For example, when meaningful features of scalar fields are defined by thresholding, a topological structure called the *merge tree* compactly encodes features for all possible thresholds.

To track features over time, existing methods commonly fix a threshold; changing this value requires expensive recomputation of the tracking information [29]. Supporting threshold changes over time or on a per-feature basis depends on specifying a large number of parameters a priori. Moreover, many existing methods

P. Oesterling (✉) • G. Scheuermann
Computer Science Department, University of Leipzig, 04009 Leipzig, Germany
e-mail: oesterling@informatik.uni-leipzig.de; scheuermann@informatik.uni-leipzig.de

C. Heine
AG Computational Topology, University of Kaiserslautern, 67653 Kaiserslautern, Germany
e-mail: cheine@cs.uni-kl.de

G.H. Weber • D. Morozov
Computational Research Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720, USA
e-mail: ghweber@lbl.gov; dmitriy@mrzv.org

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_5

correlate features via spatial overlap in consecutive time steps—an approach that can lead to ambiguities [24] and becomes computationally intractable in higher dimensions. Time-varying Reeb graphs [13, 18] require complicated distinction of cases for topological events; their number increases with the dimensionality and no case tables have been presented beyond the 3-D case.

Here, we introduce time-varying merge trees—a topological summary of time-varying scalar fields that addresses these issues as follows. Instead of using a single threshold, we track features for all thresholds, enabling threshold selection guided by the time-varying merge tree's visualization. To avoid tracking ambiguities, our algorithm records all necessary changes to the merge tree within the time interval between adjacent linearly-interpolated time steps, establishing a clean topological foundation for feature tracking. By focusing on merge trees for threshold-based feature tracking, we are able to provide a complete set of cases valid for all dimensions.

We represent time-varying merge trees as sequences of landscape profiles [21] and link hills visually to indicate structural changes. To reduce the clutter caused by showing many hills and links, we use topological simplification and represent preservation of topological feature groups by one visual link rather than many.

We demonstrate the utility of our approach by applying it to the analysis of time-varying, high-dimensional point clouds, where the merge tree of the density function captures clusters and their nesting [20].

2 Related Work

Defining and tracking features [22] is a common solution to visualizing large time-varying data sets. We focus on scalar fields, where many feature definitions are based on isosurfaces [19].

Szymczak [25] annotates contour trees for consecutive time steps to support queries of contours that evolve in particular ways or hit the boundary. Sohn and Bajaj [24] track contours using a similarity measure that considers spatial overlap of the inside and outside of contours. Ji and Shen [16] use the earth mover's distance to determine correspondence among contours.

For a family of real-valued functions on a common d-manifold without boundary, Edelsbrunner et al. [10] define Jacobi sets, which can be used to track critical points. On this basis they compute the time-varying contour tree of a function on the 3-sphere [13]. However, changes to the contour tree require detailed case analysis and the algorithm is difficult to extend to higher dimensions [13] (Mascarenhas, private communication, 2013). By restricting considerations to the merge tree, our algorithm considers fewer and simpler cases, and—more importantly—makes it independent of the domain's dimension. Instead of tracking critical points explicitly, Cohen-Steiner et al. [9] use evolving persistence diagrams to trace critical value pairs visually.

Keller and Bertram [17] present a method to compute a time-varying isosurface from a so-called hyper-Reeb graph, i.e., a Reeb graph augmented with Betti numbers indicating, e.g., genus changes. While isosurface extraction is applicable for arbitrary dimensions [2], d-dimensional regular grids of hypercubes and isosurfaces extracted as sets of $(d-1)$-dimensional simplices are quickly becoming impractical in higher dimensions, and Bhaniramka et al. [2] provide only 4-D and 5-D examples.

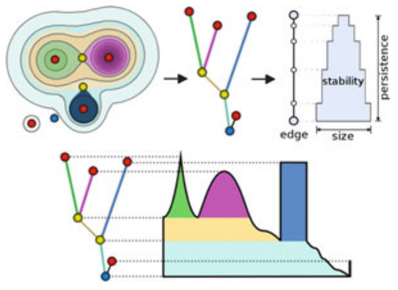
Bremer et al. [4] use the Morse-Smale complex to compute burning regions restricted to an isotherm for a range of fuel consumption thresholds. Once an appropriate fuel consumption threshold is identified, they use the Reeb graph of a 4-D space-time isosurface to track these regions over time [4, 28]. In later work, Bremer et al. [5] use the merge tree to compute statistics about burning regions in combustion simulations within a single time step. Once an appropriate threshold is identified, burning regions are tracked over time via overlap.

Visualizing tracked features can be challenging. Bremer et al. [3] survey methods and applications of topological feature tracking in molecular analysis, combustion simulations, and porous materials analysis. The results are shown as feature tracks embedded in the original domain. Similarly, Chen et al. [8] showed an embedding of the Reeb graph in the original data to track level sets of particle data. Other approaches store the tracking in a graph that is then shown using graph layout techniques. Widanagamaachchi et al. [29] present tracking graphs that update quickly when the user changes the isovalue of the tracked features. Because we compute tracking information for all features up front, we can layout the tracking information directly and show more feature properties, e.g. their size and robustness.

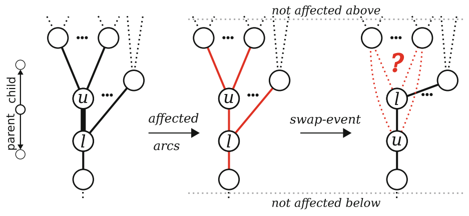
For point cloud data—the high-dimensional example application in this paper—Turkay et al. [26] present a system to explore time-varying clusters along with their structural properties, but this splits information into multiple views that need to be integrated in the analyst's mind. In contrast, our visualization technique shows all relevant metrics in the same view as the cluster evolution.

3 Background

A *superlevel set* of a function $f : \Omega \to \mathbb{R}, \Omega \subset \mathbb{R}^d$ for a value v is the set of all points $x \in \Omega$ where $f(x) \geq v$. It may consist of multiple connected components, whose evolution with varying v is of particular interest to us. Imagine the function f as a landscape, initially fully submerged by water, and the value v as the current water level. When the water is slowly drained, i.e., v is decreased, hills will start to emerge from the water, corresponding to new superlevel sets created at f's local *maxima*. When draining the water further, hills/superlevel sets will merge at points called *saddles*. The draining process stops when v reaches f's *global minimum*. Although the metaphor uses a 2-D landscape, the concepts are applicable in any dimension.



(a)



(b)

Fig. 1 (**a**) A scalar function's superlevel set boundaries and critical points indicate feature candidates. The merge tree encodes superlevel set evolution; its arcs and properties can be represented as a landscape profile. Color indicates correspondences among regions, merge tree arcs, and hills. A hill's height and width signify the corresponding superlevel set's persistence and size, respectively. The distribution of implicitly stored function values is reflected by a hill's shape; and thus its area reflects stability. The *green hill* is not very stable and the *blue hill* is maximally stable. (**b**) *Left*: If nodes u and l transpose, the merge tree needs to be reconstructed locally. *Center*: Only the arcs incident to u and l (*red*) are affected, whereas the rest of the tree remains unchanged. *Right*: Some arcs (*black*) are implicitly correct, while others (*dotted red*) need to be validated

The *merge tree* encodes changes of superlevel set connectivity. Leaves represent local maxima, inner nodes represent saddles, the root represents the global minimum, and *arcs*, i.e., directed edges, connect nodes according to the process outlined. Each node is labeled with the value v of its corresponding event. Each node and each point on an arc correspond to exactly one connected component of a superlevel set for one value v (cf. Fig. 1a).

For piecewise-linear functions on simplicial grids, Carr et al. [6] give an algorithm to compute the *augmented merge tree*: Initially, each grid vertex is represented by one node in an otherwise empty tree and one set in a union-find data structure. The algorithm processes all grid vertices u in order of decreasing function value and (1) determines the sets of u's *upper link*, i.e., grid neighbors with higher function value, (2) adds an arc from each set's lowest node to u's node, and (3) unites these sets with u's set and declares u's node as the new set's lowest node. In addition to leaves, saddles, and the root node, the augmented merge tree also consists of *regular nodes*, which have exactly one outgoing and one incoming arc. Removing all regular nodes yields the merge tree, consisting of so-called *superarcs* and *supernodes*.

Noise in the data complicates the merge tree, but it can be detected and removed by *topological simplification*. In this process, superarcs are annotated with a measure of robustness. Repeatedly, the leaf superarc of lowest robustness is removed from the tree, potentially turning a saddle into a regular node, which, together with its two incident superarcs, is then replaced by a superarc. For each region mapped onto a superarc, we use three measures: *persistence* [12], the difference between the maximum and minimum function value of a region, *size*, the number of grid vertices of a region, and *stability* [21], a summed function value distribution of a region.

The merge tree can be represented as a 2-D landscape profile [21], which is a variation of 3-D topological landscapes [15, 27]—a terrain metaphor for the more complex contour tree. A topological landscape has the same topology like its input tree and can represent structures of any dimension without structural occlusion. In a 2-D landscape profile, nesting hills represent superarcs of the merge tree; free of any occlusion and perspective distortion. Feature size is signified by width, persistence by height, and the distribution of function values in a region by the shape of the hills.

4 Overview

We visualize time-dependent, real-valued functions defined on domains of any dimension by studying how superlevel sets evolve with time. Our algorithm (1) computes the merge tree for each scalar field in the input sequence; (2) finds the sequence of structural changes that transforms each merge tree into its successor, storing these changes as *tracking records*; (3) uses topological simplification to remove noise from the merge trees, adjusting the tracking information accordingly, and (4) represents each tree as a landscape profile and translates tracking records into visual links connecting related hills. Steps (1)–(3) can be run concurrently for pairs of consecutive snapshots.

5 Merge Tree Transformation

To compute the merge tree evolution between two time steps, suppose that we computed the merge tree at each point in time for linear interpolation of the function between the two time steps. The construction algorithm outlined in Sect. 3 implies that the augmented merge tree's structure does not change as long as the ordering of grid vertices based on their function value stays the same. Instead of an infinite set of merge trees, we only need to consider the finite number of changes to the augmented merge tree that arise when vertices *transpose*, i.e., their function values' ordering changes. Moreover, because the augmented merge tree represents domain subset relations via arcs and paths, transpositions of nodes not connected by an arc do not affect the tree structure. It suffices to observe changes in the tree whenever two nodes joined by an arc transpose and their common arc *collapses*. We need to work on the augmented version of the merge tree because otherwise we would miss when a regular node pair becomes critical through a transposition.

Because reconstructing the whole merge tree from scratch after every single arc collapse would be computationally expensive, we consider how the merge tree can change after a single transposition. Figure 1b illustrates a transposition for arc (u, l), with u and l being the upper and lower nodes, respectively. There are two types of arcs: arcs not affected by the transposition that are implicitly preserved in the tree and arcs that may change, thus requiring additional validation.

Lemma 1 (Arc Lemma) *For a fixed domain and a fixed vertex order F, there is an arc (u, l), with $F(u) > F(l)$, in the merge tree of F if and only if the component of l in the domain restricted to the vertices w with $F(w) > F(l)$ contains the vertex u and does not contain any vertex w with $F(u) > F(w) > F(l)$.*

The proof of the lemma follows immediately from the algorithm used to construct merge trees, described in Sect. 3. It follows from the Arc Lemma that the transposition of u and l affects the merge tree only locally.

Property 1 Any arc that does not contain u or l remains in the merge tree.

Proof Since the only change in the order F is the transposition of vertices u and l, if the two properties of the Arc Lemma hold for an arc (x, y), with $x, y \notin \{u, l\}$, before the transposition, they continue to hold after the transposition. (And if they do not hold before, they do not hold after.) □

This property implies that the merge tree remains unchanged above all of u's and l's children, as well as below l's parent and thus only arcs incident to u and l need further validation. While it is immediate that u and l are still connected after the transposition, the Arc Lemma also implies an arc between u and l's parent node.

Property 2 u inherits l's parent node.

Proof Let p be l's parent node. The component of l before the transposition is the same as the component of u after the transposition. Since the order of nodes between l and p before, and u and p after the transposition are the same, the Arc Lemma implies that we have an arc (u, p) in the tree after the transposition. □

The validation of u's and l's child arcs depends on their connection in the underlying grid. The nodes' links do not change, but their upper links do, and thus require analysis. For node l, the change of its upper link is limited.

Property 3 l retains the arcs to all the components that remain in its upper link.

Proof For any arc (w, l), the two properties of the Arc Lemma hold before the transposition. After the transposition, the first property holds because if the component of w remains in l's upper link, then u belonged to a different component of l than w (so its removal could not have disconnected w from l). The second property holds because there is one less node between w and l. In other words, (w, l) remains an arc after the transposition. □

To understand the changes to u's children, we need to determine how its upper link is affected by the transposition. l can become a new upper link component, it can become part of an existing upper link component (a regular node), or l can combine an arbitrary number of u's previous upper link components. We need to check which of u's upper link components are in l's upper link, once l is higher than u. In other words, we determine whether l is connected to some of u's upper link components and thus becomes a regular node or a saddle. If l is not connected to any of u's upper link components, it becomes a new maximum within the upper link of u.

To achieve our goal, we start a traversal towards the merge tree's root from each grid node x in l's upper link. The traversal follows the unique path from x to the root of the tree. Node l lies on this path since x belongs to the superlevel set component of l. Node u lies on this path if and only if x falls in the superlevel set component of some node y in u's upper link (possibly, with $x = y$). The component of y in the upper link of u is represented, without loss of generality, by an arc (y, u). In this case, l inherits the arc (y, l) after the swap. Indeed, both conditions of the Arc Lemma are satisfied. If u does not lie on the path from x to the root, then, after the transposition, there is no connection in the superlevel set component of l between l and u's former upper link. In this case no arc is redirected to l. Most transpositions are between regular nodes, but only few of these produce a new maximum-saddle pair. We noticed that this can only happen when the involved regular nodes are grid neighbors. If not, we can safely swap them without performing a child traversal.

Our implementation starts with the augmented merge trees of the first time step and two lists of the grid vertices sorted descending by their values at the first and second time step. The first list will be reordered over time for correct determination of a node's (changing) upper link. The second list is to determine potential transpositions of new arcs that occur after a transposition. The time of an arc collapse is inferred from the linear interpolation between the vertices' values in the first and the second time step. We keep arc collapses in a priority queue, breaking ties based on the lexicographical order of their incident node IDs; a straightforward extension of simulation of simplicity [11].

The number of arc collapses depends on how many node pairs change their relative order in both time steps; i.e. on the structural variation of the function. In the worst case, i.e., if the node ordering is reversed, their number is bounded by $O(n^2)$ for n tree nodes. Potential push-updates to the priority queue take $O(log\ e)$, for e arcs in the queue. The traversal to validate node l's upper link depends on the number of tree nodes on the paths between l's upper link nodes and l itself. Trivially, this number is bounded by n; better bounds depend on the function itself and on grid granularity. In our experiments, however, we observed that the number of arc collapses is usually less than 5% of n^2 and that the 90th percentile of the traversal lengths is around 10% of n; while no traversal was longer than 30% of n.

6 Feature Tracking

By using a continuous transformation, we can identify structural changes of the augmented merge tree and we know the exact time and the order of all events. The challenge is now to infer the changes to the unaugmented merge tree's superarcs.

Between two time steps, superarcs may be born, die, or match with a superarc of the following merge tree. Figure 2 illustrates how we distinguish these cases.

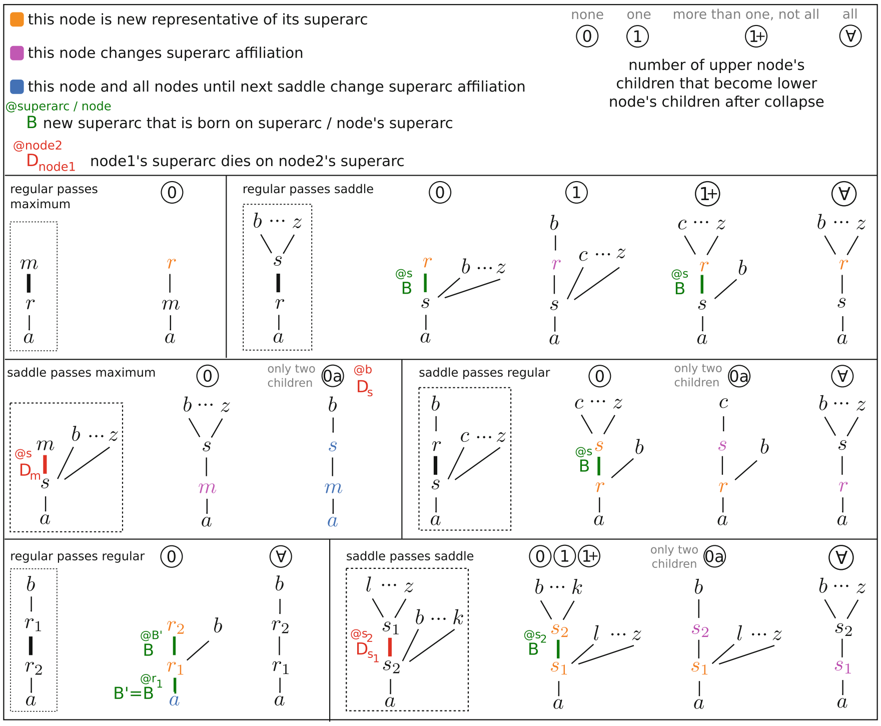

this node is new representative of its superarc
this node changes superarc affiliation
this node and all nodes until next saddle change superarc affiliation
@superarc / node
B new superarc that is born on superarc / node's superarc
@node2
D node1 node1's superarc dies on node2's superarc
none
one
more than one, not all
all
number of upper node's children that become lower node's children after collapse
regular passes maximum
regular passes saddle
saddle passes maximum
only two children
saddle passes regular
regular passes regular
saddle passes saddle

Fig. 2 Case table indicating how tracking information is affected by the nodes' types before and after the transposition. For every possible configuration (*dotted boxes*), the bold arc is about to collapse and all possible results are shown to the right. In principle, superarcs die when supernodes pass each other, and a new superarc is born when the passing node becomes a supernode

For birth and death events we keep track on which parent superarcs they take place because the tree may undergo many changes. For example, a newborn superarc may move within the tree, or it may give birth to other superarcs. Similarly, a superarc on which another superarc died may move or die. In rare circumstances, an arc gives birth to the arc it dies on. Therefore, we have to store tracking information recursively. To display superarc relations later on in the visualization, each superarc needs a representative. We use its upper supernode for this purpose. Leaf superarcs are thus represented by their maxima, and inner ones by their upper saddle node. For each transposition, we check if the superarcs of the tree are affected. If necessary, we create or destroy tracked arcs, and we record on which arcs these changes occur. If superarcs do not change, we record when regular nodes change their association to a superarc and, if necessary, update a superarc's representative to handle moving features that are represented by a different set of grid vertices in the next time step.

For every superarc we maintain a *tracking record* that stores: the initial representative, the current representative, the superarc born from, and the superarc died on. Initially, the entries for superarc "born from" and "died on" are empty. We also store for each node to which superarc it belongs. One possibility to track superarcs—including precise place of birth/death and correct times—is to consider the node types before and after an arc collapse and to handle this event according to the case table in Fig. 2. This table summarizes all possible configurations of an arc collapse, using symmetry of events (e.g. "a saddle node rises above a maximum" versus "a maximum falls below a saddle") and impossible events (e.g. "two maxima swap") to reduce the number of considered cases. Possible results of a configuration reflect the changes of *l*'s upper link. Furthermore, the table indicates in which configurations superarcs are born or die and in which cases a node's superarc affiliation changes. Tracking then consists of simply creating or updating affected tracking records after every arc collapse. For example, if before a transposition the lower node is a saddle *s* and the upper node is a regular node *r* (Fig. 2, right column, second row) and both nodes are saddles after the transposition (Case 0), we first create a new superarc with *s* as its current representative and *s*'s previous superarc as the "born from" entry. Then we set *r* to be *s*'s previous superarc's new representative and update the superarc connecting both nodes. For simplicity, events relating to the minimum of the tree are considered to be either regular or saddle events.

A function's main features naturally appear as maximum-saddle pairs of significant persistence, size, or stability. Therefore, we restrict further processing and visualization to leaf superarcs. To create pairwise relations between original and final superarcs we post-process the tracking records after the transformation as follows: If the "died on" entry for an original leaf record is empty, we associate that record's initial representative with its current representative, and store this as a *match record*. If the "died on" entry of an original leaf record is not empty, we recursively follow it until we reach the record where "died on" is empty. We associate the record's initial representative with the record's current representative found and store this as a *death record*. Finally, for each new leaf record, we recursively follow the "born from" entry, associate the found record's initial representative and the new record's current representative, and store this as a *birth record*. This gives us a set of records, classified into either match, birth, or death events, that tell us how features of the complete function relate to each other between the original and the final tree.

7 Simplification

Merge trees of noisy functions contain many small superarcs that represent features below meaningful thresholds. Topology-based simplification [7], i.e., the removal of those superarcs whose properties, e.g. their persistence, are below a user-specified threshold, has been applied successfully to reduce noise in the data.

If a superarc is removed from the tree, we also have to adjust those tracking records that have this superarc as their origin or target, i.e., as their initial or current representative. To this end, we redirect the tracking record by replacing the removed superarc by its parent superarc. Tracking records may become redundant by this process, e.g. if the target of a birth-record, or the origin of a death record is removed.

Note that for the time-varying analysis, changing the simplification threshold for the merge tree at time step i requires repeating simplification of tracking information between time steps $i - 1$ and i, and between time steps i and $i + 1$.

8 Prototype Visualization and Examples

The time-varying merge tree consists of: the merge trees at the given time steps that describe the hierarchy as well as quantitative properties for all features, the tracking information of all features over time, and exact times for structural events of the complete function.

To reduce visual complexity, we use a discrete approach that separates the depiction of structure from that of temporal evolution. We display the merge trees as occlusion-free 2-D topological landscape profiles [21], stacked in the third dimension according to their time stamp, and using orthographic projection to facilitate feature comparison. We relate features over time using visual links between the profiles, similar to standard isotracking graphs [5, 24], but showing more feature properties and, most importantly, features for all thresholds in the landscape profiles.

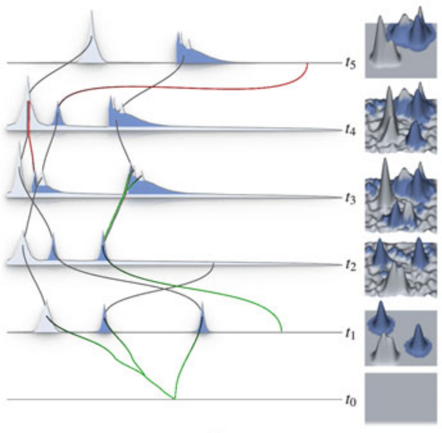
To visualize the tracking records that associate two superarcs of two subsequent merge trees, we identify both superarcs using the record's initial and current representative, identify their areas in the profiles, and use these areas' centroids as the visual link's origin and target coordinates. To distinguish record types, we use black, green, and red links for match, birth, and death records, respectively. The user can filter tracking information by selecting arbitrary parts of the profiles. Tracking information is then filtered either forwards, backwards, or in both directions in time. More sophisticated analysis is achieved by combining simplification and interactive selection. For example, small features could be excluded from simplification if they are related to the evolution of user-selected features—e.g. if noise becomes a prominent user-selected feature later on.

Although topological simplification is the primary tool for removing irrelevant hills, we further reduce the remaining links to increase visual clarity by: *link aggregation* to group multiple incoming or outgoing links per hill by type/color and let them fork if necessary; *link unification* for links of different type/color that have exactly the same origin and target hills, which could happen for fine-grained topological events in one and the same feature, e.g., if a new arc is born inside a region on which the former maximum dies; and *match-link combination* to connect hierarchical subfeatures that do not change over time with a single black link at their lowest shared saddle. A single match link between two profiles indicates that structure is preserved entirely. Finally, *reordering* saddle node children can change hill positions in the profile without changing its topology. This strategy could be used to optimize crossing links; but does not allow to switch arbitrary hills.

8.1 2-D Example

Figure 3a uses an artificial function in 2-D to illustrate the visualization of a time-varying merge tree. The grid consists of 2500 vertices and the computation of the topologically most complex time step takes less than a second. We used a machine with two 2.4 GHz Quad-Core AMD Opteron(tm) processors for all experiments. Because time steps are processed concurrently, the total time to obtain the image, including topological simplification to remove small features, is approximately 1 s.

As can be seen, the tracking is robust with respect to noise and feature shape, and moving features are recognized if their spatial distance is small enough. Typically, a moving feature is identified as a superarc whose implicit regular nodes (grid vertices) only change or as a superarc that gives birth to a new one on which it dies afterwards. Time steps t_1 and t_2 show a counterexample. Because the spatial distance of the right feature in the function is too big, the tracking detects that a new feature gets born and the old one vanishes. A higher time resolution would be



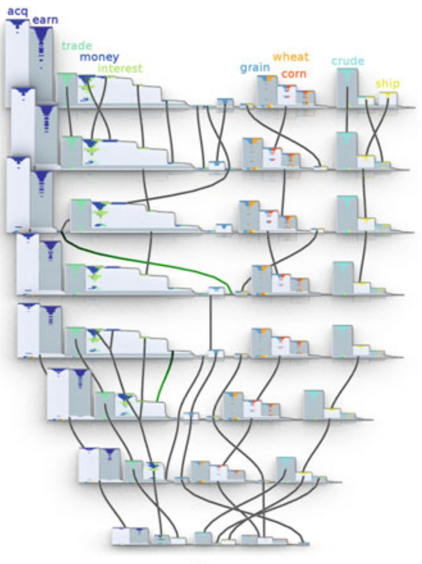


Fig. 3 (**a**) Artificial, topologically simplified 2-D function (*right*): t_0: flat function. t_1: three main features are born. t_2: all features move and noise is added. t_3: all features move, one splits, the other two grow in size and persistence, respectively. t_4: features move and join. t_5: one feature dies, another is rotated, noise is removed. (**b**) Reuters data: Hills on the profiles for each time slice show the number, nesting, importance, and homogeneity of clusters with all relevant metrics (persistence, size, and stability) in the same view. Histograms, colored by class, verify that documents primarily accumulate by class and form subclusters for related categories. Over time, tracked features of the varying merge tree show how more subclusters break apart and grow individually. The profiles grow in their width and height to reflect that new documents are added with each additional time slice

needed to correctly detect this as a fast moving feature. In general, death events are detected, but in time step t_5 a feature matches to one of the maxima produced by added noise in that area. Because noise is removed by topological simplification, the link target correctly points to the removed hill's parent hill. If a feature splits, the new one typically matches to one of the former maxima (t_3, second hill) or a new feature is born (t_3, right hill). Likewise, if features join, they first share a higher saddle and then one feature would die on the other (t_4). In t_3, the first two hills change their order because the topological landscape profiles [21] sort subtrees of each merge tree saddle by persistence to put the highest hills to the left.

8.2 High-Dimensional Example

To demonstrate the effectiveness and applicability of time-varying merge tress for high-dimensional data, we extend previous work by Oesterling et al. [20] on topology-based cluster analysis of static, high-dimensional points clouds. The authors assume the point cloud to be outcomes of a set of random variables with identical high-dimensional probability distributions. Using geometric graphs and a Gaussian kernel, they obtain an approximation of the high-dimensional probability density function whose merge tree topology they then visualize as a topological landscape. We extend this work by approximating a point cloud in space-time by its varying density function and analyze its superlevel set topology over time.

To make computation of the time-varying merge tree tractable, we construct a grid that remains unchanged over time, but still supports sampling the density functions of all time steps with sufficient accuracy. We merge all input point clouds into a single set of points, construct the neighborhood graph, and determine point densities using an appropriate Gaussian filter radius (see [20] for details) so separate all clusters.

We use categorized documents from the Reuters-21578 collection [1] that appeared on the Reuters newswire in 1987. To demonstrate accurate feature tracking in high dimensions, we extract documents for ten economy-related categories and use the *tf-idf* [23] document-term weighting to define word importances for each document's dimensions in the vector space model. Using Linear Discriminant Analysis [14], a supervised projection that uses given classification information to minimize information-loss, we project the data to a ($\#classes - 1 = 9$)-dimensional space and end up with 5309 documents, manually divided into eight time slices, 20 days each, from 02/21/1987 to 10/19/1987. Figure 3b shows the visualization of the time-varying merge tree. The total time to obtain the image is around 1 s. The most complex transformation required processing approximately 33,000 arc collapses.

Based on classification information attached to the documents, we can place colored histograms on the hills to indicate how documents of different classes are distributed across the clusters. As can be seen, dense regions primarily match to documents of a single class, while some documents of related classes are in a subcluster relationship. The valley between hills of unrelated topics is typically low as it reflects the density between the corresponding clusters. Likewise, for related topics, like “grain”,“wheat” and “corn”, the subspace spanned by the vocabulary used in these documents also contains less specific documents between the cluster centers and thus saddle densities are higher. Another insight, suggested by the rectangular hill shape, is that clusters are very compact in the sense that document densities are close to the cluster’s maximum density; hence their positioning close to the hilltops. Over time, dense regions are primarily stable, but grow in size and persistence, and occasionally split into more subclusters that become increasingly prominent.

9 Conclusion

We introduced time-varying merge trees as a compact description of time-varying scalar fields and combined landscape profiles with a tracking information overlay to support exploration of time-varying scalar fields. We further demonstrated the utility of our method using a 9-D document collection data set. Our method can inform parameter selection for in-depth feature analysis.

The visualization is currently limited to showing a few time steps in a single image, and the optimal depiction of time-varying merge trees remains an open question. Future work will also focus on reducing the runtime of the transformation, which depends on the topological variance between two time steps. While the considered events are both necessary and sufficient for the augmented merge tree, it may be possible to process fewer events for the unaugmented merge tree. We will also extend topological simplification to the time-varying merge tree, instead of simplifying time steps individually. While it is trivial to adapt our algorithm to the time-varying split tree and compute the contour tree for a given time, tracking the contour tree edges remains a topic of future work.

Acknowledgements The authors thank anonymous reviewers for valuable comments and assistance in revising the paper. This work was supported by a grant from the German Research Foundation (DFG) within the strategic research initiative on Scalable Visual Analytics (SPP 1335). This work was supported by the Director, Office of Advanced Scientific Computing Research, Office of Science, of the U.S. DOE under Contract No. DE-AC02-05CH11231 (Lawrence Berkeley National Laboratory) through the grant “Topology-based Visualization and Analysis of High-dimensional Data and Time-varying Data at the Extreme Scale”, program manager Lucy Nowell.

References

1. Bache, K., Lichman, M.: UCI machine learning repository. http://archive.ics.uci.edu/ml (2013)
2. Bhaniramka, P., Wenger, R., Crawfis, R.: Isosurface construction in any dimension using convex hulls. IEEE Trans. Vis. Comput. Graph. **10**(2), 130–141 (2004)
3. Bremer, P., Bringa, E., Duchaineau, M., Gyulassy, A., Laney, D., Mascarenhas, A., Pascucci, V.: Topological feature extraction and tracking. J. Phys. Conf. Ser. **78**(1), 012007 (2007)

4. Bremer, P.T., Weber, G.H., Pascucci, V., Day, M., Bell, J.B.: Analyzing and tracking burning structures in lean premixed hydrogen flames. IEEE Trans. Vis. Comput. Graph. **16**(2), 248–260 (2010)
5. Bremer, P.T., Weber, G.H., Tierny, J., Pascucci, V., Day, M., Bell, J.: Interactive exploration and analysis of large-scale simulations using topology-based data segmentation. IEEE Trans. Vis. Comput. Graph. **17**(9), 1307–1324 (2011)
6. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. **24**(2), 75–94 (2003)
7. Carr, H., Snoeyink, J., van de Panne, M.: Simplifying flexible isosurfaces using local geometric measures. In: Proceedings of IEEE Visualization, pp. 497–504. IEEE Computer Society, Los Alamitos (2004)
8. Chen, F., Obermaier, H., Hagen, H., Hamann, B., Tierny, J., Pascucci, V.: Topology analysis of time-dependent multi-fluid data using the Reeb graph. Comput. Aided Geom. Des. **30**(6), 557–566 (2013)
9. Cohen-Steiner, D., Edelsbrunner, H., Morozov, D.: Vines and vineyards by updating persistence in linear time. In: Proceedings of Symposium on Computational Geometry, pp. 119–126. Association for Computing Machinery, New York (2006)
10. Edelsbrunner, H., Harer, J.: Jacobi sets of multiple Morse functions. In: Cucker, F., DeVore, R., Olver, P., Suli, E. (eds.) Foundations of Computational Mathematics. Minneapolis, pp. 37–57. Cambridge University Press, Cambridge (2004)
11. Edelsbrunner, H., Mücke, E.P.: Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Trans. Graph. **9**(1), 66–104 (1990)
12. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. Discret. Comput. Geom. **28**(4), 511–533 (2002)
13. Edelsbrunner, H., Harer, J., Mascarenhas, A., Pascucci, V., Snoeyink, J.: Time-varying Reeb graphs for continuous space-time data. Comput. Geom. **41**(3), 149–166 (2008)
14. Fukunaga, K.: Introduction to Statistical Pattern Recognition, 2nd edn. Academic Press Professional, Inc., San Diego, CA (1990)
15. Harvey, W., Wang, Y.: Topological landscape ensembles for visualization of scalar-valued functions. Comput. Graph. Forum **29**(3), 993–1002 (2010)
16. Ji, G., Shen, H.W.: Feature tracking using Earth Mover's distance and global optimization. In: Proceedings of Pacific Graphics. Poster (2006)
17. Keller, P., Bertram, M.: Modeling and visualization of time-varying topology transitions guided by hyper Reeb graph structures. In: Proceedings of IASTED International Conference on Computer Graphics and Imaging, pp. 15–25 (2007)
18. Mascarenhas, A., Snoeyink, J.: Implementing time-varying contour trees. In: Proceedings of Symposium on Computational Geometry, pp. 370–371. Association for Computing Machinery, New York (2005)
19. Mascarenhas, A., Snoeyink, J.: Isocontour based visualization of time-varying scalar fields. In: Möller, T., Hamann, B., Russell, R.D. (eds.) Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration, pp. 41–68. Springer, Berlin (2009)
20. Oesterling, P., Heine, C., Jänicke, H., Scheuermann, G., Heyer, G.: Visualization of high-dimensional point clouds using their density distribution's topology. IEEE Trans. Vis. Comput. Graph. **17**(11), 1547–1559 (2011)
21. Oesterling, P., Heine, C., Weber, G.H., Scheuermann, G.: Visualizing nD point clouds as topological landscape profiles to guide local data analysis. IEEE Trans. Vis. Comput. Graph. **19**(3), 514–526 (2013)
22. Reinders, F., Post, F.H., Spoelder, H.J.: Visualization of time-dependent data with feature tracking and event detection. Vis. Comput. **17**(1), 55–71 (2001)
23. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM **18**(11), 613–620 (1975)
24. Sohn, B.S., Bajaj, C.: Time-varying contour topology. IEEE Trans. Vis. Comput. Graph. **12**(1), 14–25 (2006)

25. Szymczak, A.: Subdomain aware contour trees and contour evolution in time-dependent scalar fields. In: Shape Modeling and Applications, pp. 136–144. IEEE Computer Society, Los Alamitos (2005)
26. Turkay, C., Parulek, J., Reuter, N., Hauser, H.: Interactive visual analysis of temporal cluster structures. Comput. Graph. Forum **30**(3), 711–720 (2011)
27. Weber, G.H., Bremer, P.T., Pascucci, V.: Topological landscapes: a terrain metaphor for scientific data. IEEE Trans. Vis. Comput. Graph. **13**(6), 1416–1423 (2007)
28. Weber, G.H., Bremer, P.T., Day, M., Bell, J., Pascucci, V.: Feature tracking using Reeb graphs. In: Pascucci, V., Tricoche, X., Hagen, H., Tierny, J. (eds.) Topological Methods in Data Analysis and Visualization, pp. 241–253. Springer, New York (2011)
29. Widanagamaachchi, W., Christensen, C., Pascucci, V., Bremer, P.T.: Interactive exploration of large-scale time-varying data using dynamic tracking graphs. In: IEEE Symposium on Large Data Analysis and Visualization, pp. 9–17 (2012)

Agreement Analysis of Quality Measures for Dimensionality Reduction

Bastian Rieck and Heike Leitte

Abstract High-dimensional data sets commonly occur in various application domains. They are often analysed using dimensionality reduction methods, such as principal component analysis or multidimensional scaling. To determine the reliability of a particular embedding of a data set, users need to analyse its quality. For this purpose, the literature knows numerous quality measures. Most of these measures concentrate on a single aspect, such as the preservation of relative distances, while others aim to balance multiple aspects, such as intrusions and extrusions in k-neighbourhoods. Faced with multiple quality measures with different ranges and different value distributions, it is challenging to decide which aspects of a data set are preserved best by an embedding. We propose an algorithm based on persistent homology that permits the comparative analysis of different quality measures on a given embedding, regardless of their ranges. Our method ranks quality measures and provides local feedback about which aspects of a data set are preserved by an embedding in certain areas. We demonstrate the use of our technique by analysing quality measures on different embeddings of synthetic and real-world data sets.

1 Introduction

High-dimensional data sets are ubiquitous in most scientific disciplines today. By including more variables, natural phenomena can be modelled and understood more precisely. With an increasing number of variables, visualization for exploratory data analysis becomes essential to gain an understanding of the data. A common approach for visualizing complex high-dimensional data employs dimensionality reduction methods. For embedding their data into a lower dimension, users can choose from many algorithms, such as *principal component analysis* (PCA), *Isomap*, and *t-distributed stochastic neighbour embedding* (t-SNE).

B. Rieck (✉) • H. Leitte
IWR, Heidelberg University, Heidelberg, Germany
e-mail: bastian.rieck@iwr.uni-heidelberg.de; heike.leitte@iwr.uni-heidelberg.de

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_6

Since ground truth information is often unavailable, quality measures are required to judge how accurately a given method is able to retain a structural property of the data set such as local neighbourhoods. Quality measures usually only assess a single property of the data set. Given an embedding of a data set and several quality measures, users often want to know about compromise solutions: For example, an embedding that distorts local neighbourhoods somewhat but keeps the global structure of the data set intact might be preferable over an embedding that does not distort local neighbourhoods but completely distorts the input data at a global scale.

For an embedding of a high-dimensional data set, we are thus interested in finding out which specific properties (e.g. neighbourhoods, distances, etc.) of it are faithfully retained. To this end, we analyse the agreement of multiple quality measures on the data. Modelling each quality measure as a scalar field on the embedding, we are interested in the regions of the highest error of a quality measure and hence decompose each scalar field into regions defined by their maxima. Instead of having to compare the quality measures per point, we compare only their decompositions, which are much more stable with respect to noise. If two measures highlight the same regions as having a low quality, their resulting decompositions will be very similar, and we thus consider their behaviour on the data set to be similar. This implies that their respective properties are retained to a similar extent. To measure similarity and highlight areas in which different quality measures disagree the most, we use a similarity measure and a graph matching algorithm. Using several real-world data sets, we demonstrate how our method helps users determine which properties of a data set have been respected by an embedding.

2 Related Work

Multi-Field Data The task of comparing different scalar quality measures on a data set is a particular instance of a *multi-field* problem. In this context, several methods already permit the comparison of scalar functions. Sauber et al. [17] used gradient similarity measures and local correlation coefficients to analyse correlations in scalar fields defined over (regular) 3D or 2D grids. Their approach quickly becomes computationally infeasible with a larger amount of scalar fields. Schneider et al. [18, 19] used *contour trees* for comparing iso-surfaces in two scalar fields. To define similarity between features, they use similarity measures based on the approximated contour volume as well as information-theoretic and graph clustering methods. Their method is geared towards analysing fields for flow visualization and requires cell-based grids, whereas our method is specifically targeting unstructured data.

Scalar Field Topology and Persistent Homology Our method uses persistent homology to analyse the topological structure of quality measures on a data set. This approach is related to other methods from scalar field topology. Gerber et al. [10] used inverse regression in Morse-Smale complexes to obtain a simplified visualization of scalar functions on high-dimensional data. Their method is used for parameter studies but does not permit the comparison of multiple scalar fields. Chazal et al. [3] decompose a scalar field of density values of a manifold into basins of attraction to find stable clusters. Their algorithm does not allow for comparing clusters in different scalar fields but has well-defined stability guarantees that we will use in our approach. Correa et al. [7] use sparse subsets of the Morse-Smale complex to visualize the structure of scalar fields. This approach complements our method as a visual aid for comparison but does not visualize features in different scalar fields. Oesterling et al. [14] use *join trees* to visualize the behaviour of density functions on high-dimensional point clouds. Their visualization yields a topological landscape in which regions of similar density are highlighted, or a landscape profile that represents clusters as peaks [15]. Both methods are not specifically suited for comparing multiple scalar fields among each other. A complete decomposition of merges and splits of contours in scalar fields is given by the *contour tree* [2]. Our method requires a computation that is similar to the *join tree*, because we are only interested in the maxima of a scalar field.

Dimensionality Reduction An in-depth overview of state-of-the art dimensionality reduction methods is given by van der Maaten et al. [21]. Lee and Verleysen [11] survey numerous quality measures and show how the analysis of global quality measures helps in selecting from a set of different dimensionality reduction methods. Bertini et al. [1] survey quality measures in the context of high-dimensional data visualization. Quality measures are used to provide a broad overview of a data set. By contrast, our method assumes that the data set has already been embedded and aims on communicating which structural properties are retained with respect to the original data.

3 Quality Measures

For the subsequent analysis in Sect. 5, we shall use two groups of local quality measures: Distance-based measures and rank-based measures. The former are more stable against small changes in the embedding whereas the latter are more stable against large changes or linear scaling in the data [11]. In the following, we use pointwise definitions of all quality measures for a data set of cardinality n. We also transform their range such that high values indicate regions of low quality (hence, the functions measure *errors*). Subsequently, d_{ij} refers to the original distances in the high-dimensional space, while δ_{ij} refers to the distances in the embedded space.

Table 1 Properties that are measured by the quality measures

Measure	Property
RMSE	Average squared distance deviation
Kruskal's stress	Average squared distance deviation penalizing small deviations
Residual variance	Correlation between original and embedded distances
Rank correlation	Correlation between ranks of original and embedded distances
Neighbourhood loss	Changes in k nearest neighbours; measure of group preservation
MRRE	Extrusions and intrusions of k nearest neighbours

There are more measures available in literature [11, 21], but we have decided to select the most common ones and aimed for those that are not specifically optimized for a certain algorithm. Table 1 gives a short overview of the properties they measure.

Root-Mean-Square-Error (RMSE) RMSE measures the average squared difference between the distances: $f(x_i) = \sqrt{\sum_{j=1}^{n} \left(d_{ij} - \delta_{ij}\right)^2 / n}$

Kruskal's Stress In contrast to RMSE, this stress measure penalizes deviations in small distances more than in large distances: $f(x_i) = \sqrt{\sum_{j=1}^{n} \left(d_{ij} - \delta_{ij}\right)^2 / \sum_{j=1}^{n} \delta_{ij}^2}$

Residual Variance The residual variance measures the complement of the explained variance between d_{ij} and δ_{ij}, using the linear correlation coefficient:

$$f(x_i) = 1 - R^2(\{d_{i0}, \ldots, d_{in}\}, \{\delta_{i0}, \ldots, \delta_{in}\}) \tag{1}$$

Spearman's Rank Correlation By converting the distances d_{ij} and δ_{ij} to ranks r_{ij} and ρ_{ij}, respectively, this measure is more stable against outliers in the data and invariant to linear scaling: $f(x_i) = 1 - 6 \sum_{j=1}^{n} \left(r_{ij} - \rho_{ij}\right) / \left(n \left(n^2 - 1\right)\right)$

Neighbourhood Loss This measure is agnostic to distances and requires an enumeration of the k nearest neighbours of each point both in the original space and the embedded space, which we denote $n_k(i)$ and $\nu_k(i)$:

$$f(x_i) = 1 - |n_k(i) \cap \nu_k(i)|/k \tag{2}$$

Mean Relative Rank Error (MRRE) MRRE measures the mean amount of rank deviations using the k nearest neighbours of the point in both the original space and the embedded space. Lee and Verleysen [11] developed MRRE to penalize two common errors in embeddings, namely very distant points that *intrude* into the k-neighbourhood of a point, as well as very close points that *extrude* from such a neighbourhood.

4 Methods

In the following, we will describe the components of our method. The main algorithm is similar to the calculation of the *join tree*, but uses persistent homology to obtain a criterion for the stability of maxima. We refer the reader to Edelsbrunner and Harer [9] for a more detailed account of computational topology and persistent homology.

4.1 Scalar Field Decomposition Using Persistent Homology

Let $\mathbb{D}$ be a connected domain and $f\colon \mathbb{D} \to \mathbb{R}$ a scalar function such as a quality measure. A natural way of summarizing this function for data analysis and comparison is to detect its *peaks* and decompose the data according to the gradient of f, i.e. we decompose $\mathbb{D}$ into disjoint subsets consisting of all those points that reach a certain peak when following the gradient. This approach is also known as *mode-seeking* [4]; see Fig. 1, left, for a simple example. With discrete data, however, mode-seeking approaches are known to be very unstable. To obtain a measure of the stability of the detected peaks, we thus use *persistent homology*, an algorithm from computational topology. Persistent homology summarizes data sets using their topological features. Each topological feature is assigned a significance measure, the *persistence*.

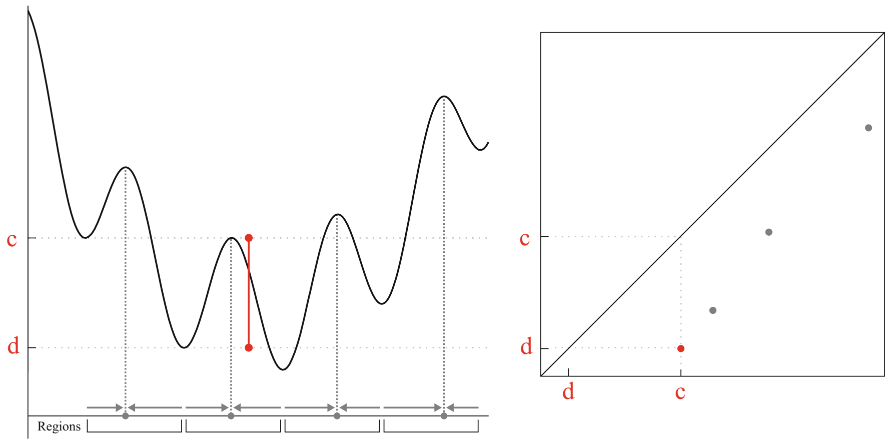


Fig. 1 *Left*: By following the steepest ascent (shown as *arrows* on the abscissa), the domain of f is decomposed into disjoint regions. The minima are the boundaries of a region. *Right*: The persistence diagram of f. The distance from the diagonal is a measure of the stability of a peak

4.1.1 The 1-Dimensional Case

Let $\mathbb{D} \subseteq \mathbb{R}$ be our domain and $f\colon \mathbb{D} \to \mathbb{R}$ a scalar function. We can use persistent homology to describe connectivity changes in the *superlevel sets* of f, i.e. sets of the form $L_c^+(f, c) = \{x \mid f(x) \geq c\}$ for $c \in \mathbb{R}$. We now traverse the function values of f in decreasing order and keep track of how the connected components of f change. When we reach a new peak in f, i.e. a maximum of f, a new connected component will be created. By contrast, when we reach a minimum, two connected components are merged into one. We always merge the "younger" connected component (the one with the smaller peak) into the "older" connected component (the one with the larger peak) to ensure consistency [9, p. 150].

By keeping track of the merges, we obtain the *persistence diagram*. It contains a point (c, d) for every connected component created at $c = f(x)$ and merged into an older connected component at $d = f(x')$. Since by definition of the superlevel sets, $c \geq d$, all points in the persistence diagram are located below the diagonal. Figure 1 illustrates this process for a simple function with several peaks. The *persistence* of a tuple (c, d) is given as $c - d \geq 0$ and serves as a measure of significance. Peaks that quickly get paired with higher peaks result from coarse samplings of a scalar function, whereas peaks with a large difference between creation and destruction may be assumed to represent real features in the data set—in Sect. 4.1.3 we describe an algorithm for automatically finding a significance threshold.

4.1.2 The High-Dimensional Case

For a discrete set of unstructured points $\mathbb{D} \subseteq \mathbb{R}^n$, we first need to approximate its connectivity before calculating persistent homology. The literature knows many *neighbourhood graphs* with different strengths and weaknesses for this purpose [6]. Since our data sets are not too sparse, we follow the approach of Chazal et al. [3] and use the *Rips graph* $\mathscr{R}_\epsilon$ of the domain $\mathbb{D}$. This graph requires a metric such as the Euclidean distance and a threshold ϵ. The Rips graph $\mathscr{R}_\epsilon$ has a vertex set of $V = \{0, 1, \ldots, |\mathbb{D}|\}$ and an edge set of $E = \{(u, v) \mid d_{uv} \leq \epsilon\}$, meaning that there is an edge between vertices u and v if their distance (measured using the metric) is less than or equal to the selected distance threshold. It endows the unstructured data set with connectivity information, which we require in order to perform mode-seeking just as in the 1-dimensional example.

We now apply a decomposition algorithm of Chazal et al. [3] to the scalar field. The algorithm requires that each vertex v of $\mathscr{R}_\epsilon$ has been assigned its corresponding scalar value $f(v)$. It consists of a *peak-seeking* and a *merge* phase:

Peak-seeking: Traverse the vertices of $\mathscr{R}_\epsilon$ in decreasing order of their function values. Connect each vertex to its neighbour with the largest function value. If the function values of all neighbours are smaller than the one of the current vertex, we have found a tentative peak. The edges that are created by this step correspond to discrete gradient lines of the scalar field in $\mathbb{D}$. When all vertices have been traversed, we have a collection of tree edges that decompose $\mathbb{D}$ into disjoint regions, similar to the regions shown in Fig. 1.

Merging: Traverse the vertices of $\mathscr{R}_\epsilon$ in decreasing order of their function values while maintaining a *union-find data structure* [5, pp. 561–568]. The root of each entry in the data structure corresponds to the peak vertex of a connected component. When arriving at an existing peak during the iteration, a new entry is added to the data structure. Upon arriving at a vertex that is not a peak, we again iterate over all neighbours. We merge neighbours that belong to peaks that are *lower* than the current peak into our component—this ensures that we always reach the highest possible peak. By contrast, we merge our current peak with the peaks of all neighbours that belong to *higher* peaks. This changes the value of the current peak, which in turn might trigger other merges with lower peaks again.

This algorithm is a reformulation of the *upper-star filtration* [9, pp. 164–165] in persistent homology. We obtain the corresponding persistence diagram from the algorithm by keeping track of the creation and destruction of components in the merge phase. The peaks of infinite persistence yield the desired decomposition of the domain. In order to have a fine-grained control about which peaks to consider significant and which peaks to prune because they are unstable, Chazal et al. [3] suggest merging peaks based on the differences in their persistence—this accounts for noise in the data. Given a threshold $\tau \in [0, \infty]$, merges in the second phase of the algorithm are only performed if the peak is lower and the persistence of the peak, i.e. the difference between the peak function value and the function value at the current vertex, is smaller than the threshold. In the algorithm above, we have $\tau = \infty$, meaning that all regions will be merged if possible. In general, the resulting regions are known to be stable [3]. Since τ affects which peaks are considered relevant and which peaks are considered noise by the algorithm, we subsequently describe an algorithm for choosing it automatically, based on the input data.

Choosing ϵ Varying ϵ controls the connectivity of the unstructured domain $\mathbb{D}$ of the data set. Very small values for ϵ result in a disconnected graph without any edges. Very large values for ϵ, on the other hand, make $\mathscr{R}_\epsilon$ the complete graph on n vertices. However, since $\mathscr{R}_{\epsilon'} \subseteq \mathscr{R}_\epsilon$ for $\epsilon' \leq \epsilon$, numerous useful choices for ϵ exist. In practice, methods such as dendrograms have proven to be effective [3, 6]. Here, we use a computationally cheap method [16] based on the average distances of points to their respective k nearest neighbours, for $k \in \{10, \ldots, 20\}$. This yields initial estimates for ϵ. We pick the smallest one such that $\mathscr{R}_\epsilon$ is still connected. If this is not possible, we add edges between the connected components of the graph. We assign these edges the average pairwise distance between the connected components, ensuring that $\tau = \infty$ results in a single region.

4.1.3 Threshold Selection

Finding a suitable threshold τ involves checking the separation of points in the persistence diagram. A result of Chazal et al. [3, Theorem 4.8] states that relevant

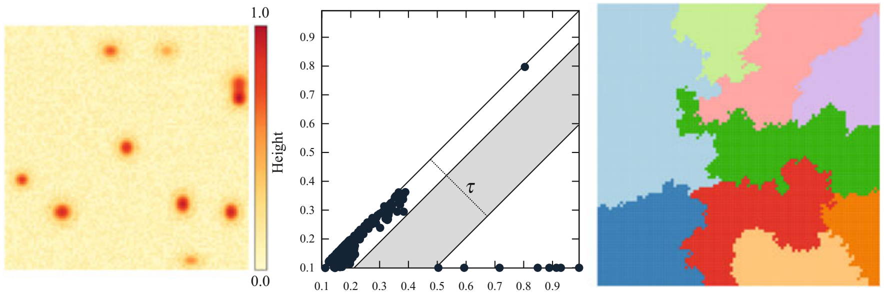


Fig. 2 A noisy scalar field (*left*), its corresponding persistence diagram (*middle*) with the region of largest separation, and the resulting decomposition (*right*). Our algorithm suggests a threshold of $\tau \approx 0.33$. The decomposition remains stable for $\tau \in [0.18, 0.48]$

peaks can be extracted from the persistence diagram if it contains a band of a certain width (that depends on ϵ) that does not contain any points. This is the largest empty region parallel to the diagonal we can draw into the persistence diagram (see Fig. 2, middle). The distance to the diagonal from any point within this region is then an admissible value for the threshold parameter τ, which remains stable over a large range.

The theorem makes assumptions about the structure and the sampling conditions of the input data—both of which are unavailable for real-world data. Nonetheless, we can apply a threshold selection process inspired by the theorem. The theorem essentially searches for the largest empty area in a persistence diagram. If this area is deemed large enough (which depends on assumptions about the input data and the function values of the Rips graph), the relevant peaks can be extracted with high probability. We can simulate this decision process by searching for the largest empty area in a persistence diagram and relating its size to the persistence values.

More precisely, we transform the coordinate system of the persistence diagram by a rotation of $\pi/4$. This makes the diagonal become the abscissa of the new coordinate system. In this transformed coordinate system, we sweep over all points by descending y-value and keep track of the vertical distance between subsequent points. Using the largest vertical distance—which is the width of the desired empty region—and the y-coordinate at which it was detected, we obtain a potential value for the threshold parameter τ. We then calculate the ratio of the width of the largest empty region to the mean width of all empty regions in the persistence diagram. In our experiments, we found that a ratio of at least four results in useful and stable thresholds for τ. Smaller ratios are indicative of much noise and may require manual selections via persistence diagrams. In each of our experiments, for instance, automated threshold selection only failed for at most one out of the six quality measures.

4.2 *Similarity Measure and Quality Visualization*

As a result of the scalar field decomposition algorithm, we are given a set of disjoint regions. We now want to compare the similarity between two such regions over different scalar fields, given the assumption that scalar fields with a similar amount of similarly placed maxima exhibit the same coarse behaviour. Given two regions $A = \{a_1, a_2, \dots\}$ and $B = \{b_1, b_2, \dots\}$, where each a and b refers to a vertex in the scalar field, we calculate their similarity using the *Jaccard index*, i.e.

$$J(A, B) = |A \cap B| / |A \cup B| \in [0, 1]. \tag{3}$$

Inspired by the bottleneck distance and Wasserstein distance calculations between persistence diagrams [9, pp. 229–236], we propose an *assignment problem* for assessing the global degree of similarity between two scalar fields. We define the cost between two regions as $1 - J(A, B)$, meaning that we want to penalize regions that do not overlap. To account for different numbers of regions in two scalar fields, we include empty dummy regions so that a region may also be matched with no region from the other scalar field. The total cost of the assignment problem serves as an indicator of how much the decompositions calculated from two scalar fields differ. The pairwise total costs between two scalar fields yields a matrix of pairwise distances. Using a 1-dimensional PCA of this matrix, we obtain a linear ordering of the scalar fields which reflects their respective distances—similar scalar fields are thus placed in proximity to each other. This allows us to read off which properties of an embedding are most likely retained.

Note that $J(\cdot, \cdot)$ cannot differentiate between all functions. It is possible to have two very different functions whose topological decompositions are very similar. We did not encounter this in our experiments, though.

Local Similarity Scatterplot To provide a local degree of similarity assessment, we select a reference scalar field. We now solve the assignment problem for each remaining scalar field and keep track of the costs for matching all regions in the reference field. We then visualize the average assignment costs of the reference field using three colours (red, orange, green; each corresponding to 33% of the value range) on the embedding. In the optimal case, all other scalar fields result in the same decomposition as the reference scalar field—the visualization will thus not highlight any region. Green regions indicate an (almost) perfect agreement with all other quality measures. Orange regions show that there are mild differences to the other measures, whereas red regions highlight regions that are severely mismatched with the remaining measures—thereby indicating that a region is unique and does not occur often in the other scalar fields.

5 Results

In the subsequent data sets, we assume that the user has chosen a specific property that needs to be minimized by the embedding, e.g. stress. With this in mind, users can choose a dimensionality reduction scheme that minimizes this measure on a global scale. We now have the additional task of finding out whether *other* measures are retained as well on the data. As a data pre-processing step, we normalize the scalar values of each quality measure to [0, 1] such that 0 represents no error (highest quality) and 1 represents the maximum error (lowest quality). This is only required to ensure that the scales of different persistence diagrams may be compared more easily.

5.1 Swiss Roll

The *Swiss roll* data set was introduced by Tenenbaum et al. [20] as an example of how non-linear embedding methods (Isomap) are able to outperform classical linear embeddings (PCA) in certain cases. The data set consists of a "curled up" plane. Isomap is one of the few algorithms capable of embedding this data set properly. We thus work with quality measures on the Isomap embedding. Figure 3, left, shows the embedding of the data set (using rainbow colours to indicate the position along the curled plane). We apply our algorithm on the scalar fields induced by the quality measures and choose the threshold automatically. All distance-based quality measures exhibit comparatively large errors along the bottom and the top of the embedded data, while the middle region contains almost no errors. The range of these errors is very small, though. This effect is caused by the unwrapping that distorts distances on a global scale. The same effect occurs somewhat less obviously in all rank-based quality measures. Here, the impact of the unwrapping is somewhat mitigated by the local neighbourhood size—although we now have the additional error source of small changes in neighbourhoods. Our automated threshold selection

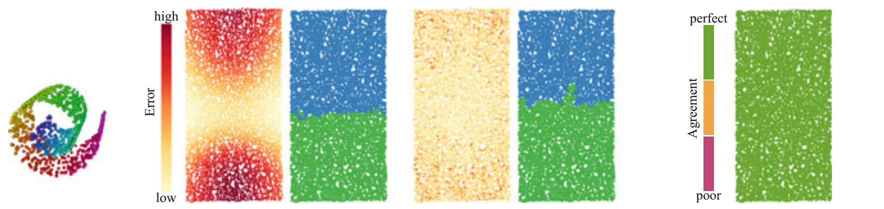


Fig. 3 Swiss roll and two example measures with their decompositions. All quality measures exhibit a large amount of errors in the bottom and top regions of the embedding. This results in a two-region decomposition. The local similarity plot hence does not change, regardless of the selected reference measure

results in the same decomposition of the data, though. The measures hence exhibit a very similar behaviour on the data set, proving that Isomap preserves their properties to the same extent. Combined with the knowledge about the small range of the errors, we thus conclude that Isomap yields a perfect embedding of the *Swiss roll*.

5.2 *Handwritten Digits*

We use the *Optical Recognition of Handwritten Digits* data set from the UCI Machine Learning Repository [13]. It consists of 5620 instances of 64-dimensional feature vectors describing the handwritten digits of multiple writers. We will compare the behaviour of different quality measures on a linear embedding (PCA) and on a non-linear embedding (t-SNE) of the data.

PCA Figure 4 shows a selection of different quality measures for this data set. They all exhibit a large spread in their value range, indicating that the errors are substantial. Our algorithm thus rates all quality measures on this data to be very

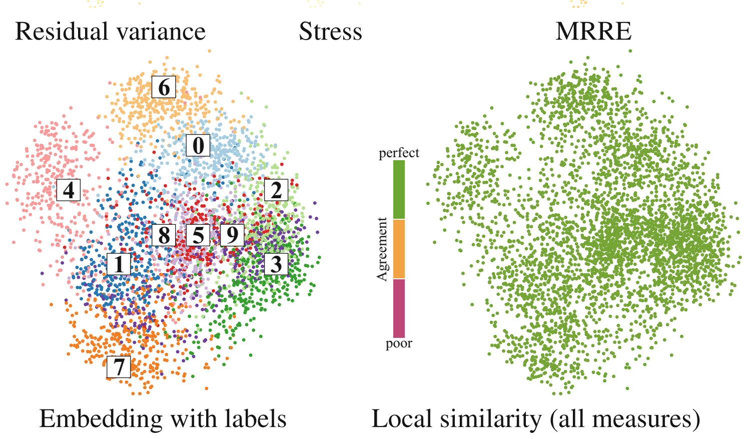


Fig. 4 Errors in the PCA projection concentrate on a single region of slightly variable size in the data. This is easily seen in the plots of residual variance and Stress, for example, but not in MRRE. Here, larger errors seem to be distributed uniformly. Upon decomposing the data, these peaks are shown to be of low persistence, resulting in one region with a single peak

similar. Except for MRRE, we observe a single region of variable size in the central region of the embedding. By contrast, MRRE appears to contain multiple smaller peaks but not a single expressive region. Our decomposition shows that due to the higher baseline error in the quality measure, the detected peaks do not have a sufficiently high persistence. Thus, the decomposition of all measures results in a single large region. By assigning labels to the data set (Fig. 4, bottom), we see that the quality measures have their largest errors around the region of the digits 5, 8 and 9, as this region is not well-separated by PCA. The embedding is thus an example of a compromise solution: The embedding favours global over local structures (as measured by MRRE), but errors accumulate in a single region.

t-SNE The relative distances of the different quality measures (Fig. 5, top) show that three measures (MRRE, rank correlation, and residual variance) are very similar: With a high baseline error, they only have one significant peak in each of the regions of the embedding. The stress measure yields a more fine-grained decomposition. Here, regions A, B, C, and D are split (instead of staying a larger region). When comparing with the labelled embedding (Fig. 5, bottom), we see that e.g. Region B corresponds to a separation of the digit 1. The higher stress values in the bottom part of this region indicate that the distances in t-SNE do not reflect the

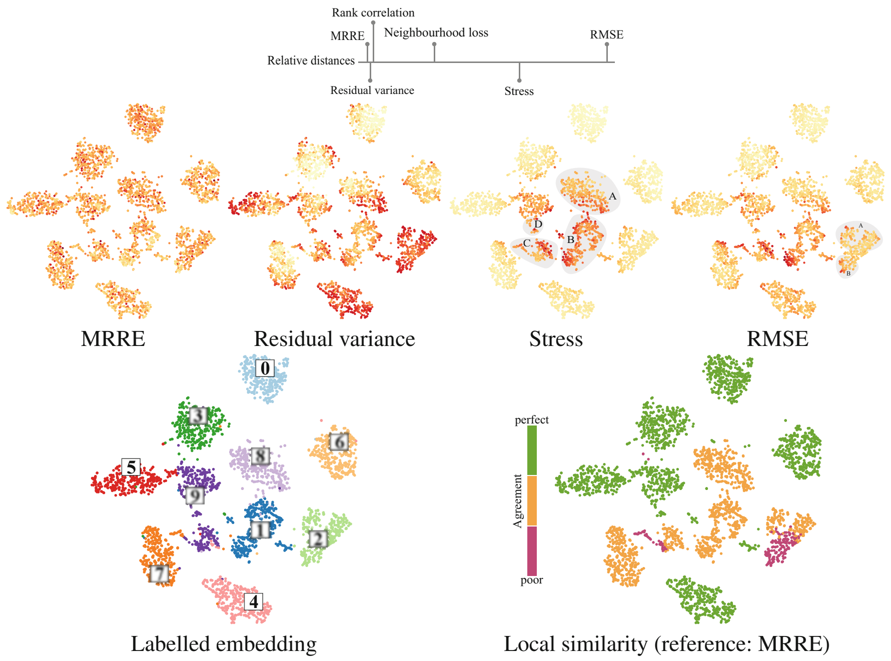


Fig. 5 Large errors in the t-SNE projection occur uniformly in the regions corresponding to the digits. This is indicated well by e.g. MRRE. RMSE, by contrast, highlights regions with a non-uniform error distribution

high-dimensional distances very well. RMSE differs the most from MRRE. Here, we see an additional split introduced by two new regions A and B. This is caused by two peaks of high persistence at the top and bottom of the marked region. The new regions correspond to the digit 2. RMSE highlights that t-SNE is distorting the distances for these points in favour of the very good global separation in the embedding.

The contested regions are highlighted in the local similarity scatterplot (Fig. 5, bottom), using MRRE as a reference measure. Orange and red highlight regions in which the other measures differ the most from MRRE: For the most part, t-SNE is able to separate the data set very well, but local distances are distorted locally for some digits. When comparing the embeddings obtained from PCA and t-SNE, we can check e.g. the residual variances to conclude that t-SNE with a residual variance of ≈ 0.55 commits smaller distance errors on average than PCA with a residual variance of ≈ 0.78, thereby offering a better global separation of the digits. Hence, t-SNE might be preferable over a PCA embedding, despite the localized errors it introduces.

5.3 Concrete Compressive Strength

The *concrete compressive strength* data from the UCI Machine Learning Repository [13] contains 1030 mixtures of 8 different concrete compounds. We use PCA to obtain an embedding because it is known to yield a good overview of this data set [12]. Since the data are known to exhibit linear structures [10], which should ideally be preserved locally, we use Neighbourhood loss as the reference quality measure. The relative distances (Fig. 6, top) indicate that there are three groups of

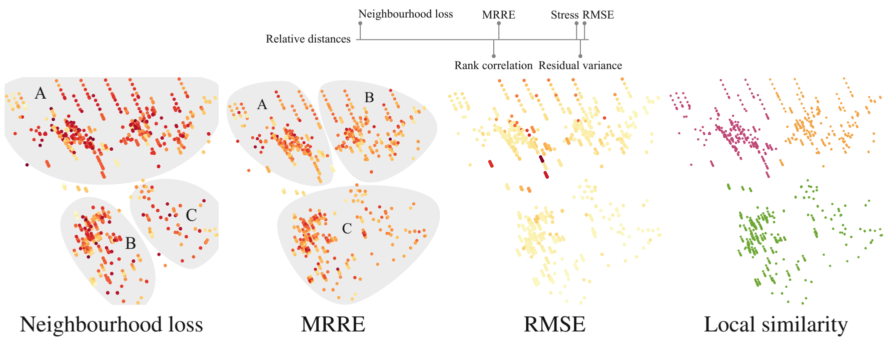


Fig. 6 Neighbourhood loss exhibits three maxima of high persistence, resulting in a decomposition in three regions. MRRE yields a different decomposition into three regions. RMSE (representing distance-based measures) exhibits large errors in the upper region only. Using neighbourhood loss as a reference measure, the local similarity scatterplot shows that the measures disagree mostly in the upper region of the embedding

quality measures on the data. The distance-based quality measures stress, RMSE, and residual variance all exhibit a single large peak around the top of the data set. Since there are no maxima of high persistence in the remaining part of the data set, our decomposition algorithm results in a single large region. By contrast, MRRE and rank correlation, decompose the top part of the data further into two regions A and B because the corresponding peaks are separated by lower values. Neighbourhood loss results in a complementary decomposition, showing that the top part of the data is dominated by a single peak of high persistence, while the bottom part decomposes into two regions B and C.

The local similarity scatterplot indicates the agreement of quality measures with respect to neighbourhood loss. We can see that the measures disagree on both the upper regions, indicating that their error distributions are very different. This leads us to question the quality of the embedding at these areas—which are rated very differently by the rank-based measures, while the distance-based measures mostly agree.

6 Conclusion

We introduced a method for comparing the behaviour of different quality measures for dimensionality reduction algorithms. Our method currently decomposes scalar fields according to their maxima only. For future work, we plan on evaluating whether the inclusion of minima would further increase the expressive power. We also want to evaluate the effects of different neighbourhood graph type approximations [6]. Furthermore, we want to investigate how different measures for feature relevance in scalar fields, such as *topological saliency* [8], can improve the results. Last, we want to look for alternatives to the Jaccard index for measuring the similarity of decompositions.

References

1. Bertini, E., Tatu, A., Keim, D.: Quality metrics in high-dimensional data visualization: an overview and systematization. IEEE Trans. Vis. Comput. Graph. **17**(12), 2203–2212 (2011)
2. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. **24**(2), 75–94 (2003)
3. Chazal, F., Guibas, L.J., Oudot, S.Y., Skraba, P.: Persistence-based clustering in Riemannian manifolds. J. ACM **60**(6), 41:1–41:38 (2013)
4. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Trans. Pattern Anal. Mach. Intell. **17**(8), 790–799 (1995)
5. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 3rd edn. MIT Press, Cambridge (2009)
6. Correa, C., Lindstrom, P.: Towards robust topology of sparsely sampled data. IEEE Trans. Vis. Comput. Graph. **17**(12), 1852–1861 (2011)

7. Correa, C., Lindstrom, P., Bremer, P.T.: Topological spines: a structure-preserving visual representation of scalar fields. IEEE Trans. Vis. Comput. Graph. **17**(12), 1842–1851 (2011)
8. Doraiswamy, H., Shivashankar, N., Natarajan, V., Wang, Y.: Topological saliency. Comput. Graph. **37**(7), 787–799 (2013)
9. Edelsbrunner, H., Harer, J.: Computational Topology: An Introduction. American Mathematical Society, Providence, RI (2010)
10. Gerber, S., Bremer, P.T., Pascucci, V., Whitaker, R.: Visual exploration of high dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. **16**(6), 1271–1280 (2010)
11. Lee, J.A., Verleysen, M.: Quality assessment of dimensionality reduction: rank-based criteria. Neurocomputing **72**(7–9), 1431–1443 (2009)
12. Lee, J.H., McDonnell, K.T., Zelenyuk, A., Imre, D., Mueller, K.: A structure-based distance metric for high-dimensional space exploration with multidimensional scaling. IEEE Trans. Vis. Comput. Graph. **20**(3), 351–364 (2014)
13. Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
14. Oesterling, P., Heine, C., Jänicke, H., Scheuermann, G., Heyer, G.: Visualization of high-dimensional point clouds using their density distribution's topology. IEEE Trans. Vis. Comput. Graph. **17**(11), 1547–1559 (2011)
15. Oesterling, P., Heine, C., Weber, G.H., Scheuermann, G.: Visualizing nD point clouds as topological landscape profiles to guide local data analysis. IEEE Trans. Vis. Comput. Graph. **19**(3), 514–526 (2013)
16. Rieck, B., Mara, H., Leitte, H.: Multivariate data analysis using persistence-based filtering and topological signatures. IEEE Trans. Vis. Comput. Graph. **18**(12), 2382–2391 (2012)
17. Sauber, N., Theisel, H., Seidel, H.P.: Multifield-graphs: an approach to visualizing correlations in multifield scalar data. IEEE Trans. Vis. Comput. Graph. **12**(5), 917–924 (2006)
18. Schneider, D., Wiebel, A., Carr, H., Hlawitschka, M., Scheuermann, G.: Interactive comparison of scalar fields based on largest contours with applications to flow visualization. IEEE Trans. Vis. Comput. Graph. **14**(6), 1475–1482 (2008)
19. Schneider, D., Heine, C., Carr, H., Scheuermann, G.: Interactive comparison of multifield scalar data based on largest contours. Comput. Aided Geom. Des. **30**(6), 521–528 (2013)
20. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319–2323 (2000)
21. van der Maaten, L.J.P., Postma, E.O., van den Herik, H.J.: Dimensionality reduction: a comparative review. Technical Report 005, Tilburg University (2009)

Part III
Scalar Field Topology

Fast Similarity Search in Scalar Fields using Merging Histograms

Himangshu Saikia, Hans-Peter Seidel, and Tino Weinkauf

Abstract Similarity estimation in scalar fields using level set topology has attracted a lot of attention in the recent past. Most existing techniques match parts of contour or merge trees against each other by estimating a best overlap between them. Due to their combinatorial nature, these methods can be computationally expensive or prone to instabilities. In this paper, we use an inexpensive feature descriptor to compare subtrees of merge trees against each other. It is the data histogram of the voxels encompassed by a subtree. A small modification of the merge tree computation algorithm allows for obtaining these histograms very efficiently. Furthermore, the descriptor is robust against instabilities in the merge tree. The method is useful in an interactive environment, where a user can search for all structures similar to an interactively selected one. Our method is conservative in the sense that it finds all similar structures, with the rare occurrence of some false positives. We show with several examples the effectiveness, efficiency and accuracy of our method.

1 Introduction and Related Work

Finding structural similarities in scalar fields is of prime importance when one needs to analyze repeating patterns or symmetric arrangements in the data. To this end, several methods involving feature-based analysis using the topology formed by level-sets have been proposed. The *merge tree* is one such arrangement of features which traces the connectivity evolution of sub/super-level sets in the data. It is easy to see that similar structures exhibit similar level-set arrangements and hence similar branchings in the merge tree, each such branch or subtree representing a unique structurally important region. The *contour tree* [2] is an extension of the merge tree as it contains the combined information for both sub and super-level sets.

H. Saikia (✉) • T. Weinkauf
KTH Royal Institute of Technology, Stockholm, Sweden
e-mail: saikia@kth.se; weinkauf@kth.se

H.-P. Seidel
MPI Informatik, Saarbrücken, Germany
e-mail: hpseidel@mpi-inf.mpg.de

The *Reeb Graph* [6] is a generalized form of the contour tree to include non-simply connected manifolds. The *Morse-Smale Complex* [7] is a segmentation of the data into regions of uniform gradient flow. These topological arrangements give rise to graph representations like the extremum graph [17] or topological spines [5].

Beketayev et al.[1] compare two merge trees by comparing all of their possible branch decompositions. This method provides an accurate ϵ-match with respect to noisy perturbations, but is not practical for interactive applications. This is because computing all possible branch decompositions has exponential complexity and a memoized solution that the authors propose still involves a higher order polynomial runtime. Precisely, $O(n^5)$ for comparing two trees of n nodes each for a given threshold ϵ.

Thomas et al. [16, 18] present methods to visualize all symmetric structures in a scalar field using the contour tree. In [16], the authors cluster the different branches in the branch decomposition tree of the contour tree to display symmetric arrangements. Using a single branch decomposition tree, however, is less robust against noise. In [18] they extract iso-surfaces using the contour tree and cluster them in a feature space. In another work,[17], the authors achieve similar results using the extremum graph.

Saikia et al. [12] perform a similarity search for any structurally significant region as given by a subtree in the merge tree by first pre-computing the similarity of all possible subtrees to all other existing subtrees. The method involves computing branch decomposition trees for every subtree and overlaying them in the best way possible. The similarity is then computed as the minimum cost of this overlay. Although the method is fast owing to a memoized algorithm to compute and compare branch decomposition trees, the result can be affected by perturbations that lead to a different order in the hierarchy of branching.

We employ the method by Saikia et al. [12] in the sense that we compare all subtrees of a merge tree against each other. However, in this paper, instead of overlaying branch decomposition trees obtained from the subtrees, we describe every subtree using a feature vector. This is given by the intensity distribution of the member voxels within a subtree region and is thus appropriately termed as a histogram. These histograms can be efficiently computed using just a small modification to the merge tree computation algorithm, exploiting the fact that the tree is created bottom-up. This augmented algorithm allows us to compute the histograms for every subtree on-the-fly, i.e., while the merge tree is being computed. The merge tree computation is done by a single sweep through a sorted (by function value) array of the voxels in the data as described in [2, 15]. Augmented contour tree algorithms have also been used before for topological simplification by computing local geometric measures such as volume in [3].

Histograms have had a long standing use in data visualization [8], automatic transfer function generation [9, 10] and various volume rendering techniques [14, 20]. Histograms have also been used in shape retrieval, where they are defined on the distances from the barycentric center of a simplicial mesh to its surface triangles [19]. This distance metric is known as a *cord*. A histogram presents itself as a simple and powerful statistical representation of the data distribution. It is also shown in

[4, 13] that the histogram has a close relationship with isosurface area. However, we do not focus on a precise calculation of the distribution within a region for the sake of simplicity and efficiency.

In the following sections, we provide some background and notation in Sect. 2, the method to compute merging histograms in Sect. 3, how similarity search is performed using the new method in Sect. 4, show a few results obtained using our method in Sect. 5 and conclude after some evaluation and discussion of our method in Sect. 6.

2 Background and Notation

Given a Morse scalar field $f : \mathbb{R}^n \rightarrow \mathbb{R}$, and any value c in the range of the function f, super-level sets L_c^+ and sub-level sets L_c^- are defined as follows

$$L_c^+ = \{\mathbf{x}|f(\mathbf{x}) \geq c\} \tag{1}$$

$$L_c^- = \{\mathbf{x}|f(\mathbf{x}) \leq c\}. \tag{2}$$

For the sake of convenience let us only talk about super-level sets from here on. Each super-level set can have one or more disconnected components. These components are born at the *maxima*, merge with other components at *saddles*, and finally merge and disappear at the *global minimum*.

We define a region $\mathscr{R}$ as the set of voxels belonging to any component *just* before it merges with another component. A is the representation of the connectivity evolution of these regions. Every birth or merge is represented as a node in this tree, and these nesting relationships are denoted by edges. Every region is represented as a non-empty subtree, the largest region being the entire merge tree. In case of super-level sets, the merge tree is often called a .

We denote a merge tree as $M = (\mathscr{N}, \mathscr{E})$ where $\mathscr{N} = \{n_1, n_2, \ldots, n_p\}$ are the nodes and $\mathscr{E} = \{e_1, e_2, \ldots, e_{p-1}\}$ are the edges of the tree. Note that a tree with p nodes has $p - 1$ edges.

Figure 1 illustrates the idea of regions and edges. Region $\mathscr{R}_A$ is given by $\{e_A\}$ and corresponds to the entire blue area, $\mathscr{R}_B$ by $\{e_B\}$—the entire red area and $\mathscr{R}_C$ by $\{e_A, e_B, e_C\}$—the entire green, red and blue areas together.

3 Merging Histograms

3.1 Histograms as Feature Descriptors

Our objective is to compare subtree regions of a merge tree. Given that these regions are similar in data value, size and also to a certain extent the geometry, a

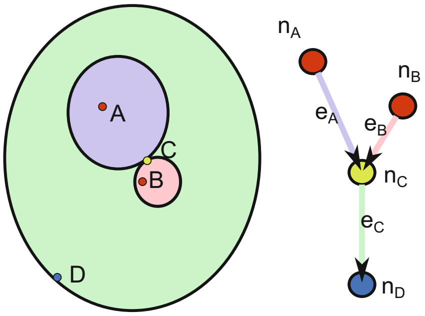


Fig. 1 A region in a scalar field and its join tree $J = (\mathcal{N}, \mathcal{E})$. Here $\mathcal{N} = \{n_A, n_B, n_C, n_D\}$ and $\mathcal{E} = \{e_{A\to C}, e_{B\to C}, e_{C\to D}\}$ or simply $\{e_A, e_B, e_C\}$. The two *red* nodes are maximas and leaf nodes, the *yellow* node is a saddle, and the *blue* node is the global minimum and the root. The edges signify the corresponding areas in the same color and the *arrows* show the unidirectional path from every maximum to the global minimum

of the distribution of the member voxels in the data range is a good measure for comparison. This is because a histogram roughly encapsulates the surface area of a region at various iso-levels. This gives us a good measure of the distribution of intensity in the region.

Given a scalar function f, and a subtree region $\mathcal{R}$, the histogram is computed by binning all voxels (or pixels for 2D) in this region into a number of bins by their function value.

Figure 2 shows an illustration of this feature descriptor. Two structurally different subtree regions of a dataset can be distinguished from each other by comparing their individual histograms.

3.2 Computation

The good thing about constructing these histograms is that they can be computed incrementally during the construction of the merge tree itself. The part *not* marked in red in Algorithm 1 shows the classic merge tree computation. Let us look closely at the three different cases encountered while sweeping through the sorted data, and how the histograms have to be modified during this process.

Case 1: extremum This is the *if* clause in line 10 of Algorithm 1. In this case a new node n_i is added to the set of nodes and a new component in the union-find c_i starts. The initial histogram is set to zero for all bins, i.e., $\mathbf{h}_i = \mathbf{0} = [0, \ldots, 0]$.

Case 2: regular voxel This is the *else-if* clause in line 15 of Algorithm 1. Here the current voxel is added to the component it belongs to. The appropriate histogram bin has to be incremented for the corresponding component. Let us assume the histogram bin that this point falls into is b. Then, $h_{i,b} \leftarrow h_{i,b} + 1$, where c_i is the component to which this voxel belongs to.

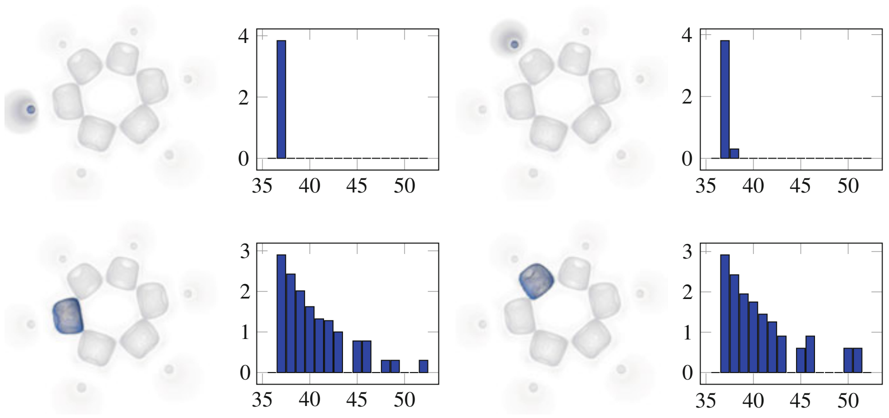


Fig. 2 Four different subtree regions in the benzene dataset along with their corresponding histogram feature vectors. The x-axis in the plots shows the corresponding histogram bins and the y-axis shows the log scaled values of the total number of voxels in each bin. A total of 100 bins were used in all cases. The x-range with only non-zero values is shown. As can be seen, the regions corresponding to the hydrogen atoms have similar histograms as do the regions corresponding to the carbon atoms

Case 3: saddle This is the *else* clause in line 19 of Algorithm 1. This is the case when a point is in contact with two or more edges. All edges pertaining to every component are added to the set of edges. A new component is constructed and a new node corresponding to this voxel is added to the set of nodes. The histogram corresponding to this component has to be initialized using the sum of all histograms pertaining to all of the components in set C_G. Thus, for a saddle node n_i and number of bins B we have $\mathbf{h}_i = [h_{i,1}, h_{i,2}, \ldots, h_{i,B}]$, where each bin is given by

$$h_{i,b} = \sum_{c_j \in C_G} h_{j,b} \tag{3}$$

The only modification that needs to be done to the algorithm is to incorporate these three cases. Thus, when the merge tree is computed, the feature vectors are also pre-computed as a result.

4 Similarity Search Using Merging Histograms

After the computation is performed using Algorithm 1, in addition to the merge tree, we also obtain the feature descriptors for every subtree region in the form of a histogram of values. Using these feature descriptors we compare every subtree region with each of the others and store the results in a distance matrix. Later we

```
Data: The scalar field $f$, with vertices $\mathbf{x}_1, \ldots, \mathbf{x}_m$ in sorted order.
Result: If $f(\mathbf{x}_i) \geq f(\mathbf{x}_j)$ for $i < j$ then Join Tree $J = (\mathscr{N}, \mathscr{E})$. If $f(\mathbf{x}_i) \leq f(\mathbf{x}_j)$ for $i < j$ then
        Split Tree $S = (\mathscr{N}, \mathscr{E})$. Also the set $H = \{h_1, \ldots, h_i, \ldots\}$ containing all histograms
        corresponding to every edge $e_i \in \mathscr{E}$ and subtree region $\mathscr{R}_i$.
begin
    $\mathscr{N} := \emptyset$ , $\mathscr{E} := \emptyset$ , $H := \emptyset$, UnionFind $U$
    for $i \leftarrow 1$ to $m - 1$ do
        Set of neighbors of $\mathbf{x}_i$ : $G = \{\mathbf{g}_1, \ldots, \mathbf{g}_p\}$
        Set of components containing $G$ : $C_G := \emptyset$
        for $j \leftarrow 1$ to $p$ do
            $C_G \leftarrow C_G \bigcup findComponent_U(\mathbf{g}_j)$
        end
        $b := bin(f(\mathbf{x}_i))$                                  // Finding the bin value.
        if $|C_G| = 0$ then
            $c_v \leftarrow createNewComponent_U(\mathbf{x}_i)$
            $\mathscr{N} \leftarrow \mathscr{N} \bigcup \{n_i\}$
            $\mathbf{h}_v = [0, \ldots, 0]$                                   // Initializing.
            $h_{v,b} \leftarrow h_{v,b} + 1$
        else if $|C_G| = 1$ then
            $C_G = \{c_v\}$
            $addMemberToComponent_U(c_v, \mathbf{x}_i)$
            $h_{v,b} \leftarrow h_{v,b} + 1$                             // Incrementing the bin.
        else
            $C_G = \{c_a, \ldots, c_k\}$
            $\mathscr{N} \leftarrow \mathscr{N} \bigcup \{n_i\}$
            $\mathscr{E} \leftarrow \mathscr{E} \bigcup \{e_{a \to i}, \ldots, e_{k \to i}\}$
            $c_v \leftarrow createNewComponent_U(\mathbf{x}_i)$
            $H \leftarrow H \bigcup \{\mathbf{h}_a, \ldots, \mathbf{h}_k\}$                  // Adding to output.
            $\mathbf{h}_v = \mathbf{h}_a + \ldots + \mathbf{h}_k$                                  // Merging.
        end
    end
    $\mathscr{N} \leftarrow \mathscr{N} \bigcup \{n_m\}$
end
```

Algorithm 1: An augmented version of the classic merge tree algorithm to account for merging histograms. The augmented parts are shown in red.

can reference any of these subtree regions by means of interactively selecting it, and querying for its best matches in the distance matrix.

4.1 Distance Measure

The distance measure between two histograms should be as discriminative as possible. To compare two histograms we use the L_2-norm of the log-scaled bin values. Log scaling helps to smooth the histograms a little bit and make the comparison function slightly robust to noise. Thus, the distance d between two subtree regions $\mathscr{R}_i$ and $\mathscr{R}_j$ can be given as

$$d(\mathscr{R}_i, \mathscr{R}_j) = \sum_{b \in [1,B]} ||g(h_{i,b}) - g(h_{j,b})|| \tag{4}$$

where g is given by

$$g(x) = \begin{cases} 0, & \text{if } x = 0 \\ \log x, & \text{otherwise.} \end{cases} \tag{5}$$

4.2 *Querying the Distance Matrix*

A distance matrix is then constructed using the distance values for every pair of subtree regions computed using Eq. 4. Any row and column in this matrix refers to a subtree region. As has been seen before in Sect. 2 the number of subtree regions is equal to the number of edges in the merge tree. This shows that the time complexity of computing a distance matrix and its size are dependent only on the merge tree and not on the size of the data.

In an interactive setting, a user picks any voxel in the dataset. Since every voxel is contained within an edge, the corresponding edge can be queried for. And since every edge corresponds to a unique subtree, we can immediately identify which subtree region is selected by the user. Once this region is known, similar regions to it can be queried simply by looking for the smallest values in the corresponding row of the distance matrix. A distance threshold slider also allows the user to increase or decrease the threshold and show correspondingly more or lesser close matches.

5 Results

Now we show a few results obtained using our method. Figure 3 shows a volume rendering of the Benzene data set and two different search regions. As can be seen, the sixfold symmetry in the molecule is evident from the closest matches to the selected subtree region. Figure 4 shows the EMDB-1603 data set and a few different search regions. The dataset had nearly 38,000 edges in its merge tree, which were then simplified to around 900 edges by eliminating low persistent edges (below 2%). The particle exhibits a ninefold symmetry as seen in all three selections and their closest matches. As can be observed, interesting structures which could otherwise not be seen clearly are apparent when singled out. Figures 5 and 6 show two other protein datasets alongwith some interesting self-similar structures. Figure 7 shows another EMDB dataset with a helical structure. This is identified in the selected subtree region and its closest matches. Figure 8 shows another complicated dataset where the symmetric arrangements are revealed during exploration.

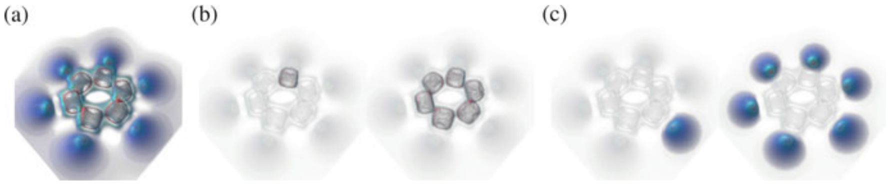


Fig. 3 Scalar field depicting the potential around a Benzene molecule. Two different sixfold symmetry regions are seen. (**a**) Full volume rendering. (**b**) First selected region and its closest matches. (**c**) Second selected region and its closest matches

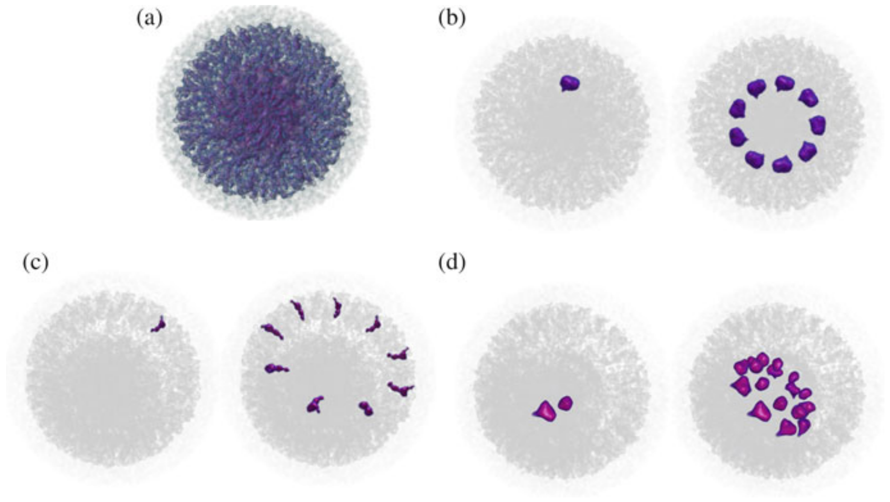


Fig. 4 EMDB-1603. A cryo-electron microscopy reconstruction of a recombinant active ribonucleoprotein particle of influenza virus. The ninefold symmetry is apparent in the matchings shown. Different transfer functions are used for the three different selections for better visibility. (**a**) Full volume rendering. (**b**) First selection and its corresponding best matches. (**c**) Second selection visualized at a slightly different angle and its best matches. (**d**) Third selection and its best matches

All EMDB data sets are obtained from the Protein Data Bank Japan (pdbj.org) online archive. The results are all rendered using the Voreen volume rendering engine (voreen.uni-muenster.de).

6 Discussion

6.1 Runtime Comparison with Tree Overlay Methods

Computing feature vectors on the fly while computing the merge tree itself leads to a more efficient implementation as opposed to performing the two steps sequentially

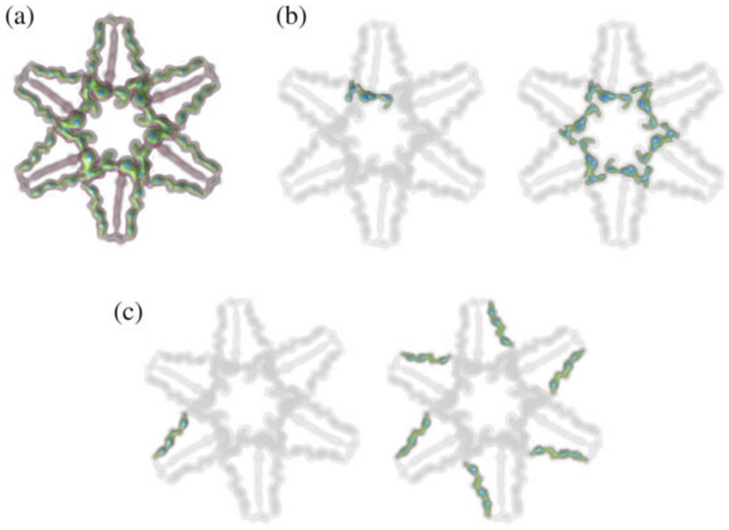


Fig. 5 EMDB-1201. The myosin V inhibited state obtained by cryo-electron tomography. There exists a sixfold symmetry. (**a**) Full volume rendering. (**b**) first selection and its best matches. (**c**)Second selection and its best matches

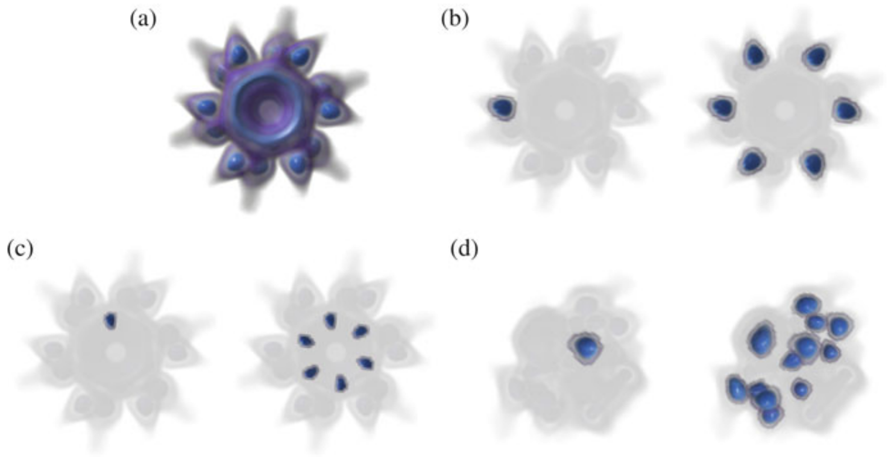


Fig. 6 EMDB-1706. Cryo-electron reconstruction of Lactococcal phage p2 baseplate. There exists a sixfold symmetry. (**a**) Full volume rendering. (**b**) First selection and its closest matches. (**c**) Second selection and its matches. (**d**) Selection in (**b**) from a different angle. A few more best matches are shown to reveal a duplicate sixfold symmetric pattern

as in the Extended Branch Decomposition Graph method in [12]. There is almost no overhead of running the augmented merge tree computation algorithm as opposed to the classic algorithm, as can be seen in Table 1.

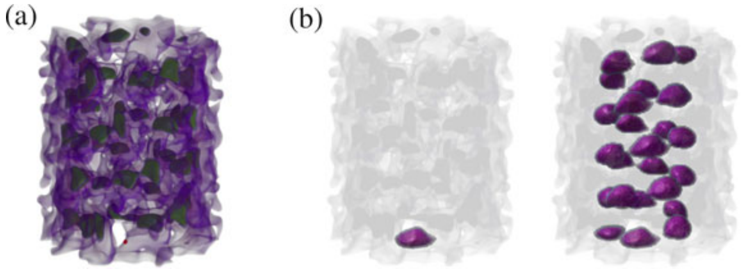


Fig. 7 EMDB-2400. MuB is an AAA+ ATPase that forms helical filaments to control target selection for DNA transposition. (**a**) Full volume rendering. (**b**) A selection and its closest matches. The helical structure is apparent

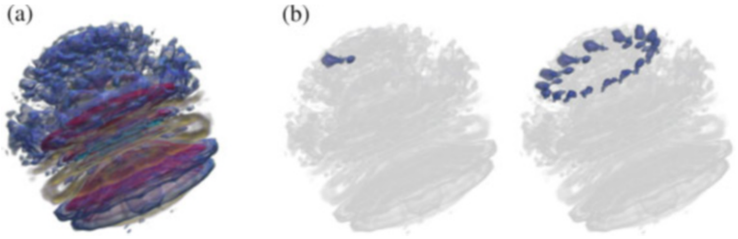


Fig. 8 EMDB-5300. Structural Diversity of Bacterial Flagellar Motors: Campylobacter jejuni. (**a**) Full volume rendering. (**b**) A selection and its closest matches.

For searching a subtree region, a distance matrix has to be constructed and this can be achieved in $O(n^2B)$ time as opposed to $O((n\log n)^2)$ in [12]. Note that both methods, the one in this paper and the method in [12], compare all subtrees to all others and then allow for similarity searching interactively in real-time. Comparing two subtree regions using our method is very fast as it just compares the two individual histograms in $O(B)$ time. Hence our method is orders of magnitude faster. This can be seen in Table 2.

6.2 Robustness

Merging histograms perform well under small perturbations in the data. This can be immediately observed from the fact that there is no ordering of hierarchy in this representation unlike a branch decomposition tree. Since every ROI is defined only by its complete underlying structure and not on the precise order in which its containing iso-contours evolved, this method is immune to slight changes in the merge tree due to noise (see Fig. 9).

For more complicated branchings however, the histogram cannot accurately represent the branching hierarchy. This means that two trees with extremely

Table 1 Merge tree computation times (in milliseconds) for various data sets—without histograms and with histograms of bin sizes 10 and 100

			Merge tree computation time in *ms*		
				Augmented algorithm 1	
Dataset	Vertices	Edges in join tree	Classic algorithm	With 10 bins (% increase)	With 100 bins (% increase)
Benzene	101^3	23	678	752 (10.9)	700 (3.2)
Neghip	64^3	252	148	157 (6.1)	156 (5.1)
EMDB-1603	160^3	38,671	4934	5470 (10.9)	6336 (28.4)
EMDB-1201	180^3	41,169	1589	1873 (17.9)	2531 (59.3)
EMDB-1706	130^3	155	935	1112 (18.9)	1141 (22.0)
EMDB-2400	128^3	2003	1973	2189 (11.0)	2083 (5.6)
EMDB-5300	60^3	3685	191	210 (10.0)	237 (24.1)

As can be seen there is only a slight overhead to adding histogram information to the computation phase. All operations were performed on a machine with a 2.66 GHz Intel Xeon processor and 12 GB main memory

Table 2 Total feature computation and comparison times (in milliseconds) for various data sets—using the eBDG method in [12] and the histogram method

			Feature computation and comparison time in *ms*	
Dataset	Vertices	Edges in join tree (after simplification)	eBDG approach	Our approach
Benzene	101^3	23	506	489
Neghip	64^3	252	268	142
EMDB-1603	160^3	38,671 (910)	54,487	6572
EMDB-1201	180^3	41,169 (198)	13,955	3749
EMDB-1706	130^3	155	672	702
EMDB-2400	128^3	2003	764,629	3632
EMDB-5300	60^3	3685	6,805,101	7081

As can be seen, with more number of edges the eBDG method requires far more time to compare all features against each other than the histogram method. All operations were performed on a machine with a 2.3 GHz Intel i7 processor and 16 GB main memory

different branchings (and hence underlying topological structure) might end up having very similar histograms (see Fig. 10). This might result in false negatives. However, note that similar subtrees have similar histograms, i.e., the method finds similar structures independent of their complexity.

6.3 Histogram Resolution and Comparison

The bin number reflects the resolution at which we sample our data. The higher the bin number, the greater the resolution. In our examples we observe that a bin number

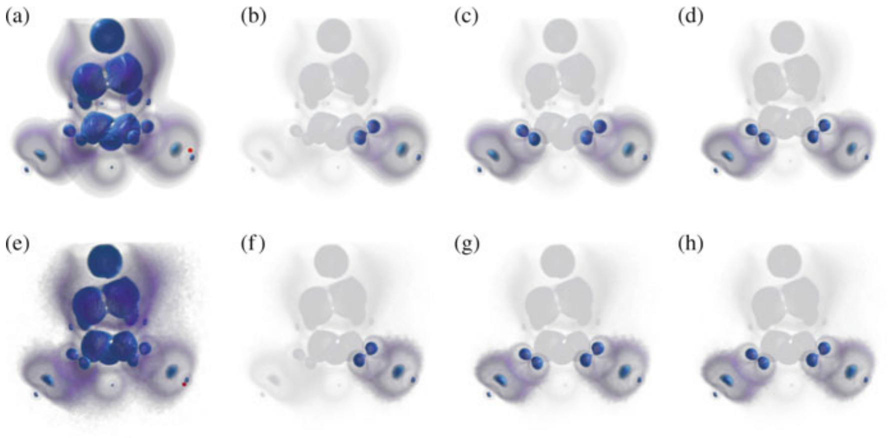


Fig. 9 Neghip dataset. The results are in accordance with the eBDG method described in [12]. The matching results are stable even in presence of noise. (**a**) Full volume rendering. (**b**) A selection. (**c**) Similar structures to (**b**) using merging histograms. (**d**) Similar structures to (**b**) using eBDG. (**e**) 5% random noise added to (**a**). (**f**) Same selection. (**g**) Similar structures to (**f**) using merging histogram. (**h**) Similar structures to (**f**) using eBDG.

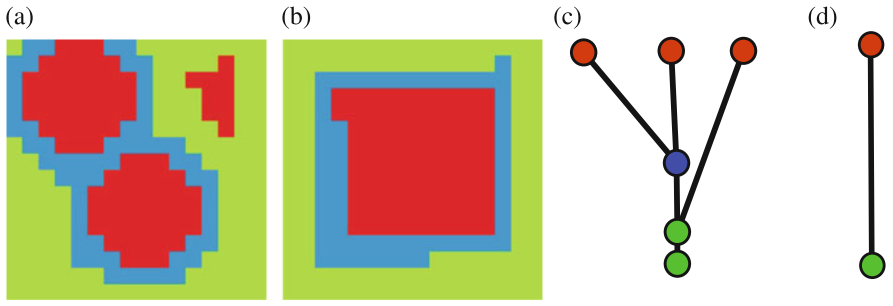


Fig. 10 An example of false positives in the histogram approach. Two simple yet topologically different datasets were designed to have the exactly same pixel distribution (*Red* = 83, *Blue* = 57 and *Green* = 116) and hence identical histograms. (**c**) and (**d**) show their corresponding join trees. As can be seen, a tree overlay method will be able to find the differences between these datasets but the histogram method will not. (**a**) Simple dataset A. (**b**) Simple dataset B. (**c**) Join tree for A. (**d**) Join tree for B

of 100 is a good estimate for most cases, and using a higher number does not alter the results much. In our implementation, we do not impose a necessary condition on the bin number. Sometimes it may so happen that two non-overlapping iso-valued regions are assigned to the same bin due to the resolution being too low. This issue can be addressed in future work. Another alternative binning strategy would be to assign more bins to more dense parts of the dataset and vice versa, thereby choosing a non-linear binning mechanism based on the intensity distribution of the entire dataset.

The current implementation considers applications which require finding similar regions at *nearly the same* iso-range. For applications which require finding similar structures at different iso-ranges, the method can be modified to finding the best overlap between histograms which minimizes the distance between them. This can be achieved by using standard dynamic programming techniques such as the Earth Mover's Distance [11].

7 Conclusion

Interactive similarity search in scalar fields using merge trees present a lot of useful possibilities like real-time exploration of the data, and reveal interesting patterns in volume renderings that are otherwise hard to see. Finding such patterns involve the general idea of finding similar subtrees to the corresponding subtree of the underlying pattern. Instead of trying to compare these subtrees, we focused on comparing feature descriptors of the subtree regions themselves and showed that a faster method can be used to achieve similar results.

We presented merging histograms—a feature descriptor defined for all subtrees of a merge tree, which was shown to be easily computed on-the-fly as part of the merge tree computation step. We presented a few simple modifications to the merge tree computation algorithm to achieve this. The comparison was shown to be quite discriminative and robust to small perturbations.

Computing these self-similarities between all pairs of subtree regions very quickly, provides for a rich interactive possibility.

A direction for future work could be to display self-similar structures at various iso-levels automatically without any user intervention.

References

1. Beketayev, K., Yeliussizov, D., Morozov, D., Weber, G.H., Hamann, B.: Measuring the distance between merge trees. In: Topological Methods in Data Analysis and Visualization III, pp. 151–165. Springer, Berlin (2014)
2. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. In: Proceedings ACM-SIAM Symposium on Discrete Algorithms, SODA '00, pp. 918–926. Society for Industrial and Applied Mathematics, Philadelphia (2000)
3. Carr, H., Snoeyink, J., van de Panne, M.: Simplifying flexible isosurfaces using local geometric measures. In: Proceedings IEEE Visualization, pp. 497–504. IEEE Computer Society (2004)
4. Carr, H., Brian, D., Brian, D.: On histograms and isosurface statistics. IEEE Trans. Vis. Comput. Graph. **12**(5), 1259–1266 (2006)
5. Correa, C., Lindstrom, P., Bremer, P.T.: Topological spines: a structure-preserving visual representation of scalar fields. IEEE Trans. Vis. Comput. Graph. **17**(12), 1842–1851 (2011)
6. Doraiswamy, H., Natarajan, V.: Efficient algorithms for computing Reeb graphs. Comput. Geom. **42**(6–7), 606–616 (2009)

7. Günther, D., Reininghaus, J., Seidel, H.P., Weinkauf, T.: Notes on the simplification of the Morse-Smale complex. In: Topological Methods in Data Analysis and Visualization III, pp. 135–150. Springer, Berlin (2014)
8. Ioannidis, Y.: The history of histograms (abridged). In: Proceedings International Conference on Very Large Data Bases (VLDB), vol. 29, pp. 19–30. VLDB Endowment (2003)
9. Kindlmann, G.L.: Semi-automatic generation of transfer functions for direct volume rendering. In: Proceedings IEEE Symposium on Volume Visualization, pp. 79–86 (1998)
10. Lundström, C., Ljung, P., Ynnerman, A.: Local histograms for design of transfer functions in direct volume rendering. IEEE Trans. Vis. Comput. Graph. **12**(6), 1570–1579 (2006)
11. Rubner, Y., Tomasi, C., Guibas, L.J.: A metric for distributions with applications to image databases. In: Proceedings International Conference on Computer Vision, pp. 59–66. IEEE (1998)
12. Saikia, H., Seidel, H.P., Weinkauf, T.: Extended branch decomposition graphs: structural comparison of scalar data. Comput. Graph. Forum **33**(3), 41–50 (2014)
13. Scheidegger, C.E., Schreiner, J.M., Duffy, B., Carr, H., Silva, C.T.: Revisiting histograms and isosurface statistics. IEEE Trans. Vis. Comput. Graph. **14**(6), 1659–1666 (2008)
14. Sereda, P., Vilanova Bartroli, A., Serlie, I.W.O., Gerritsen, F.A.: Visualization of boundaries in volumetric data sets using LH histograms. IEEE Trans. Vis. Comput. Graph. **12**, 208–218 (2006)
15. Tarasov, S.P., Vyalyi, M.N.: Construction of contour trees in 3D in O(n log n) steps. In: Proceedings Annual Symposium on Computational Geometry, SCG, pp. 68–75. ACM, New York (1998)
16. Thomas, D.M., Natarajan, V.: Symmetry in scalar field topology. IEEE Trans. Vis. Comput. Graph. **17**(12), 2035–2044 (2011)
17. Thomas, D.M., Natarajan, V.: Detecting symmetry in scalar fields using augmented extremum graphs. IEEE Trans. Vis. Comput. Graph. **19**(12), 2663–2672 (2013)
18. Thomas, D., Natarajan, V.: Multiscale symmetry detection in scalar fields by clustering contours. IEEE Trans. Vis. Comput. Graph. **20**(12), 2427–2436 (2014)
19. Tung, T., Schmitt, F.: Augmented Reeb graphs for content-based retrieval of 3D mesh models. In: Proceedings Shape Modeling Applications, pp. 157–166. IEEE (2004)
20. Younesy, H., Möller, T., Carr, H.: Visualization of time-varying volumetric data using differential time-histogram table. In: Proceedings International Workshop on Volume Graphics, pp. 21–224. IEEE (2005)

Morse-Smale Analysis of Ion Diffusion in Ab Initio Battery Materials Simulations

Attila Gyulassy, Aaron Knoll, Kah Chun Lau, Bei Wang, Peer-Timo Bremer, Michael E. Papka, Larry A. Curtiss, and Valerio Pascucci

Abstract Ab initio molecular dynamics (AIMD) simulations are increasingly useful in modeling, optimizing and synthesizing materials in energy sciences. In solving Schrödinger's equation, they generate the electronic structure of the simulated atoms as a scalar field. However, methods for analyzing these volume data are not yet common in molecular visualization. The Morse-Smale complex is a proven, versatile tool for topological analysis of scalar fields. In this paper, we apply the discrete Morse-Smale complex to analysis of first-principles battery materials simulations. We consider a carbon nanosphere structure used in battery materials research, and employ Morse-Smale decomposition to determine the possible lithium ion diffusion paths within that structure. Our approach is novel in that it uses the wavefunction itself as opposed distance fields, and that we analyze the 1-skeleton of the Morse-Smale complex to reconstruct our diffusion paths. Furthermore, it is the first application where specific motifs in the graph structure of the complete 1-skeleton define features, namely carbon rings with specific valence. We compare our analysis of DFT data with that of a distance field approximation, and discuss implications on larger classical molecular dynamics simulations.

1 Introduction

First principles (ab initio) simulations of molecular structures, employing density functional theory (DFT) or Hartree-Fock (HF) methods, are increasingly common in materials science applications. Unlike classical dynamics simulations, they solve

A. Gyulassy (✉) • A. Knoll • B. Wang • V. Pascucci
SCI Institute, University of Utah, Salt Lake City, UT, USA
e-mail: jediati@sci.utah.edu; knolla@sci.utah.edu; beiwang@sci.utah.edu; pascucci@sci.utah.edu

K.C. Lau • M.E. Papka • L.A. Curtiss
Argonne National Laboratory, Argonne, IL, USA
e-mail: kclau@anl.gov; papka@anl.gov; curtiss@anl.gov

P.-T. Bremer
Lawrence Livermore National Laboratory, Livermore, CA, USA
e-mail: bremer5@llnl.gov

the Schrödinger equation to more accurately determine molecular geometry. They compute the wavefunction, the electronic structure of the molecules they simulate, which consists of scalar fields of component molecular orbitals. Most analysis of such simulations is carried out on the resulting molecular geometry as opposed to the wavefunction. However, the wavefunction offers advantages: it is the closest we can come to "ground truth" concerning the structure of the molecule, and its volumetric representation can be used for topological analysis of scalar fields.

We consider an application in materials science: determining the charge capacity of simulated battery anode structures. The material in question is a sphere of carbon sculpted from a solid block of graphite, heated to a high temperature (2500 Kelvin) via molecular dynamics, and then annealed. The resulting "nanosphere" resembles ordinary graphitic soot, but possesses numerous channels that can accommodate lithium ion electrolyte. These channels form as a result of defects in the graphite structure, which in turn change the coordination number (valence) of their component carbon atoms. Higher average coordination number indicates more defect sites, and structures that are better-suited as battery anodes. While effective, simple statistical analysis does not fully quantify the charge capacity of these structures. To do that, we must understand the paths that lithium ions may diffuse through inside the carbon nanosphere structure.

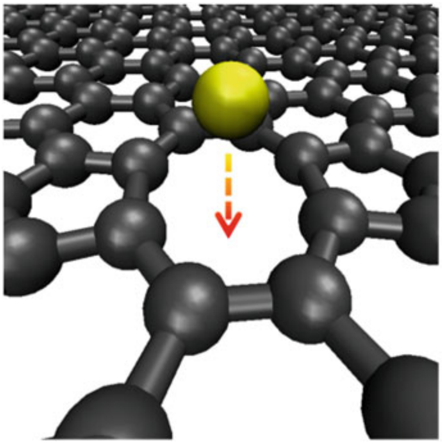
The goal of this work is to use Morse-Smale analysis to tackle this problem, analyzing the scalar field of the wavefunction itself from DFT computation. We compute a 1-skeleton of the Morse-Smale complex to determine likely ion diffusion paths in the nanosphere, identifying defect sites in the carbon structure through which lithium ions may pass. We present a model for these features, and examine statistical properties of the topology to establish a methodology to analyze such data. We apply this technique to extract and compare results from the DFT-computed wavefunction to those of a distance field.

2 Background

Our scientific goal is to examine the structure of carbon nanospheres throughout the heating and annealing process, and analyze the suitability for the resulting structure as a battery anode material. To do this at a relatively small scale (hundreds of atoms), but ensure higher physical accuracy, we conduct an ab initio molecular dynamics (AIMD) simulation, specifically DFT computation using the VASP code [18].

In experiments, the monodispersed carbon nanospheres that are used for a sustainable lithium energy storage electrode can be synthesized effectively by autogenic reactions of hydrocarbon precursors (e.g. polyethylene from plastic waste, etc.) at high temperature and pressure, enabling synthesis of a battery anode from recycled materials [29]. From the reported studies, the unique carbon microstructures (i.e. layered graphitic motifs and sufficient carbon defects) are critical in promoting lithium diffusion into and out from the interior of carbon nanospheres for a practical lithium ion battery operations (i.e. capacity, voltages and charging rates). In order to understand how Li ion can be diffused into or from (or intercalated into or de-intercalated from) carbon microstructures, the Li ion diffusion energy barriers along the diffusion paths within the carbon microstructures of an electrode have to be determined at atomistic level. To accurately quantify the Li ion diffusion dynamics, DFT nevertheless remains the preferred choice in theoretical descriptions at the atomistic level. In addition to computing physically accurate atom geometry, solving the Schrödinger equation explicitly generates the electronic structure (electron or charge density cloud), volume data on which we can apply topological analysis.

From the reported studies [29], the interlayer Li diffusion paths are most probably determined by the presence of large n-membered ring (e.g. $n > 6$) defect sites due to the extensive thermal graphitization and significant carbon dislocations during the high temperature synthesis of carbon. To model the anticipated Li ion diffusion energy barrier through the n-membered rings, the atomistic simulation is carried out based on DFT calculations with plane wave basis sets as implemented in the VASP code [18]. All the DFT calculations were spin-polarized and carried out using the gradient corrected exchange-correlation functional of Perdew, Burke and Ernzerhof (PBE) [26] under the projector augmented wave (PAW) method, with plane wave basis sets up to a kinetic energy cutoff of 400 eV. For a Li-ion diffusion barrier, as the size of the n-membered ring increases, the barrier for a Li ion to diffuse through decreases significantly, as confirmed by the DFT study in this work (Fig. 1) and previous studies [35]. For the larger 9-membered ring, the Li diffusion energy barrier is even smaller, i.e. 0.15 eV, slightly smaller than the reported value (i.e. 0.5 eV) in carbon nanotubes [24].



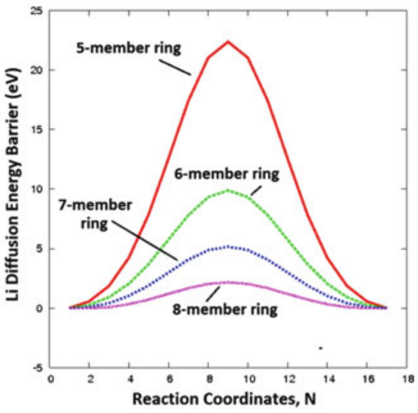


Fig. 1 *Left:* the symmetric feature of the energy barriers are due to the Li ion diffusion into and out of the center of n-membered ring at a single graphene sheet. *Right:* the Li ion diffusion energy barrier (in eV) obtained from DFT calculation

In addition to large n-membered rings of carbon defects, the Li diffusion is also facilitated through the intra-layer in-plane diffusion within the layered graphite present in carbon nanospheres [19]. From the computed free energy surface explored by our CPMD metadynamics simulation, we found the Li in-plane diffusion barrier at arbitrary direction in between the graphitic layers is comparatively small (i.e. 8 kcal/mol = 0.35 eV), which is consistent with the reported values (i.e. 0.26–0.50 eV) [30]. Thus, it is reasonable to assume that the diffusion dynamics of Li ions are driven by inter-layer diffusion through n-membered rings (referred to as *defects* in the perfect C-6 carbon ring structure) as well as intra-layer movement.

We can pair these insights with topological analysis to identify Li-accessible tunnels inside the nanosphere, The approach of our work is to use the Morse-Smale complex to define an initial skeleton of minima and 1-saddles, count the size of carbon rings (number of maxima) adjacent to that skeleton to determine which tunnels an ion would likely pass through, and thus extract the diffusion path skeleton. Moreover, the 1-skeleton represents both inter and intra-layer diffusion paths, making it well-suited for the analysis of this particular structure.

2.1 Morse-Smale Complex

The following provides a brief introduction to the Morse-Smale complex, which we use to identify features in the DFT data, and topological simplification, used to study the function at multiple scales and reason about the stability of the identified features.

Morse Functions and the Morse-Smale (MS) Complex Let f be a real-valued smooth map $f : \mathbb{M} \rightarrow \mathbb{R}$ defined over a compact d-manifold $\mathbb{M}$. A point $p \in \mathbb{M}$ is critical when $|\nabla f(p)| = 0$, i.e. the gradient is zero, and is non-degenerate when its Hessian (matrix of second partial derivatives) is non-singular. The function f is a *Morse function* if all its critical points are non-degenerate and no two critical points have the same function value. In this case the *Morse Lemma* states that there exists local coordinates around p such that f has the following *standard form*: $f_p = \pm x_1^2 \pm x_2^2 \cdots \pm x_d^2$. The number of minus signs in this equation gives the *index* of critical point p. In three-dimensional functions, minima are index-0, 1-saddles are index-1, 2-saddles are index-2, and maxima are index-3.

An integral line in f is a path in $\mathbb{M}$ whose tangent vector agrees with the gradient of f at each point along the path. The integral line passing through a point p is the solution to $\frac{\partial}{\partial t}L(t) = \nabla f(L(t)), \forall t \in \mathbb{R}$, with initial value $L(0) = p$. Each integral line has an origin and destination at critical points of f, at $t = \pm\infty$. *Ascending* and *descending* manifolds are obtained as clusters of integral lines having common origin and destination respectively. The descending manifolds of f form a cell complex that partitions $\mathbb{M}$; this partition is called the *Morse* complex. Similarly, the ascending manifolds also partition $\mathbb{M}$ in a cell complex. A Morse function f is a *Morse-Smale function* if ascending and descending manifolds of its critical points only intersect transversally. An index-i critical point has an i-dimensional descending manifold and a $(d-i)$-dimensional ascending manifold. The simply-connected cells formed by the intersections of ascending and descending manifolds form the cells of the Morse-Smale (MS) complex. A three-dimensional MS complex is a cell complex where cells of dimension zero through three are called nodes, arcs, quads, and crystals, respectively. Each arc is a 1-manifold bounded by two nodes, 0-manifolds, each quad is a 2-manifold bounded by arcs, and finally, each crystal of the MS complex is bounded by quads. Cells of the MS complex satisfy several combinatorial properties: end points of arcs are critical points whose indices differ exactly by one; quads contain exactly four arcs on their boundary (although some might be repeated); and the closure of the boundary of a crystal contains a collection of quads, arcs, saddles and exactly one minimum and one maximum. The *1-skeleton* of the MS complex is formed by the nodes and arcs, representing much of the connectivity information of the complex.

Topological Simplification A function f is simplified by repeated cancellation of pairs of critical points connected by an arc in the MS complex. The local change in the MS complex indicates a smoothing of the gradient vector field and hence of the function f. Forman [9] showed how a cancellation could be achieved in a discrete gradient field by reversing the gradient path between two critical cells. Gyulassy et al. [12] provided a full characterization of cancellation operations in terms of how they affect the connectivity of the complex and the geometry of the ascending/descending manifolds, operating solely on the combinatorial structure of the complex. Each cancellation operation removes a pair of critical points, reconnects arcs of the complex, and merges their ascending and descending manifolds with their neighbors geometry. Repeated application of cancellations in order of *persistence*, the absolute difference in function value of the canceled critical points, results in a hierarchy of MS complexes and a multi-resolution representation of features. Gyulassy et al. [14] described data structures and search algorithms to reconstruct the ascending and descending manifolds of any critical point at any stage of simplification, allowing rapid browsing of the extracted features at multiple scales.

3 Related Work

We review works that are most relevant to our proposed techniques. Most of the geometric and topological methods discussed here originate from molecular shape analysis, in particular, in the detection of protein cavities. A cavity is an empty space enclosed by the molecule, and it includes voids (without openings that allow access to the surrounding solvent), pockets and tunnels (with openings). A tunnel connects multiple surface sites through pathways; while a pocket connects a site in the interior with a surface site. We refer the reader to Lidow et al. [22, 23] for illustrations of these scenarios. In our case diffusion paths correspond to the pockets and tunnels of a given molecule. Most of the geometry-based cavity detection algorithms rely on the computation of Voronoi diagram, its dual graph Delaunay triangulation, weighted Delaunay triangulation or its close relative alpha shapes. Because of the large number of work in the field, we include a few significant ones.

Voronoi diagram-based techniques that have a special focus on the analysis and visualization of tunnels include works in [22, 23], tools MOLE [28] and CAVER [27], where the work in [23] detects structures from a molecular dynamics trajectory.

Alpha shapes are closely related to alpha complexes which are subcomplexes of the Delaunay triangulation of the point set. Alpha shape theory [6, 8] has been used for the detection of protein cavities [7, 20, 21] and has been featured in tools such as CAST [5, 20], Proshape [17], CAVER [27] and MolAxis [34]. A related concept that is similar to Alpha shape, but better in terms of remaining connected for all resolutions, called Beta shape [16], has been proposed to define cavities [15] in molecules.

A few works exist that employ Morse complex or Morse-Smale complex in shape analysis. Sousbie used Morse complexes to compute filamentary structures in of the cosmic web [31]. Topological spines [4], used as visual representations that preserve the topological and geometric structure of a scalar field, have been developed based on the extraction of sparse subsets of the Morse-Smale complex. The works in [3, 25] decompose the protein surfaces into segmented features for the analysis of protein-protein interactions. However these works are restricted to study surface geometry of the proteins, while our proposed technique focuses on both surface and interior structures. On the other hand, Bajaj et. al. [2] model structural features of molecules by computing stable and unstable manifolds of the critical points of the distance function induced by the iso-surface. Their work is most relevant to our algorithm in a sense that both algorithms rely on properties associated with stable and unstable manifolds of critical points of distance functions to the surface of the molecule; and both detect pockets and tunnels simultaneously. The work in [2] forms pockets and tunnels by clustering and merging adjacent stable manifolds of critical points based on their scalar value; such an approach is not applicable to diffusion in carbon nanospheres, since, as we show in Sect. 6, there is no scalar threshold (either of the DFT field or distance field) that distinguishes between ring structures of different valences. The full MS complex has been used to identify atomic structures on volume data, identifying the atoms and bonds in a C4H4 molecule and orbitals of a hydrogen atom [11]. Günther et al. [10] refined topological analysis of electron density, by using a derived gradient for identification of both covalent and non-covalent bonds.

In the chemistry literature, relatively little analysis is carried out on the wave-function field itself, as opposed to atom geometry. Bader analysis [1] decomposes charge density into regions of uniform gradient each associated with one atom, for example using Voronoi partitioning. It is similar to Morse theory in that it uses gradient descent to partition the scalar field, but would not help in identifying tunnels between 1-saddles and minima.

Our proposed technique uses persistence simplification to separate noise from features, and similar simplification algorithms have been employed to simplify surface features based on MSCs [3, 4, 25, 33] or alpha shapes [32].

4 Li Diffusion Paths in DFT Data

In this section, we first discuss our approach to handling DFT data, then present the model for identifying features in such data, defining features according to the structure of the MS complex. Next, we present justification and validation for the model, analysing the stability of the extracted features. Finally, we discuss details of the methodology, corner cases, and remaining ambiguities.

4.1 Preparation of DFT Data

DFT simulations employ a variety of different functionals and strategies to maximize accuracy and reduce computation time. As interior orbitals (1s, 2s) typically remain static throughout a simulation, it is common to ignore their computation entirely. A side-effect of this optimization is that electron density is ill-defined at the nuclei: whereas the true wavefunction (e.g., obtained through x-ray crystallography or computation of all orbitals) would exhibit maxima at the atomic nuclei, our data instead show empty space (minima) at the nuclei.

As our analysis relies entirely on pre-existing connections within the MS complex to define the skeleton, it is challenging to correct this phenomenon *post facto*. We found it simplest to modify the scalar field itself to restore maxima at the carbon nuclei. To do this, we created a field of summed radial distance functions $G = \sum g_i$, $g_i(d) = Z\, exp(-(d/r)^2)$, where Z=6 is the atomic number (maximum possible electron density) and r = 0.36 Ångstrom is half the covalent radius of sp^2 carbon (graphite). Then, given our DFT all-electron density field F, our augmented field is $F' = sup\{F, G\}$. This allows us to correctly identify maxima at the nuclei in the MS complex, and is shown in Fig. 2.

We use standard techniques from discrete Morse theory to compute an initial MS complex [13]. Our analysis entails every component of the MS complex taken together, necessitating coherent simplification to remove low-persistence features and artifacts due to discretizing a function onto a mesh. One challenge in using DFT data is the exponential nature of the electron density as a function of distance from an atom. For instance, the features of the 732-atom nanosphere dataset we use in our results span 9 orders of magnitude; the difference between minima and 1-saddles, and 2-saddles and maxima occurs on the scales of 10^{-4} and 10^2, respectively. To obtain a coherent view of the structure, we apply a straightforward strategy of computing the MS complex on the DFT field, but then rescaling critical points using the *log* function before applying persistence simplification.

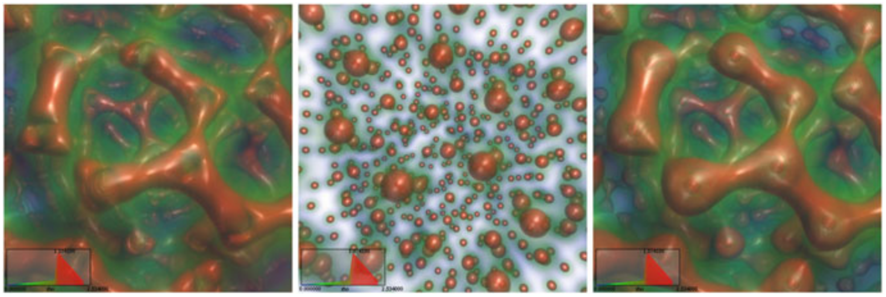


Fig. 2 Modification of DFT data to restore maxima at the nuclei. *Left:* original DFT data F. *Center:* RDF impulses G. *Right:* augmented $F' = sup\{F, G\}$

4.2 Theoretical Model for Analyzing Electron Density

We follow a standard model relating the critical points of electron density to various features of the carbon nanosphere. For each of the following features, we identify the component of the MS complex that corresponds to it, and illustrate the features in Fig. 3.

Carbon atoms: It is well-understood in chemistry that the electron density reaches a maximum at atom locations, and in our model, we use maxima of the electron density field as a proxy for carbon atoms.

Covalent bonds: The covalent bonds between carbon atoms also appear as relatively high values, and typically a high-valued 2-saddle connecting two maxima in the 1-skeleton of the MS complex indicates a bond.

Voids: Voids are minima of the electron density field, and represent locally minimal energy configurations. In our model we identify voids as low-valued minima in the MS complex.

Free Li diffusion path: The intra-layer in-plane regions are characterized by regions where Li ions have a comparatively small energy barrier to diffusion, corresponding to low electron density values. We model this as the network connecting voids in the 1-skeleton corresponding to very low electron densities.

N-member rings: The bonds and atoms in a Carbon ring form a circular ridge-line in space, that forms the boundary of a disk. The disk itself separates low-valued voids, and corresponds to the ascending 2-manifold emanating from a 1-saddle in the MS complex. To account for possible pinching of the boundary of a disk, we compute the valence of a ring as the number of carbon atoms in the largest simple cycle on the boundary.

Defect sites: Defect sites, allowing Li to diffuse across layers, are a carbon rings with valence > 6, and are identified as 1-saddles that are not part of free Li diffusion paths and are also connected to > 6 bond 2-saddles in the MS complex.

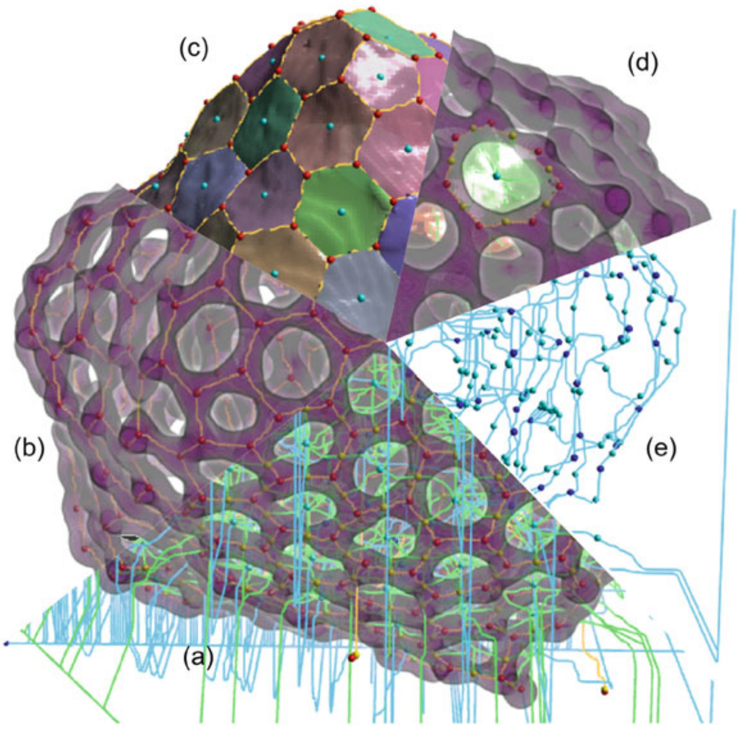


Fig. 3 The full MS complex (a) of the 732-atom nanosphere provides the structure for identifying features. Carbon atoms (*red spheres*) and covalent bonds (*yellow arcs*) are identified as maxima, high-valued 2-saddles, and the arcs between them (b). Rings are the boundary of ascending 2-manifolds emanating from 1-saddles (c), and rings with valence > 6 are identified as defect sites (d). Free Li ion diffusion paths (e) are the 1-skeleton connecting voids (*blue spheres*) with low density value

4.3 Model Validation: Feature Stability

The model presented in Sect. 4.2 is missing some key information needed for practical feature extraction: the MS complex must first be simplified to a user-supplied threshold, a second threshold is needed to identify the 2-saddles corresponding to covalent bonds, and finally, a third threshold determines free Li ion diffusion paths. Ideally, one can arrive at stable thresholds for each of these without a priori knowledge, by simply examining the distribution and persistence of features of the computed MS complex. The goal is to be able to identify such features and then relate them back to what we know about the data, as a strong validation of the approach. In the following, we describe this methodology as it is applied to the 732-atom DFT data.

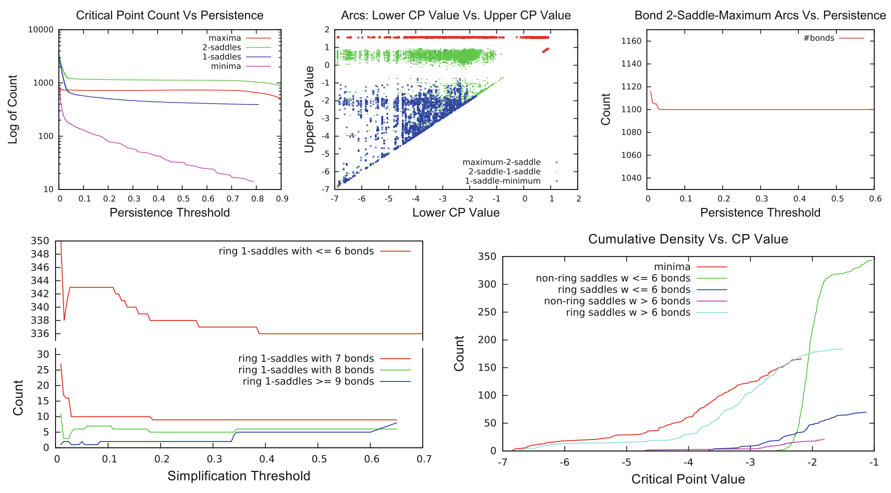


Fig. 4 Analysis of the feature set encoded by the MS complex leads to insights about the location and stability of features. Counting critical points as a function of persistence simplification threshold (**a**) shows that carbon atoms can be extracted in a stable manner. Plotting each arc of the complex with a point whose coordinates are the values of the critical points (**b**) reveals that the 2-saddle-maximum connections corresponding to covalent bonds are well separated from other features, and furthermore stable with respect to simplification (**c**). Counting the number of n-valence rings reveals that barrier 1-saddles are stable in a smaller range of simplification (**d**). Finally, we identify a conservative estimate for threshold free diffusion paths by identifying flatter regions in the cumulative density function of critical points (**e**) while avoiding any barrier rings

We begin with identification of Carbon atoms; in our model, they are maxima of the MS complex. Figure 4a shows the count of critical points of the MS complex as a function of persistence simplification threshold. A large stable threshold is identified in the range [0.03:0.6]. In our example data, we find exactly 732 high-valued maxima in this threshold range, corresponding exactly to the carbon atoms in the atomistic simulation. For simplification thresholds less than 0.03, low-valued maxima appear on some of the covalent bonds, and for thresholds greater than 0.6, high-valued maxima begin to merge together.

Next, we validate our model for high-valued 2-saddles representing covalent bonds by finding a stable threshold that distinguishes between 2-saddle-maximum arcs of the complex. In Fig. 4b, we show each arc of the MS complex as a point, with coordinates given by the lower and upper critical points of the arc. A clear structure is apparent from this figure, notably the separation of high-valued 2-saddle-maximum arcs from the other arcs in the dataset. Selecting the 2-saddle-maximum arcs in the box 2-saddle=[-0.05:2.0] provides a stable threshold, that corresponds to the covalent bonds in the data. This is furthermore shown in Fig. 4c to be stable for the same simplification threshold range as the carbon atoms, yielding exactly 1110 covalent bonds in these threshold ranges.

The rings that do not allow Li ions to diffuse, those with valence ≤ 6, can be identified by the count of 2-saddles corresponding to covalent bonds. Intuitively, these are given by 1-saddles at some simplification threshold. The count of barrier rings as a function of persistence is shown in Fig. 4d. This plot indicates that a smaller stable threshold, as 1-saddles are lost after a persistence simplification threshold of 0.11.

So far, we have established that carbon atoms, covalent bonds, and barrier rings can be extracted in a very stable manner, with simplification threshold of 0.06 ± 0.03. This stability aligns with expectation, due to the regular structure imposed by the chemistry and physics of the annealed nanosphere. We make the logical step that any 2-saddles that are *not* covalent bonds must be part of an intra-layer region, where Li ions can diffuse relatively easily. Furthermore, since the ascending 2-manifolds of 1-saddles form topological disks that fill in the interior of covalently bonded carbon rings, any 1-saddle adjacent to a non-bond 2-saddle (and any minimum attached to that) must also be part of the free diffusion region. Finally, we consider the case where based on the electron density value alone we can determine if the Li ions may diffuse. In Fig. 4e we show cumulative density functions of minima and 1-saddles, categorized by whether they are ring, non-ring, defect, or non-defect points. To pick the threshold that guarantees that Li may diffuse freely, we must pick a threshold below the lowest ring saddle values with ≤ 6 bonds, which occurs at value -3.7. The most permissive, yet relatively stable threshold occurs around -4.

5 Comparison with Distance Field

Production-scale nanosphere simulations are expected to produce results on the order of one million atoms, making DFT data impossibly expensive to compute. One alternative we explore in this section is to use the atom positions to create a distance field. In particular, we generate a signed distance field, that is the negative of the minimum distance from a grid vertex to a Carbon atom, preserving the convention of identifying carbon atoms as maxima. The same data-driven approach is applied from Sect. 4.3 to obtain stable threshold of 0.1 for persistence simplification, -0.9 for identification of covalent bonds, and -5.0 to identify free paths. Figure 5 shows a direct comparison of features extracted from the DFT data and the distance field. There is a one-to-one correspondence between Carbon atoms and covalent bonds, between the two fields (a, e). The first differences appear in the identification of ring saddles (b, f), where certain 1-saddles appear as free paths in the DFT field, and instead appear as defect sites in the distance field. This difference can be noted in (c, g), where a defect present in the distance field (g) is instead identified as a free path connecting the exterior to the interior of the nanosphere (d). In both fields, every void in the interior of the nanosphere is completely connected by free diffusion paths.

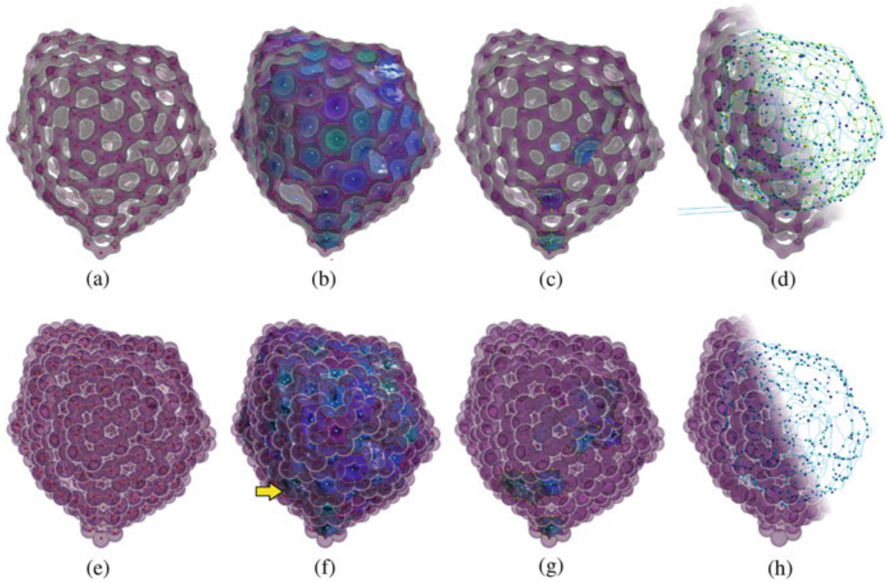
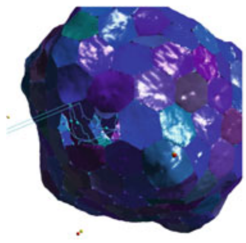
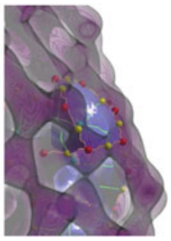
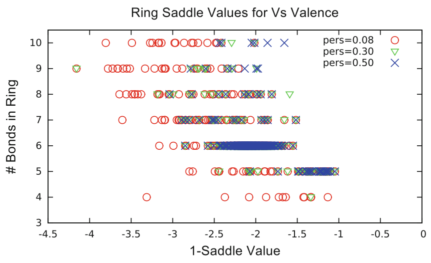


Fig. 5 A visual comparison of features of the DFT data (*top row*) with features of the distance field (*bottom row*). The exact same set of carbon atoms and bonds are found in each (**a**, **e**). While the majority of rings are the same, the DFT data classifies some defect rings as intra-layer free diffusion paths. For instance (**f**) displays a ring (*yellow arrow*) not present in (**b**). This difference can be accounted for with (**c**, **d**, **g**, **h**), where (**g**) shows a defect detected in the distance field, whereas (**d**) shows the same site as part of an intra-layer free path. Other defects sites match (**c**, **g**), and the free paths in both (**d**) and (**g**) completely connect every void in the interior.

6 Discussions

We apply our data-driven approach to the 732-atom DFT field, and found a stable persistence simplification threshold of 0.06, a stable selection threshold for bonds of > -0.05, and a conservative estimate of free paths with value lower than -4. With these thresholds, we found that the entire interior of the 732-atom DFT field is accessible to Li ion diffusion. Figure 6a illustrates that according to the definitions we presented the free paths directly connect the exterior of the nanosphere with every layer. Figure 6b demonstrates a scenario where the ascending 2-manifold of a 1-saddle does not have a simple circle as its boundary. To avoid incorrectly classifying this as a fault, we perform an additional check on the ring, only counting the atoms that form a simple cycle. Such configurations arise when carbon atoms are bonded in the third dimension outside of the plane of the carbon ring. Finally, we justify using the ring valence as opposed to selecting a threshold for diffusion (as was done in [2]). Figure 6c shows the function values for various valence rings. Even for large simplification thresholds, rings identified as valence 5 or 6 may have *lower* function value than higher valence rings.

(a) (b) (c)

Fig. 6 In the DFT field, free paths were identified between all layers of the nanosphere (**a**). A corner case illustrates the need to compute the length of simple cycles when counting the valence of a ring (**b**). The distribution of values for rings of varying valence shows that there are no scalar thresholds that separate features (**c**)

7 Conclusion/Future Work

We have presented a new technique for analyzing Li ion diffusion paths in carbon nanospheres that utilizes identifying motifs (rings) in the 1-skeleton of the MS complex. We have further shown that feature analysis is stable, and appropriate thresholds can be identified from the topology of the data itself without a priori knowledge. Finally, we compared the analysis of the DFT data to a distance field, showing similarities, but also differences in the classification of free path vs. defect sites. Further study is needed to understand the implications on qualitative measurements of Li storage capacity of nanospheres. We plan to use this methodology to understand the energy storage properties of large-scale nanospheres.

Acknowledgements This research was supported as part of the Center for Electrical Energy Storage (CEES), an Energy Frontier Research Center (EFRC) and Argonne Leadership Computing Facility under the U.S. Department of Energy, Office of Science, Office of Basic Energy (Award Number DE-AC02-06CH11357). This work was also supported in part by the NSF CISE ACI-0904631 at the University of Utah.

References

1. Bader, R.F.: Atoms in Molecules. Wiley, New York (1990)
2. Bajaj, C., Gillette, A., Goswami, S.: Topology based selection and curation of level sets. In: Hege, H.C., Polthier, K., Scheuermann, G. (eds.) Topology-Based Methods in Visualization II, Mathematics and Visualization, pp. 45–58. Springer, Heidelberg (2009)
3. Cazals, F., Chazal, F., Lewiner, T.: Molecular shape analysis based upon the Morse-Smale complex and the Connolly function. In: Proceedings of 19th Annual Symposium on Computational Geometry, pp. 351–360 (2003)
4. Correa, C.D., Lindstrom, P., Bremer, P.T.: Topological spines: A structure-preserving visual representation of scalar fields. IEEE Trans. Vis. Comput. Graph. **17**(12), 1842–1851 (2011)

5. Dundas, J., Ouyang, Z., Tseng, J., Binkowski, A., Turpaz, Y., Liang, J.: CASTp: computed atlas of surface topography of proteins with structural and topographical mapping of functionally annotated residues. Nucleic Acids Res. **34**(2), W116–W118 (2006)
6. Edelsbrunner, H., Fu, P.: Measuring space filling diagrams and voids. Technical Report UIUC-BI-MB-94-01, University of Illinois at Urbana-Champaign (1994)
7. Edelsbrunner, H., Koehl, P.: The geometry of biomolecular solvation. Comb. Comput. Geom. **52**, 243–275 (2005)
8. Edelsbrunner, H., Mucke, E.P.: Three-dimensional alpha shapes. ACM Trans. Graph. **13**(1), 43–72 (1994)
9. Forman, R.: A user's guide to discrete Morse theory. Séminare Lotharinen de Combinatore, vol. 48 (2002)
10. Günther, D., Boto, R.A., Contreras-Garcia, J., Piquemal, J.P., Tierny, J.: Characterizing molecular interactions in chemical systems. IEEE Trans. Vis. Comput. Graph. **20**(12), 2476–2485 (2014)
11. Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P.T., Hamann, B.: Topology-based simplification for feature extraction from 3D scalar fields. In: Proceedings of IEEE Visualization 2005 (VIS'05), pp. 535–542. IEEE, New York (2005)
12. Gyulassy, A., Bremer, P.T., Pascucci, V., Hamann, B.: A practical approach to Morse-Smale complex computation: scalability and generality. IEEE Trans. Vis. Comput. Graph. **14**(6), 1619–1626 (2008)
13. Gyulassy, A., Bremer, P.T., Pascucci, V.: Computing Morse-Smale complexes with accurate geometry. IEEE Trans. Vis. Comput. Graph. **18**, 2014–2022 (2012)
14. Gyulassy, A., Kotava, N., Kim, M., Hansen, C.D., Hagen, H., Pascucci, V.: Direct feature visualization using Morse-Smale complexes. IEEE Trans. Vis. Comput. Graph. **18**(9), 1549–1562 (2012)
15. Kim, D.S., Sugihara, K.: Tunnels and voids in molecules via voronoi diagram. In: 2012 Ninth International Symposium on Proceedings of Voronoi Diagrams in Science and Engineering (ISVD), pp. 138–143. IEEE, New York (2012)
16. Kim, D.S., Cho, Y., Sugihara, K., Ryu, J., Kim, D.: Three-dimensional beta-shapes and beta-complexes via quasi-triangulation. Comput. Aided Des. **42**(10), 911–929 (2010)
17. Koehl, P., Levitt, M., Edelsbrunner, H.: Proshape: understanding the shape of protein structures. Software at biogeometry. duke.edu/software/proshape (2004)
18. Kresse, G., Furthmüller, J.: Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. Comput. Mater. Sci. **6**(1), 15–50 (1996)
19. Lau, K.C., Lu, J., Low, J., Peng, D., Wu, H., Albishri, H.M., Al-Hady, D.A., Curtiss, L.A., Amine, K.: Investigation of the decomposition mechanism of lithium bis(oxalate)borate (LiBOB) salt in the electrolyte of an aprotic Li–O_2 battery. Energ. Technol. **2**(4), 348–354 (2014)
20. Liang, J., Edelsbrunner, H., Fu, P., P., S.H., Subramaniam, S.: Analytical shape computation of macromolecules: II. Inaccessible cavities in proteins. Proteins Struct. Funct. Genet. **33**(1), 18–29 (1998)
21. Liang, J., Woodward, C., Edelsbrunner, H.: Anatomy of protein pockets and cavities. Protein Sci. **7**(9), 1884–1897 (1998)
22. Lindow, N., Baum, D., Hege, H.: Voronoi-based extraction and visualization of molecular paths. IEEE Trans. Vis. Comput. Graph. **17**(12), 2025–2034 (2011)
23. Lindow, N., Baum, D., Bondar, A., Hege, H.: Dynamic channels in biomolecular systems: Path analysis and visualization. In: 2012 IEEE Symposium on Proceedings of Biological Data Visualization (BioVis), pp. 99–106. IEEE, New York (2012)
24. Meunier, V., Kephart, J., Roland, C., Bernholc, J.: Ab initio investigations of lithium diffusion in carbon nanotube systems. Phys. Rev. Lett. **88**(7), 075506 (2002)
25. Natarajan, V., Wang, Y., Bremer, P.T., Pascucci, V., Hamann, B.: Segmenting molecular surfaces. Comput. Aided Geom. Des. **23**(6), 495–509 (2006)
26. Perdew, J.P., Burke, K., Ernzerhof, M.: Generalized gradient approximation made simple. Phys. Rev. Lett. **77**(18), 3865 (1996)

27. Petřek, M., Otyepka, M., Banáš, P., Košinová, P., Koča, J., Damborský, J.: CAVER: a new tool to explore routes from protein clefts, pockets and cavities. BMC Bioinf. **7**, 316 (2006)
28. Petřek, M., Košinová, P., Koča, J., Otyepka, M.: MOLE: a voronoi diagram-based explorer of molecular channels, pores, and tunnels. Structure **15**(11), 1357–1363 (2007)
29. Pol, V.G., Wen, J., Lau, K.C., Callear, S., Bowron, D.T., Lin, C.K., Deshmukh, S.A., Sankaranarayanan, S., Curtiss, L.A., David, W.I., et al.: Probing the evolution and morphology of hard carbon spheres. Carbon **68**, 104–111 (2014)
30. Song, J., Ouyang, B., Medhekar, N.V.: Energetics and kinetics of Li intercalation in irradiated graphene scaffolds. ACS Appl. Mater. Interfaces **5**(24), 12968–12974 (2013)
31. Sousbie, T.: The persistent cosmic web and its filamentary structure-I. theory and implementation. Mon. Not. R. Astron. Soc. **414**(1), 350–383 (2011)
32. Sridharamurthy, R., Doraiswamy, H., Patel, S., Varadarajan, R., Natarajan, V.: Extraction of robust voids and pockets in proteins. In: Hlawitschka, M., Weinkauf, T. (eds.) EuroVis - Short Papers, pp. 67–71. The Eurographics Association (2013)
33. Weiss, K., Iuricich, F., Fellegara, R., De Floriani, L.: A primal/dual representation for discrete morse complexes on tetrahedral meshes. Comput. Graphics Forum **32**(3pt3), 361–370 (2013)
34. Yaffe, E., Fishelovitch, D., Wolfson, H.J., Halperin, D., Nussinov, R.: MolAxis: efficient and accurate identification of channels in macromolecules. Proteins **73**, 72–86 (2008)
35. Yao, F., Gunes, F., Ta, H.Q., Lee, S.M., Chae, S.J., Sheem, K.Y., Cojocaru, C.S., Xie, S.S., Lee, Y.H.: Diffusion mechanism of lithium ion through basal plane of layered graphene. J. Am. Chem. Soc. **134**(20), 8646–8654 (2012)

Piecewise Polynomial Reconstruction of Scalar Fields from Simplified Morse-Smale Complexes

Léo Allemand-Giorgis, Georges-Pierre Bonneau, and Stefanie Hahmann

Abstract Morse-Smale complexes have been proposed to visualize topological features of scalar fields defined on manifold domains. Herein, three main problems have been addressed in the past: (a) efficient computation of the initial combinatorial structure connecting the critical points; (b) simplification of these combinatorial structures; (c) reconstruction of a scalar field in accordance to the simplified Morse-Smale complex. The present paper faces the third problem by proposing a novel approach for computing a scalar field coherent with a given simplified MS complex that privileges the use of piecewise polynomial functions. Based on techniques borrowed from shape preserving design in Computer Aided Geometric Design, our method constructs the surface cell by cell using piecewise polynomial curves and surfaces. The benefit and limitations of using polynomials for reconstruction surfaces from topological data are presented in this paper.

1 Introduction

Morse-Smale (MS) complexes are combinatorial structures encoding topological features of smooth scalar fields by connecting their critical points. The discrete Morse theory [11] has been used to define MS complexes for scalar data defined on meshes.

Because of their ability to characterize the topology of level sets, MS complexes have found many applications in visualization [18]. Each cell in a MS complex is a set of integral lines of a discrete gradient flow (the equivalent of streamlines of a smooth gradient field) connecting a critical point with some value to another critical point with a higher value. Along an integral line, the data is monotonic, strictly increasing. Inside a 2d-cell, there are no critical points, and the level sets have the topology of an open interval. Previous works have their focus on the efficient computation of MS complexes [25, 26], on the simplification and hierarchical representation of MS complexes [2, 5] or on the extraction of features using MS complexes [18]. The simplification of a MS complex is a combinatorial operation

L. Allemand-Giorgis • G.-P. Bonneau • S. Hahmann (✉)
Université Grenoble Alpes, CNRS, LJK, INRIA, Grenoble, France
e-mail: name@inria.fr

consisting in deleting two adjacent 0d-cells in the complex. All adjacent cells to the deleted nodes affected by the simplification are either deleted or unified. Since the simplification is a purely combinatorial operation, for visualization purposes it is thus often required to compute new scalar data that correspond to the simplified complex. The scalar field reconstruction problem is the topic of the present work. The main requirements on the scalar field reconstructed from a simplified MS complex are the following

- interpolation of the scalar field values at the critical points
- bijection between the nodes of the MS complex and the critical points
- bijection between arcs of the MS complex and 1-cells.

This problem has been solved in the past by applying to each 1d- and 2d-cell a discrete laplacian smoothing in the original mesh with the addition of constraints in order to ensure the monotonicity of data inside the cells [2, 30]. In contrary to previous mesh-based approaches, the present work investigates the use of smooth surface representations.

In the fields of Computer Aided Geometric Design and Approximation Theory *shape preserving methods* seek to approximate a scalar or geometric function while preserving some "shape properties" such as the sign, the monotonicity or the convexity of the given data. Most of these methods produce piecewise polynomial Bézier and B-spline functions, parametric curves and surfaces.

In the present work we propose to reconstruct scalar fields from simplified MS complexes in 2D, based on shape-preserving methods. Instead of using discrete laplacian smoothing on the original mesh, our approach is to use piecewise polynomial curves and surfaces. Specifically, we first approximate data along 1d-cells by computing monotonic B-splines [19], and we then show how to fill-in 2d-cells with monotonic triangular Bézier patches based on our previous work on monotonic interpolation of gridded data [1]. The final resulting scalar field is a piecewise polynomial approximation of the original data that is coherent with the simplified MS complex. We present different approaches for parameterizing and computing these piecewise polynomial surfaces.

2 Related Work

Topology-based / Computational topology methods have become very effective for analysis, visualization and simplification of features in scientific data sets and geometric shape models. Most of these methods make use of results from Morse theory which dates back to the nineteenth century. It became useful in scientific visualization and computer graphics with a first efficient algorithm by Edelsbrunner et. al [5] for computing the MS complex for piecewise linear scalar fields. The MS complex is a topological representation of a scalar function which decomposes the domain of a function into regions having uniform gradient flow. It consists of the critical points (minima, maxima, saddles) and the separatrices connecting them.

Efficient algorithms to compute the MS complex for 2D and 3D scalar fields can be found in [14–17, 26].

Topological simplification is used for denoising, feature discrimination and smoothing of scalar fields. Algorithms apply, inter alia, the notion of topological persistence [6] for iteratively cancelling pairs of critical points [2, 7] in MS complexes.

Closely related to topology-driven simplification algorithms are methods that reconstruct a function obeying the simplified topological data. A setup similar to ours are [2, 30] in the sense that the MS complex is simplified and a corresponding scalar function is computed by fully constraining the simplified complex. In [2] a MS complex is iteratively simplified by cancelling pairs of critical points ordered by persistence resulting in a hierarchical representation. The discrete embedding of boundary and the interior of the cells is smoothly reconstructed cell by cell using constrained Laplacian optimization. In [30] a constrained bi-Laplacian optimization is applied globally on the entire domain. Bi-Laplacians increase the order of continuity from C^0 to C^1 but loose the maximum principle, i.e. loose the important monotonicity property of the resulting function inside the MS cells. Inequality constraints are thus added to enforce the scalar field to be monotonic inside each cell. In [20] only the position and value of extremas are constrained to compute smooth shapes by minimizing a non-linear functional similar to [30]. They further provide an explicit control over the topology of a scalar field. As an extension of [20, 30] Günther et al. [12] present a method for reconstructing iteratively a smooth scalar field from a set of selected extrema by combining non-linear optimization based on a monotonicity graph with topological simplification. Both methods [12, 20] however do not constrain the entire MS complex, but use it to satisfy the monotonicity constrains in the optimization.

However, a big challenge of these numerical reconstruction approaches is the stability in the optimization process. This may create additional critical points in the output preventing it from strictly conforming to the input constraints. Additionally, the overall optimization process might be computationally expensive resulting in extensive running times. Tierny et al. [28, 29] propose an interactive combinatorial algorithm to edit a scalar field on a surface with a user-prescribed extrema that does not build on the computation of a topological structure such as MS complex or contour tree.

The major differences to our work are twofold: first, we provide an explicit polynomial representation of the surface patches and the cell boundaries.

This explicit representation has two advantages: it allows for an arbitrarily dense sampling of the resulting function. It further allows for exact evaluation and computation of differential properties of the scalar function for post-processing applications.

Second, our resulting surface exactly conforms to the input MS complex, meaning that no undesired additional critical points occur as it may be the case with numerical optimization methods.

Our method does not depend on the density/size of the discretization of the input function.

3 Background and Overview

In this section, the required theoretical background on MS complexes is given, for scalar fields defined in a planar domain. First, Sect. 3.1 presents a summary of Morse theory, then Sect. 3.2 explains how MS complexes can be simplified. For further background about Morse theory, the reader is refereed to [22, 27].

3.1 Morse Theory

Let $f : D \to \mathbb{R}$ be a differentiable function, where D is a bounded open subset of $\mathbb{R}^2$. A point $p \in D$ is called a non-degenerated critical point of f if $\nabla f(p) = 0$ and the determinant of the Hessian of f in p is non zero. f is called a Morse function if its critical points are non degenerated. There are three kinds of critical points depending on the sign of the two eigenvalues of the Hessian matrix. The number of negative eigenvalues is called the index of the critical point. If the index is zero then p is a minimum, if the index is one, then p is a saddle point and if all eigenvalues are negative, then p is a maximum.

A C^1 curve $l : \mathbb{R} \to D$ is called an integral line of f if $(\forall t \in \mathbb{R})\ \frac{\partial}{\partial t} l(t) = \nabla f(l(t))$. It is a path which follows the gradient of the function, so the values of f along this line parameterized by t are strictly increasing. The limit of l when $t \to -\infty$ (resp. $t \to +\infty$) is called the origin (resp. destination) of l. The origin and the destination of an integral line are critical points of f, and $\textit{index}(\text{destination}) > \textit{index}(\text{origin})$. The union of a critical point $\{p\}$ with the set of all points of integral lines which share the origin p is called the ascending manifold of p. Analogously, the union of a critical point $\{q\}$ with the set of all points of integral lines which have the same destination q is the descending manifold of q. The set of all ascending manifolds and the set of all descending manifolds form two distinct partitions of D.

The MS complex is the partition of the domain D using the sets of integral lines sharing the same origin and destination. In other words, each set in this partition is the intersection of an ascending and a descending manifold. It has the structure of a CW-complex, with cells of three types, as illustrated in Fig. 1a:

- 0-cells : these cells are critical points of the function,
- 1-cells : these cells are specific integral lines called separatrices,
- 2-cells : these cells are always bounded by a cycle of four 1-cells. Two of these 1-cells link a minimum with two saddles, and the two other 1-cells link the two saddles with a maximum. There is no critical point in the interior of a 2-cell.

In the present paper, we do not work with differentiable functions, but with discrete scalar fields sampling a differentiable function at the vertices of a triangulation D. We therefore rely on Forman's discrete Morse Theory [11] which generalizes MS complexes using a combinatorial gradient vector field. In this setting, integral lines and cells of the MS complex are union of domain simplices (e.g. vertices, edges and triangles).

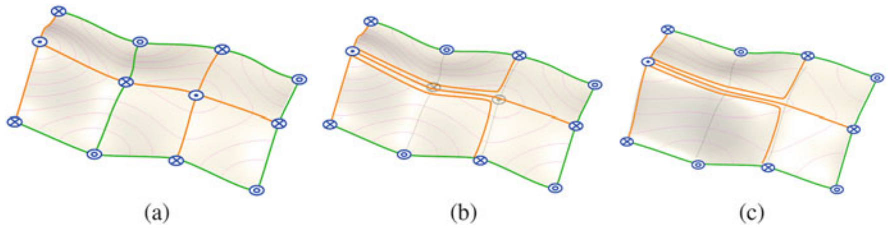


Fig. 1 (**a**) A scalar function on a rectangular domain with its MS complex. It has six 2-cells and seventeen 1-cells. The lines of steepest ascent (resp. descent) from the saddle points are shown in orange (resp. *green*) (**b**) a pair of adjacent critical points in *gray* is removed. As a consequence a simplified topology is produced with four new 2-cells and three new 1-cells. (**c**) a function is reconstructed which MS complex matches the simplified topology (see different isoline pattern)

3.2 Simplification

To explain the simplification of MS complexes, we use the canonical mapping from a MS complex to an undirected graph which maps 0-cells to nodes and 1-cells to edges. The 2-cells then correspond to cycles in the graph. The simplification of the MS complex is obtained by sequentially removing pairs of adjacent nodes in the graph [5, 13]. The edges of least persistence are removed first [4, 23]. Let $\{p, q\}$ be an edge in the graph such that $index(q) = index(p) + 1$. Removing the two nodes in the graph implies the removal of all edges incident to them, thereby creating a cycle of length strictly larger than four. New edges are then inserted between each node previously adjacent to p with the same index value than q except q and each node previously adjacent to q with the same index value than p except p. Each of these new edges replace three removed edges. These new edge insertions ensure that all new cycles are of length four and connect one minimum, one saddle, one maximum and a second saddle. This process is illustrated in Fig. 1a, b with $index(p) = 1$, i.e. when p is a saddle point and q is a maximum. In this case a new edge is inserted between each saddle point that was adjacent to q and distinct from p, and the maximum that was adjacent to p and distinct from q.

The previous process is purely combinatorial. It builds a new simplified topology from the topology of the MS complex, but does not produce a new embedding. A trivial embedding is obtained by associating to each new edge the union of the three 1-cells it replaces, and to each new cycle of union of previous 2-cells. But this trivial embedding does not correspond to a MS complex since there are critical points along the new 1-cells, as illustrated in Fig. 1b. Therefore a new function has to be reconstructed from the original function and from the simplified topology such that the topology of the MS complex of this reconstructed function matches the simplified topology, as illustrated in Fig. 1c. The function reconstruction is the subject of the present paper, as explained in the next section.

3.3 Overview

Our method takes as input an initial discrete scalar field $f : D \subset \mathbb{R}^2 \to \mathbb{R}$, where D is a planar triangulation. The scalar field values f, also called height values, are given at the vertices of D. We also take as input the persistence threshold, so that in the simplified MS complex, the value of least persistence should be larger than the input threshold. In practice the persistence threshold is specified as a percentage of the maximum persistence. 0% mean no simplification and 100% means maximum simplification.

We first use the recent parallel algorithm by Shivashankar et al. [26] to compute the discrete MS complex of f and to simplify its topology, as explained in Sect. 3.2. Our method is also working with output from other algorithms [2, 7].

As explained in Sect. 3.2, the simplification of the MS Complex is a purely combinatorial process which removes critical points and produces a simplified topology connecting the remaining critical points. And the trivial embedding of this simplified topology consists in 1-cells and 2-cells inside which the original scalar field has critical points, as illustrated in Fig. 1b. Therefore we need to reconstruct a new scalar field whose MS complex matches exactly the simplified topology, i.e. it should be strictly monotonic along the simplified 1-cells, and should contain no critical points inside the simplified 2-cells.

In order to reconstruct such a scalar field we proceed in two steps. In the first step we traverse each 1-cell and compute a smooth uniform quadratic spline which is C^1-continuous, strictly monotonic and approximates the original scalar field values. This is described in Sect. 4. In the second step we reconstruct a scalar field inside each 2-cell, which interpolates the four monotonic uniform quadratic splines along its boundaries and contains no critical points, as explained in Sect. 5. To compute this scalar field, we first parameterize the domain of the 2-cell into the unit square, and then compute a monotonic piecewise cubic triangular Bézier surface defined in the unit square, which interpolates the height values of the uniform quadratic splines along the edges of the square.

As a consequence our reconstructed scalar field is defined by piecewise polynomial functions along the 1-cells and inside the 2-cells.

4 Monotonic Smoothing of the 1-Cells

The first part of our method consists in reconstructing smooth and strictly monotone curves along the new 1-cells. Before explaining the details we must point out two facts. First, the 1-cells of the simplified MS complex in the domain are known to have a jagged shape prohibiting a smooth surface reconstruction [2, 30]. Second, it happens often that two 1-cells, having the same maximum as destination but distinct saddles as origin, merged before the maximum. In the other direction, two 1-cells, having the same minimum as origin and distinct saddles as destination, may share the same geometry before getting disconnected. Such merging or disconnection occur at special vertices called *junction points*. Figure 3-right shows two junction points between a minimum and three distinct saddles. These junction points segment the corresponding 1-cells into several pieces. We first compute new scalar field values at the junction points, so that subsequent processing may be done independently on each piece of 1-cells. Then for each piece we compute a piecewise quadratic uniform B-spline that is smooth, strictly monotonic and approximates the input scalar field values.

1. **Scalar field value at junction points**. If we follow a simplified 1-cell with junction points, the sequence of scalar field values at these junction points is not strictly monotone. Assigning iteratively a new value by linear interpolation of the critical end points as proposed in [30] may run in conflicts in case a junction point is shared by more than two 1-cells. We therefore propose to compute the junction point values globally by solving the following linear programming problem:

$$\sum |\tilde{f}(\mathbf{p}_j) - f(\mathbf{p}_j)| \rightarrow min \quad \text{subject to}$$
$$max\_lower(\mathbf{p}_j) \leq \tilde{f}(\mathbf{p}_j)$$
$$\tilde{f}(\mathbf{p}_j) \leq min\_larger(\mathbf{p}_j)$$

   where f is the initial scalar field, $\tilde{f}$ the unknown value and $\mathbf{p}_j$ the junction points in D. $max\_lower(p_j)$ (resp. $min\_larger(p_j)$) denotes the maximum (resp. minimum) height value of all junctions and critical points adjacent to p_j and are required to be lower (resp. larger) than $\tilde{f}(p_j)$.

   The idea is to compute a new function value $\tilde{f}(\mathbf{p}_j)$ as close as possible to $f(\mathbf{p}_j)$ for all junction points, while satisfying the monotonicity requirements along all 1-cells.
2. **Monotone reconstruction of the scalar field along 1-cells**. We now compute a monotone piecewise polynomial function along each part of the 1-cells by applying a monotone B-spline smoothing [19] between two adjacent points (junction or critical) of the MS complex.

   Let $(\mathbf{x}_i)_{i=0\ldots n}$ be the $n+1$ vertices of a part of a 1-cell in the domain D between two junction or critical points x_0 and x_n. Let $(t_i)_{i=0\ldots n}$ be the associated normalized arc length parameters, so that $t_0 = 0 < t_1 < \cdots < t_n = 1$. New scalar field values have already been computed in $\mathbf{x}_0$ and $\mathbf{x}_n$, and we have $\tilde{f}(\mathbf{x}_0) < \tilde{f}(\mathbf{x}_n)$.

   We now compute a quadratic uniform B-spline function $g_\alpha : [0,1] \rightarrow \mathbb{R}$ that is strictly increasing, and approximates the input scalar field values. Let $g_\alpha(t) = \sum_{i=0}^{k+1} \alpha_i N_i^2(t)$, where $t \in [0,1]$, $\alpha_i \in \mathbb{R}$ are the control points, $0 = u_{-2} = u_{-1} = u_0 < \ldots < u_k = u_{k+1} = u_{k+2} = 1$ the uniform knots and N_i^2 the B-spline basis functions of degree 2 defined on the knot sequence $\{u_j\}_{j=-2,\ldots,k+2}$. We set

$\alpha_0 = \tilde{f}(\mathbf{x}_0)$ and $\alpha_{k+1} = \tilde{f}(\mathbf{x}_n)$ to ensure endpoint interpolation and compute the remaining coefficients α_i, $i = 1, \ldots, k$ as follows:

$$\sum_{i=1}^{n-1} |f(\mathbf{x}_i) - g_\alpha(t_i)| \to min \quad \text{subject to} \tag{1}$$

$$g'_\alpha(u_0) = 0 \tag{2}$$

$$g'_\alpha(u_k) = 0 \tag{3}$$

$$g'_\alpha(u_i) \geq 0, \; i = 1, \ldots, k-1 \tag{4}$$

Condition (1) ensures approximation. (2) and (3) ensure zero gradients at the extremities and finally (4) ensures monotonicity. Figure 2 shows two results with the same input data but different numbers k of control-points.

As the reconstructed surface, see next section, will be composed of cubic triangular Bézier patches, we first convert the quadratic B-spline function g_α into quadratic Bézier curves by repeated knot insertion and then apply a degree elevation to obtain k cubic Bézier curves. See [8] for more details about Bézier and B-spline curves. Note that the parameter k represents the number of curve segments along the spline curve. The higher k the more flexible is the curve and the better it is able to approximate a point set. In contrary, the smaller k is the smoother is the resulting curve. In our experiments we choose either $k = 4$ or $k = 7$, see discussion in Sect. 6.

3. **Smoothing of 1-cells in the domain**. As we aim to reconstruct the simplified MS complex with piecewise polynomial functions, we further smooth the jagged 1-cells in the domain by approximating in a least square sense each piece of 1-cell by a quadratic uniform parametric B-spline curve $\mathbf{c}(t) = \sum_{i=0}^{k+1} \mathbf{c}_i N_i^2(t) \subset \mathbb{R}^2$, $t \in [0, 1]$ with the same number of segments k and knot vector as for the monotone height function g_α. Endpoints interpolation (critical points or junction

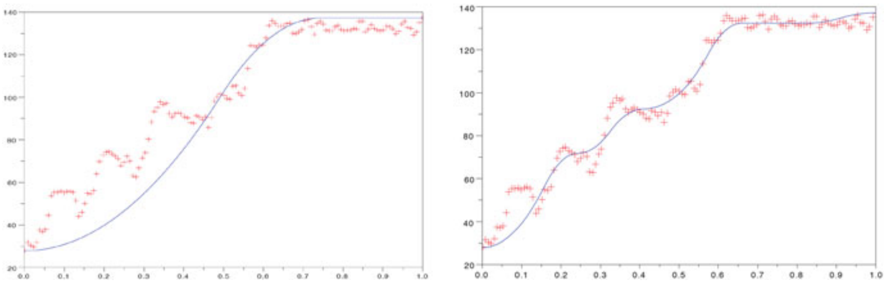


Fig. 2 Height values (*red dots*) are approximated by a *monotonic* quadratic B-spline function (*blue curve*). *Left*: Approximation with $k = 4$ curve segments. *Right*: Approximation with $k = 12$ curve segments

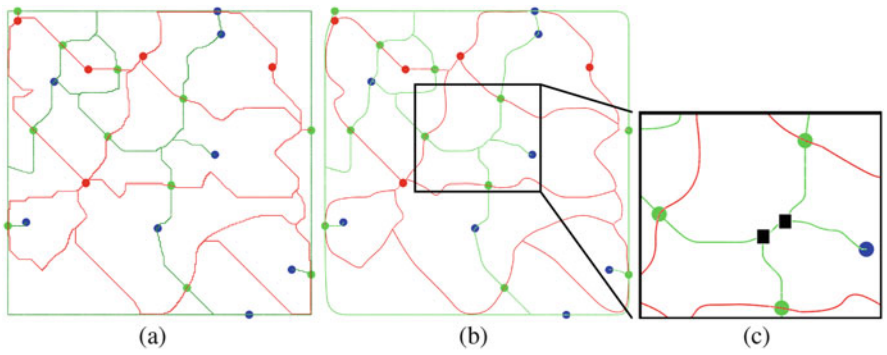


Fig. 3 Smoothing of the 1-cells in the domain. (**a**) original jagged 1-cells. (**b**) smoothing by uniform quadratic B-splines. (**c**) zoom into three 1-cells (*green curves*) descending from 3 distinct saddles (*green dots*) to the same minimum (*blue dot*). The three 1-cells merge at two junction points (*black dots*)

points $\mathbf{p}_i, \mathbf{p}_j$) is enforced by setting $\mathbf{c}(0) = \mathbf{p}_i$ and $\mathbf{c}(1) = \mathbf{p}_j$. After knot insertion and degree elevation we get again k cubic Bézier curve segments. Figure 3 illustrates this smoothing for a real MS complex. The final monotone smooth height function can be written as $(\mathbf{c}(t), g_\alpha(t))$.

5 Monotone Interpolation Inside the 2-Cells

In the previous section we have explained how to compute smooth monotonic piecewise polynomial functions that approximate the input scalar field values on each 1-cell of the simplified MS complex.

In this section we go further and described our method for computing for each 2-cell a piecewise polynomial surface that interpolates its boundary 1-cells and has no interior critical points. The construction is done independently for each 2-cell. We begin by parameterizing the 2-cell onto the unit square $[0, 1]^2$ (Sect. 5.1). Then a monotonic surface defined in the unit square is computed (Sect. 5.2). Combining the monotone surface and the parameterization, a monotone reconstruction of the 2-cell is computed, which interpolates the 1-cells, has the 0-cells as critical points and no other critical points.

5.1 Parametrization of a 2-Cell

A 2-cell is a closed connected subdomain $\Omega \subset D \subset \mathbb{R}^2$ bounded by four pieces of 1-cells. We need to parameterize Ω in the unit square before computing the monotonic interpolant in the next section. In other words, a diffeomorphism $\Phi : \Omega \to [0,1]^2$ needs to be defined. Such a diffeomorphism maps functions without critical points on the unit square to functions without critical points on Ω. If F is a function without critical points on the unit square, then $\tilde{f} = F \circ \Phi : \Omega \to \mathbb{R}$ is a function without critical points on Ω. We compute the parameterization Φ using the following procedure.

First, we sample points on $\partial\Omega$ by evaluating all Bézier curves bounding Ω at a fixed number c of parameter values. Usually, we set $c = 5$. Then, a conforming Delaunay triangulation is computed which has all boundary points as constraints. The number of inserted points in the interior of Ω depends on the expected triangle quality. We set the *triangle quality* parameter so that the size of inner triangles is similar to the average sample point distance on the boundary. We use the CGAL [3] library to compute the triangulation.

A parameter value $(u, v) \in [0,1]^2$ is computed for each vertex of the triangulation as follows. First, the boundary curves are mapped to the border of the unit square by associating the critical points (resp. junction points) with the corners of the square: the minimum is mapped to the parameter $(0,0)$, the maximum to $(1,1)$ and the two saddle points to the remaining corners. In some cases $\partial\Omega$ does not contain the minimum or the maximum of the 2-cell because the boundary 1-cells merge, as illustrated in the right image of Fig. 3. In these cases, we map to the corner $(0,0)$ (resp. $(1,1)$) the junction point that is closest to the missing minimum (resp. maximum) along $\partial\Omega$. Between each corner along $\partial\Omega$, we assign the parameter that was used to sample the boundary Bézier curves. Then Floater's Mean Value Coordinates [10] algorithm is applied to compute a parameterization of the triangulation of Ω in the unit-square, constrained by the parameter values on $\partial\Omega$. The parameterization process is illustrated in Fig. 4.

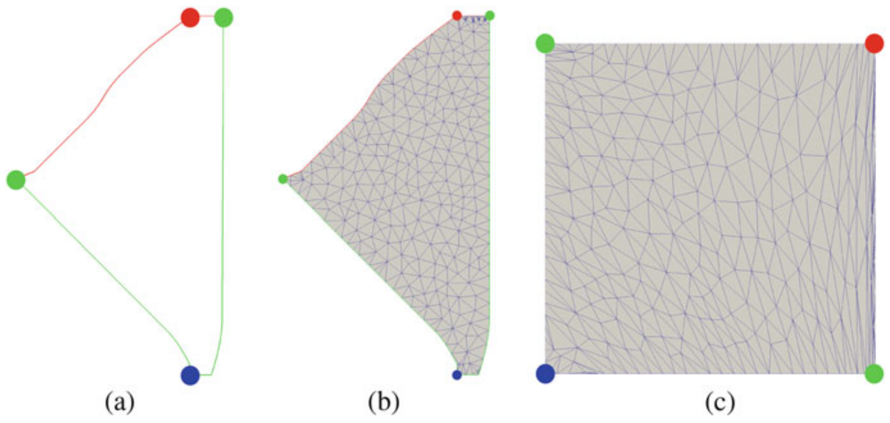


Fig. 4 (**a**) Boundary of a 2-cell, the *red point* is a maximum, the *blue point* is a minimum and the two *green points* are saddles. (**b**) triangulation of the 2-cell. (**c**) parameterization of the 2-cell

5.2 Monotonic Scalar Field on the Unit Square

In Sect. 4 we have computed smooth 1-cells and new scalar field values strictly increasing along these 1-cells. In the previous section we have parameterized each 2-cell in the unit square. This parameterization maps the boundary 1-cells to the edges of the unit square. We now explain how to compute a scalar field on the unit square that interpolates the given increasing scalar field values on the edges of the domain and that has no critical point in the interior of the domain.

We apply techniques from [1], where an algorithm is developed for interpolating a grid of scalar values by a monotonic C^1-continuous piecewise cubic triangular Bézier surface. The interpolant of [1] is uniquely defined by interpolated scalar values and by two partial derivatives at the vertices of a uniform grid. Sufficient conditions on the interpolated values and the partial derivatives are derived to ensure that the interpolant has no interior critical points. More precisely, the interpolated values are supposed to be strictly increasing along the diagonals of the grid. Moreover if no partial derivative is known, then [1] provides an algorithm that computes partial derivative values leading to a monotonic interpolant.

The monotone interpolant [1] is applied as follows. If there is one or more junction points along one of the boundary 1-cells, the other 1-cells are subdivided,[1] so that the same number of cubic Bézier curves is used in the four 1-cells. We initialize a uniform grid that has in each direction a number of edges equals to the number of cubic Bézier curves. Along the boundaries of the grid, we prescribe the end-points of the cubic Bézier curves computed in Sect. 4 and the first derivative of these cubic Bézier curves at their end-points. To compute the scalar values to be interpolated at the interior points of the grid, a Laplace equation with the boundary values as constraints is solved. In almost all cases the resulting values are strictly increasing along the grid diagonals. If not, we interpolate linearly the scalar values along the diagonal. Then, the interior partial derivatives are computed by using the results of [1]. In the end, we get a piecewise cubic triangular Bézier function that has no interior critical points, and that interpolates the strictly increasing 1-cells computed in Sect. 4. Such an interpolant is shown in Fig. 5a. Its reparameterization in the 2-cell domain, as explained in Sect. 5.1, is shown in Fig. 5b.

There is a particular case, called *pouch cells* which occurs when the two saddle points are merged. To handle this type of cells as well, we adapt our reconstruction process by only changing the parameterization step.

[1] Subdivision of Bézier curves cuts them into several Bézier curves defining exactly the same curve [8].

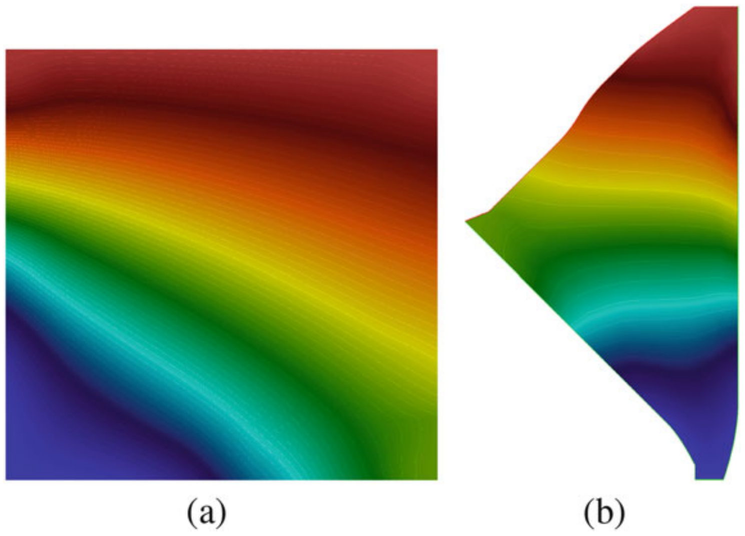


Fig. 5 (**a**) Colormap of the piecewise cubic scalar field in the unit square. It has no critical points as can be seen from the isolines. (**b**) the same scalar field after reparameterization in the 2-cell domain

6 Results

We have implemented our algorithm and tested it on three types of functions. All the experiments were performed on a machine with an Intel Xeon W3520 CPU which provides 8 GB RAM. The MS complexes of the input function and its simplified versions were computed with [26] and preprocessed as explained in Sect. 3.3. We visualize the initial and reconstructed functions with color contour plots (linear color scale) augmented with a regularly sampled set of isolines. The MS complexes are visualized with the ascending (red curves) and descending 1-cells (green curves) together with the critical points, i.e. saddles (green dots), maxima (red dots) and minima (blue dots).

The first example is a hand-made scalar data set that has been created using Shepard's scattered data interpolation method [24]. Given a set of N randomly chosen position and function values $\{\mathbf{x}_i; f_i\}$, the method computes a function $F_\mu(\mathbf{x}) = \sum_{k=0}^{N} f_k \|\mathbf{x} - \mathbf{x}_i\|^{-\mu} / \sum_{j=0}^{N} \|\mathbf{x} - \mathbf{x}_j\|^{-\mu}$ such that $F(\mathbf{x}_i) = f_i$ with zero-gradients at the input points when $\mu > 1$. Our so-called *Shepard* data set has then been built by taking randomly 50 data points and values and by evaluating F_2 on a 250×250 regular grid. The resulting function has at least these 50 extrema. A simplification of 25% is applied and the resulting MS complex reconstructed. Figure 6 shows the

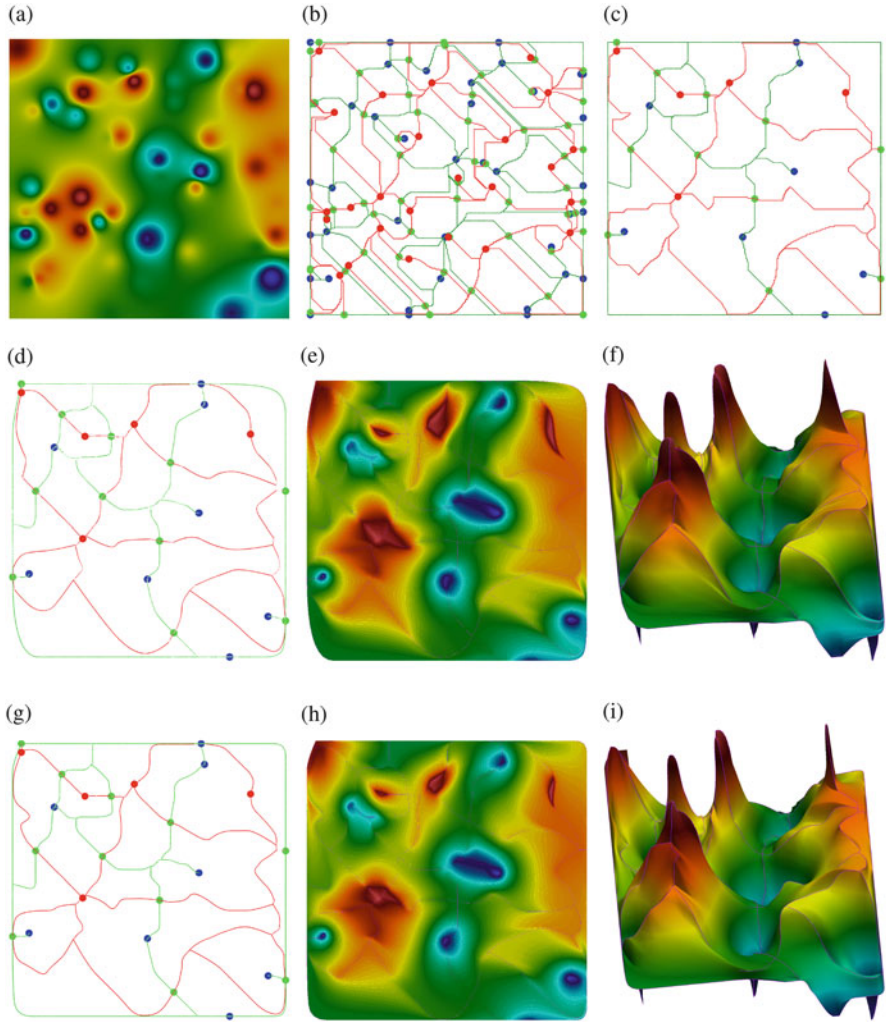


Fig. 6 Applying our algorithm on a synthetic example that has been generated using Shepard's scattered data interpolation. We tested our algorithm with two different values of k (number of Bézier curves per 1-cell, grid size of Bézier patches). In both cases our method reconstructs a piecewise smooth scalar field that exactly obeys the topological structure of the MS complex. (**a**) Initial function, (**b**) initial MS complex, (**c**) simplified MS complex, (**d**) smoothed MS complex with $k = 4$ Bézier curves per 1-cell, (**e**) reconstruction, (**f**) reconstruction, (**g**) smoothed MS complex with $k = 7$ Bézier curves per 1-cell, (**h**) reconstruction ($k = 4$), (**i**) reconstruction ($k = 7$)

initial function (a) and its MS complex (b). A 25% simplification is applied (c). The middle and bottom line of Fig. 6 show two different reconstructions of the same simplified MS complex. They differ in the number k of curve segments composing each piece of the 1-cells (see Sect. 4).

Table 1 Measured performances and model statistics

Model			MSC			1-cells		2-cells		
	#faces	#V	simplif. level	#CP	#2-cells	k	time I[a]	time II Delaunay +param.	time III geometry	#Δ Bézier patches
Shepard	124,002	62,500	25%	25	22	4	0,13	8,43	1,15	6672
			25%	25	22	7	0,13	8,80	0,39	19,024
MtRainier	280,296	140,913	2%	69	53	7	0,31	9,64	2,16	43,300
Trigo	19,602	10,000	0%	31	24	7	0,03	2,11	0,11	6240

[a]Time I is the computation time for smoothing and reconstruction of 1-cells

The role of k is twofold. On one hand it governs the smoothness and flexibility of the curves. The smaller k is, the smoother are the curves; but a higher value gives more flexibility to the curves and thus better approximates the height values. On the other hand, the number of Sibson patches is equal (or sometimes proportional) to k^2 per 2-cell. The middle row shows a reconstruction with $k = 4$, the bottom row with $k = 7$. The first step of our algorithm is shown in (d) and (g), where the jagged 1-cells of the simplified MS complex are smoothed by parametric B-spline curves in the plane, and *monotone* B-spline height functions that best approximate the initial scalar field along the 1-cells. The final reconstructions are shown in (e,f) and (h,i). Even though there is no much difference visible in the final reconstruction contour plots, it can be observed in the 1-cells reconstructions that the curves in the plane have less undulations for $k = 4$ in (d), but that the boundary 1-cells are less well approximated compared to $k = 7$ in (g), see the green curves at the bottom left and top right corners.

The model statistics and computation times are listed in Table 1. As expected, re-parameterization is the most costfull step. Triangulating the domain of each 2-cell followed by a parameterization takes 95% of the total computation time. It follows further that the influence of k on the computation time is neglectable, even though the number of Sibson patches is proportional to k^2.

In the second example is a real terrain data set of the *Mont Rainier*.[2] It consists of 140148 vertices defining a $458\times$ 306 grid. This data set is more complex in the sense that the initial function is not smooth and has critical points that are not well distributed over the domain. Figure 7 shows the initial function (a), its MS complex (b) with 1931 critical points. After 2% simplification, the MS complex (c) has 69 critical points. Figure 7d, e, f show the reconstruction with the smoothed 1-cells and final function.

[2]http://data.geocomm.com/catalog/.

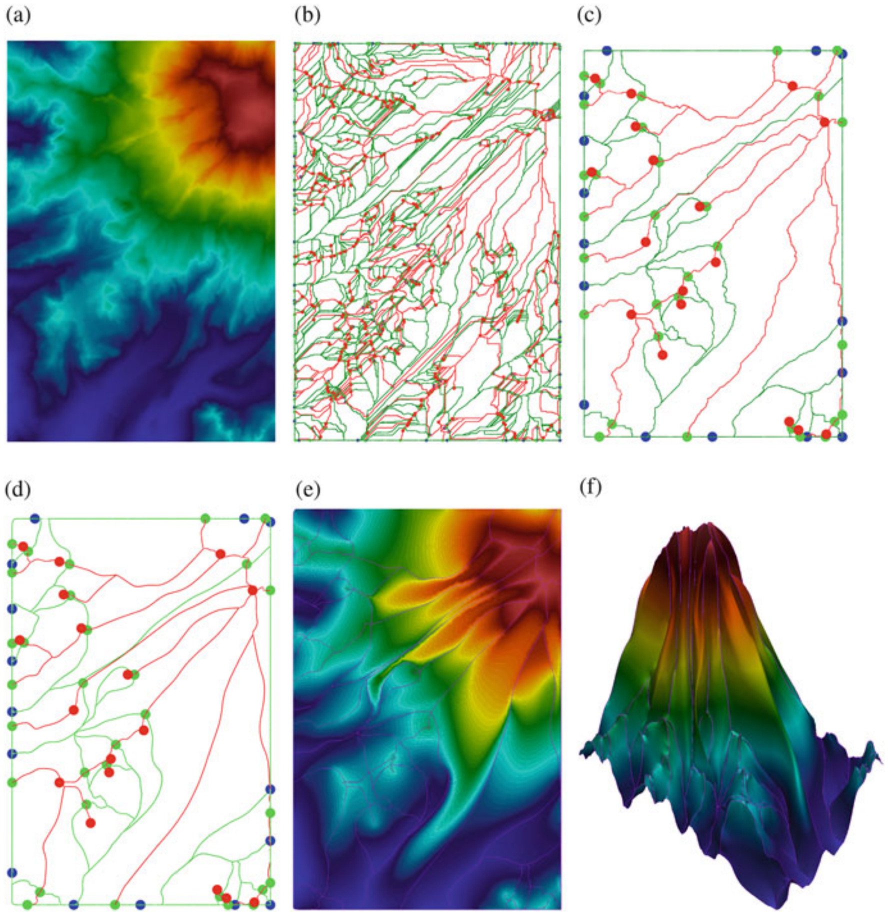


Fig. 7 The terrain data set of Mt Rainier (**a**) has 1931 critical points (**b**). The simplified MS complex with 69 critical points is reconstructed. (**a**) Initial function, (**b**) initial MS complex, (**c**) simplified MS complex, (**d**) smoothed MS complex, (**e**) reconstruction, (**f**) reconstruction

The last data set is sampled on a 100×100 grid from the mathematical function $a \cdot \sin(x/a) \cdot \sin(y/a)$ with $a = 10$.
The function has 31 critical points in the domain $[-49, 50]^2$. Here we did not compute any simplification (0%), but we applied our algorithm to reconstruct a function from the initial MS complex, see Fig. 8.

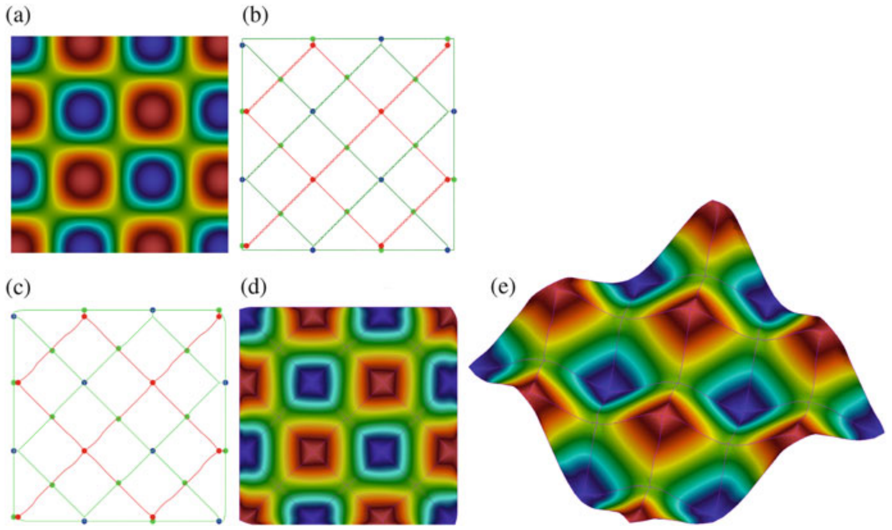


Fig. 8 Reconstruction of data sampled from $\sin(x) \cdot \sin(y)$. The original data set (**a**) has been reconstructed (**d**, **e**) without topological simplification. (**a**) Initial function, (**b**) MS complex, (**c**) smooth MS complex, (**d**) reconstruction, (**e**) reconstruction

7 Conclusion

This paper experienced for the first time the use of geometric modeling techniques to MS complex reconstruction. The main contribution is the application of a shape preserving spline surface. In particular we approximate the jagged 1-cells on the complex with smooth B-splines. We introduce a C^1 monotone Sibson spline interpolant and combine it with a reparameterization to reconstruct the 2-cells individually. The resulting surface obeys completely the given MS complex. The fact to provide an explicit closed form representation of the reconstructed function may be an advantage when follow-up evaluations of the function and any higher order derivatives are required. The independence of the present method of the mesh or grid on which the initial function is defined on may also be advantageous in case where the grid size is very large.

Beside these achievements we noticed several limitations of our method. First, the lack of existing C^1 reparameterisation techniques makes our resulting surface being only piecewise C^0 continuous at the end. Recent techniques in isogeometric analysis, where the problem of parameterizing a domain defined by boundary curves is also a central concern, enable to parameterize an arbitrary domain up to some restriction using C^1 splines [9, 21, 31]. However, none of these techniques can actually deal with arbitrary domains as they occur in MS complexes. A C^1 parameterization method for arbitrary domains would improve the general smoothness of our results. We could then develop the Sibson interpolant further to reach globally C^1 reconstructions.

Second, regarding computation times, the parameterization step seems to be the bottleneck of the presented method. As the reconstruction of the 2-cells is done independently for each 2-cell, the algorithm could be parallelized to improve this part of the method. We believe however, that it would be more efficient to overcome the parameterization step completely and to work directly on the domain. The triangular spline approximation approach in [21] could be a further avenue to explore in the future.

Acknowledgements This research was partially funded by the ERC advanced grant EXPRESSIVE (no. 291184).

References

1. Allemand-Giorgis, L., Bonneau, G.P., Hahmann, S., Vivodtzev, F.: Piecewise polynomial monotonic interpolation of 2D gridded data. In: Bennett, J., Vivodtzev, F., Pascucci, V. (eds.) Topological and Statistical Methods for Complex Data: Tackling Large-Scale, High-Dimensional, and Multivariate Data Spaces, pp. 73–91. Springer, Berlin/Heidelberg (2015)
2. Bremer, P.T., Edelsbrunner, H., Hamann, B., Pascucci, V.: A topological hierarchy for functions on triangulated surfaces. IEEE Trans. Vis. Comput. Graph. **10**(4), 385–396 (2004)
3. CGAL: Computational Geometry Algorithms Library (2017). http://cgal.org
4. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. In: FOCS, pp. 454–463. IEEE Computer Society (2000)
5. Edelsbrunner, H., Harer, J., Zomorodian, A.: Hierarchical Morse complexes for piecewise linear 2-manifolds. In: Symposium on Computational Geometry, pp. 70–79. ACM (2001)
6. Edelsbrunner, H., Letscher, D., Zomorodian, A.: Topological persistence and simplification. Discret. Comput. Geom. **28**(4), 511–533 (2002)
7. Edelsbrunner, H., Harer, J., Zomorodian, A.: Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. Discret. & Computat. Geom. **30**(1), 87–107 (2003)
8. Farin, G.E.: Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide. Academic, San Diego (1997)
9. Farin, G.E., Hansford, D.: Discrete Coons patches. Comput. Aided Geom. Des. **16**(7), 691–700 (1999)
10. Floater, M.S.: Mean value coordinates. Comput. Aided Geom. Des. **20**(1), 19–27 (2003)
11. Forman, R.: A user's guide to discrete morse theory. In: Proceedings of the 2001 International Conference on Formal Power Series and Algebraic Combinatorics. Advances in Applied Mathematics, special volume, p. 48 (2001)
12. Günther, D., Jacobson, A., Reininghaus, J., Seidel, H., Sorkine-Hornung, O., Weinkauf, T.: Fast and memory-efficient topological denoising of 2D and 3D scalar fields. IEEE Trans. Vis. Comput. Graph. **20**(12), 2585–2594 (2014)
13. Gyulassy, A.: Combinatorial construction of Morse-Smale complexes for data analysis and visualization. Ph.D. Thesis, University of California, Davis (2008)
14. Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P., Hamann, B.: A topological approach to simplification of three-dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. **12**(4), 474–484 (2006)
15. Gyulassy, A., Natarajan, V., Pascucci, V., Hamann, B.: Efficient computation of Morse-Smale complexes for three-dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. **13**(6), 1440–1447 (2007)
16. Gyulassy, A., Bremer, P., Hamann, B., Pascucci, V.: A practical approach to Morse-Smale complex computation: scalability and generality. IEEE Trans. Vis. Comput. Graph. **14**(6), 1619–1626 (2008)

17. Gyulassy, A., Bremer, P., Pascucci, V.: Computing Morse-Smale complexes with accurate geometry. IEEE Trans. Vis. Comput. Graph. **18**(12), 2014–2022 (2012)
18. Gyulassy, A., Kotava, N., Kim, M., Hansen, C.D., Hagen, H., Pascucci, V.: Direct feature visualization using Morse-Smale complexes. IEEE Trans. Vis. Comput. Graph. **18**(9), 1549–1562 (2012)
19. He, X., Shi, P.: Monotone B-spline smoothing. J. Am. Stat. Assoc. **93**(442), 643–650 (1998)
20. Jacobson, A., Weinkauf, T., Sorkine, O.: Smooth shape-aware functions with controlled extrema. Comput. Graph. Forum **31**(5), 1577–1586 (2012)
21. Jaxon, N., Qian, X.: Isogeometric analysis on triangulations. Comput. Aided Des. **46**, 45–57 (2014)
22. Morse, M.: The Calculus of Variations in the Large, vol. 18. American Mathematical Society, Providence (1934)
23. Robins, V.: Computational topology at multiple resolutions: foundations and applications to fractals and dynamics. Ph.D. Thesis, University of Colorado, Boulder (2000)
24. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of Symposium on Geometry Processing, pp. 517–524. ACM, New York (1968)
25. Shivashankar, N., Natarajan, V.: Parallel computation of 3D Morse-Smale complexes. Comput. Graph. Forum **31**(3), 965–974 (2012)
26. Shivashankar, N., Maadasamy, S., Natarajan, V.: Parallel computation of 2D Morse-Smale complexes. IEEE Trans. Vis. Comput. Graph. **18**(10), 1757–1770 (2012)
27. Smale, S.: On gradient dynamical systems. Ann. Math. **74**(1), 199–206 (1961)
28. Tierny, J., Günther, D., Pascucci, V.: Optimal general simplification of scalar fields on surfaces. In: Bennett, J., Vivodtzev, F., Pascucci, V. (eds.) Topological and Statistical Methods for Complex Data: Tackling Large-Scale, High-Dimensional, and Multivariate Data Spaces, pp. 57–71. Springer, Berlin/Heidelberg (2015)
29. Tierny, J., Pascucci, V.: Generalized topological simplification of scalar fields on surfaces. IEEE Trans. Vis. Comput. Graph. **18**(12), 2005–2013 (2012)
30. Weinkauf, T., Gingold, Y.I., Sorkine, O.: Topology-based smoothing of 2D scalar fields with $\underline{C}^1$-continuity. Comput. Graph. Forum **29**(3), 1221–1230 (2010)
31. Xu, G., Mourrain, B., Duvigneau, R., Galligo, A.: Parameterization of computational domain in isogeometric analysis: methods and comparison. Comput. Methods Appl. Mech. Eng. **200**(23), 2021–2031 (2011)

Part IV
Vector and Tensor Field Topology

Topological Extraction of Escape Maps in Divergence-Free Vector Fields

Ronald Peikert, Gustavo Machado, and Filip Sadlo

Abstract An escape map is the partial mapping from seed points to exit points of streamlines in a bounded domain. The escape map is piecewise continuous, and a topological segmentation of the domain boundary yields the regions and curve segments on which it is continuous. Escape maps have recently been introduced in the context of studying the connectivity of coronal holes. Computation of escape maps faces the problem of exponentially diverging streamlines, where standard adaptive streamsurface methods can fail. As a tool to detect such places and to guide escape map computation, a technique based on isoclines has recently been proposed (Machado et al., IEEE Trans. Vis. Comput. Graph. 20(12):2604–2613, 2014). We show in this paper that, in the case of a divergence-free vector field, boundary switch connectors can be used as a purely topological alternative, which to the best of our knowledge is the first practical application of boundary switch connectors. We provide a systematic approach to the topological segmentation of 3D domain boundaries for divergence-free vector fields. Finally, we explore an alternative approach based on streamtubes and targeted at robustness in escape map computation. Simulation results as well as a synthetic vector field are used for validation.

1 Introduction

In a static vector field $\mathbf{v}(\mathbf{x})$ on a domain $D \subset \mathbb{R}^n$ $(n = 2, 3)$, the *streamline* $\Phi(\mathbf{x}, t)$ is a function of the seed point $\mathbf{x}$ and the integration time t. Its value is either a point in D, or *undefined* if the streamline leaves the domain in time less than t. Keeping

R. Peikert (✉)
ETH Zurich, Zürich, Switzerland
e-mail: peikert@inf.ethz.ch

G. Machado
University of Stuttgart, Stuttgart, Germany
e-mail: Gustavo.Machado@vis.uni-stuttgart.de

F. Sadlo
University of Heidelberg, Heidelberg, Germany
e-mail: filip.sadlo@iwr.uni-heidelberg.de

the integration time fixed leads to the *flow map*, $\Phi : D \rightarrow D$, which is a partial map because the streamline function can be undefined at $(\mathbf{x}, t)$.

In analogy to the flow map, we can define the *escape map* $\Psi^{+} : \partial D \rightarrow \partial D$ as a partial map from the domain boundary to itself. The integration time is the smallest positive t for which $\Phi(\mathbf{x}, t)$ is a point on ∂D. We find it advantageous not to require that the streamline exits the domain at this point, even though the term "escape map" would suggest it. The streamline can also just be tangent to the boundary and stay within the domain. The map Ψ^{-} is analogous, but takes the negative time with smallest absolute value.

The escape maps are *piecewise* continuous, and it is an interesting goal to find the maximal subsets of ∂D on which they are continuous. We will refer to such a partitioning, caused by the discontinuities of Ψ^{+} and Ψ^{-}, as the *topological segmentation* of ∂D. Its segments can be points, curve segments, or (for $n = 3$) two-dimensional regions. Having segmented ∂D, the escape map can now be also understood as a mapping from the set of segments to itself, which is again a partial map. The set of segments is usually finite, especially if $\mathbf{v}(\mathbf{x})$ is given in discretized form.

In a practical escape map problem [12], an additional partitioning of ∂D into regions $\partial D_1, \partial D_2, \ldots$ may be specified (e.g., "map regions", "escape regions"). In that case, the segmentation has to be refined, such that all segments are either fully contained in the interior, the exterior, or the boundary of any ∂D_i. The typical question "What is $\left(\Psi^{+}(\partial D_1) \cup \Psi^{-}(\partial D_1)\right) \cap \partial D_2$?" can then be answered by determining a subset of the set of segments.

The theoretic background of our problem is *vector field topology* (VFT) [6, 10, 15]. A systematic analysis of VFT in 3D was done by Asimov [2], discussing in particular 1D and 2D manifolds of saddles and saddle-type periodic orbits. Invariant tori, not mentioned by Asimov, can also be part of a 3D segmentation, but will not be relevant for us, as they cannot intersect the domain boundary. VFT as a tool for visualizing magnetic fields has been studied by Cai et al. [4], focusing on saddle connectors and on chaotic fields. An extension of VFT for representing dipoles in magnetic fields has recently been derived by Bachthaler et al. [3]. We will assume that the magnetic field contains no dipoles within the computational domain, therefore first-order VFT will be sufficient for our purpose. VFT on bounded domains has been studied by de Leeuw and van Liere [5] and Scheuermann et al. [17] in 2D and by Weinkauf et al. [20] in 3D, who analyzed the manifolds of boundary switch curves. Given this background, our task roughly consist of segmenting the bounded 3D vector field, intersecting it with the boundary, and exhaustively applying the maps $\Psi^{\pm}$. We show that such a segmentation contains elements of a relatively small number of types, at least in our chosen setting of divergence-free fields.

2 Escape Maps in Divergence-Free Vector Fields

A first, coarse segmentation is given by the sign of the normal component of $\mathbf{v}(\mathbf{x})$ on ∂D. At *outflow regions* of ∂D, where $\mathbf{v}(\mathbf{x})$ has an outward normal component, the map Ψ^+ is undefined, while at *inflow regions*, Ψ^- is undefined. The remaining parts of ∂D, where the flow is tangential, are particularly interesting. If we exclude structurally unstable degeneracies, then the normal component on ∂D has non-degenerate zero contours. By this, there are no critical points on ∂D, and the flow is tangential only on *boundary switch points* (if $n = 2$) or *boundary switch curves (BSCs)* (if $n = 3$). BSCs consist of *outbound segments*, where both Ψ^+ and Ψ^- are undefined, and *inbound segments*, where both of them are defined [20]. On the *inout points* between such segments, the flow direction crosses the BSC. On these points exactly one of the two escape maps is defined.

For the full segmentation, the topology of $\mathbf{v}(\mathbf{x})$ in the interior has to be considered, too. To make discussion simpler, we will only consider the three-dimensional case and focus our interest on a combined segmentation for both Ψ^+ and Ψ^-. Since, at each point of ∂D, at least one of the escape maps is undefined, except for points on a BSC, we will use the symbol Ψ to denote the existing one.

In addition, we will restrict ourselves to *divergence-free vector fields*, which simplifies a number of things. Firstly, on ∂D the escape map cannot be undefined in regions, but only on curve segments and points. This is because open subregions of inflow or outflow regions have a nonzero flux through ∂D, and in a divergence-free field this flux is conserved. Secondly, there are neither sinks/sources nor attracting/repelling periodic orbits, so we are left with saddles and saddle-type periodic orbits, as far as topological features in the interior are concerned. Invariant tori, which could normally also act as attractors and thereby contribute to the topological segmentation, cannot do this in a divergence-free field (as can be seen again with the flux argument). Thirdly, by excluding degeneracies, we can assure that special streamlines such as boundary switch connectors or 1D manifolds of saddles always reach ∂D and that special streamsurfaces such as those seeded at BSCs reach ∂D with exceptions being just a finite number of isolated points.

2.1 Escape Maps from Streamline Sampling

A naive method would be to sample ∂D with streamlines and check where they intersect the domain again. Since in the divergence-free vector field only a null set of streamlines will stagnate in the interior, this method will give a good picture of the escape map in terms of its appearance, though not its topology.

For demonstration on a practical example, we will use in the following the coronal holes simulation data [16]. The data are given on a domain that is confined by two concentric spheres. Hence, the boundary consists naturally of two parts ∂D_1 and ∂D_2, and as they are spheres, no region boundaries are introduced by the

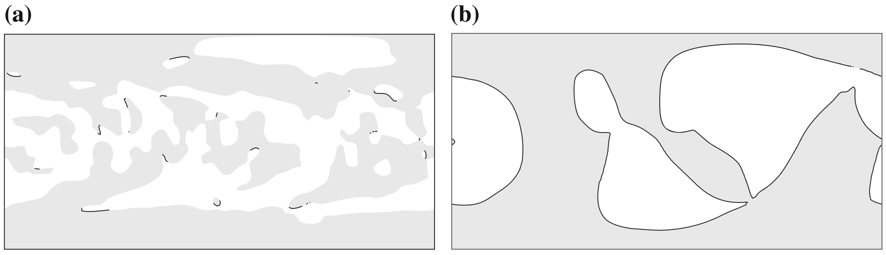


Fig. 1 Boundary segmentation for Coronal Holes data sets (spherical coordinate plots). Inflow and outflow regions are shown in two different *gray shades*. Inbound BSCs are shown as *black curves*. (**a**) Inner boundary ($r = 1.0$). (**b**) Outer boundary ($r = 2.5$)

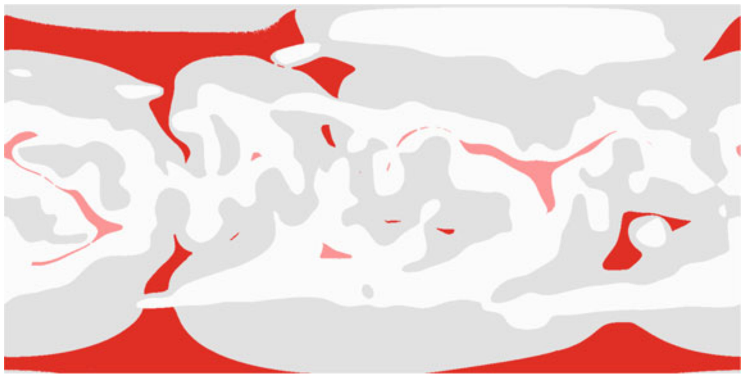


Fig. 2 Escape map from regularly seeded streamlines. *Red regions* are "escape regions" (connecting to the outer sphere)

partitioning. In the time step shown in Fig. 1, the BSCs on the outer boundary are rather simple, consisting of just two connected components, and are inbound BSCs up to a tiny segment (in the upper right corner). We will use the same time step, named Carrington rotation 2128, throughout this paper.

Figure 2 shows the regions of Fig. 1b mapped onto the inner sphere by $\Psi^{\pm}$. Especially the thin structures (which are of particular interest in the underlying research project) are poorly captured by this approach. They can be better resolved by seeding more streamlines, but finally, the exact connectivity of the regions cannot be derived from such images. Antiochos [1] conjectured that disconnected regions cannot exist within a single inflow or outflow region.

2.2 Escape Maps from Full Topological Segmentation

Given the definition of the escape map in the previous section, a straightforward method is to first compute the full topological segmentation of D, and then find all pairs of segments that map onto each other under either Ψ^+or Ψ^-. This requires the extraction of the full set of critical points and periodic orbits, which may be too much of an effort, e.g., if either an "escape boundary" or a "map boundary" [12] has been specified that is a relatively small subset ∂D_i of ∂D. Furthermore, streamsurface tracing can be numerically challenging, which can make this approach unpractical.

2.3 Our Streamtube-Based Approach

We propose a method that is inspired by the sampling approach, but that is topological. Instead of an unorganized set of streamlines, we will use the BSCs as seed curves. BSCs separate inflow from outflow regions and therefore generate *streamtubes* that enclose sets of streamlines growing in the same direction. We use the term streamtube for streamsurfaces with closed seed curves, to be distinguished from stream polygons [18] sometimes also called streamtubes. If the BSCs consist of n simply-connected closed curves, there are $2n$ streamtubes to be traced, because from each such curve we can integrate in backward or forward time. We call these the *stable* and *unstable manifolds* (W^s and W^u), respectively, in analogy to the (un)stable manifolds of saddle points. Tracing these streamtubes will not give the final result (the escape map of a specified set of regions), but it is an essential first step.

If applied to our main example, where there are just two closed curves on the outer boundary, there are four streamtubes to be traced, and the result as seen on the inner boundary is shown in Fig. 3. BSCs are zero-level isolines of the normal velocity component, and are therefore closed lines. Assuming trilinear interpolation, exact BSCs (consisting of hyperbola segments) are easily found using the *Asymptotic Decider* method [13].

As seed points we used the points given by the isoline method plus additional points on the exact isoline, as long as the distance between neighbors at either end of the streamlines exceeds a given threshold. For the discontinuities (shown as the fine lines in Fig. 3) this means that the seed interval is iteratively reduced down to machine precision.[1]

[1]We used 80-bit `long double` arithmetic.

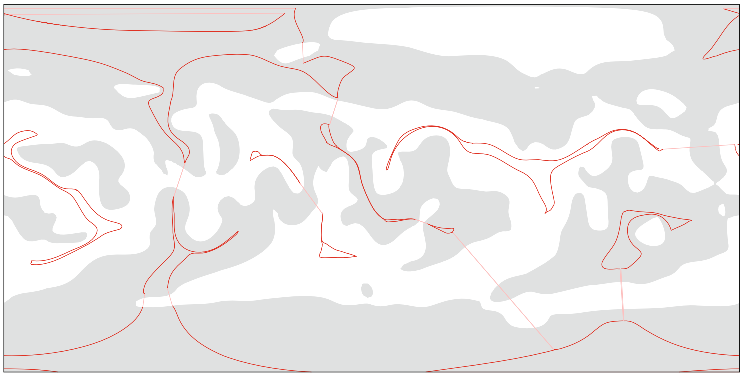


Fig. 3 BSCs of the outer sphere (*black lines* in Fig. 1b) mapped under $\Psi^{\pm}$ to the inner sphere. *Light red lines* mark discontinuities of the map

The mapped BSCs in Fig. 3 reveal more detail than the regions in Fig. 2, and they seem to consist of chains of closed curves. At many places the mapped BSCs (red lines) seem to fully enclose a region and then jump (as a pair of coinciding light red lines) to the next such region. At this stage, two problems still exist. Firstly, the discontinuities could be overestimated, because machine precision is not sufficient to resolve a gap. An illustrative example for this phenomenon is given in Sect. 3. This would mean that the correct region extends beyond what the red lines suggest. Secondly, some of the regions are not yet fully enclosed by curve segments.

There are two possible reasons for discontinuities in the escape map of a BSC. The first one is that in the sequence of streamlines, a streamline can be tangential to ∂D, which is a cause for an abrupt change of topological behavior, see Fig. 4. In this case, the streamline must have hit another BSC, which also means that it is a *boundary switch connector* [20].

The second reason for a discontinuity is a branching that happened because either a saddle point or a saddle-type periodic orbit has been hit. In this case, the 2D (un)stable manifold of the saddle, or one of the 2D manifolds of the periodic orbit, must intersect the seed curve. Similar to the case of the *saddle connector* [19] and the *boundary switch connector*, which are intersections of manifolds of two saddles or two BSCs, respectively, we have here the intersecting manifolds of one BSC and one saddle. As a working term, we will refer to this as a *mixed connector*. This can be seen in Fig. 5. On the entire "trace" of the saddle (the green curve including points P and Q), the escape map is undefined, because streamlines converge to the saddle point. This means that the (black) BSC is punctured into open curve segments which map to disjoint (red) curves. However, points infinitesimally close to the saddle trace are mapped to points infinitesimally close to one end of the saddle's 1D manifolds. By including these limit points, we could also say that closed curves are mapped to pairs of closed curves.

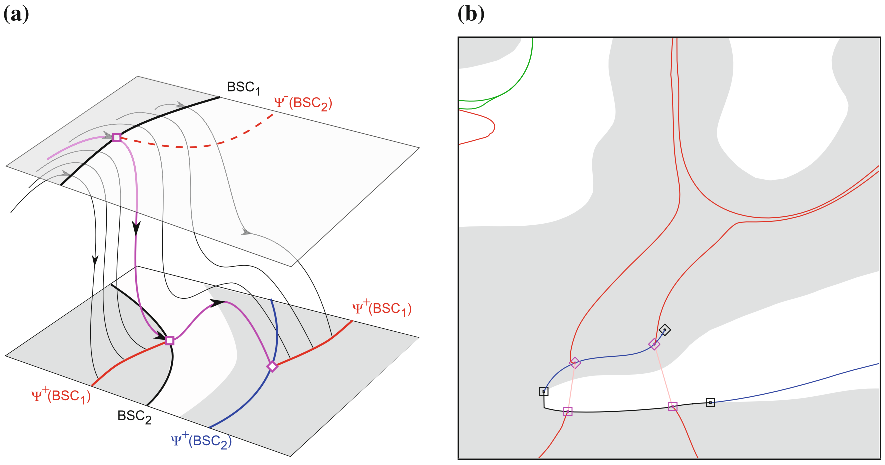


Fig. 4 First kind of discontinuity of a mapped BSC. The (*red solid*) curve hits another BSC and "jumps" over an inflow region. *Color coding* of point types $1/1'$: *black boxes/diamonds*, $2/2'$: *magenta boxes/diamonds*, $3/3'$: *cyan boxes/diamonds*, 4: *green boxes*. (**a**) Sketch of two BSCs whose manifolds intersect in a (*magenta*) boundary switch connector. (**b**) Real data, close-up on inner sphere. Inbound BSCs (*black*), BSCs mapped from outer (*red*) and inner (*blue*) boundary. *Light colors* are used to indicate discontinuities

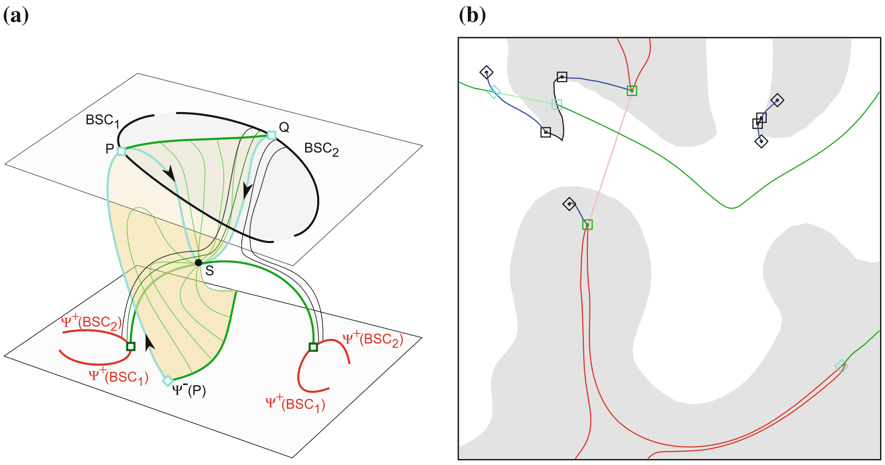


Fig. 5 Second kind of discontinuity of a mapped BSC. The mapped BSC (*red solid curve*) consists of two parts, each of it converging to an endpoint of a 1D manifold (*green boxes*). *P* and *Q* are endpoints of mixed connectors, and the mapped *P* (*cyan diamond*) is an endpoint of the saddle trace (*green curve*). (**a**) Sketch of a BSC whose manifold intersects that of a saddle *S* in a (*cyan*) mixed connector. (**b**) Real data, close-up on inner sphere

We can recapitulate that the (un)stable manifolds of BSC can hit another BSC or a saddle point, which causes discontinuities in their "traces" on ∂D. A similar statement can be made for the other type of 2D manifolds occurring in topological

Table 1 Possible segment types in topological boundary segmentation

	Curve segment		Possible endpoints
A	Inbound BSC segment	1	Inout point
		2	Endpoint of boundary switch connector
		3	Endpoint of mixed connector
B	Mapped inbound BSC segment	4	Endpoint of saddle's 1D manifold
		1'	Mapped inout point (under $\Psi^{\pm}$, only one is defined)
		2'	Mapped type 2 point (under the $\Psi^{\pm}$ not mapping to a type 2 point)
		3'	Mapped type 3 point (under $\Psi^{\pm}$, only one is defined)
		2	Endpoint of boundary switch connector
C	Saddle trace segment[a]	3	Endpoint of mixed connector
		3'	Mapped type 3 point
		4	Endpoint of saddle's 1D manifold

[a]Intersection of 2D (un)stable manifolds of saddles or saddle-type periodic orbits with ∂D

segmentations, namely (un)stable manifolds of saddles (or saddle-type periodic orbits). Such a manifold can also hit a BSC and lead to a discontinuous curve on ∂D. An example can be seen in Fig. 5a, where the (green) saddle trace "jumps" from point P to point $\Psi^{-}(P)$, and it also appears in Fig. 5b (fine green line). The remaining case is a stable 2D manifold of a saddle intersecting the unstable 2D manifold of another saddle. Their intersection curve, the saddle connector, cannot intersect ∂D. However, another property of this configuration is that all 1D manifolds of the two saddles act as limit curves of the 2D manifolds of the other saddle. On ∂D this means that the saddle traces end at the endpoint of the 1D manifolds.

To summarize, the regions of the topological segmentation of ∂D can be bounded by the three types of curve segments and the seven types of points listed in Table 1.

2.4 Escape Map Computation

The steps of our method can now be briefly described. We exclude the case of saddle-type periodic orbits, which we did not encounter in our examples.

1. Extract all saddle points and compute only their 1D manifolds. Endpoints are part of the segmentation. They can either attach to a curve segment later or can otherwise puncture the surrounding region.
2. Compute the BSCs on ∂D and identify their inbound and outbound segments.

3. Trace streamtubes (represented as linked lists of streamlines) seeded at BSCs (represented as sets of simply-connected closed curves). At outbound portions of the seed curve, integration stops immediately, but these are needed to maintain the closed front curve. Additional streamlines are inserted (i.e., seeded at correctly interpolated points on the seed curve) as long as gaps remain at either end of the streamtube.
4. Mark all places on BSCs where seeding had to be refined to the maximum as endpoints of either a boundary switch connector or a mixed connector.
5. Locate the places where streamlines seeded at points from step 4 abruptly diverge. At BSCs the location is easily found, because one of the two diverging streamlines exits D nearby. Also saddle points should be easy to identify, given the precomputed list of saddle points and reasonably precise arithmetic, because streamlines attain a speed minimum near the saddle point. In our two examples, no ambiguities arose in this step.
6. At locations found on BSCs, compute the boundary switch connector (for details see below). Attach its two endpoints to the corresponding curve segments. Map each endpoint with its second $\Psi^{\pm}$ and attach also the resulting points.
7. At saddle points found in step 5, compute the mixed connector (for details see below), and use its endpoint for puncturing the seed curve. Attach endpoints of 1D manifolds to the corresponding curve.
8. Check for all remaining saddles whether the ends of their 1D manifolds lie inside one of the seed curves. In such a case, and unless the other end point lies inside the same seed curve,[2] the trace of the 2D manifold must be computed, too. This trace is then a closed curve lying inside the streamtube. The inside of the trace must be subtracted from the region that is mapped by the streamtube. Two examples of such closed saddle traces can be seen in Fig. 6 (shown as green curves).

With the exception of those required in step 8, it is not necessary to compute 2D manifolds of saddles, unless a full segmentation is desired (Figs. 6, 7). If, as in our main example, only escape maps of inflow and outflow regions are needed, complete extraction of the saddles' 2D manifolds is not required.

Most of the above steps are numerically unproblematic, as these require at most the solution of ODEs. Only steps 6 and 7 are more challenging. While endpoints of the two types of connectors are known from steps 4 and 5, it is not trivial to actually compute the connector. The connector is needed for verification of step 5, which is basically just a proximity-based matching. The problem is ill-posed because these streamlines intersect ∂D tangentially, and it is also difficult to exactly hit a saddle point with a shooting approach. Our solution for the boundary switch connector is to integrate from both endpoints and then verify that the two curves coincide in the

[2]This rule is taken from [12].

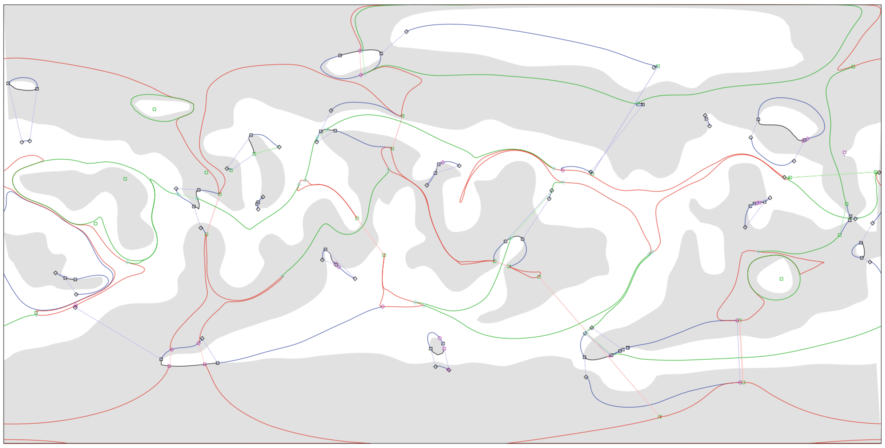


Fig. 6 Full segmentation of the inner boundary ($r = 1.0$). *Color scheme* as in Fig. 4

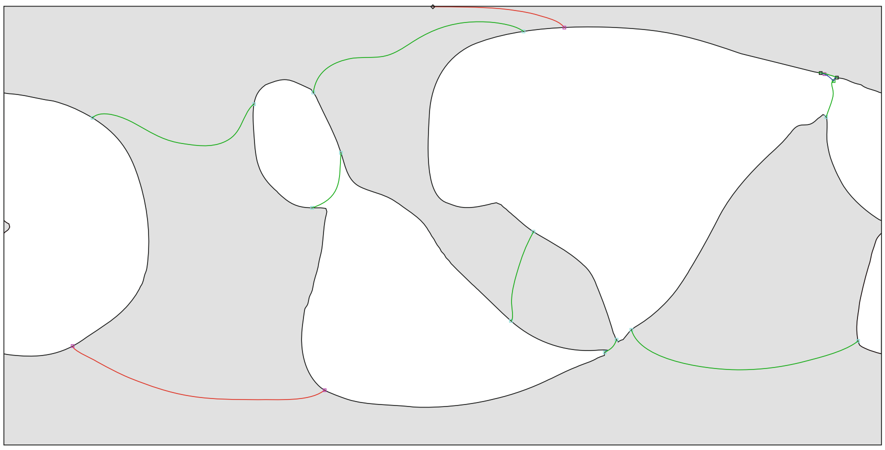


Fig. 7 Full segmentation of the outer boundary ($r = 2.5$)

interior of D. This worked well, even in the synthetic example in Sect. 3, where the direct numerical approach failed completely.

A second problem in steps 6 and 7 is attaching the found endpoints to the corresponding curves (mapped BSCs or saddle traces). Due to exponentially diverging streamlines, gaps can exist, where no more samples can be created by refining the seed curve. In Figs. 6 and 7 such gaps exist but are too small to be seen. In our synthetic example, a large gap occurs (see Fig. 8) between the upper endpoints of the (red) streamlines and the endpoint of the boundary switch connector (a point lying on the black BSC, labeled B' in Fig. 9. A way to create additional samples is to track the discontinuity by integrating streamlines backward. A pair of such streamlines seeded at both sides of the predicted curve (at sufficient distance) keep staying on different sides of the BSC's or saddle's 2D manifold, and so their two endpoints will reflect the discontinuity. Iterative bisection of the distance between the two seed points then results in a point of the curve.

3 Challenges in Streamsurface Tracing

In order to study a pattern observed in an application (coronal holes), we use the following synthetic divergence-free vector field.

$$\mathbf{v} = \begin{pmatrix} u(x,y,z) \\ v(x,y,z) \\ w(x,y,z) \end{pmatrix} = \begin{pmatrix} z - (ay)^2 \\ c \\ x \end{pmatrix}$$

with good parameter choices of $a = \frac{1}{4}$ and $c = \frac{1}{5}$. For $c \neq 0$, the field has neither critical points nor periodic orbits. Domain boundaries are the two parallel planes $z = \pm 1$. If a finite domain is desired, boundaries $x = \pm W$ and $y = \pm L$ may be used in addition, for some large enough values of W and L, such as $W = 3$ and $L = 6$.

The idea of this synthetic field is to move a 2D saddle along a parabola such that it enters the top boundary (at the point $0, -\frac{1}{a}, 1$) and leaves it again (at $0, \frac{1}{a}, 1$). The 2D saddle lies in the xz-plane and has its separatrices in the diagonal. The constant speed c along the y axis is small enough, such that the flow is determined mostly by the saddle, especially at the domain boundaries. On the other hand, it is large enough to guarantee a non-stiff ODE.[3]

While there exists neither a critical point nor a periodic orbit in D, the observed pattern of streamlines (Fig. 8a) resembles the one of a saddle-type periodic orbit. This so-called *streamsurface bifurcation* consists of two streamsurfaces intersecting in a *bifurcation line* [7, 11, 15]. Streamlines of one streamsurface converge exponentially to the bifurcation line, while those of the other diverge exponentially from it. In our example, the bifurcation line can be analytically calculated[4] resulting in the streamline $z = (ay)^2 + 2(ac)^2,\ x = 2a^2cy$. The two associated streamsurfaces are obtained by seeding in the vicinity to this curve. Figure 9 shows that this pair of streamsurfaces provides a good approximation of the escape map.

We will show now that the escape map can be expressed in terms of VFT, despite the fact that VFT cannot express the bifurcation line, which is mainly responsible

[3]The constant speed in y direction can be made more generic by adding a tiny linear term. Then a saddle point exists far outside of the domain.

[4]See "Appendix: Analytic Derivations for the Synthetic Example" for a derivation.

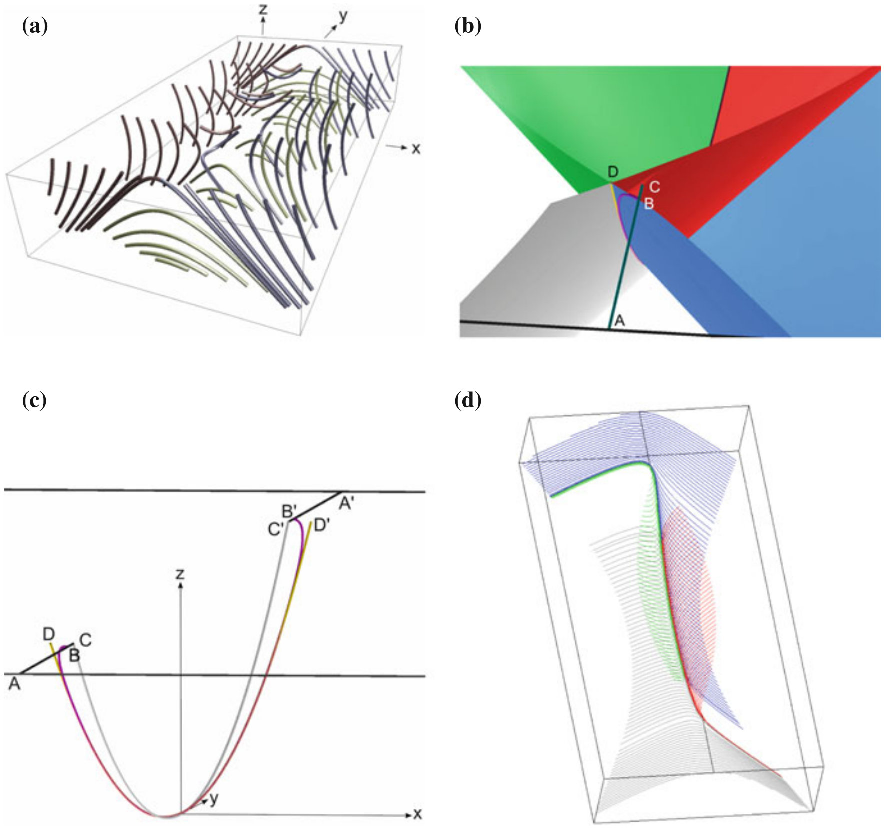


Fig. 8 Synthetic vector field. Point labels are as in Fig. 9. (**a**) Overview of the vector field. (**b**) Manifolds computed from BSC and from bifurcation line. The deviation is visible in the two *red surfaces*, which almost coincide, except near points *B* and *D*. (**c**) Deviation of bifurcation line (*yellow*) from boundary switch connector (*magenta*). Also shown: inbound BSC (*black*), 2D saddles in *xz*-slices (*gray*). (**d**) Gap between *red and gray* (or *blue and green*) streamlines not resolvable within machine precision

for the shape of the escape map. In our field containing neither critical points nor periodic orbits, all topological structure must be due to the BSCs. The most interesting BSC is of course the one that divides the top boundary into an inflow region ($x < 0$) and an outflow region ($x > 0$). A closer inspection of this line reveals that it consists of an outbound segment at $|y| < \frac{1}{a}$ and two inbound segments at $|y| > \frac{1}{a}$. Only inbound segments of BSCs can generate 2D manifolds inside D and thus potentially solve our problem. At first sight it seems now that these BSC segments generate manifolds which also reside mainly in the two outer regions $|y| > \frac{1}{a}$. The manifolds observed in the inner region $|y| < \frac{1}{a}$ would then have to be generated by some other BSC. There are other BSCs on ∂D, but they can be shown to have no effect on the streamline behavior near our region of interest. Also

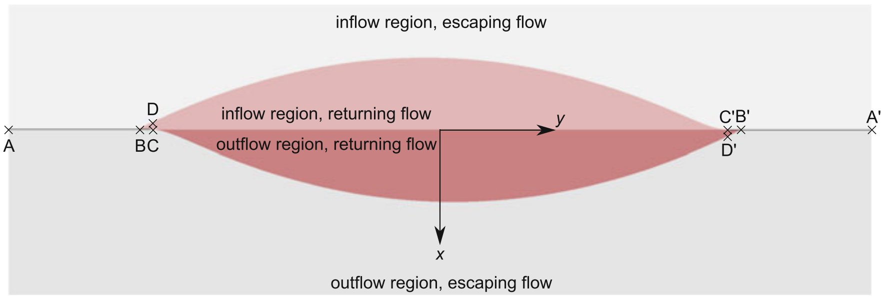

inflow region, escaping flow
D
inflow region, returning flow
y
C'B'
A'
A
BC
outflow region, returning flow
D'
x
outflow region, escaping flow

Fig. 9 The region colored in *red* is the return region (the region mapped to itself by the escape map Ψ. The BSC is $\overline{AA'}$, but its center part $\overline{CC'}$ is an outbound segment. $\Psi^{\pm}$ maps the segments $\overline{AB}$ and $\overline{A'B'}$ to remote parts of ∂D, while the small remaining segments are mapped to the large curves bounding the return region: $\Psi^{+}(\overline{BC}) = \widehat{B'C}$ and $\Psi^{-}(\overline{B'C'}) = \widehat{BC'}$. The bifurcation line has exit points at D and D', hence the return region derived from it would erroneously end at these points, i.e., not extend to B and B'

the boundary of the top face has to be considered, but it has no effect either. Coming back to the BSC ($x = 0, |y| > \frac{1}{a}, z = 1$) it turns out that a small piece of it in fact generates the rather large manifolds in question.

These exponentially diverging streamlines make streamsurface integration difficult,[5] as is shown in Fig. 8d. Even though in our simple vector field, streamlines can be calculated analytically, machine precision does not suffice to resolve values on the BSC near the point $y = -c - \sqrt{(1/a)^2 - c^2}$, which for our choice of parameters is $-\frac{1}{5}(1 + \sqrt{399}) \approx -4.195$. Figure 8d shows the result for 64-bit floating point arithmetic. The gap is reduced to about 20% its size if 80-bit arithmetic is used. This, as a side remark, disproves the quite common practice of parametrizing a streamsurface with the parameter of the seed curve. In order to compute the missing streamlines in between, one would either need arbitrary precision arithmetic or a different type of adaptivity than simply refining the sampling on the seed curve. However, refinement schemes such as Hultquist's method [8] are not an option either. This is illustrated by Fig. 8c, where the boundary switch connector and the bifurcation line (red and yellow streamlines) significantly differ at their two end points, but approach each other to the small distance of $\approx 1.63 \times 10^{-10}$ when they intersect the z-axis. Any attempt to start integration at this point on the boundary switch connector would erroneously lead to the endpoint of the bifurcation line. Correct results are obtained by two-sided streamline integration as described in Sect. 2.4.

[5]We are faced with a stiff problem in the sense that reducing the step size does not help. ODEs in both examples are not stiff, and adaptive RK4 is sufficient. Comparison of `double` and `long double` calculations shows that the former are accurate and just leave a wider unresolved gap.

4 Conclusions

Our proposed method is limited to *divergence-free* vector fields. We assume that streamline integration does neither stagnate at a sink or source nor converge to a periodic orbit. If data are not "sufficiently divergence-free", and in particular contain such sink/source-type features, a divergence-cleaning step must be done beforehand. Furthermore, in the presence of chaos or just strong spiraling at focus saddle points or periodic orbits, the geometric complexity of the mapped regions can be too much to be correctly captured by our technique.

We considered the case of *saddle-type periodic orbits* only in the theoretic analysis, but not in the implemented method. In general, finding periodic orbits is a difficult task. A bounding technique [21] and scalar indicators [9] have been proposed for steering the search. We did not encounter periodic orbits in our examples, but we can assume that our exhaustive seed curve refinement would provide reasonably good detection also of periodic orbits. Thus the search aspect is less important, and our method of choice would be to sample a Poincaré section at the found location of diverging streamlines and get the periodic orbit as its fixed point [14]. The details on how to get the correct segments of their traces and use them for maintaining closed streamtubes would be a subject of future research.

Appendix: Analytic Derivations for the Synthetic Example

By solving ODEs analytically, streamlines can be shown to have the parametric representation

$$\begin{aligned}
y(t) &= y_0 + ct \\
x(t) &= (x_0 - 2a^2cy_0)\cosh(t) + (z_0 - a^2(2c^2 + y_0^2))\sinh(t) + 2a^2cy(t) \\
z(t) &= (x_0 - 2a^2cy_0)\sinh(t) + (z_0 - a^2(2c^2 + y_0^2))\cosh(t) + a^2y(t)^2 + 2a^2c^2.
\end{aligned}$$

An explicit form for the *bifurcation line* is obtained by choosing initial conditions such that all exponential terms (i.e., the hyperbolic functions) vanish, giving the locally slowest streamline in both forward and backward direction. It is the parabola

$$x(y) = 2\,a^2cy, \;\; z(y) = 2\,a^2c^2 + a^2y^2,$$

which has its lowest point at

$$(0, 0, 2a^2c^2).$$

The *boundary switch connector* cannot be easily given in explicit form. It can be represented as the streamline seeded at the point

$$(0, 0, 2\,a^2c^2 - \left(a^2c^2\left(\tau^2 + 2\,\tau + 2\right) - 1\right)\mathrm{e}^{-\tau}),$$

where τ is the solution near 20.97498436 of

$$\left(\frac{\tau^2}{2} + 1 - \frac{1}{2a^2c^2}\right)\tanh(\tau) = \tau.$$

The two curves differ significantly at their two exit points (in the plane $z = 1$), but approach each other to the small distance of $\approx 1.63 \times 10^{-10}$ when they intersect the z-axis. A precise version of this value has been computed using a precision of 100 decimal digits, and seeding forward and backward (analytically computed!) streamlines from this point reach the exit points with an error of less than 10^{-70}.

References

1. Antiochos, S.K., DeVore, C.R., Karpen, J.T., Mikić, Z.: Structure and dynamics of the Sun's open magnetic field. Astrophys. J. **671**(1), 936–946 (2007)
2. Asimov, D.: Notes on the topology of vector fields and flows. Technical Report RNR-93-003. NASA Ames Research Center (1993)
3. Bachthaler, S., Sadlo, F., Weeber, R., Kantorovich, S., Holm, C., Weiskopf, D.: Magnetic flux topology of 2D point dipoles. Comput. Graphics Forum **31**(3), 955–964 (2012)
4. Cai, D.S., Lembege, B., Nishikawa, K.I.: Visualization of tangled vector field topology and global bifurcation of magnetospheric dynamics. In: Usui, H., Omura, Y. (eds.) Advanced Methods for Space Simulations, pp. 145–166. Terrapub, Tokyo (2007)
5. de Leeuw, W., van Liere, R.: Collapsing flow topology using area metrics. In: Proceedings of IEEE Visualization, pp. 149–154. IEEE, New York (1999)
6. Helman, J., Hesselink, L.: Representation and display of vector field topology in fluid flow data sets. IEEE Comput. **22**(8), 27–36 (1989)
7. Hornung, H., Perry, A.E.: Some aspects of three-dimensional separation. Part I. Streamsurface bifurcations. Z. Flugwiss. Weltraumforsch. **8**, 77–87 (1984)
8. Hultquist, J.P.M.: Constructing stream surfaces in steady 3D vector fields. In: Proceedings of IEEE Visualization, pp. 171–178. IEEE, New York (1992)
9. Kasten, J., Reininghaus, J., Reich, W., Scheuermann, G.: Toward the extraction of saddle periodic orbits. In: Bremer, P.-T., Hotz, I., Pascucci, V., Peikert, R. (eds.) Topological Methods in Data Analysis and Visualization III. Mathematics and Visualization, pp. 55–69. Springer, Berlin, Heidelberg (2014)
10. Laramee, R.S., Hauser, H., Zhao, L., Post, F.H.: Topology-based flow visualization, the state of the art. In: Hauser, H., Hagen, H., Theisel, H. (eds.) Topology-Based Methods in Visualization, pp. 1–19. Springer, Berlin, Heidelberg (2007)
11. Machado, G., Sadlo, F., Ertl, T.: Local extraction of bifurcation lines. In: Bronstein, M., Favre, J., Hormann, K. (eds.) Proceedings of Vision, Modeling and Visualization (VMV), pp. 17–24. The Eurographics Association, Goslar (2013)
12. Machado, G., Sadlo, F., Müller, T., Ertl, T.: Escape maps. IEEE Trans. Vis. Comput. Graph. **20**(12), 2604–2613 (2014)

13. Nielson, G.M., Hamann, B.: The asymptotic decider: resolving the ambiguity in marching cubes. In: Proceedings of IEEE Visualization, pp. 83–91 (1991)
14. Peikert, R. Sadlo, F.: Flow topology beyond skeletons: visualization of features in recirculating flow. In: Hege, H.C., Polthier, K., Scheuermann, G. (eds.) Topology-Based Methods in Visualization II, pp. 145–160. Springer, Berlin, Heidelberg (2009)
15. Perry, A.E., Chong, M.S.: A description of eddying motions and flow patterns using critical-point concepts. Annu. Rev. Fluid Mech. **19**(1), 125–155 (1987)
16. Predictive science modeling support for helioseismic and magnetic imager solar dynamics observatory. http://www.predsci.com/hmi/home.php
17. Scheuermann, G., Hamann, B., Joy, K.I., Kollmann, W.: Localizing vector field topology. In: Post, F.H., Nielson, G.M., Bonneau, G.P. (eds.) Data Visualization. The Springer International Series in Engineering and Computer Science, vol. 713, pp. 19–35. Springer US, New York (2003)
18. Schroeder, W.J., Volpe, C.R., Lorensen, W.E.: The stream polygon: a technique for 3D vector field visualization. In: Proceedings of IEEE Visualization, pp. 126–132. IEEE, New York (1991)
19. Theisel, H., Weinkauf, T., Hege, H.C., Seidel, H.P.: Saddle connectors – an approach to visualizing the topological skeleton of complex 3D vector fields. In: Proceedings of IEEE Visualization, pp. 225–232. IEEE, New York (2003)
20. Weinkauf, T., Theisel, H., Hege, H.C., Seidel, H.P.: Boundary switch connectors for topological visualization of complex 3D vector fields. In: Proceedings of Symposium on Data Visualisation 2004 (VISSYM ’04), pp. 183–192. The Eurographics Association, Goslar (2004)
21. Wischgoll, T., Scheuermann, G.: Locating closed streamlines in 3D vector fields. In: Proceedings of Symposium on Data Visualisation 2002 (VISSYM ’02), pp. 227–232. The Eurographics Association, Goslar (2002)

Compute and Visualize Discontinuity Among Neighboring Integral Curves of 2D Vector Fields

Lei Zhang, Robert S. Laramee, David Thompson, Adrian Sescu, and Guoning Chen

Abstract This paper studies the discontinuity in the behavior of neighboring integral curves. The discontinuity is measured by a number of selected attributes of integral curves. A variety of attribute fields are defined. The attribute value at any given spatio-temporal point in these fields is assigned by the attribute of the integral curve that passes through this point. This encodes the global behavior of integral curves into a number of scalar fields in an Eulerian fashion, which differs from the previous pathline attribute approach that focuses on the discrete representation of individual pathlines. With this representation, the discontinuity of the integral curve behavior now corresponds to locations in the derived fields where the attribute values have sharp gradients. We show that based on the selected attributes, the extracted discontinuity from the corresponding attribute fields may relate to a number of flow features, such as LCS, vortices, and cusp-like seeding curves. In addition, we study the correlations among different attributes via their pairwise scatter plots. We also study the behavior of the combined attribute fields to understand the spatial correlation that cannot be revealed by the scatter plots. Finally, we integrate our attribute field computation and their discontinuity detection into an interactive system to guide the exploration of various 2D flows.

1 Introduction

Vector field analysis is a ubiquitous approach that is employed to study a wide range of dynamical systems for applications such as automobile and aircraft engineering, climate study, and earthquake engineering, among others. There is a large body of

L. Zhang • G. Chen (✉)
University of Houston, Houston, TX, USA
e-mail: lzhang38@uh.edu; gchen16@uh.edu

R.S. Laramee
Swansea University, Wales, UK
e-mail: rlaramee@gmail.com

D. Thompson • A. Sescu
Mississippi State University, Starkville, MS, USA
e-mail: dst@cavs.msstate.edu; asescu@cavs.msstate.edu

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_11

work on generating reduced representations to aid the understanding of complex and large-scale flow data sets, by classifying integral curves based on their individual attributes. These methods typically first classify the integral curves into different clusters based on a similarity measure [9], then compute representative curves for each cluster [24]. Due to the discrete representation of these methods, there is no guarantee that important flow features will be captured.

In this paper, we introduce a number of attribute fields that encode global behaviors of the individual integral curves measured by certain geometric and physical properties. The scalar attribute value at each spatio-temporal position is derived from the attribute value of the integral curve that passes through it. With this Eulerian representation, spatio-temporal positions correlated by the same integral curves will have similar attribute values, while those neighboring points traversed by integral curves that possess different behavior will have different attribute values. Figure 1 provides a number of example attribute fields. The 1D plots (Fig. 1 right) show the attribute values along the seeding line segments (i.e., the red segments in Fig. 1 left). They exhibit some cliff-like sharp changes (highlighted by the blue arrows), which correspond to certain *discontinuities* in the corresponding attribute field. This discontinuity may be closely related to certain flow features, such as flow separation, as shown in Fig. 1a.

We consider a number of scalar attributes as discussed by Pobitzer et al. [12] and Shi et al. [18]. To study the correlation between the attribute fields generated from these attributes, we compute their pairwise scatter plots (Fig. 5). Our results indicate that some attribute fields are highly correlated. Therefore, the set of attribute fields can be reduced. This echoes the results presented by Pobitzer et al. [12].

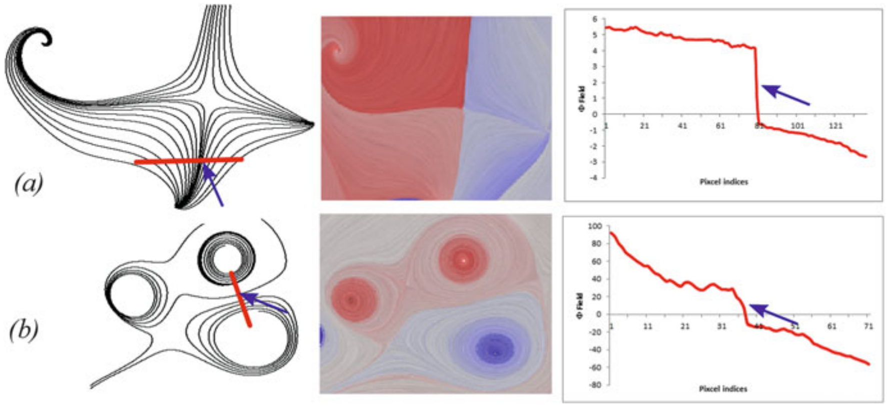


Fig. 1 The illustration of the relation between the attribute field and a number of well-known flow features, including the flow separation (**a**) and vortices (**b**). The *left column* shows the vector fields illustrated by streamlines, the *middle column* shows the rotation field, while *right column* shows the plots of the rotation field of the streamlines intersecting with given seeding line segments (shown in *red*). Note that the discontinuities (sharp gradients) in the rotation field indicate the flow features

More importantly, we find that the magnitude of the gradient of the attribute field that measures the rate of change between neighboring positions exhibits strong correlation with the FTLE fields of the same flows. This coincides with the results reported by Shi et al. [17]. Furthermore, we compute a *super attribute field* that combines multiple attribute fields to study their spatial correlation. That is, we see whether two attribute fields have similar configurations (e.g., local extrema, ascending and descending trends) at the same spatial positions.

Finally, we integrate the attribute field computation and the discontinuity detection into an interactive system to support the exploration of various flow behavior via the visualization of the discontinuity structure of a chosen attribute field or the derived super attribute field. We have applied our framework to a number of synthetic and real-world 2D vector fields to demonstrate its utility.

2 Related Work

There is a large body of literature on the analysis and visualization of flow data. Interested readers are encouraged to refer to recent surveys [4, 11] that provide systematic classifications of various analysis and visualization techniques.

Among many vector field analysis techniques, vector field topology is a powerful tool that provides a flow segmentation strategy based on the origin and destination of individual streamlines. Since its introduction to the visualization community [7], vector field topology has received extensive attention for the identification of different topological features [2, 19, 23]. Recently, Morse decomposition [3] and combinatorial vector field topology [13] have also been introduced for a more stable construction and representation of vector field topology. The theory and computation of vector field topology does not apply to unsteady flows. Users typically opt for the identification of *Lagrangian Coherent structures (LCS)*, i.e., curves (2D) or surfaces (3D) in the domain across which the flux is negligible, as an alternative. LCS can be extracted as the ridges of the *Finite Time Lyapunov Exponent (FTLE)* of the flow [5, 16]. Similar to these conventional flow structure analysis techniques, our method also aims to reveal certain flow structures that indicate the boundaries of individual flow regions with different behaviors as described by specific integral curve attributes.

Salzbrunn and Scheuermann introduced *streamline predicates* that classify streamlines by interrogating them as they pass through certain user-specified features, e.g., vortices [14]. Later, this approach was extended to classify pathlines [15]. At the same time, Shi et al. presented a data exploration system to study different characteristics of pathlines based on their various attributes [18]. Pobitzer et al. demonstrated how to choose a representative set of pathline attributes for flow data exploration based on a statistics-based dimension reduction method [12]. McLoughlin et al. [10] introduced the streamline signature, based on a set of curve-based attributes, to guide the effective seeding of 3D streamlines. In contrast to the above described integral curve classification and selection techniques that treat the individual integral curves as discrete units, our method derives a number of scalar fields throughout the entire flow domain to encode the global behaviors of integral curves. This allows us to study the discontinuity in the behaviors of neighboring integral curves via some well-established edge detection techniques, such as the Canny edge detector [1] and a discrete gradient operator.

3 Vector Field Background and Trajectory Attributes

Consider a 2-manifold $\mathbb{M} \subset \mathbb{R}^2$. A vector field can be expressed as an ordinary differential equation (ODE) $\dot{\mathbf{x}} = V(\mathbf{x}, t)$ or a map $\varphi : \mathbb{R} \times \mathbb{M} \to \mathbb{R}^2$. There are a number of curve descriptors that depict different aspects of the translational property in vector fields.

- A *streamline* is a solution to the initial value problem of the ODE system confined to a given time t_0: $\mathbf{x}_{t_0}(t) = \mathbf{p}_0 + \int_{t_0}^{t} V(\mathbf{x}(\eta); t_0)d\eta$.
- *Pathlines* are the paths of the massless particles released in the flow domain at a given time t_0: $\mathbf{x}(t) = \mathbf{p}_0 + \int_{t_0}^{t} V(\mathbf{x}(\eta); \eta)d\eta$.
- A *streakline*, $\tilde{\mathbf{s}}(t)$, is the connection of the current positions of the particles, $\mathbf{p}_{t_i}(t)$, that are released from position p_0 at consecutive time t_i.

There are a number of features in *steady flows*, $V(\mathbf{x})$, that are of interest. A point $\mathbf{x}_0$ is a *fixed point* (or *singularity*) if $V(\mathbf{x}_0) = 0$, and a trajectory is a *periodic orbit* if it is closed. Hyperbolic fixed points, periodic orbits and their connectivity define the *vector field topology* [2]. *Vortices* are another important flow feature that is of interest to domain experts. There is no unified definition for vortices. Informally, a vortex is a region where the flow particles are rotating around a common axis (reduced to a point in 2D). In this work, we consider streamlines with larger winding angles than a user-specified threshold, say 2π, to be within vortices. In unsteady flows, topology is not well-defined. One typically looks for certain *coherent structures* that correspond to structures in the flow that are present for a *relatively* long time. The LCS, i.e., the ridges of the FTLE field, is one such coherent structure [5]. Another feature is singularity path that depicts the trajectory of a singularity in an unsteady flow [21].

3.1 Attribute Fields

Consider an integral curve, $\mathscr{C}$, starting from a given spatio-temporal point $(\mathbf{x}, t_0)$, the attribute field value at this point is computed as:

$$\mathscr{F}(\mathbf{x}, t_0) = \mathscr{F}(\mathscr{C}(\mathbf{x})|_{t_0}^{t_0+T}) \tag{1}$$

where $\mathscr{C}(\mathbf{x})|_{t_0}^{t_0+T}$ denotes an integral curve, i.e., either a streamline, a pathline, or a streakline starting at time t_0 with an integral time window $[t_0, t_0+T]$. $\mathscr{F}(\cdot)$ indicates certain attribute of interest of $\mathscr{C}$. Note that, for the rest of the paper, we consider only forward integration of the integral curves. Backward integration can be considered similarly. In practice, an integral curve $\mathscr{C}$ is represented by N integration points P_i and $(N-1)$ line segments (P_i, P_{i+1}). We then define a number of attribute fields based on Eq. (1) using the integral curves attributes discussed in the work [12, 18].

The attribute fields we investigated in this paper with their abbreviation are rotation field Φ, length field Ł, average particle velocity field $avgV$, non straight velocity field nsV, relative start end distance field $seDist$, average direction field $avgDir$, acceleration $acce$, curl, Hunt's Q and λ_2 [12]. Figure 3 illustrates a number of attribute fields derived from the Double Gyre flow [16]. Specifically, the rotation field Φ describes the accumulated winding angle changes along the integral curves, which is defined as $\Phi_{\mathscr{C}} = \sum_{i=1}^{N-1} d\theta_i$, where $d\theta_i = (\angle(\overrightarrow{P_iP_{i+1}}, \overrightarrow{X}) - \angle(\overrightarrow{P_{i-1}P_i}, \overrightarrow{X})) \in (-\pi, \pi]$ represents the angle difference between two consecutive line segments. $\overrightarrow{X}$ is the X axis of the XY Cartesian space. $d\theta_i > 0$ if the vector field at P_i is rotating counter-clockwise with respect to the vector field at P_{i-1}, while $d\theta_i < 0$ if the rotation is clockwise. Figure 1 shows a number of example Φ fields. The computation of the other attribute fields can be performed using a similar accumulation process based on their definitions [10, 16].

3.1.1 2D and 3D Attribute Fields

If the attribute field is computed based on streamlines, it is a 2D field. Figure 4b shows the rotation field of a synthetic steady flow based on streamlines. To visualize the attribute fields, we utilize a blue-white-red color coding scheme with blue corresponding to negative values and red for positive values. Pathlines or streaklines-based attribute fields are 3D fields. That is, given any spatio-temporal position $(\mathbf{x}, t_0)$, its attribute value is determined by the pathline (or streakline) starting from this position and following the forward flow direction (Eq. (1)). Figure 2 (upper) shows a volume rendering of the pathline-based Φ field of the Double Gyre flow [16] within the time range $[0, 10]$. For the rest of the paper, we focus on the behaviors of the attribute fields at specific time steps, i.e., the cross sections of the 3D field (Fig. 2 (bottom)). Figure 3 shows a number of attribute fields of the Double Gyre flow. Note that the average direction field $avgDir$ (Fig. 3b) measures the angle between the vector pointing from the starting point to the end point of an integral curve and the X axis. The range of the $avgDir$ field is $[0, 2\pi)$. The pathline tracing starts at $t = 0$ with an integral time window size $T = 10$. Figure 3e shows a Φ field computed based on the streaklines. Figure 3f shows the Φ field obtained using the backwardly traced pathlines.

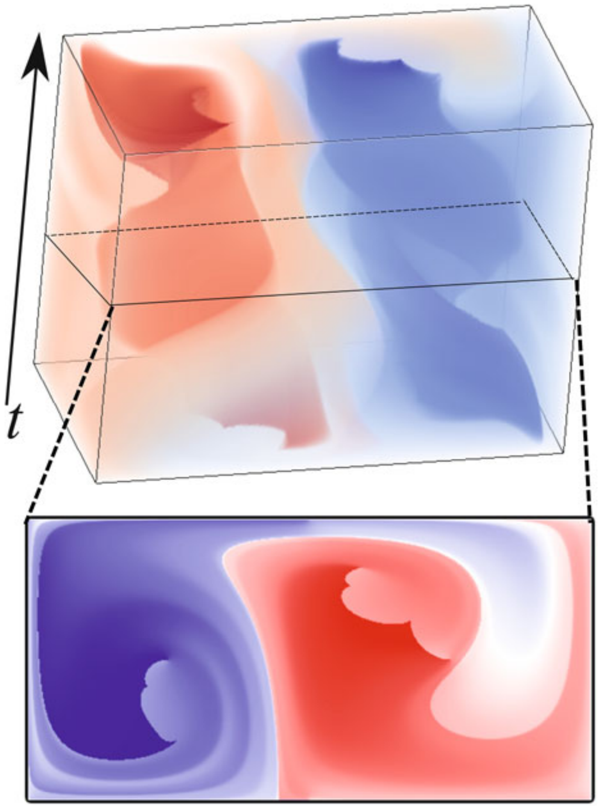


Fig. 2 The volume rendering (*upper*) of the pathline-based Φ field of the Double Gyre flow. The *bottom* shows one slice at $t = 5$

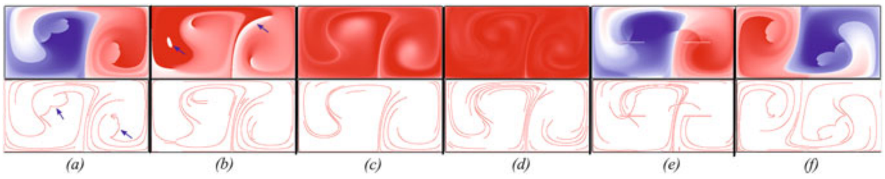


Fig. 3 Illustration of a number of attribute fields derived from the Double Gyre flow and their detected edges. (**a**)–(**d**) show the attribute fields Φ, Ł, $avgDir$ and *acceration*, $avgDir$ computed from pathlines, respectively. (**e**) is the rotation field Φ from streaklines. (**f**) is the rotation field Φ from pathlines using backward integration. The parameters of Canny edge detector are $\sigma = 2.0$, $\alpha = 0.3$, $\beta = 0.8$

3.2 *Discontinuity in Attribute Fields*

3.2.1 Discontinuity Extraction

As shown in Fig. 1, the attribute fields may contain discontinuities that correspond to the sharp gradients in the integral curve behavior. These discontinuities in the attribute fields are similar to the edges in a digital image. The gradient of the attribute field may be able to locate these discontinuities (Fig. 4f), but may require non-intuitive thresholds to reveal the salient ridges. Therefore, we opt for the more robust Canny edge detector [1] to locate this discontinuity in the attribute fields, which can be converted into 2D images. The Canny edge detector has three input parameters: σ—the standard deviation of the Gaussian smoothing filter, α—the low threshold and β—the high threshold. The lower row of images in Fig. 3 shows the detected edges from the corresponding attribute fields of the Double Gyre flow. Note that for the average direction field $avgDir$, the field values of the two neighboring pathlines may be (or be close to) 0 and 2π respectively, as highlighted with the arrows in Fig. 3b. But it does not indicate the discontinuity because they are in the same (or be close to) direction. Therefore, this case is filtered out in the Canny edge detector.

3.2.2 Relation to Flow Features

Steady Flow Features Many discontinuities (i.e., edges identified by the edge detector) of these attribute fields share characteristics with certain well-known flow features. For example, Fig. 4 compares the discontinuity structure of the rotation field Φ of a synthetic steady flow to its topology (Fig. 4a), which is illustrated via a set of integral curves that end or start from saddles, i.e., *separatrices*—a special type of streamline. The Φ field is not continuous across the separatrices if the accumulation is performed using an infinite time window. This is because an arbitrarily small perturbation in the direction other than the flow direction will result in another integral curve with a length much different from the separatrix,

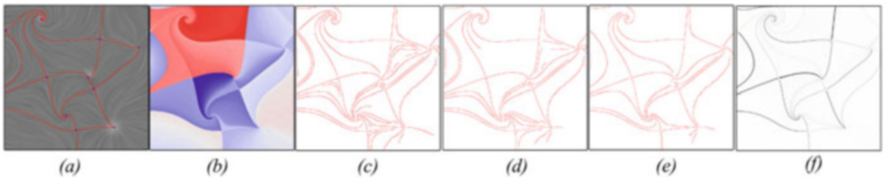


Fig. 4 Discontinuity detection on a Φ field derived from a synthetic steady flow using the Canny edge detector with different combinations of parameters. (**a**) The differential topology with LIC as the background; (**b**) Φ field; ((**c**)–(**e**)) Detected edges with different parameters of the Canny edge detector: (**c**)—$\sigma = 3.0$ $\alpha = 0.3$ $\beta = 0.8$, (**d**)—$\sigma = 3.0$ $\alpha = 0.6$ $\beta = 0.8$, (**e**)—$\sigma = 3.0$ $\alpha = 0.3$ $\beta = 0.86$; (**f**) the gradient of Φ field

making the Φ field accumulated using Eq. (1) discontinuous at separatrices. With different parameters, different levels of detail of the discontinuity in the Φ field can be revealed (Fig. 4c–e). Figure 4f shows the gradient of the Φ field, which does not provide a clean discontinuity structure.

LCS Lagrangian Coherent Structures (LCS) are defined as the ridges of the corresponding FTLE field. It indicates the regions of the domain with relatively large separation. Compared to LCS, it appears that the edges detected from all the attribute fields of the Double Gyre flow encode at least part of this information. This is also true for the other data sets that we have investigated (i.e., Figs. 10 and 11). The discontinuity may be observed at the ridges of transportation, i.e., LCS due to a similar reason to the separatrices in steady flow. A pathline seeded on the ridges may have behaviors different from its neighboring pathlines caused by the separation, leading to the discontinuity in the attribute fields.

Cusp Seeding Curves The cusp seeding curve has been discussed in [22] to reduce self-intersecting pathlines in the pathline placement. These cusp seeding curves of the Double Gyre flow can be identified from the discontinuities in the rotation field Φ as shown in Fig. 3a. This cusp-like behavior in pathlines is caused by the abrupt change in the pathline direction, i.e., almost angle of π difference between the previous and current directions, which is in turn caused by the intersection of the pathlines with the paths of singularities.

Singularity Path Singularity paths reveal the trajectories of fixed points in an unsteady flow. Among all the attribute fields studied, only the Φ field computed based on streaklines encodes such information. See Fig. 3e for an example where the paths of the two vortices of the Double Gyre flow are revealed by the edges detected from the streakline-based Φ field. This is because singularity paths will induce the cusp-like behavior in pathlines (Fig. 12), also discussed in [22]. This cusp-like behavior corresponds to a large local angle change, which in turn leads to a large change, i.e., discontinuity in the Φ field. In addition, the temporal behavior, i.e., the translation of the singularities can only be captured by measuring the attributes of particles released at the same position at consecutive times, i.e., streaklines.

4 Correlation Among Different Attribute Fields

Considering the large number of attributes that can be used to describe various flow behaviors, it will be interesting to see how their corresponding attribute fields are correlated. In this section, we study their correlation using two approaches, i.e., a correlation study via their pairwise scatter plots and a spatial correlation study via combinations of attribute fields.

4.1 Correlation Study via Pairwise Scatter Plots

There are different attributes that can be used to characterize the behaviors of integral curves, as discussed in [12]. To understand how well they correlate with one another, we construct a scatter plot matrix based on the Double Gyre flow, as shown in Fig.5. Each of the entries of this matrix shows a scatter plot with two attributes as its X and Y axes. Based on this matrix, we find the following useful relations.

Length Field Ł vs. Average Particle Velocity Field *avgV* These two attributes show a strong linear relationship (entry highlighted by the purple box in the matrix). This is because the arc-length of each integral curve is equal to the sum of the velocity magnitude, multiplied by the integration step-size, measured along this curve.

Φ* vs. Curl vs. λ_2 vs. *Q These four attributes are also closely related, as they all measure the accumulation of the amount of local flow rotation along integral curves. While *Q* value is always negative, the other attributes can be both positive and negative. The patterns shown in the plots *w.r.t Q* (i.e., row *Q*) are generally very clear with little noise, which indicates *Q* could be a good attribute to be considered for this data set. All the plots *w.r.t Φ* (row *Φ*) and curl (row curl) exhibit certain symmetric patterns. Between these two, the plots associated with *Φ* tend to reveal cleaner patterns with less noise. This indicates that *Φ* field may be an

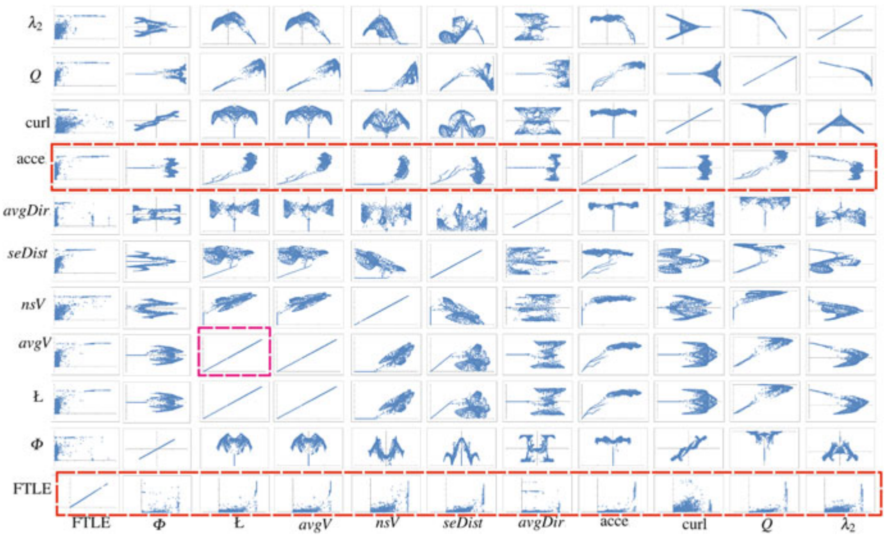


Fig. 5 The scatter plot matrix of different attribute fields of the Double Gyre flow. Note that the scatter plots associated with FTLE shows the correlation among the magnitude of the gradient of the individual attribute fields with the FTLE field

important attribute field that encodes different flow information for subsequent data exploration.

***FTLE* vs. the Gradient of the Attribute Field** The strong correlation between the gradient of the attribute fields with the *FTLE* field is illustrated in scatter plots (row *FTLE*), as both the *FTLE* field and the gradient operator measure the amount of change between neighboring positions. In fact it supports the discussion of the relation between the discontinuity in the attribute fields and the *FTLE* structure in Sect. 3.2.2 and the results of [17], i.e., the transportation structure of certain materials (e.g., some flow attributes) matches closely with the FTLE structure.

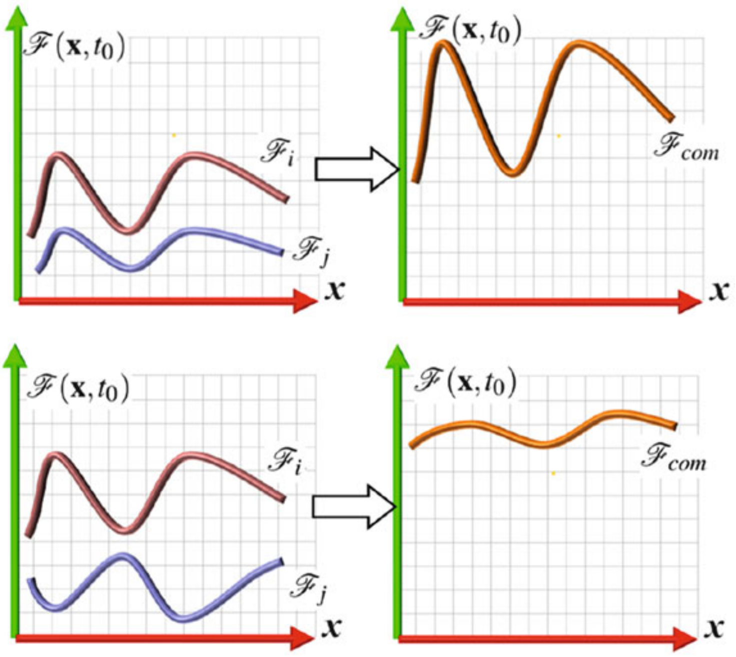
Acceleration Field *aace* vs. Other Attribute Fields The scatter plots of the acceleration field, which is computed by integrating the acceleration magnitude along a set pathlines, and the remaining attribute fields (raw *acce*) generally display clear patterns. In particular, when the value of the *acce* field is small, the other attributes tend to be small. When the value of the *acce* field is increasing and becomes sufficiently large, the other attribute values tend to large values as well. This is consistent with the knowledge that the acceleration, a result of the external force based on Newton Second Law, is the source of many different flow behaviors, such as flow separation and rotation. However, this relation is not true between *acce* and λ_2 or Q. That is, the smaller the *acce* value, the larger the absolute values of λ_2 and Q. This in fact matches the result of the work [8] that utilizes the local minima of the acceleration field to detect vortex cores.

Note that all the above discussions are based on the experiments with the Double Gyre flow. In the future, we plan to further validate them with other flow datasets.

4.2 Spatial Correlation via Combined Attribute Fields

To study the spatial correlation of different attribute fields, i.e., whether they have local maxima or minima at similar locations, or whether they have similar discontinuities at the same location, we need to perform certain spatial correlation analysis. This information cannot be easily obtained from the above scatter plots. Therefore, in the following we combine certain attribute fields of interest to define a *super attribute field*. We hope to obtain information about the spatial correlation of the selected attribute fields by studying the behavior of their combined field. Figure 6 provides some simple 1D examples to illustrate the logic behind the strategy of combined attribute field. If the selected attribute fields have similar behaviors (Fig. 6 upper left), the combined attribute field amplifies the similar behaviors (Fig. 6 upper right). While if selected attribute fields have different behaviors (Fig. 6 bottom left), the combined attribute weakens this difference. Here we combine only two attribute fields at a time, as combining more than two fields will complicate the study of their pairwise correlation.

Another reason we opt for the study of the combined attribute fields is to understand the behavior of the discontinuities in different attribute fields. To achieve



$\mathscr{F}(\mathbf{x}, t_0)$
$\mathscr{F}_i$
$\mathscr{F}_j$
x
$\mathscr{F}(\mathbf{x}, t_0)$
$\mathscr{F}_{com}$
x
$\mathscr{F}(\mathbf{x}, t_0)$
$\mathscr{F}_i$
$\mathscr{F}_j$
x
$\mathscr{F}(\mathbf{x}, t_0)$
$\mathscr{F}_{com}$
x

Fig. 6 The combined attribute field may strengthen the difference if the selected attributes have similar spatial behavior (*upper*) and vice versa (*bottom*)

that, one can simply overlap the detected edges from different attribute fields, as shown in Fig. 7a. However, the detected edges from the individual fields are independent of each other. With this simple overlapping, it is difficult to know whether their corresponding attribute fields have similar behavior or not (i.e., both are descending, or one is descending while the other is ascending) at the locations that exhibit sharp change. This information may be revealed in the combined attribute field.

Assume $\mathscr{F}_i, i = 1, 2, \ldots, n$ represent the attribute fields introduced in Sect. 3.1. We study three combination strategies to compute a super attribute field $\mathscr{F}_{com}$.

Linear combination is defined as $\mathscr{F}_{com} = \mathscr{F}_i + \mathscr{F}_j$, where $\mathscr{F}_i$ and $\mathscr{F}_j$ are selected attribute fields from the attribute fields pool. However, if one of the selected attribute fields has a much larger value range, the super field will be dominated by this attribute field. Figure 7b is the result of combined super field from the rotation field Φ ($[-11.73, 11.73]$) and the length field Ł ($[0., 2.8]$), which shows mostly the features of the rotation field.

Weighted combination is employed to address the issue of the simple combination. Here, $\mathscr{F}_{com} = \alpha\widehat{\mathscr{F}_i} + \beta\widehat{\mathscr{F}_j}$, where $\alpha + \beta = 1$ and satisfies $0 \leq \alpha \leq 1, 0 \leq \beta \leq 1$. $\widehat{\mathscr{F}_i}$ and $\widehat{\mathscr{F}_j}$ are the normalized values of the attribute field $\mathscr{F}_i$ and $\mathscr{F}_j$, respectively. Figure 8a shows the super fields computed using the weighted

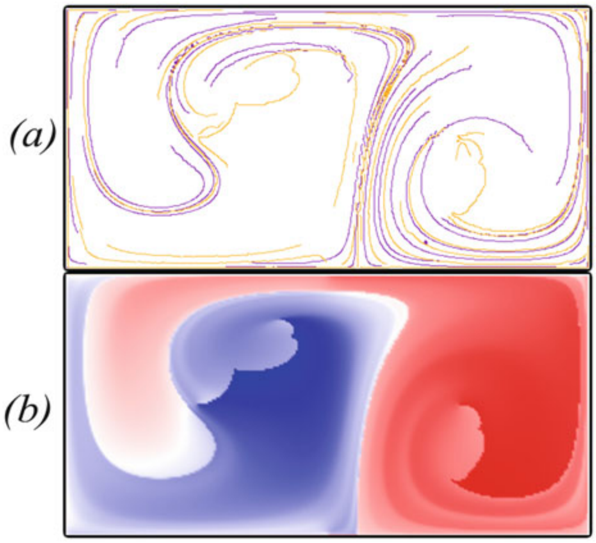


Fig. 7 (**a**) Overlap of edges in Φ (*yellow*) and Ł (*purple*) fields; (**b**) simple combination of Φ and Ł fields

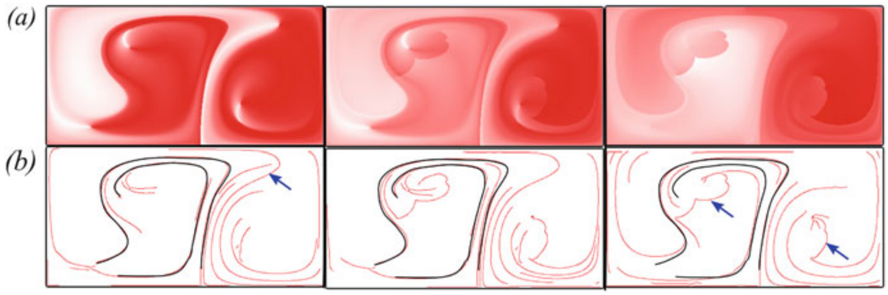


Fig. 8 Illustration of weighted combination of Φ and $avgDir$ field. (**a**) Combined attribute field; (**b**) edges detected from the super field. The *dark gray curves* on top are stable edges that do not change with weights. The weights of Φ and $avgDir$ from let to right are $\alpha_1 = 0.1, \alpha_2 = 0.9$; $\alpha_1 = 0.5, \alpha_2 = 0.5$; $\alpha_1 = 0.9, \alpha_2 = 0.1$, respectively. The parameters of Canny edge detector are $\sigma = 1.0, \alpha = 0.3, \beta = 0.8$

combination of the Φ and $avgDir$ fields of the Double Gyre flow, with the weight for the Φ field being 0.1, 0.5, and 0.9, respectively. With this weighted combination, we can further identify the discontinuity structure in the super field that is non-sensitive to the choices of weights. That is, no matter what weight combination is selected, the derived super field always contains this discontinuity, which is composed of *stable edges*. Figure 8b shows these stable edges as gray curves super-imposed onto the edges extracted from the corresponding super field. In this example, nine super fields, in which the weight of the Φ field is $\alpha_1 = 0.1, 0.2, \ldots, 0.9$, respectively, were generated to identify the stable edge.

Different from stable edges, the *common edges* indicate those edges that are exhibited by most of the attribute fields, as shown in Fig. 10c. Complementary to the common edges, the *unique edges* only arise in certain attribute fields. The arrow in Fig. 8b highlights the unique edges only possessed by the $avgDir$ field and the rotation field respectively.

5 System Overview and Implementation

Figure 9 shows the framework of our attribute fields based flow structure exploration. This framework consists of two main processes.

Precomputation Given the input vector field, we first compute integral curves from the sampled positions forwardly and backwardly with a user-specified time window T. The integral curves are stored as a series of spatio-temporal points. We then compute the local attributes at each spatio-temporal point along the integral curves. The attributes of these integral curves are accumulated from the local attributes and assigned to their corresponding starting points. This step can be preprocessed.

Interaction With the above pre-computed results, the user can choose to inspect the flow structure revealed by the discontinuities of a single attribute field of interest. Changing the parameters of the Canny edge detector reveals different levels of details of the structure (Fig. 4b–d). The user can also choose from the list of the available attribute fields and the desired combination scheme to compute a super attribute field to study the correlation and combination of different attribute fields. Again, the Canny edge detector can be applied to reveal the discontinuity structure in the obtained super field (Figs. 7 and 8).

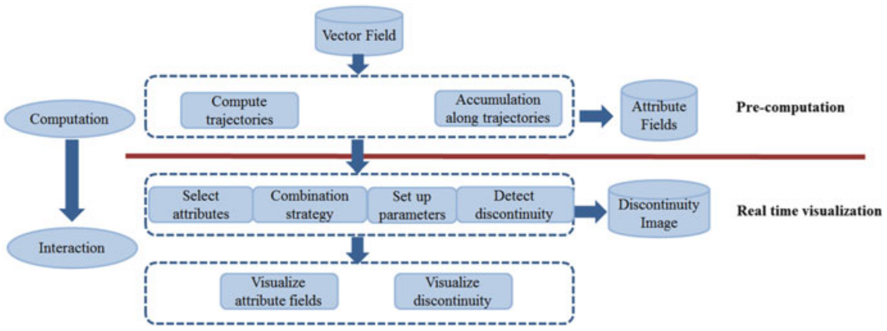


Fig. 9 Pipeline of attribute fields computation and the discontinuity detection. Attribute fields are pre-computed, while the discontinuity is detected in real time during the interactions

6 Results and Applications

We have applied our attribute field based analysis and exploration framework to a number of synthetic and real-world 2D vector fields. The cost of pre-computation of attribute fields depends on the resolution of the spatio-temporal domain and the time window of integral curve integration. Pathline-based attribute field computation requires 10–32 s for the data sets considered in this paper, while streakline-based attribute field computation requires about 4–15 min. All processing times are measured on a PC with an Intel Core i7-3537U CPU and 8 GB RAM.

The first example is the Double Gyre flow with a spatial resolution of 256×128, which has been shown earlier. For the second example, we consider a dynamical system defined by the forced-damped Duffing oscillator [6] with the constant spatial divergence operator -0.25, which is a non-area-preserving. We choose a spatial resolution of 800×600 and a time window $T = 5$. The attribute fields of the system and the corresponding detected edges are shown in Fig. 10a. The detected edges from each attribute field encode the LCS information. Figure 10c upper shows a super field generated from an equally weighted combination of all six attribute fields. Figure 10c bottom illustrates the detected common edges.

Another example is a simulation of a 2D unsteady flow behind a square cylinder with a Reynolds number of 160 [20]. We use a spatial resolution of 400×50 to compute the attribute fields. The time window for this data set is 3. Figure 11 shows the attribute fields and the corresponding detected edges. While the edges detected in all of the attribute fields encode at least part of the LCS of the flow, the non straight velocity field nsV (Fig. 11b) also reveals the swirling behavior of the flow clearly.

Our approach has the potential to reveal the cusps in the spatial projection of pathlines and streaklines [22]. Figure 12a shows the spatial projection of a set of pathlines seeded on the cusp seeding curve detected from the Φ field, while Fig. 12b are streaklines seeded on the singularity path extracted from a streakline-based Φ field. Both the pathlines and streaklines show cusp-like characteristics. Interestingly, the locations of cusp-like characteristic on the sample pathlines reveal the singularity path (the green dashed line in Fig. 12a), while those on streaklines indicate the cusp seeding curve (the green dashed curve in Fig. 12b). This attribute

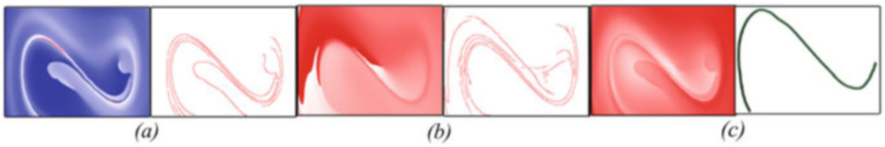


Fig. 10 Results of the forced-damped Duffing system. (**a**)–(**b**) Φ and $avgDir$ fields and their detected edges.The parameters of Canny edge detector are $\sigma = 1.8$, $\alpha = 0.3$, $\beta = 0.9$. (**c**) A super field using the equally weighted combination of Φ, $avgDir$, L, nsV and $seDist$

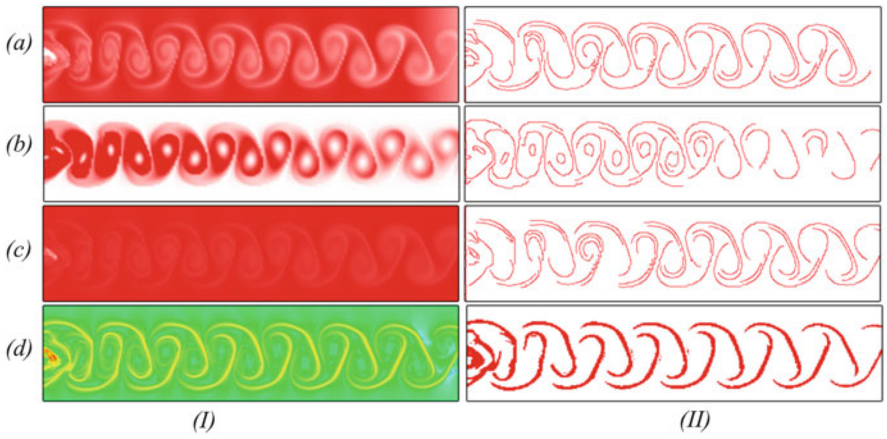


Fig. 11 Attribute fields of the flow behind cylinder and detected edges. (**a**) $avgV$ field; (**b**) nsV field; (**c**) Ł field; (**d**) *FTLE* and LCS. The parameters of Canny edge detector are $\sigma = 1.0, \alpha = 0.3, \beta = 0.8$

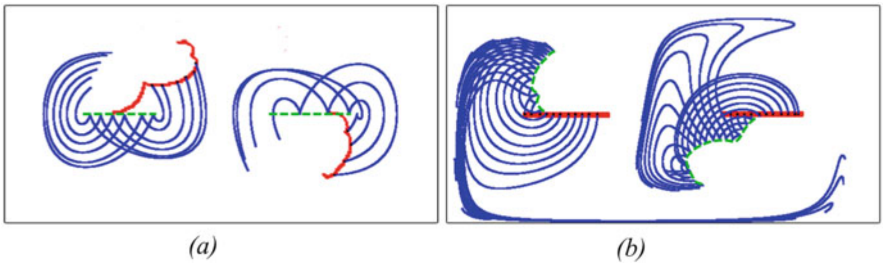


Fig. 12 Pathline and streakline seeding. (**a**) Pathlines seeded on the cusp seeding curve. (**b**) Streaklines seeded on the singularity path

field based discontinuity extraction can be valuable for a variety of applications, such as pathline and streakline placement [22], flow domain segmentation and flow pattern search, which we will explore in future work.

7 Conclusion

In this paper, we introduce a number of derived fields to encode various attributes of the integral curves. In this way, the discontinuity of the behavior between neighboring integral curves can be studied. We show that this discontinuity may be closely related to a number of flow features. We also study different strategies to combine individual attribute fields to form a super attribute field to study the spatial correlation of the attribute fields. We integrate the attribute field computation and the discontinuity extraction into an interactive visualization system to aid the exploration of flow structure. In the future, we plan to extend this work to handle higher-dimensional vector fields. We also plan to have an in-depth investigation on the rigorous description between the relation of the discontinuity in these attribute fields and those well-defined flow features.

Acknowledgements We thank Jackie Chen, Mathew Maltude, Tino Weinkauf for the data. This research was in part supported by NSF IIS-1352722 and IIS-1065107.

References

1. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-8**(6), 679–698 (1986)
2. Chen, G., Mischaikow, K., Laramee, R.S., Pilarczyk, P., Zhang, E.: Vector field editing and periodic orbit extraction using Morse decomposition. IEEE Trans. Vis. Comput. Graph. **13**(4), 769–785 (2007)
3. Chen, G., Mischaikow, K., Laramee, R.S., Zhang, E.: Efficient Morse decompositions of vector fields. IEEE Trans. Vis. Comput. Graph. **14**(4), 848–862 (2008)
4. Edmunds, M., Laramee, R.S., Chen, G., Max, N., Zhang, E., Ware, C.: Surface-based flow visualization. Comput. Graph. **36**(8), 974–990 (2012)
5. Haller, G.: Lagrangian coherent structures and the rate of strain in two-dimensional turbulence. Phys. Fluids **13**(11), 3365–3385 (2001)
6. Haller, G., Sapsis, T.: Lagrangian coherent structures and the smallest finite-time Lyapunov exponent. Chaos **21**(2), 023115 (2011)
7. Helman, J.L., Hesselink, L.: Representation and display of vector field topology in fluid flow data sets. IEEE Comput. **22**(8), 27–36 (1989)
8. Kasten, J., Reininghaus, J., Hotz, I., Hege, H.C.: Two-dimensional time-dependent vortex regions based on the acceleration magnitude. IEEE Trans. Vis. Comput. Graph. **17**(12), 2080–2087 (2011)
9. Lu, K., Chaudhuri, A., Lee, T.Y., Shen, H.W., Wong, P.C.: Exploring vector fields with distribution-based streamline analysis. In: Proceedings IEEE Pacific Visualization Symposium, Sydney, pp. 257–264 (2013)
10. McLoughlin, T., Jones, M.W., Laramee, R.S., Malki, R., Masters, I., Hansen, C.D.: Similarity measures for enhancing interactive streamline seeding. IEEE Trans. Vis. Comput. Graph. **19**(8), 1342–1353 (2013)
11. Pobitzer, A., Peikert, R., Fuchs, R., Schindler, B., Kuhn, A., Theisel, H., Matkovic, K., Hauser, H.: The state of the art in topology-based visualization of unsteady flow. Comput. Graph. Forum **30**(6), 1789–1811 (2011)
12. Pobitzer, A., Lez, A., Matkovic, K., Hauser, H.: A statistics-based dimension reduction of the space of path line attributes for interactive visual flow analysis. In: Proceedings IEEE Pacific Visualization Symposium, pp. 113–120. IEEE, Piscataway (2012)
13. Reininghaus, J., Lowen, C., Hotz, I.: Fast combinatorial vector field topology. IEEE Trans. Vis. Comput. Graph. **17**, 1433–1443 (2011)
14. Salzbrunn, T., Scheuermann, G.: Streamline predicates. IEEE Trans. Vis. Comput. Graph. **12**(6), 1601–1612 (2006)
15. Salzbrunn, T., Garth, C., Scheuermann, G., Meyer, J.: Pathline predicates and unsteady flow structures. Vis. Comput. **24**(12), 1039–1051 (2008)
16. Shadden, S., Lekien, F., Marsden, J.: Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. Physica D **212**(3), 271–304 (2005)

17. Shi, K., Theisel, H., Weinkauf, T., Hege, H.C., Seidel, H.P.: Visualizing transport structures of time-dependent flow fields. IEEE Comput. Graph. Appl. **28**(5), 24–36 (2008)
18. Shi, K., Theisel, H., Hauser, H., Weinkauf, T., Matkovic, K., Hege, H.C., Seidel, H.P.: Path line attributes - an information visualization approach to analyzing the dynamic behavior of 3D time-dependent flow fields. In: Hege, H.C., Polthier, K., Scheuermann, G. (eds.) Topology-Based Methods in Visualization II. Mathematics and Visualization, pp. 75–88. Springer, Berlin (2009)
19. Theisel, H., Weinkauf, T., Seidel, H.P.: Grid-independent detection of closed stream lines in 2D vector fields. In: Proceedings Conference on Vision, Modeling and Visualization, pp. 421–428 (2004)
20. Weinkauf, T., Theisel, H.: Streak lines as tangent curves of a derived vector field. IEEE Trans. Vis. Comput. Graph. **16**(6), 1225–1234 (2010)
21. Weinkauf, T., Theisel, H., Van Gelder, A., Pang, A.: Stable feature flow fields. IEEE Trans. Vis. Comput. Graph. **17**(6), 770–780 (2011)
22. Weinkauf, T., Theisel, H., Sorkine, O.: Cusps of characteristic curves and intersection-aware visualization of path and streak lines. In: Peikert, R., Hauser, H., Carr, H., Fuchs, R. (eds.) Topological Methods in Data Analysis and Visualization II. Mathematics and Visualization, pp. 161–176. Springer, Berlin (2012)
23. Wischgoll, T., Scheuermann, G.: Detection and visualization of closed streamlines in planar fields. IEEE Trans. Vis. Comput. Graph. **7**(2), 165–172 (2001)
24. Yu, H., Wang, C., Shene, C.K., Chen, J.H.: Hierarchical streamline bundles. IEEE Trans. Vis. Comput. Graph. **18**(8), 1353–1367 (2012)

Decomposition of Vector Fields Beyond Problems of First Order and Their Applications

Wieland Reich, Mario Hlawitschka, and Gerik Scheuermann

Abstract In our paper, we discuss generalized vector field decompositions that mainly have been derived from the classical Helmholtz-Hodge-decomposition. The ability to decompose a field into a kernel and a rest respectively to an arbitrary vector-valued linear differential operator allows us to construct decompositions of either toroidal flows or flows obeying differential equations of second (or even fractional) order and a rest. The algorithm is based on the fast Fourier transform and guarantees a rapid processing and an implementation that can be directly derived from the spectral simplifications concerning differentiation used in mathematics.

1 Introduction

Vector field decompositions build an important branch in the topology-based and non-topology-based methods of visualization of flows in many sciences, but particular in physics and hydrodynamics. A widely used form is the Helmholtz-Hodge-decomposition, which can be interpreted as the best possible approximation by a field of divergence-free and curl-free nature. Even though there are plenty of existing algorithms with many different characteristics, most of them are rigid in the sense that they only allow an analysis respectively to the operator ∇.

In our paper, we extend the technique to a broader class of decompositions, which are able to deal with a larger amount of applications, where the classical Helmholtz-Hodge decomposition would give only very limited insight into the structural patterns of the underlying dynamical system. For that purpose, we make use of the Fourier transform of vector fields, which has the nice property to simplify the Helmholtz-Hodge-decomposition to an orthogonal decomposition. Inside the spectral domain we are able to construct a decomposition axis that is equivalent to a more (or less) restrictive operator than ∇ is in the spatial domain.

W. Reich (✉) • M. Hlawitschka • G. Scheuermann
University of Leipzig, Leipzig, Germany
e-mail: reich@informatik.uni-leipzig.de; hlawit@informatik.uni-leipzig.de; scheuermann@informatik.uni-leipzig.de

In Sect. 3, we are discussing the basics of the spectral variant of the Helmholtz-Hodge decomposition and our ideas for generalizations. The results are presented in Sect. 4 and the paper finishes with a conclusion and a list of open problems in Sect. 5.

2 Related Work

Both the works of Tong et al. [21] and of Polthier and Preuss [16] propose a Helmholtz-Hodge-decomposition on vector fields, that relies on the computation of the curl-free and divergence-free parts by a variational approach. Petronetto et al. present a meshless algorithm in [15].

Stam [20] uses a method which is based on the discrete Fourier transform to filter the mass-conserving part of particle movement in his fluid solver that has its theoretical foundation in the Semi-Langrangian scheme. Of all the publications mentioned in this section, it has to be seen as that of most relevance and most influence to our work. A survey on Helmholtz-Hodge-decompositions was recently published by Bhatia et al. [1].

In the history of flow visualization, numerous other vector field decompositions had proven their worth. Luchtenburg et al. [13] uses a Proper Orthogonal decomposition (POD), while Wiebel et al. [24] refine the classical Galilei-transform to a localized flow. Knight and Mallinson [11] compute dual stream functions, which is a decomposition of an incompressible flow into a cross-product of two gradient fields.

All of the named methods were designed to perform on vector fields. However, there also exist important techniques for second order tensors, such as the asymmetric tensor decomposition by Zhang et al. [25] and the polar decomposition, which is, when applied to the Jacobian of a flow map, the principle of detecting Lagrangian coherent structures in the work of Haller [7].

Like in the work from Reddy and Chatterji [17], the Fourier transform used by us is the basis of a huge number of filtering and registration processes in image processing. The generalization of scalar-valued data to Clifford-numbers by Ebling and Scheuermann [3] showed that linear and shift-invariant filters can be directly carried forward to vector fields using a Clifford-convolution.

A general overview on flow visualization is given in [22]. More details for using the Fourier transform to simplify partial differential equations can be found in [4].

3 Mathematical Foundations

3.1 The Helmholtz-Hodge-Decomposition

The Fundamental Theorem of Vector Calculus states that any sufficiently smooth[1] vector field v on an unbounded domain that decays in its norm faster than $1/r$ as $r \to \infty$ can be written as the superposition of a curl-free and divergence-free vector field

$$v = -\nabla \Phi \; + \; \nabla \times A. \tag{1}$$

In the past decades, many attempts were made to design fitting algorithms to perform the Helmholtz-Hodge-Decomposition on discrete vector fields on bounded domains. A survey on these methods can be found in [1]. Widely used techniques are iterative solvers for the potentials Φ and A, which are demanding the Neumann boundary condition [16], stating that the curl-free part has to be perpendicular to the boundary everywhere, while the divergence-free part is parallel. The residual of those operations, which can be described as $v - \nabla \times A + \nabla \Phi$, is considered to be the harmonic rest, which is curl-free and divergence-free at the same time. Recent publications [2] present novel boundary conditions which allow an improved and more natural approach to the interaction of all decomposition parts with the boundary.

Nevertheless and similarly to Stam [20], our paper can be regarded as a contribution to variations of Helmholtz-Hodge-Decompositions that are executed not on necessarily differentiable, but square-integrable functions. The two main reasons are:

- Square-integrable functions can be Fourier-transformed, and therefore provide a very easy way to perform a differentiation and decomposition in the spectral domain, allowing to replace ∇ by any other vector-valued linear position-independent differential operator, even by operators that cannot be expressed in a closed form in the spatial domain.
- The algorithm in this paper benefits highly from the extreme speed of the most up-to-date libraries [5, 6] for performing a fast Fourier transform (FFT). The algorithm often needs to be tested using many different parameter configurations, which makes speed a prime importance.

The disadvantage of the FFT is that it allows only a few types of possible boundary conditions, which are either periodic (the dataset repeats itself infinitesimally times in each spatial dimension), or, which can be seen as a special case, rapidly decaying. However, there are possible workarounds like continuating the data beyond the boundary to a rapidly decaying function, minimizing artifacts that would occur due to the rapid change of the field and its derivatives.

[1]In general, a two times continuously differentiable vector field is sufficient, as these are the partial derivatives of the highest order occurring in the proof.

Further, spectral differentiation can be sensitive to high frequency noise. but we rarely did experience that as being a problem in our experiments, since the spectrum of vector fields associated with fluid flows is generally composed of low frequencies, with the exception of highly turbulent regions.

3.2 *The Spectral Helmholtz-Hodge-Decomposition*

A useful tool in mathematics and physics for solving linear partial differential equations with constant coefficients is the Fourier transform, as it provides a simplification to an algebraic equation [4]. Further, it reduces a convolution to a pure multiplication and it therefore plays a substantial role in signal- and image processing [17].

If we consider a scalar square-integrable function $f(x)$ and its Fourier transform

$$\hat{f}(k) = \int_{-\infty}^{\infty} f(x) e^{-2\pi ikx} dx, \tag{2}$$

then its spatial derivative can be computed in the spectral domain by

$$2\pi ik \cdot \hat{f}(k). \tag{3}$$

Similarly, if we have a square-integrable vector field $v(x_1, x_2, x_3)$ and its Fourier transform $\hat{v}(k_1, k_2, k_3)$, then its Jacobian matrix in the spectral domain is of the form

$$2\pi i \cdot \begin{pmatrix} k_1 \cdot \hat{v}_1 & k_2 \cdot \hat{v}_1 & k_3 \cdot \hat{v}_1 \\ k_1 \cdot \hat{v}_2 & k_2 \cdot \hat{v}_2 & k_3 \cdot \hat{v}_2 \\ k_1 \cdot \hat{v}_3 & k_2 \cdot \hat{v}_3 & k_3 \cdot \hat{v}_3 \end{pmatrix}. \tag{4}$$

Moreover, the transforms of divergence and curl of v can be expressed as

$$< 2\pi i \overrightarrow{k}, \hat{v}(\overrightarrow{k}) > \text{ and } 2\pi i \overrightarrow{k} \times \hat{v}(\overrightarrow{k}) \tag{5}$$

with $\overrightarrow{k}$ being $(k_1, k_2, k_3)^T$.

Comparable to [3], we consider the Fourier transform of a three-dimensional vector field as three independent real transforms of a scalar-valued three-dimensional function. These partial derivatives in the spectral domain do always exist, even if the original vector field is not differentiable everywhere.

If we decompose $\hat{v}(k_1, k_2, k_3)$ into a field, which is parallel to the wave-vector $\overrightarrow{k}$ by calculating

$$\hat{v}_{\parallel} =< \hat{v}, \frac{\overrightarrow{k}}{||\overrightarrow{k}||} > \cdot \frac{\overrightarrow{k}}{||\overrightarrow{k}||} \tag{6}$$

and a perpendicular part

$$\hat{v}_{\perp} = \hat{v} - \hat{v}_{\parallel}, \tag{7}$$

then we get

$$< 2\pi i \overrightarrow{k}, \hat{v}_{\perp} >= 0 \text{ and } 2\pi i \overrightarrow{k} \times \hat{v}_{\parallel} = 0. \tag{8}$$

Due to the linearity of the Fourier transform, we can transform both parts back into the spatial domain and obtain a uniquely determined decomposition into a curl-free and divergence-free field. There is no harmonic part, because $\hat{v}$ satisfies both equations at the same time only for position $\overrightarrow{k} = (0, 0, 0)^T$, which is a zero set in the context of square-integrable fields. A property that will get lost in the discretization later on. However, the constructive character of the proof allows us to implement the formulas almost directly in Sect. 3.4.

Readers interested in a deeper-going discussion of the Hodge-Decomposition in the spectral domain might also have a look in Littlejohn's notes [12]. The resulting field components of $\hat{v}$ are denoted as longitudinal and transversal parts in his texts. Figure 1 illustrates the decomposition results of the simple synthetic vector field which is generated by the polar differential equations $r' = \sin \pi r$ and $\varphi' = 1$ and contains a periodic orbit on each circle in the plane having the radius of a natural number. The analytic decomposition of that vector field can simply be calculated by setting one of these equations to zero.

3.3 *The Generalized Spectral Decomposition*

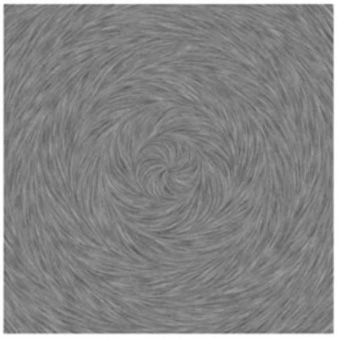
The question arises whether we can replace the operator ∇ (respectively $2\pi i \overrightarrow{k}$) by a differential operator $L \cdot \nabla$ (respectively $2\pi i L \cdot \overrightarrow{k}$) and still obtain a mathematically valid and practically meaningful decomposition. For the scope of this paper, we will restrict ourselves on differential operators with constant coefficients, i.e., L will be a 3×3 -matrix with constant entries.

3.3.1 Differential Operators of First Order

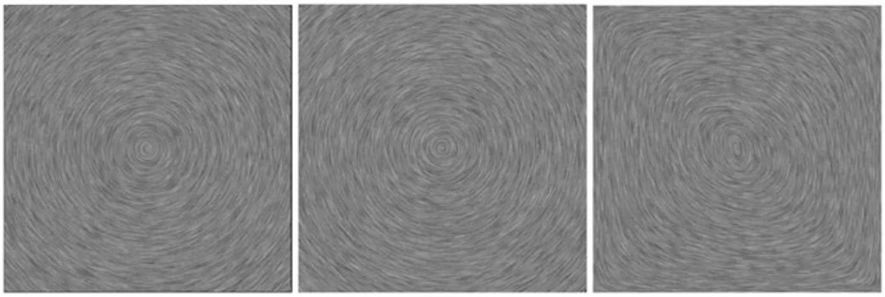
Generalizing formula (1) leads to a decomposition of the field v respectively to an operator L by

$$v = v_{kernel} + v_{rest}, \tag{9}$$

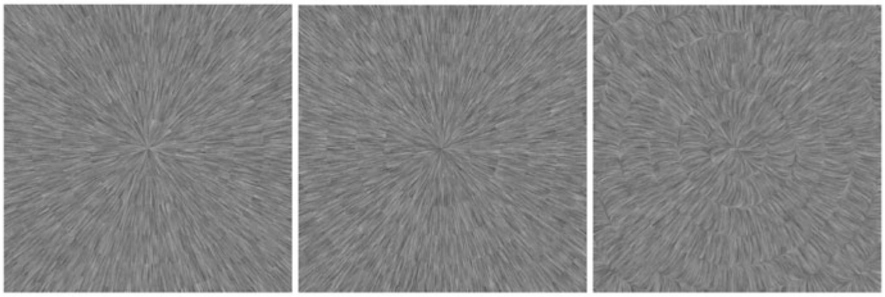
with $< L \cdot \nabla, v_{kernel} > = 0$ and $(L \cdot \nabla) \times v_{rest} = 0$ everywhere.



(a)



(b)



(c)

Fig. 1 (**a**) Visualizing the synthetic field defined by $r' = \sin \pi r$ and $\varphi' = 1$ using a LIC [19], (**b**) *from left to right*: the analytic divergence-free part, the FFT-based divergence-free part, the divergence-free part using the method of Tong et al. [21], (**c**) the irrotational parts in the same order, the FFT-based method preserves the symmetric structures better than a decomposition with orthonal/parallel boundary conditions

The existence and uniqueness of that decomposition for square-integrable functions follows directly from the proof in the last section, we just have to replace the axis of the orthogonal decomposition in the spectral domain by $L \cdot \overrightarrow{k}$. It is easy to see that if L is the identity matrix, we will get the classical Helmholtz-Hodge-decomposition.

Consider the divergence-free vortex field

$$v(x_1, x_2, x_3) = \begin{pmatrix} -x_2 \\ x_1 \\ 1 \end{pmatrix}, \tag{10}$$

as another example which has trivial Helmholtz-Hodge-decomposition.

However, it is possible to further characterize the field by its scalar products with the operators

$$\begin{pmatrix} 0 \\ -\frac{\partial}{\partial x_3} \\ \frac{\partial}{\partial x_2} \end{pmatrix}, \begin{pmatrix} \frac{\partial}{\partial x_3} \\ 0 \\ -\frac{\partial}{\partial x_1} \end{pmatrix}, \text{ and } \begin{pmatrix} -\frac{\partial}{\partial x_2} \\ \frac{\partial}{\partial x_1} \\ 0 \end{pmatrix}. \tag{11}$$

Only the third vector-valued operator leads to a non-vanishing component v_{rest}.

The associated linear operators L that transform ∇ are

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \text{ and } \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}. \tag{12}$$

It could be insufficient to only examine the field with respect to these operator triplets, which are aligned to the basis of the Euclidian space and, like in this special case, are just a projection of the curl of the underlying vector field v. A simple solution to that shortcoming is to multiply the matrix L with another matrix, e.g., a rotational operator, that generates a coordinate transform.

3.3.2 Differential Operators of Second Order

One might argue that decompositions with respect to first order operators can be easily achieved by any of the already existing algorithms performing a Helmholtz-Hodge-decomposition, since the identity

$$0 = < L \cdot \nabla, v > = < \nabla, L^T \cdot v > \tag{13}$$

holds. Consequently, we just need to multiply the field with the transposed matrix L^T and perform a Helmholtz-Hodge-decomposition on the result. There are two reasons why we do not recommend that. First of all, it is not clear how the matrix-multiplication affects boundary conditions that are more sophisticated or restrictive than a periodic or rapidly decaying vector field. Moreover, one would lose the already gained foundation to bring a much broader variety of differential operators into the concept, e.g., those of second order.

For the latter intention, we need to express the vector-valued differential operators

$$\begin{pmatrix} \frac{\partial^2}{\partial x_1^2} \\ \frac{\partial^2}{\partial x_2^2} \\ \frac{\partial^2}{\partial x_3^2} \end{pmatrix} \tag{14}$$

and

$$\begin{pmatrix} \frac{\partial^2}{\partial x_2 \partial x_3} \\ \frac{\partial^2}{\partial x_1 \partial x_3} \\ \frac{\partial^2}{\partial x_1 \partial x_2} \end{pmatrix}. \tag{15}$$

in the spectral domain, which leads, when allowing first order terms as well, to the decomposition axis

$$L_1 \cdot 2\pi i \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} - L_2 \cdot 4\pi^2 \begin{pmatrix} k_1^2 \\ k_2^2 \\ k_3^2 \end{pmatrix} - L_3 \cdot 4\pi^2 \begin{pmatrix} k_2 k_3 \\ k_1 k_3 \\ k_1 k_2 \end{pmatrix}, \tag{16}$$

where L_1, L_2 and L_3 are matrices with constant real values, that might represent a transform of spatial coordinates. As a side note, the factor $2\pi i$ can be left out in the decomposition axis for the first order equations, because it vanishes in the normalization process. For the equations of mixed order, it is essential because it gives both terms the proper weighting.

A more challenging case appears when the vector field has to be decomposed by a PDE that cannot be expressed by the inner or outer product with a single vector, e.g., consider the classical case of $\nabla \times (\nabla \times v)$. The expression of this term in the spectral domain has the symmetric form

$$-4\pi^2 \cdot \begin{pmatrix} -k_2^2 - k_3^2 & k_1 k_2 & k_1 k_3 \\ k_1 k_2 & -k_1^2 - k_3^2 & k_2 k_3 \\ k_1 k_3 & k_2 k_3 & -k_1^2 - k_2^2 \end{pmatrix} \cdot \hat{v}, \tag{17}$$

which means that, if we want to "split" the part w from a vector field v, that fulfills $\nabla \times (\nabla \times w) = 0$, we need to remove three components from all vectors in the spectral domain with each component being defined by a (normalized) row of that matrix.

3.3.3 Fractional Differentials

Fractional differentials have not been very common in the analysis of vector fields for visualization issues so far. The main idea for a scalar function f is to find a linear operator T so that

$$T^{\alpha}(f(x)) = \frac{\partial}{\partial x} f(x) \tag{18}$$

is fulfilled for a given real number α. On the basis of difference quotients or linear approximations of functions, this seems like an impossible task and it is not clear whether these operators do exist at all.

However, for a square-integrable function $f(x)$ with a known Fourier transform $\hat{f}(k)$, it follows directly that in the case $\alpha = 2.0$

$$T(\hat{f}(k)) = \sqrt{2\pi i k} \cdot \hat{f}(k) \tag{19}$$

is a solution.

Similarly, one can define fractional divergence and fractional curl, and the corresponding decomposition would be performed respectively to the decomposition axis

$$\begin{pmatrix} k_1^{\alpha} \\ k_2^{\alpha} \\ k_3^{\alpha} \end{pmatrix}. \tag{20}$$

Fractional divergence and fractional curl have a substantial role in studying fluids in porous media as to be seen in Sect. 4.2. More details in the theory of fractional derivatives are given in [14].

3.4 *Differentiation of Discrete Data with the Fast Fourier Transform*

Our Fourier transforms have always been carried out by the fftw3-library [5], which did an excellent service regarding the computational times. Even on discrete vector fields with several millions of data points, all necessary operations (forward and backward transform for every component of the 3D vector field) were computed in less than a minute. An alternative GPU-based method is also available [6].

While the algorithm-pipeline consisting of a forward transform, an orthogonal decomposition relative to a vector class as in given in (16), and a backward transform has already been clearly pointed out, there are still some inherent differences between the continuous and the discrete Fourier transform.

The coordinate system is not centered in the discrete spectral domain (aliasing phenomenon) so we have to shift all k_i by number of sample points of the corresponding dimension i. We strongly recommend to an inexperienced user to read Johnson's technical report "Notes on FFT-based differentiation" [10] for further details.

Moreover, the harmonic part of the classical Helmholtz-Hodge-decomposition is no longer a zero set. In spectral domain it is the value of $\hat{v}[\overrightarrow{k}]$ for $\overrightarrow{k} = [0, 0, 0]^T$, which is the only vector being perpendicular and parallel at the same time to any other vector. Nevertheless, we always assign that value, which represents the average of the field in the spectral domain, to the v_{kernel}-part, as we would not want to loose an incompressible fluid's mean direction, by the best approximation to a divergence-free field. In Fig. 2 is shown how rigid the harmonic part of the FFT-based algorithm is, when we exclude it as sole third part, because it cannot contain a fixed point of saddle nature as the method by Bhatia et al. [2].

4 Applications

4.1 Toroidal Magnetic Fields

The flow induced by a vector field on (or inside) a torus can develop an immense amount of different structures. First, there are possible irrotational types, e.g., the gradient of the height field. Further, there might be an infinite number of closed streamlines, e.g., the variational field of a small ring moving along the torus. At last, even a chaotic behaviour is possible, e.g. streamlines intersecting a Poincaré-section-plane [9] will never do it at the same location.

Sanderson et al. analyze the magnetic field of a toroidal fusion reactor in [18] by an integration-based technique. Due to the fact that magnetic fields are naturally divergence-free, applying a classical Helmholtz-Hodge-decomposition to such a dataset seems pointless. However, the flow can still be decomposed into two rotational parts, a so called Toroidal-Polodial-decomposition. This is where the generalized spectral decomposition comes into the game. We had chosen the differential operator $\frac{\partial v_1}{\partial x_2} - \frac{\partial v_2}{\partial x_1}$ instead of ∇ for a torus located in the x_1-x_2-plane and obtained two perfectly orthogonal vector fields, both having a non-trivial rotational structure, which is illustrated in Fig. 3.

Eventually both parts of the movement can be analyzed and filtered separately and the spectral decomposition can be the foundation for useful tools for the visualization of magnetic fields as occurring in the publication by Sanderson et al. [18].

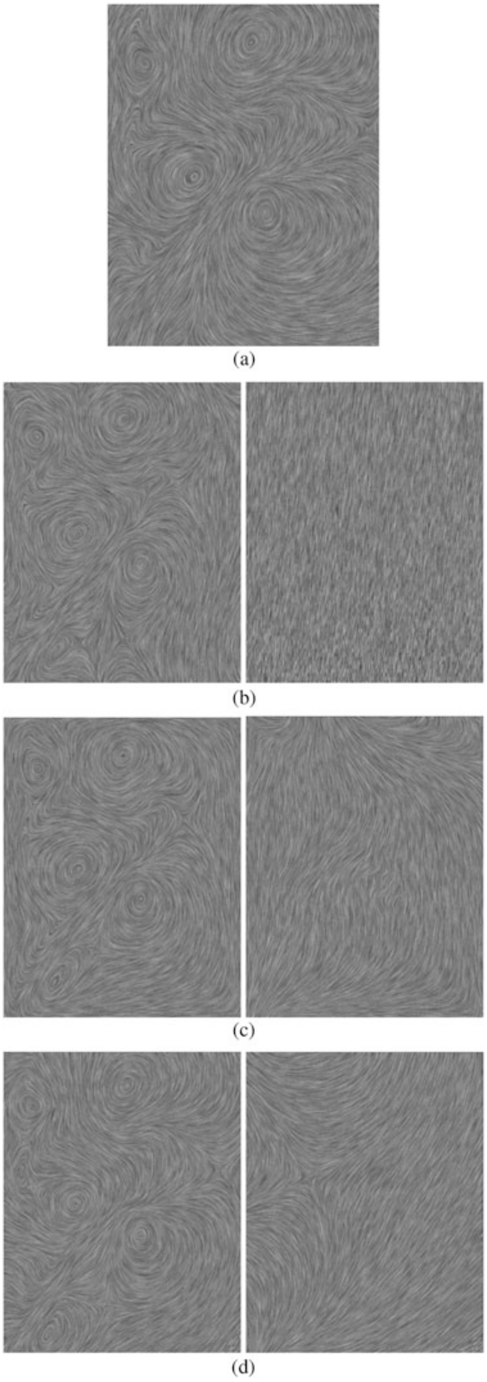


Fig. 2 (**a**) A planar incompressible CFD-simulation, (**b**) the divergence-free part (*left*) and harmonic part (*right*) computed in the spectral domain, (**c**) the divergence-free part (*left*) and harmonic part (*right*) computed by the method of Tong et al. [21], (**d**) the divergence-free part (*left*) and harmonic part (*right*) computed by the method of Bhatia et al. [2]

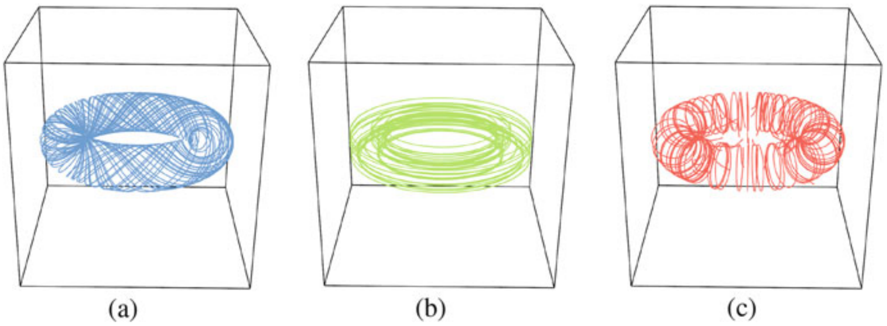


Fig. 3 An invariant torus in a 3D-flow with chaotic behavior on its surface: (**a**) *blue* streamlines started on the surface of the torus, (**b**) *green* streamlines on the toroidal part v_{kernel}, (**c**) *red* streamlines on the poloidal part v_{rest}

4.2 *Flow Through Porous Media*

In their recent text about groundwater hydrology [23] Wheatcraft and Meerschaert are developing the theory that a flux through a porous medium does not need to obey the classical mass conservation laws. Instead they suggest the improvement, that, instead of a divergence of zero, a fractional divergence of zero is a more appropriate model. Therefore, the degree of the fractional derivatives used describes the heterogeneity of the medium.

Consequently, it would be of great interest to analyze not only the best possible approximation of fluid flow data by divergence-free vector fields (which is part of the classical Helmholtz-Hodge-decomposition), but also by fields of vanishing fractional divergence. With the methods proposed in our paper it is easily put in execution.

In Fig. 4 we analyzed the influence of the parameter α when computing a field of free fractional divergence. Vector fields of vanishing fractional divergence are extremely similar in their visual appearance as classic divergence-free fields and can often not be distinguished from them without a closer look at their derivatives. It also seems to be a relation between the magnitude of the vectors and the areas where the classic divergence ($\alpha = 0$) differs most from zero. However, it is remarkable, that the mean value of the classic divergence in all three parts was extremely close to zero (-3.7×10^{-7}, -3.6×10^{-7} and -3.9×10^{-7}). The regions of non-zero divergence cancel each other out and are also often found pairwise. After all, groundwater flow still carries many properties of incompressible flows.

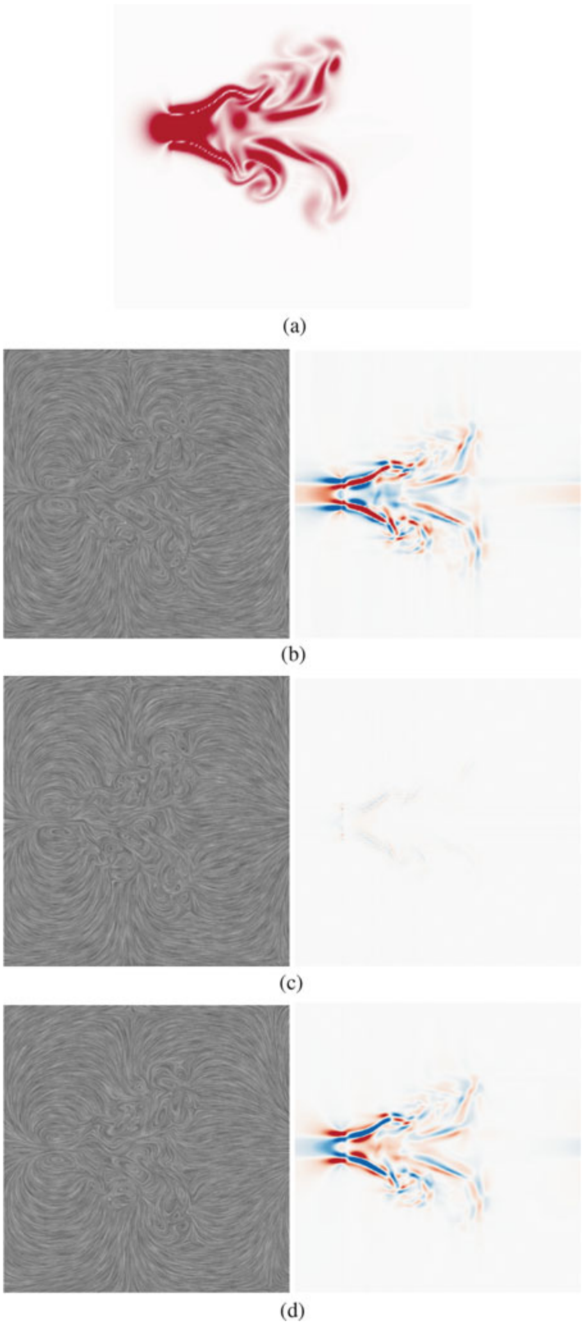


Fig. 4 A planar CFD-simulation, (**a**) The magnitude of the vectors, (**b**) approximation by field of fractional divergence with $\alpha = 0.8$, (**c**) approximation by a divergence-free field with $\alpha = 1.0$, (**d**) approximation by field of fractional divergence with $\alpha = 1.2$, all subfigures from (**b**) to (**d**) are accompanied by the color mappings of the classic divergence

5 Conclusion and Future Work

We presented a novel, more general approach to decomposing vector fields in the spectral domain and showed that the classical Helmholtz-Hodge-decomposition is a special case of the method. At the same time, the result is exactly the homogeneous solution of a partial differential equation, which creates a stronger relationship to mathematics than the manual filter design in [3], which is simply a multichannel-extension of the convolution from image processing.

Moreover, all decompositions of first order have been demystified by Eq. (13), while the gate to a much broader variety of decomposition types, e.g., those of fractional order, has been opened.

As the algorithm depends on an orthogonal decomposition and a FFT only, it is very convincing regarding the computational times. The limitations of the FFT-based method is clearly the lack of a user-given boundary condition, as the algorithm treats the domain as having no spatial boundaries and the data is getting repeated along each coordinate axis. On the other hand, the FFT-based method excels the accuracy of the other methods if the data contains mainly periodic or symmetric features. The limited possible configurations of boundary conditions associated with the FFT might be engaged by finding improved localized variants of the algorithm, such as wavelets [8]. That could also be the key to include position-dependent operators in the equations, such as helicity density or acceleration.

Acknowledgements We like to thank Steven Schlegel for his help in creating the images. We also thank Harsh Bhatia providing results of his decomposition of our datasets and the fruitful discussion.

References

1. Bhatia, H., Norgard, G., Pascucci, V., Bremer, P.T.: The Helmholtz-Hodge decomposition - a survey. IEEE Trans. Vis. Comput. Graph. **19**(8), 1386–1404 (2013)
2. Bhatia, H., Pascucci, V., Bremer, P.T.: The natural Helmholtz-Hodge decomposition for open-boundary flow analysis. IEEE Trans. Vis. Comput. Graph. **20**, 1566–1578 (2014)
3. Ebling, J., Scheuermann, G.: Clifford Fourier transform on vector fields. IEEE Trans. Vis. Comput. Graph. **11**(4), 469–479 (2005)
4. Farlow, S.: Partial Differential Equations for Scientists and Engineers. Dover Books on Advanced Mathematics. Dover, New York (1993)
5. Frigo, M., Johnson, S.G.: The design and implementation of FFTW3. Proc. IEEE **93**(2), 216–231 (2005)
6. Govindaraju, N.K., Lloyd, B., Dotsenko, Y., Smith, B., Manferdelli, J.: High performance discrete Fourier transforms on graphics processors. In: Proceedings of the 2008 ACM/IEEE Conference on Supercomputing, SC ’08, pp. 2:1–2:12. IEEE, Piscataway, NJ (2008)
7. Haller, G.: Distinguished material surfaces and coherent structures in three-dimensional flows. Phys. D **149**, 248–277 (2001)
8. Harouna, S.K., Perrier, V.: Helmholtz-Hodge decomposition on [0, 1] d by divergence-free and curl-free wavelets. In: Curves and Surfaces. Lecture Notes in Computer Science, vol. 6920, pp. 311–329. Springer, Berlin (2010)

9. Hirsch, M., Smale, S., Devaney, R.: Differential Equations, Dynamical Systems and an Introduction to Chaos, 2nd edn. Elsevier, Amsterdam (2004)
10. Johnson, S.G.: Notes on FFT-based differentiation. Technical Report, MIT Applied Mathematics, Massachusetts Institute of Technology (2011)
11. Knight, D., Mallinson, G.: Visualizing unstructured flow data using dual stream functions. IEEE Trans. Vis. Comput. Graph. **2**(4), 355–363 (1996)
12. Littlejohn, R.: The classical electromagnetic field Hamiltonian. Lecture Notes. Technical Report, UC Berkeley Physics, University of California (2012). http://bohr.physics.berkeley.edu/classes/221/1112/notes/hamclassemf.pdf
13. Luchtenburg, D., Noack, B., Schlegel, M.: An introduction to the POD galerkin method for fluid flows with analytical examples and matlab source codes. Technical Report, Department for Fluid Dynamics and Engineering Acoustics, Berlin Institute of Technology (2009). http://berndnoack.com/publications/Luchtenburg2009romfc.PDF
14. Miller, K., Ross, B.: An Introduction to the Fractional Calculus and Fractional Differential Equations. A Wiley Interscience Publication. Wiley, New York (1993)
15. Petronetto, F., Paiva, A., Lage, M., Tavares, G., Lopes, H., Lewiner, T.: Meshless Helmholtz-Hodge decomposition. IEEE Trans. Vis. Comput. Graph. **16**(2), 338–342 (2010)
16. Polthier, K., Preuss, E.: Identifying vector field singularities using a discrete Hodge decomposition. In: Visualization and Mathematics III, pp. 112–134. Springer, Berlin (2002)
17. Reddy, B.S., Chatterji, B.N.: An FFT-based technique for translation, rotation, and scale-invariant image registration. IEEE Trans. Image Process. **5**(8), 1266–1271 (1996)
18. Sanderson, A.R., Chen, G., Tricoche, X., Pugmire, D., Kruger, S., Breslau, J.A.: Analysis of recurrent patterns in toroidal magnetic fields. IEEE Trans. Vis. Comput. Graph. **16**(6), 1431–1440 (2010)
19. Stalling, D., Hege, H.C.: Fast and resolution independent line integral convolution. In: ACM SIGGRAPH, pp. 249–256 (1995)
20. Stam, J.: Stable fluids. In: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99, pp. 121–128. ACM/Addison-Wesley, New York (1999)
21. Tong, Y., Lombeyda, S., Hirani, A.N., Desbrun, M.: Discrete multiscale vector field decomposition. ACM Trans. Graph. **22**(3), 445–452 (2003)
22. Weiskopf, D., Erlebacher, B.: Overview of flow visualization. In: The Visualization Handbook, pp. 261–278 (2005)
23. Wheatcraft, S.W., Meerschaert, M.M.: Fractional conservation of mass. Adv. Water Resour. **31**(10), 1377–1381 (2008)
24. Wiebel, A., Garth, C., Scheuermann, G.: Computation of localized flow for steady and unsteady vector fields and its applications. IEEE Trans. Vis. Comput. Graph. **13**(4), 641–651 (2007)
25. Zhang, E., Yeh, H., Lin, Z., Laramee, R.S.: Asymmetric tensor analysis for flow visualization. IEEE Trans. Vis. Comput. Graph. **15**(1), 106–122 (2009)

Maximum Number of Degenerate Curves in 3D Linear Tensor Fields

Yue Zhang, Yu-Jong Tzeng, and Eugene Zhang

Abstract 3D symmetric tensor fields have a wide range of applications in science and engineering. The topology of a symmetric tensor field, which consists of degenerate curves, can provide critical insights into the behaviors of the tensor fields. Existing methods to extract degenerate curves make some assumptions of the maximum number of degenerate curves in a cell without validating this bound. In this paper, we study the maximum number of degenerate curves in a linear tensor field to contribute to accurate and efficient extraction methods.

1 Introduction

3D symmetric tensor fields have a wide range of applications, such as solid and fluid mechanics, medical imaging, and computer graphics. The topology of 3D tensor fields consists of degenerate tensors (with repeating eigenvalues), which in 3D form curves. Robust extraction and analysis of degenerate curves can play a key role in domain applications.

Existing methods for degenerate curve extraction usually detect the intersection points between the degenerate curves and the boundary faces of each cell in the mesh representing the field. Such an approach assumes that every degenerate curve intersecting a cell must also intersect its boundary. Moreover, it is usually although implicitly assumed that there is only one intersection point per cell face as described

Y. Zhang (✉)
School of Electrical Engineering and Computer Science, 3117 Kelley Engineering Center, Oregon State University, Corvallis, OR 97331, USA
e-mail: zhangyue@oregonstate.edu

Y.-J. Tzeng
School of Mathematics, University of Minnesota, 126 Church Street, Minneapolis, MN 55455, USA
e-mail: ytzeng@umn.edu

E. Zhang
School of Electrical Engineering and Computer Science, 2111 Kelley Engineering Center, Oregon State University, Corvallis, OR 97331, USA
e-mail: zhange@eecs.oregonstate.edu

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_13

in [6] and [8]. It is not clear whether these assumptions are indeed correct. Some approaches perform subdivision on the cell faces to look for additional degenerate curve intersection points; however, it is not clear when the subdivision should stop, and where the subdivision should occur in the cell faces.

All of the above issues are due to a lack of understanding in the maximum number of degenerate curves that can occur inside a cell. Given an interpolation scheme inside a cell, the problems can be stated as follows:

1. Given a polynomial tensor field, how many degenerate curves are in the field?
2. Is it possible for a degenerate curve of a polynomial tensor field to form a loop that is bounded?
3. Is it possible that two degenerate curves of the same type (linear/planar) can intersect?

The answers to the above questions are key to robust and efficient extraction of tensor field topology. In this paper, we focus on the simplest setting, i.e., when the tensor field is linear. We show that in this case there are at least one and at most four degenerate curves in the field. Moreover, up to one degenerate loop can occur, although it is not a structurally stable case. Furthermore, two degenerate curves can intersect at a triple-degenerate point, although, again, this is a structurally unstable case.

2 Previous Work

There has been much work on the topic of 2D and 3D tensor fields for medical imaging and scientific visualization. We refer the readers to the recent survey by Kratz et al. [5]. Here we only refer to the research most relevant to this chapter.

Delmarcelle and Hesselink [1, 2] introduce the topology of 2D symmetric tensor fields as well as conduct some preliminary studies on 3D symmetric tensors in the context of flow analysis. Hesselink et al. later extend this work to 3D symmetric tensor fields [4] and study the degeneracies in such fields. Zheng et al. [7] point out that triple degeneracy, i.e., a tensor with three equal eigenvalues, are not structurally stable features. They further show that double degeneracies, i.e., tensors with only two equal eigenvalues, form lines in the domain. In this work and subsequent research [8], they provide a number of degenerate curve extraction methods based on the analysis of the discriminant function of the tensor field. Tricoche et al. [6] convert the problem of extracting degenerate curves in a 3D tensor field to that of finding the ridge and valley lines of an invariant of the tensor field, thus leading to a more robust extraction algorithm. Both of these methods assume that any degenerate curve that intersects a cell must intersect its boundary. In this paper we show that even for a linear tensor field, it is possible to have a degenerate loop which can completely stay inside a cell and two degenerate curves of the same linear/planar types can intersect at a double degenerate point (although these cases are structurally unstable).

3 Background on Symmetric Tensors and Tensor Fields

In this section we review the most relevant background on tensors and tensor fields.

A 3D (symmetric) tensor T has three real-valued *eigenvalues*: $\lambda_1 \geq \lambda_2 \geq \lambda_3$. A tensor is *degenerate* if there are repeating eigenvalues. There are two types of degenerate tensors, corresponding to three repeating eigenvalues (*triple degenerate*) and two repeating eigenvalues (*double degenerate*), respectively. The *discriminant* of a tensor is defined as $\prod_{1\leq i<j\leq 3}(\lambda_i - \lambda_j)^2$. A tensor is degenerate if and only if its discriminant is zero. An alternative way to describe a degenerate tensor is through its eigenvalues and eigenvectors as follows: $T = (\lambda_n - \lambda_r)ee^T + \lambda_r I$ where λ_n is the non-repeating eigenvalue of T, λ_r the repeating eigenvalue, and e a unit eigenvector corresponding to λ_n. Note that this representation is unique up to the orientation of e. There are two types of double degenerate tensors: (1) linear ($\lambda_n = \lambda_1$ and $\lambda_r = \lambda_2 = \lambda_3$) and (2) planar ($\lambda_n = \lambda_3$ and $\lambda_r = \lambda_1 = \lambda_2$).

The trace of a tensor $T = (t_{ij})$ is $trace(T) = \sum_{i=1}^{3} \lambda_i$. T can be uniquely decomposed as $D + A$ where $D = \frac{trace(T)}{3}\mathbb{I}$ ($\mathbb{I}$ is the three-dimensional identity matrix) and $A = T - D$. The *deviator* A is a *traceless* tensor, i.e., $trace(A) = 0$. Note that T is degenerate if and only if A is degenerate. Consequently, it is sufficient to study the set of traceless tensors, which is closed under matrix addition and scalar multiplication.

A *tensor field* is a tensor-valued function over some domain $\Omega \subset \mathbb{R}^3$. The topology of a tensor field is defined as the set of *degenerate points*, i.e., points in the domain where the tensor field becomes degenerate.

There are three types of degenerate points, *triple degenerate*, *linear degenerate*, and *planar degenerate*. Zheng et al. [8] point out that while triple degeneracies can exist, they are structurally unstable, i.e., can disappear under arbitrarily small perturbations. In contrast, linear and planar degenerate points are structurally stable, i.e., they persist under small enough perturbations in the tensor field. Moreover, under structural stable conditions such points form curves, along which the tensor field is either always linear degenerate or always planar degenerate. While it is possible that linear and planar degenerate points form surfaces and volumes as well as be isolated points, such scenarios do not persist under arbitrarily small perturbation in the field, i.e., structurally unstable.

A degenerate curve can be extracted by finding the zeroth levelset of the discriminant function [7]. In one such approach, degenerate points are first found on the faces of the cells in the mesh on which the tensor data is represented. Given a face, one starts with a random point in the plane containing the face and perform the Newton-Raphson method on the discriminant function. Once all degenerate points are found on the faces of the cells, they are connected by straight line segments that approximate the degenerate curves between these points. The tangent to the degenerate curves at the points on the faces are used to remove ambiguity in how these points are connected. In a more refined approach, numerical integration is used to actually trace out the segments between degenerate points on the faces.

Degenerate curves can also be extracted with the realization that they are a subset of the *crease lines* of an invariant function of the tensor field [6]. A point p_0 is a *ridge point* of a scalar function $\mathbf{f}$ if $\bigtriangledown \mathbf{f}(p_0) \times e_2 = \bigtriangledown \mathbf{f}(p_0) \times e_3 = 0$ where e_2 and e_3 are the eigenvectors corresponding to negative eigenvalues $\lambda_2 \geq \lambda_3$ of the Hessian of $\mathbf{f}$. A *valley point* p_0 satisfies that $\bigtriangledown \mathbf{f}(p_0) \times e_1 = \bigtriangledown \mathbf{f}(p_0) \times e_2 = 0$ where e_1 and e_2 are the eigenvectors corresponding to positive eigenvalues $\lambda_1 \geq \lambda_2$ of the Hessian of $\mathbf{f}$. A crease line can then be extracted using the well-known *Marching Cubes* method.

4 3D Linear Symmetric Tensor Fields

A 3D symmetric, traceless linear tensor field has following form $LT(x, y, z) = T_0 + xT_x + yT_y + zT_z$ where T_0, T_x, T_y, and T_z are symmetric, traceless matrices. We now consider the problem of determining the maximum number of degenerate curves in a 3D linear tensor field.

4.1 Homogenous Linear Tensor Fields

We first consider the case when $T_0 = 0$. Under the assumption, $LT(-x, -y, -z) = -LT(x, y, z)$ and $(0, 0, 0)$ is a triple degenerate point. Furthermore, a point (x, y, z) is degenerate if and only if (kx, ky, kz) is also degenerate for any $k \neq 0$. Moreover, if (x, y, z) is a linear degenerate point, then (kx, ky, kz) is a linear degenerate point for $k > 0$ and a planar degenerate point for $k < 0$.

The above results indicate that if a point $p = (x, y, z)$ is a degenerate point, then any point on the ray emanating from the origin and containing p is also a degenerate point. Moreover, the linear/planar classification along the ray does not change. In contrast, the ray in the opposite direction from the origin has the opposite linear/planar classification. Figure 1 (left) shows this. The origin is a triple degenerate point, colored in gray. Yellow curves consist of linear degenerate points and green curves consist of planar degenerate points. Note that the results shown in Fig. 1 (left and middle) are based on degenerate curve extraction method of Zheng et al. [8].

A fundamental question is how many such degenerate ray pairs exist given a linear tensor field where $T_0 = 0$. This is equivalent to the following question. Given 3×3 traceless, symmetric matrices T_x, T_y, and T_z, how many solutions exist such that $xT_x + yT_y + zT_z$ is degenerate? We first note that the set of all traceless and symmetric tensors with configuration $\begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{12} & t_{22} & t_{23} \\ t_{13} & t_{23} & -t_{11} - t_{22} \end{pmatrix}$ form a five dimensional linear space

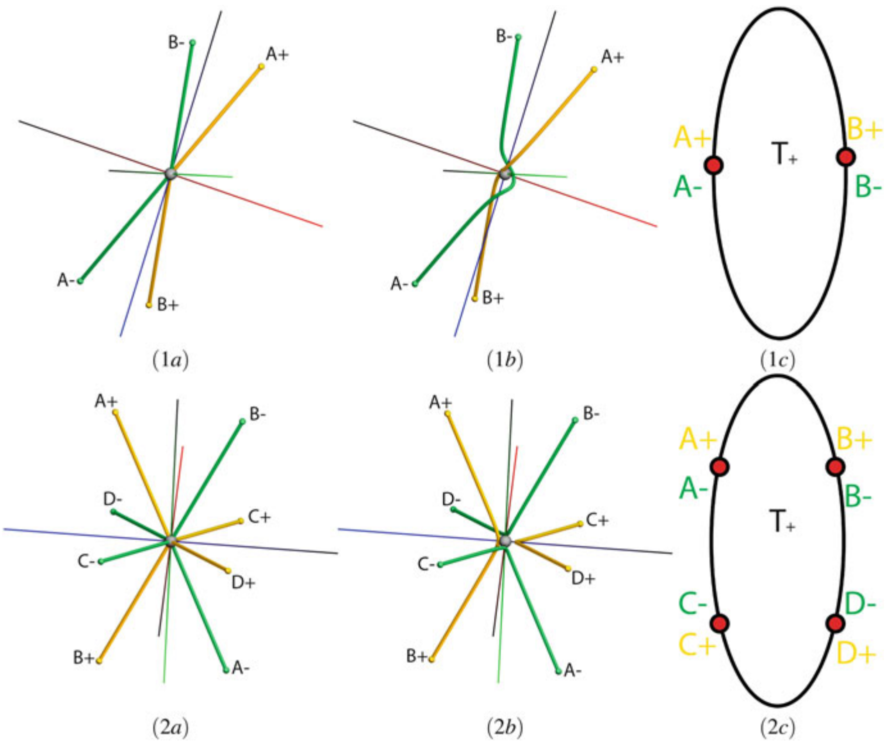


Fig. 1 Two pairs of 3D linear tensor fields. The ones on the *left* are homogeneous tensor fields, i.e., zero T_0. The fields in the *middle* have a non-zero T_0 component. The *colored curves* are the degenerate curves: *yellow* for linear degenerate tensors, and *green* for planar degenerate tensors. Notice that degenerate curves are straight lines (i.e., rays) in homogeneous tensor fields (*left*), which intersect at the origin, a triple degenerate point. The corresponding non-homogeneous tensor fields (*middle*) have the same number of degenerate curves as the homogeneous ones. Moreover, the asymptote limit points of the degenerate curves are the same as the corresponding points in the homogeneous ones (see the corresponding labels). However, the degenerate curves in non-homogeneous tensor fields do not intersect at the origin. It is interesting to note that the union of all degenerate points in the structurally stable, non-homogeneous case (*middle*) can be parameterized by an ellipse minus the *red points* (*right*). More details on this can be found in Sect. 4.2

$\mathbb{T}$ spanned by the basis $T_{11} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{pmatrix}$, $T_{22} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$, $T_{12} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$, $T_{13} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$, and $T_{23} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$. Any tensor in this space can be expressed as $t_{11}T_{11}+t_{22}T_{22}+t_{12}T_{12}+t_{13}T_{13}+t_{23}T_{23}$ for some $t_{11}, t_{12}, t_{13}, t_{22}, t_{23} \in \mathbb{R}$. Note that the subspace spanned by T_x, T_y, and T_z can be of *zero*, *one*, *two*, and *three*-dimensional space in the set of 3D symmetric, traceless tensors.

In the zero-dimensional space, we have $T_x = T_y = T_z = 0$. The tensor field is triply degenerate everywhere in the domain. This case is structurally unstable.

In the one-dimensional case, without loss of generality assume that $LT(x,y,z) = (x + ky + lz)T_x$ for some $k, l \in \mathbb{R}$. If T_x is degenerate, then $LT(x,y,z)$ is degenerate everywhere, i.e., the set of degenerate points for the whole volume. When T_x is non-degenerate, then $LT(x,y,z)$ is non-degenerate everywhere except on the plane $x + ky + lz = 0$, where it is triply degenerate. Neither case is structurally stable.

In the two-dimensional case, without loss of generality assume that $LT(x,y,z) = (x + kz)T_x + (y + lz)T_y$ for some $k, l \in \mathbb{R}$. In this case, the tensor field is triply degenerate on the intersection of the planes $x + kz = 0$ and $y + lz = 0$. Recall that this is unstable. In addition, if there exists $m, n \in \mathbb{R}$ such that $mT_x + nT_y$ is degenerate, it is straightforward to verify that $LT(x,y,z)$ is also degenerate for the plane $n(x + kz) = m(y + lz)$. This is still a structurally unstable case since the degenerate points should form curves [7].

We now focus on the last case, i.e., T_x, T_y, and T_z are linearly independent. Note that the tensor field $LT(x,y,z)$ leads to an injective map $LT : \mathbb{R}^3 \to \mathbb{T}$. The image of this map, $U = \{xT_x + yT_y + zT_z | x, y, z \in \mathbb{R}\}$, is a linear three-dimensional subspace of $\mathbb{T}$. Furthermore, LT is an isomorphism between $\mathbb{R}^3$ and U. Consequently, LT is also an isomorphism between the set of degenerate points of $LT(x,y,z)$ and $U \bigcap D$ where $D \subset \mathbb{T}$ is the set of all degenerate tensors. This isomorphism allows us to reformulate the problem of finding $(x,y,z) \in \mathbb{R}^3$ such that $xT_x + yT_y + zT_z$ is degenerate to the problem of finding the intersections of U and D.

Recall that U is a three-dimensional (codimension-two) subspace of $\mathbb{T}$, i.e., there exist two *linear*, *homogeneous* functions F_0 and G_0 of t_{ij}'s such that any element in U can be expressed as $t_{11}T_{11} + t_{22}T_{22} + t_{12}T_{12} + t_{13}T_{13} + t_{23}T_{23}$ such that $F_0(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) = G_0(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) = 0$. Note that $F_0(kt_{11}, kt_{22}, kt_{12}, kt_{13}, kt_{23}) = kF_0(t_{11}, t_{22}, t_{12}, t_{13}, t_{23})$ and $G_0(kt_{11}, kt_{22}, kt_{12}, kt_{13}, kt_{23}) = kG_0(t_{11}, t_{22}, t_{12}, t_{13}, t_{23})$ as both are homogenous polynomials. Moreover, F_0 and G_0 are linearly independent.

In contrast, D is a non-linear subspace of $\mathbb{T}$ consisting of tensors of the following format $k \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} \begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix} - \frac{k}{3} I$ for some $k \in \mathbb{R}$ and some unit vector $\begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix}$. This is equivalent to $k \begin{pmatrix} \alpha^2 - \frac{1}{3} & \alpha\beta & \alpha\gamma \\ \alpha\beta & \beta^2 - \frac{1}{3} & \beta\gamma \\ \alpha\gamma & \beta\gamma & \gamma^2 - \frac{1}{3} \end{pmatrix}$ where $\alpha^2 + \beta^2 + \gamma^2 = 1$. Note that when $k > 0$, the tensor T is linear, i.e., one repeating negative eigenvalue and one positive eigenvalue. When $k < 0$, T is planar, i.e., one repeating positive eigenvalue and one negative eigenvalue.

A degenerate tensor in U therefore must satisfy

$$F_0\left(\alpha^2 - \frac{1}{3}, \beta^2 - \frac{1}{3}, \alpha\beta, \alpha\gamma, \beta\gamma\right) = 0 \tag{1}$$

$$G_0\left(\alpha^2 - \frac{1}{3}, \beta^2 - \frac{1}{3}, \alpha\beta, \alpha\gamma, \beta\gamma\right) = 0 \tag{2}$$

$$\alpha^2 + \beta^2 + \gamma^2 = 1 \tag{3}$$

which is equivalent to

$$F_0\left(\frac{2\alpha^2 - \beta^2 - \gamma^2}{3}, \frac{2\beta^2 - \alpha^2 - \gamma^2}{3}, \alpha\beta, \alpha\gamma, \beta\gamma\right) = 0 \tag{4}$$

$$G_0\left(\frac{2\alpha^2 - \beta^2 - \gamma^2}{3}, \frac{2\beta^2 - \alpha^2 - \gamma^2}{3}, \alpha\beta, \alpha\gamma, \beta\gamma\right) = 0 \tag{5}$$

$$\alpha^2 + \beta^2 + \gamma^2 = 1 \tag{6}$$

We define $f_0(\alpha, \beta, \gamma) = F_0\left(\frac{2\alpha^2-\beta^2-\gamma^2}{3}, \frac{2\beta^2-\alpha^2-\gamma^2}{3}, \alpha\beta, \alpha\gamma, \beta\gamma\right)$ and $g_0(\alpha, \beta, \gamma) = G_0\left(\frac{2\alpha^2-\beta^2-\gamma^2}{3}, \frac{2\beta^2-\alpha^2-\gamma^2}{3}, \alpha\beta, \alpha\gamma, \beta\gamma\right)$. Both f_0 and g_0 are *homogeneous* quadratic polynomials of α, β and γ. Determining the number of rays is equivalent to finding the solutions to $f_0(\alpha, \beta, \gamma) = g_0(\alpha, \beta, \gamma) = 0$ on the unit sphere $\alpha^2 + \beta^2 + \gamma^2 = 1$ since each solution corresponds to a degenerate ray. Notice that if (α, β, γ) is a solution to the aforementioned system, so is $(-\alpha, -\beta, -\gamma)$; however, both solutions correspond to the same degenerate tensor. To remove this double-counting of solutions, we simply consider the solutions to the system $f_0(\alpha, \beta, \gamma) = g_0(\alpha, \beta, \gamma) = 0$ on $\mathbb{R}P^2$, the two-dimensional real projective space.

To make our analysis more complete, we instead consider finding solutions in the space $\mathbb{C}P^2$, the two-dimensional *complex projective space*. This can be answered by the following version of Bézout's theorem from algebraic geometry [3].

Theorem 1 *Let f_0 and g_0 be two homogeneous polynomials in three variables of degree d and e, respectively. Let C_f and C_g be the curves defined by $f_0 = 0$ and $g_0 = 0$ in the complex projective space $\mathbb{C}P^2$. Assume that C_f and C_g do not have any common component, then they intersect at exactly $d * e$ points in $\mathbb{C}P^2$, counted with multiplicity.*

According to Bézout's theorem, there are two scenarios: (1) $f_0(\alpha, \beta, \gamma)$ and $g_0(\alpha, \beta, \gamma)$ have a common non-constant component, or (2) $f_0(\alpha, \beta, \gamma)$ and $g_0(\alpha, \beta, \gamma)$ do not have a common non-constant component.

In the first case, both $f_0(\alpha, \beta, \gamma)$ and $g_0(\alpha, \beta, \gamma)$ can be factored into the product of two linear polynomials with possibly complex coefficients. Let $f_0(\alpha, \beta, \gamma) = f_1(\alpha, \beta, \gamma) f_2(\alpha, \beta, \gamma)$ and $g_0(\alpha, \beta, \gamma) = g_1(\alpha, \beta, \gamma) g_2(\alpha, \beta, \gamma)$ where f_1, f_2, g_1 and g_2 are homogeneous linear polynomials of α, β, and γ. Note that f_0 and g_0 cannot have two common components as in that case f_0 and g_0 become linearly dependent, thus violating our assumption. Consequently, f_0 and g_0 can share precisely one non-constant factor. Without loss of generality, assume that $f_1 = g_1$ and f_2 and g_2 are linearly independent. Note that $f_1 = g_1$, f_2 and g_2 must all be real-valued polynomials. Assume that $f_1 = g_1$ are complex-valued, then f_2 must be the conjugate of f_1 and g_2 the conjugate of g_1, i.e., f_2 and g_2 become linearly dependent, again violating our assumption. Consequently, the solutions to $f_0 = g_0 = 0$ consists of a line ($f_1 = 0$) and one point (the solution to $f_2 = g_2 = 0$). By letting k vary, the set of degenerate points include a plane (derived from the aforementioned line) and a line (derived from the aforementioned point). This is a structurally unstable case.

In the second scenario, f_0 and g_0 contain no common factor, and the above system of equations has four complex roots. This scenario is structurally stable. Recall that we are only concerned with real-valued solutions. Given that f_0 and g_0 are both real-valued quadratic polynomials, complex solutions must appear in pairs. This implies the following five possibilities: (1) zero real solution, (2) one repeating real solution, (3) two distinct real solutions, (4) one repeating real solution and two distinct real solutions and lastly (5) four distinct real solutions. Note that each solution corresponds to two degenerate rays in the domain of the tensor field. Consequently, there can be zero, two, four, six, and eight degenerate rays, with the cases two and six degenerate rays being structurally unstable. Half of the degenerate rays are linear, while the other half are planar. Note that this bound (eight degenerate rays or four degenerate lines) is exact (shown in Fig. 1 (2a)).

However, the case of zero degenerate curves is impossible due to the fact that the tensor field $xT_x + yT_y + zT_z$ is traceless. To see this, we first rewrite f_0 and g_0 as $f_0 = \begin{pmatrix}\alpha & \beta & \gamma\end{pmatrix} M_{f_0} \begin{pmatrix}\alpha \\ \beta \\ \gamma\end{pmatrix}$ and $g_0 = \begin{pmatrix}\alpha & \beta & \gamma\end{pmatrix} M_{g_0} \begin{pmatrix}\alpha \\ \beta \\ \gamma\end{pmatrix}$ where M_{f_0} and M_{g_0} are 3×3 traceless symmetric matrices. Note that any orthonormal change of basis in the space of (α, β, γ) ensure that both matrices after the change of basis remain traceless and symmetric while one of them becomes diagonal. Without loss of generality, we will assume that M_{f_0} is already diagonal.

We now consider two new variables $\alpha' = \frac{\alpha}{\gamma}$ and $\beta' = \frac{\beta}{\gamma}$. Recall that F_0 and G_0 are linear functions of its components, i.e., $F_0(ka, kb, kc, kd, de) = kF_0(a, b, c, d, e)$ and $G_0(ka, kb, kc, kd, ke) = kG_0(a, b, c, d, e)$ for $k \in \mathbb{R}$. Let $k = (\frac{1}{\gamma})^2$. The above equations is now

$$F_0(\frac{2\alpha'^2 - \beta'^2 - 1}{3}, \frac{2\beta'^2 - \alpha'^2 - 1}{3}, \alpha'\beta', \alpha', \beta') = 0 \tag{7}$$

$$G_0(\frac{2\alpha'^2 - \beta'^2 - 1}{3}, \frac{2\beta'^2 - \alpha'^2 - 1}{3}, \alpha'\beta', \alpha', \beta') = 0 \tag{8}$$

They can be written as

$$\begin{pmatrix}\alpha' & \beta' & 1\end{pmatrix} M_{f_0} \begin{pmatrix}\alpha' \\ \beta' \\ 1\end{pmatrix} = 0 \tag{9}$$

$$\begin{pmatrix}\alpha' & \beta' & 1\end{pmatrix} M_{g_0} \begin{pmatrix}\alpha' \\ \beta' \\ 1\end{pmatrix} = 0 \tag{10}$$

where $M_{f_0} = \begin{pmatrix} f_{11} & 0 & 0 \\ 0 & f_{22} & 0 \\ 0 & 0 & -f_{11} - f_{22} \end{pmatrix}$ is diagonal and $f_{11} > 0, f_{22} \geq 0$, and $f_{11} \geq f_{22}$. The above system (Eqs. (9) and (10)) has the same solutions as

$$\begin{pmatrix}\alpha' & \beta' & 1\end{pmatrix} M_{f_0} \begin{pmatrix}\alpha' \\ \beta' \\ 1\end{pmatrix} = 0 \tag{11}$$

$$\begin{pmatrix}\alpha' & \beta' & 1\end{pmatrix} (M_{g_0} + sM_{f_0}) \begin{pmatrix}\alpha' \\ \beta' \\ 1\end{pmatrix} = 0 \tag{12}$$

for any $s \in \mathbb{R}$. Consequently, we can select s such that Eq. (12) has no constant term. Therefore, we further assume that $M_{g_0} = \begin{pmatrix} g_{11} & g_{12} & g_{13} \\ g_{12} & -g_{11} & g_{23} \\ g_{13} & g_{23} & 0 \end{pmatrix}$, i.e., $f_{11}\alpha'^2 + f_{22}\beta'^2 = f_{11} + f_{22}$ and $g_{11}\alpha'^2 + 2g_{12}\alpha'\beta' - g_{11}\beta'^2 + 2g_{13}\alpha' + 2g_{23}\beta' = 0$. The former represents an ellipse, which can degenerate into two parallel lines $\alpha = \pm 1$ when $f_{22} = 0$. The latter is a hyperbola ($-g_{11}^2 - g_{12}^2 \leq 0$). The origin is on one sheet of the hyperbola and is also enclosed by the ellipse (or the two lines in the degenerate case). We can show that there is at least one intersection point between the hyperbola $g_0 = 0$ and the ellipse $f_0 = 0$, i.e, one degenerate ray pair. If there is one solution, it must have a multiplicity of two. Figure 1 (1a and 2a) show the cases of two degenerate curves and four degenerate curves, respectively. It is worth noting that in the case the ellipse, or the hyperbola, or both curves degenerate into straight lines, the intersection is still not empty, i.e., there is at least one real solution to the system and thus a degenerate ray in the original tensor field.

4.2 Non-homogenous Linear Tensor Fields

We now consider the case when $T_0 \neq 0$. In this case, $LT(x, y, z) = T_0 + xT_x + yT_z + zT_z$ introduces an affine map from $\mathbb{R}^3$ to $\mathbb{T}$. As with the case of $T_0 = 0$, there are a number of scenarios to consider.

First, if T_0 is in the span of T_x, T_y, and T_z, i.e., there exist $m, n, p \in \mathbb{R}$ such that $T_0 = mT_x + nT_y + pT_z$, then $LT(x, y, z) = (x + m)T_x + (y + n)T_y + (z + p)T_z$. With a change of coordinate systems $x' = x + m$, $y' = y + n$, $z' = z + p$, $LT(x', y', z')$ is homogeneous and our previous analysis applies.

If T_0 is not in the span of T_x, T_y, and T_z, there are four cases: T_x, T_y, and T_z span a *zero*, *one*, *two*, or *three*-dimensional space in the set of 3D symmetric, traceless tensors. Using similar arguments, we can show that the first three cases are structurally unstable. The last case, when T_0, T_x, T_y, and T_z are linearly independent inside $\mathbb{T}$, is what we are mostly interested in and is the most difficult. Here, we consider the intersection of the set of degenerate tensors and the space of $U = \{T_0 + xT_x + yT_y + zT_z | x, y, z \in \mathbb{R}\}$ inside $\mathbb{T}$.

First, we comment that the set U is still a three-dimensional plane inside $\mathbb{T}$ that does not contain the origin of $\mathbb{T}$. Consequently, U can again be characterized by two first-degree (inhomogeneous) polynomials F and G of t_{ij}'s such that U consists of tensors of the form $t_{11}T_{11} + t_{22}T_{22} + t_{12}T_{12} + t_{13}T_{13} + t_{23}T_{23}$ where $F(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) = 0$ and $G(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) = 0$. For reasons to be made clear soon, we still define F_0 and G_0 to be the polynomials characterizing the related set $U = \{xT_x + yT_y + zT_z | x, y, z \in \mathbb{R}\}$. Therefore, $F_0(t_{11} - t_{0,11}, t_{22} - t_{0,22}, t_{12} - t_{0,12}, t_{13} - t_{0,13}, t_{23} - t_{0,23}) = F(t_{11}, t_{22}, t_{12}, t_{13}, t_{23})$ where $T_0 = t_{0,11}T_{11} + t_{0,22}T_{22} + t_{0,12}T_{12} + t_{0,13}T_{13} + t_{0,23}T_{23}$. Consequently, $F(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) = F_0(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) + \mu_F$ where μ_F is derived from T_0 and the coefficients of F_0. Similarly, $G(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) = G_0(t_{11}, t_{22}, t_{12}, t_{13}, t_{23}) + \mu_G$ where μ_G is derived from T_0 and the coefficients of G_0. Note that μ_F and μ_G are not zero simultaneously, for otherwise T_0 is in the span of T_x, T_y, and T_z. In addition, the solutions to $F = 0$ and $G = 0$ are the same as $aF + bG = 0$ and $cF + dG = 0$ where the determinant of $\begin{pmatrix} a & b \\ c & d \end{pmatrix} \neq 0$. Without loss of generality, we assume that $\mu_F \neq 0$. Consequently, we choose $G' = G - \frac{\mu_G}{\mu_F}F = G_0 - \frac{\mu_G}{\mu_F}F_0$, which is again homogeneous. Therefore, from now on we assume that U can be characterized by a homogeneous polynomial $G_0 = 0$ and an inhomogeneous polynomial $F = 0$.

Second, U can also be characterized by the solutions to the following equations:

$$k \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} \begin{pmatrix} \alpha & \beta & \gamma \end{pmatrix} - \frac{k}{3} I = T_0 + xT_x + yT_y + zT_z \tag{13}$$

for some $k \in \mathbb{R}$ and some unit vector $\begin{pmatrix}\alpha & \beta & \gamma\end{pmatrix}$. As before we assume $\alpha^2 + \beta^2 + \gamma^2 = 1$. We would like to connect the case of $T_0 \neq 0$ to that of $T_0 = 0$. This leads to

$$k \begin{pmatrix}\alpha \\ \beta \\ \gamma\end{pmatrix} \begin{pmatrix}\alpha & \beta & \gamma\end{pmatrix} - \frac{k}{3} I - T_0 = xT_x + yT_y + zT_z \tag{14}$$

As in the case of $T_0 = 0$, we can convert F, F_0 and G_0 to be functions of α, β, and γ under the assumption that $\alpha^2 + \beta^2 + \gamma^2 = 1$. Let the corresponding functions be f, f_0 and g_0, respectively. Both f_0 and g_0 are homogeneous polynomials whose corresponding quadratic forms are traceless symmetric matrices M_{f_0} and M_{g_0}. In contrast, M_f, the quadratic form of $f = f_0 + \frac{\mu_F}{k}(\alpha^2 + \beta^2 + \gamma^2)$, is not traceless. Moreover, f is also a function of k. We are looking for the number of solutions of $f = 0$, $g_0 = 0$, and $\alpha^2 + \beta^2 + \gamma^2 = 1$ for any given k. To see this, we can further simplify the situation by assuming $\mu_F = 1$ and finding the proper basis such that M_{g_0} has the form $\begin{pmatrix} g_{11} & 0 & 0 \\ 0 & g_{22} & 0 \\ 0 & 0 & -g_{11} - g_{22}\end{pmatrix}$ where $g_{11} > 0$, $g_{22} \geq 0$ and $g_{11} \geq g_{22}$. The system we are solving is:

$$f_0(\alpha, \beta, \gamma) = -1/k \tag{15}$$

$$g_{11}\alpha^2 + g_{22}\beta^2 = (g_{11} + g_{22})\gamma^2 \tag{16}$$

$$\alpha^2 + \beta^2 + \gamma^2 = 1 \tag{17}$$

which is equivalent to

$$f_0(\alpha, \beta, \gamma) = -1/k \tag{18}$$

$$(2g_{11} + g_{22})\alpha^2 + (g_{11} + 2g_{22})\beta^2 = (g_{11} + g_{22}) \tag{19}$$

$$\alpha^2 + \beta^2 + \gamma^2 = 1 \tag{20}$$

Fixing k, this is a system of two quadratic equations $f = 0$ and $g_0 = 0$ in the complex project plane $\mathbb{C}P^2$. However, the number of real solutions is a function of k. Note that when $k = \pm\infty$, $f = f_0$ and there are either two or four real-valued solutions (counted with multiplicity). When $k = 0$, there is no real-valued solution since $T_0 \neq 0$ and T_0, T_x, T_y, and T_z are linearly independent. As k decreases from ∞ to 0, it is possible some complex-valued solutions become real-valued. A degenerate loop can form if these two families of solutions at some point become complex-valued simultaneously. There are two scenarios: (1) $g_{22} > 0$, and (2) $g_{22} = 0$. We will discuss them separately, next.

The first case, i.e., $g_{22} > 0$, is structurally stable. Furthermore, it is straightforward to verify that f and g_0 have no common factor for any $k \in \mathbb{R}$. In this case, using the Bézout's theorem we see that there are zero, one, two, three, or four real solutions for any fixed k. Notice that it is now possible to have zero real solutions since M_f is not traceless. However, the number of real solutions is a function of k. This leads to the question exactly how many degenerate curves exist when $T_0 \neq 0$.

Equation (19) represents an elliptical cylinder and Eq. (20) represents the unit sphere. Since $g_{22} > 0$ their intersection consists of two identical ellipses, corresponding to $\gamma > 0$ and $\gamma < 0$, respectively. The projection $(\alpha, \beta, \gamma) \rightarrow (\alpha, \beta)$ maps the ellipses to the ellipse described by Eq. (19), which is contained in the unit circle since $0 < \frac{g_{11}+g_{22}}{2g_{11}+g_{22}} < 1$ and $0 < \frac{g_{11}+g_{22}}{g_{11}+2g_{22}} < 1$. Recall that $\left(\alpha\ \beta\ \gamma\right)$ and $\left(-\alpha\ -\beta\ -\gamma\right)$ lead to the same tensor, we only need to consider one ellipse, i.e., $\gamma > 0$. We refer to the ellipse as T^+, and here we can use this ellipse to parameterize the degenerate points in the tensor field. Note that every point on T^+ corresponds to $f_0(\alpha, \beta, \sqrt{1-\alpha^2-\beta^2}) = -\frac{1}{k}$ for some $k \in [-\infty, +\infty]$. Moreover, every degenerate point is mapped to a point on T^+. The map is bijective.

When $k = \infty$, under structurally stable conditions our previous analysis for the homogeneous case has shown that there are either two or four real solutions to the system, denoted by p_i (i between 1 and the number of solutions) (Fig. 1 (1c and 2c): red dots). The degenerate curves in the tensor field, i.e., the solutions to the system when $T_0 \neq 0$, correspond bijectively to segments between consecutive dots p_i's. This means that the number of degenerate curves when $T_0 \neq 0$ is the same as the number of degenerate curves when $T_0 = 0$ (two opposite degenerate rays are considered as one degenerate curve). More interestingly, the structurally stable condition $g_{22} > 0$ implies that every segment between p_i's is homeomorphic to its corresponding degenerate curve via Eq. (13) and the assumption $\gamma > 0$. Consequently, it is impossible to have a finite degenerate loop, since segments are simply connected while loops are not. Consequently, all degenerate curves must end at ∞ (Fig. 1).

The second scenario, i.e., $g_{22} = 0$, is numerically unstable. In this case, Eq. (19) becomes $2\alpha^2 + \beta^2 = 1$, or equivalently $\alpha^2 = \gamma^2$ since $\alpha^2 + \beta^2 + \gamma^2 = 1$. There are two important observations in this case. First, both $(0, 1, 0)$ and $(0, -1, 0)$ are on the ellipse $2\alpha^2 + \beta^2 = 1$ and correspond to the same tensor. These two points divide the ellipse into the left and right halves. Second, assuming that f_0 and g_0 have no common factor, i.e., there are only two or four real solutions in the corresponding case of $T_0 = 0$ (counting multiplicity), we have the right half of the ellipse satisfying $\alpha = \gamma$ and the left half satisfying $\alpha = -\gamma$. By Bézout's theorem, there are at most two real solutions on each half of the ellipse for $f_0 = g_0 = 0$.

Assume that f and g_0 do not share any common factor for any $k \in \mathbb{R}$. In the case of two real solutions when $k = \pm\infty$, they must situate on half of the ellipse (assume the right half without the loss of generality). Consequently, the left half of the ellipse including the end points $(0, 1, 0)$ and $(0, -1, 0)$ correspond to a loop since the two end points are mapped to the same $p_0 = (x, y, z)$. Notice that p_0 is also on another degenerate curve connecting the two infinite points corresponding to the two solutions of $f_0 = g_0 = 0$. In this case, k is not constant along the loop. Otherwise, $F = F_0 + \frac{\alpha^2+\beta^2+\gamma^2}{k}$ will share a common factor with $\alpha^2 - \gamma^2 = 0$.

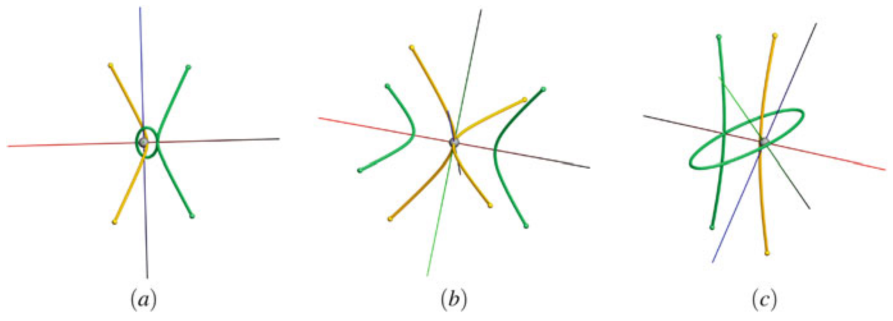


Fig. 2 The tensor field in (**a**) contains a degenerate loop. However, this is a structurally unstable case as it requires two degenerate curves to intersect. The tensor field in (**b**) contains two degenerate curves that intersect at a triple-degenerate point. Again, this is a structurally unstable case. Finally, the tensor field in (**c**) also contains a loop, although this is a different case from the one shown in (**a**)

Figure 2a shows one such case. In the case of four real solutions, there are two solutions on each half of the ellipse. In this case, two degenerate curves of the same type will intersect at a point corresponding to $(0, 1, 0)$ and $(0, -1, 0)$ while the other two degenerate curves do not intersect. Note that the intersection point is *not* triple degenerate. See Fig. 2b for one such example tensor field.

If f and g_0 have a common factor for some $k_0 \in \mathbb{R}$ (k_0 could be ∞). In this case, a degenerate loop can occur (Fig. 2c), along which $k = k_0$ is constant. Note that this is different from the aforementioned case of a degenerate loop where f and g_0 share no common factor for any k. In that case, there is degenerate loop along which k is not constant.

To summarize, under structurally stable conditions the number of degenerate curves in a tensor field $LT(x, y, z) = T_0 + xT_x + yT_y + zT_z$ is the same as the field $xT_x + yT_y + zT_z$. No degenerate loops exist and no two degenerate curves intersect. However, the pathological cases can occur under structurally unstable cases, such as the existence of a degenerate loop or two intersecting degenerate curves. However, even in those cases the number of degenerate curves for the non-homogeneous case is the same as that in the homogeneous case.

5 Conclusion

In this paper we address a fundamental question regarding the topology of 3D linear tensor fields, i.e., the number of degenerate curves in the field. We enumerate all possible scenarios and show that under structurally stable conditions there are at least one and at most four degenerate curves in a linear 3D tensor field, all of which must end in ∞. Degenerate loops and intersecting degenerate curves of the same type can occur, although only under structurally unstable conditions.

In the future we plan to expand our analysis to polynomial tensor fields. Furthermore, we plan to study bifurcations in 3D tensor fields.

Acknowledgements We wish to thank Christine Escher for valuable discussions concerning Veroness maps. We are also grateful for the constructive comments from our anonymous reviewers. Yue Zhang and Eugene Zhang are partially supported by National Science Foundation Awards IIS 15662236 and 1619383, respectively.

References

1. Delmarcelle, T., Hesselink, L.: Visualizing second-order tensor fields with hyperstream lines. IEEE Comput. Graph. Appl. **13**(4), 25–33 (1993)
2. Delmarcelle, T., Hesselink, L.: The topology of symmetric, second-order tensor fields. In: Proceedings IEEE Visualization, pp. 140–147 (1994)
3. Fulton, W.: Algebraic Curves. Mathematics Lecture Note Series. W.A. Benjamin, Reading, MA (1974)
4. Hesselink, L., Levy, Y., Lavin, Y.: The topology of symmetric, second-order 3D tensor fields. IEEE Trans. Vis. Comput. Graph. **3**(1), 1–11 (1997)
5. Kratz, A., Auer, C., Stommel, M., Hotz, I.: Visualization and analysis of second-order tensors: Moving beyond the symmetric positive-definite case. Comput. Graph. Forum **32**(1), 49–74 (2013)
6. Tricoche, X., Kindlmann, G., Westin, C.F.: Invariant crease lines for topological and structural analysis of tensor fields. IEEE Trans. Vis. Comput. Graph. **14**(6), 1627–1634 (2008)
7. Zheng, X., Pang, A.: Topological lines in 3D tensor fields. In: Proceedings IEEE Visualization, pp. 313–320 (2004)
8. Zheng, X., Parlett, B.N., Pang, A.: Topological lines in 3D tensor fields and discriminant Hessian factorization. IEEE Trans. Vis. Comput. Graph. **11**(4), 395–407 (2005)

Part V
Coherent Structures

Hierarchical Watershed Ridges for Visualizing Lagrangian Coherent Structures

Mingcheng Chen, John C. Hart, and Shawn C. Shadden

Abstract Lagrangian coherent structures provide insight into unsteady fluid flow, but their construction has posed many challenges. These structures can be characterized as ridges of a field, but their local definition utilizes an ambiguous eigenvector direction that can point in one of two directions, and its ambiguity can lead to noise and other problems. We overcome these issues with an application of a global ridge definition, applied using the hierarchical watershed transformation. We show results on a mathematical flow model and a simulated vascular flow dataset indicating the watershed method produces less noisy structures.

1 Introduction

The successful visualization of a large complex scientific dataset often relies on the ability to emphasize structure hidden within it. This is particularly true of flow datasets that in their most basic form contain a velocity vector at each point in space, essentially doubling the dimensionality of the dataset, which confounds an observer's ability to perceive the data as a whole. Moreover, in unsteady flow applications, instantaneous rate of change information becomes less directly relevant to visualize, as the more salient flow information is contained by *Lagrangian* measures that intrinsically incorporate the integrated flow behavior.

A variety of analysis techniques can simplify a flow dataset by recognizing and displaying structures representing similar flow characteristics. The recent and compelling method of Lagrangian coherent structures [11, 20] reveals the boundaries of regions of shared characteristics for unsteady fluid flow.

M. Chen • J.C. Hart (✉)
University of Illinois at Urbana-Champaign, Champaign, IL, USA
e-mail: mchen50@illinois.edu; jch@illinois.edu

S.C. Shadden
University of California, Berkeley, CA, USA
e-mail: shadden@berkeley.edu

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_14

Lagrangian coherent structures can be defined in a variety of different ways (e.g. [5, 6, 21]), but have been commonly visualized as the ridges of a scalar field, such as the finite-time Lyapunov exponent (FTLE), indicating the divergence of neighboring pathlines in a time-varying flow. Ridges are features typically derived from the second derivatives of the field, and so for common datasets are susceptible to noise and other issues [17].

A current commonly used approach to extract ridges from datasets are local, based on a marching cubes fit of the local ridge configuration in a cell from the FTLE data at the vertices. The local definition of a ridge is often based on a matrix eigenvector, which only indicates the orientation of a line, but the ambiguity created by the fact that $\mathbf{e}$ and $-\mathbf{e}$ are both equally valid eigenvectors can lead to an orientation ambiguity when detecting the ridge surface numerically. This ambiguity can manifest as spurious false positives and other noise in the ridge surface extracted by local methods such as marching ridges often used for LCS extraction [17].

Watershed methods provide a global approach to extract topological structures from datasets. Sahner et al. [16] describe both a non-global "continuous" watershed approach that traces ridges as separatrices in the Morse structure, as well as a global "discrete" watershed transformation, and use the global watershed transformation to extract vortex and strain skeletal surfaces. We similarly propose and demonstrate watershed separatrix surface extraction for the visualization of flow structure, but for LCS instead of vortex/strain skeletal surfaces, and using a hierarchical watershed to filter out spurious details, to more clearly define the boundaries between neighboring watersheds as a global sea level rises.

These global watersheds can miss some spurious ridge features and can also produce small disjoint ridges due to noise and small field undulation. As stated in Sahner et al. [16], every watershed boundary corresponds to a height ridge or valley, but they do not necessarily coincide and furthermore ridges and valleys might exist that lack corresponding watershed boundaries.

This paper specializes the watershed approach for extracting ridges in FTLE data. We apply a region merging criteria similar to topological persistence that ranks ridges based on their configuration relative to neighboring ridges and valleys. This new filtering enabled by a global approach yields improved LCS extraction from scalar field data and better visualization of unsteady flow structure.

2 Lagrangian Coherent Structures

Let $\mathbf{v}(\mathbf{x}, t)$ represent a time-varying velocity function. We denote the flow map $\Phi(\mathbf{x}, t_0, T)$, which takes $\mathbf{x}$ to its new position at time t_0+T by integrating the velocity to trace the point along its trajectory

$$\Phi(\mathbf{x}, t_0, T) = \mathbf{x} + \int_0^T \mathbf{v}(\Phi(\mathbf{x}, t_0, t), t)dt. \tag{1}$$

The Cauchy-Green strain tensor is the positive definite matrix

$$\mathbf{C}(\mathbf{x}, t_0, T) = \left[\frac{\partial \Phi(\mathbf{x}, t_0, T)}{\partial \mathbf{x}}\right]^T \left[\frac{\partial \Phi(\mathbf{x}, t_0, T)}{\partial \mathbf{x}}\right], \tag{2}$$

and measures finite-time strain of infinitesimal line elements in the fluid. The maximum separation rate is achieved when $\Delta\mathbf{x}$ is parallel to the major eigenvector of $\mathbf{C}(\mathbf{x}, t_0, T)$. The finite-time Lyapunov exponent (FTLE) measures this maximum separation rate as

$$f(\mathbf{x}) = \frac{\ln \lambda_{max}(\mathbf{C}(\mathbf{x}, t_0, T))}{2|T|}, \tag{3}$$

for points $\mathbf{x}$ at a given time t_0 over a given time interval T. Lagrangian coherent structures are often obtained as ridges of FTLE, but their specific definition relies on the particular definition of "ridge" that is used.

The height ridges of a scalar field f are defined as the points satisfying

$$\frac{\partial f}{\partial \mathbf{e}_1} = 0 \tag{4}$$

$$\frac{\partial^2 f}{\partial \mathbf{e}_1^2} < 0 \tag{5}$$

where f is a scalar function and $\mathbf{e}_1$ is either the minimum [2] or largest magnitude [9] eigenvector of the Hessian of f. The C-ridges of a scalar field f are defined similarly, except $\mathbf{e}_1$ is the major eigenvector of the Cauchy-Green tensor (2) [18] based on "normally hyperbolic" LCS [6]. In this work we do not target hyperbolic LCS per se, but the more generic FTLE ridge.

Lagrangian coherent structures can be revealed by a continuation method that tracks the surface from one or more seed points placed at FTLE local maxima [18]. The surface grows from these seed points by integrating a tangent plane orthogonal to the major eigenvector of the Cauchy-Green tensor. While the algorithm is shown to be quite efficient, it required at least one seed point on every LCS component, and multiple seeds on the same component could lead to duplicated surfaces.

LCS can also be revealed through a marching ridges technique [17]. Marching ridges [4] is a variant of the marching cubes isosurface technique [10] used when the orientation needed to define the isosurface is inconsistently specified. An eigenvector $\mathbf{e}$ represents an axis, without preference of $\mathbf{e}$ or $-\mathbf{e}$. Ridge surfaces formulated from eigenvectors often must choose one of these two directions $\mathbf{e}$ or $-\mathbf{e}$ for each eigenvector $\mathbf{e}$. Marching ridges strives to consistently choose eigenvector directions to define an orientable isosurface, but can fail especially when sorted eigenvectors change their order across a single cell.

LCS can also be extracted as a subset of raw features [12] satisfying

$$\det\left(\mathbf{H}^0\mathbf{g}|\ldots|\mathbf{H}^{n-1}\mathbf{g}\right) = 0. \tag{6}$$

The matrix **H** is the Hessian of the scalar function f, but can also be interchanged with the Cauchy-Green tensor. Since (6) does not rely on eigenvectors, raw features can be extracted as an ordinary isosurface, e.g. using marching cubes, except where they may contain non-manifold self intersections. These self intersections can confound the use of raw features to find LCS, as can the numerical instability of (6).

All of these approaches rely on a local definition of ridges and LCS, which makes them susceptible to noise and other algorithm specific issues, such as surface duplication or non-orientability. A global approach would overcome these issues by defining ridges as region boundaries by growing the regions they bound instead of tracking the boundaries between regions.

3 Watershed Segmentation

In image processing, "watershed" methods have long been used for image segmentation [14]. These techniques outline the objects depicted in an image by finding ridges in the image pixel values. For LCS and FTLE ridge extraction, such global watershed methods can reduce the false positives and non-orientability of previous local approaches.

A variety of methods can be applied to a scalar field to separate it into watershed[1] ridges and regions. A region can be defined as the points that flow to the same local minimum but this can be inefficient to compute.

It is more efficient to increment a sea-level threshold value from the global minimum value to the global maximum value. When this threshold value passes a local minimum, it creates a region that grows as the threshold increases. Neighboring regions grow into each other, identifying ridges where they meet.

The Vincent-Soille (V-S) algorithm [22] runs in linear time (proportional to the number of datapoints), and classifies all of the datapoints in an dataset as either ridge or region, labeling non-ridge datapoints by the region to which they belong. The V-S algorithm first bucket sorts the datapoints, then floods each bucket of datapoints in order from least to greatest. Ridges form when a datapoint in the current bucket has neighbors belonging to two regions, but the points in the bucket often form thick regions. Hence a (linear) distance transform is applied to the buckets to compute the distance from the nearest previously defined region (using a circular queue). In

[1]The term watershed comes from hydrology, where it denotes a drainage basin region. Some texts that apply it to dataset analysis incorrectly use it to refer to the ridges separating these basins, and call the basins "catchment basins." To avoid confusion, we will refer to *ridges* that separate *regions*.

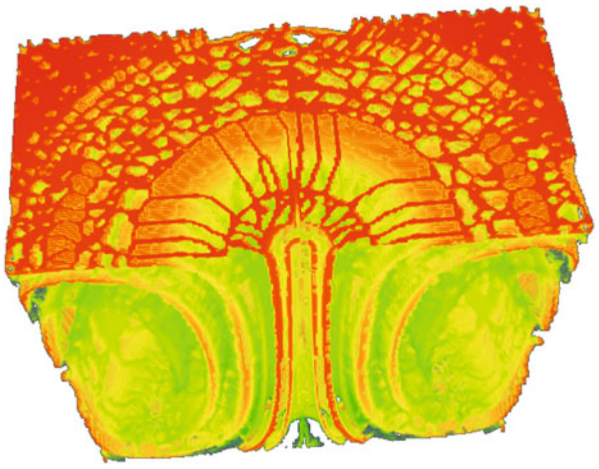


Fig. 1 Oversegmentation of the LCS of a simple convection cell flow due to variation and noise in the FTLE field

the computation of this distance transform, datapoints are assigned to their closest region. If a datapoint is not connected to a closest region, then it forms a new region. If a datapoint is equidistant to multiple regions, then it is classified as a ridge.

When applied to image segmentation, the watershed method typically oversegments, yielding many small regions. When used in LCS applications this leads to a distracting number of insignificant ridges due to noise and subtle variation in the FTLE field, as shown in Fig. 1. Hierarchical watershed methods merge similar regions to form progressively coarser segmentations and have been useful for discerning the important features in a dataset.

One method for constructing a watershed hierarchy is the waterfall transformation [1]. It constructs a graph consisting of nodes representing each ridge segment. The node's value is set to the difference between the median values of the two regions its corresponding ridge separates. Each pair of these nodes is connected with an edge if their corresponding ridge segments border the same region. Then the next level higher in this hierarchical watershed is the watershed of these ridge nodes, using the average ridge value for each node.

Alternatively, regions can be merged based on similarities in their level and/or the characteristics of the ridge separating them. To provide better control for the application of LCS extraction, we utilized this region merging approach to filter unnecessary FTLE ridges.

Our criteria to define a criterion for merging neighboring regions resembles the notion of topological persistence [3]. The persistence of a topological feature indicates how robust it is to perturbation. A well-chosen perturbation in the dataset could remove a ridge, merging the regions it separated into a single region. From the Morse theory viewpoint, this perturbation would merge a saddle point with the minimum of one of the regions. The persistence of a ridge is thus the difference between the lowest point on the ridge (its saddle point) and the larger of its two neighboring minima.

We set a persistence threshold and merge regions separated by ridges that do not meet this threshold. We accelerate this merging with a union-find data structure.

This approach is similar to scale-space hierarchical methods that smooth datasets before performing the watershed transform, using e.g. Gaussian smoothing [8]. Such techniques smooth the data with increasing filter widths to produce coarser levels of the watershed hierarchy. These smoothing operations merge neighboring regions because they cancel saddle-minima pairs.

4 Polygonization

The implicit function theorem shows that the isosurface of a regular isovalue of an analytic field function is a manifold. However, the ridges arising from processing FTLE are not necessarily so, and can include non-manifold junctions that require special methods for surface extraction [7, 13]. The "crease surfaces" analysis [19] for example shows that ridge surfaces consist of manifold patches that meet at non-manifold junctions where the Hessian is degenerate.

We utilize a variation of marching cubes for polygonization of the ridge surfaces. The watershed transform labels each voxel value with a region, and ridges arise in cells whose eight corner vertices (where the voxel values are evaluated) lie in two or more disjoint regions. If a cell's vertices lie in only two regions, we use ordinary marching cubes to polygonize the cell.

For cells that straddle three or more regions, we implement a variation of multiple material marching cubes [23]. For each of the six cell faces, we add a face center vertex and insert a pair of triangles to separate any edges whose vertices lie in separate regions. We then add a vertex at the cell center to connect these triangles. Figure 2 demonstrates the case where all eight cell corners belong to different regions. There are two cases where a vertex at the voxel face center is not needed, as shown for the front face of each example in Fig. 3.

Since the cell corners indicate only the region, and not a scalar value, the vertices used to polygonize a cell are inserted at the center of edges, faces and the cell. This leads to a blocky cuberille appearance of the resulting surface as shown in Fig. 4. We remove these distracting visual artifacts through a smoothing process. We implemented a constrained Laplacian smoothing through conjugate gradient minimization of the energy functional

$$E(\{\mathbf{x}_i\}) = \sum_i ||\mathbf{x}_i - \bar{\mathbf{x}}_i||^2 + \lambda||\mathbf{x}_i - \mathbf{x}'_i||^2 \tag{7}$$

where $\bar{\mathbf{x}}_i$ is the centroid of the vertices neighboring vertex $\mathbf{x}_i$ and $\mathbf{x}'_i$ is its original position in the cuberille polygonization. The parameter λ indicates how much the original position is respected, which we set to 0.01 in our experiments.

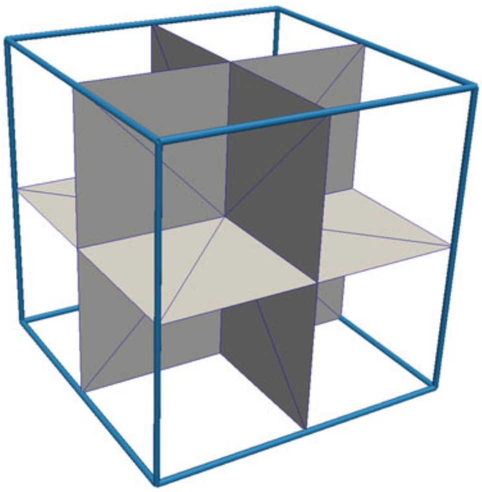


Fig. 2 Polygonization of a cell whose eight corners lie in eight different regions

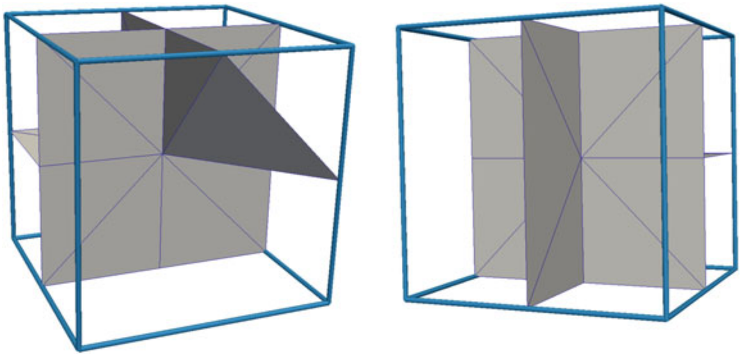


Fig. 3 Two cases where a vertex is not needed at the center of a cell face, shown for the frontmost face

The non-manifold surfaces that arise from multiple region marching cubes require special care for proper smoothing. Branch points whose neighbors may represent ridges between several different regions can confound the smoothing process, as shown in Fig. 5(left).

For each vertex, we find the maximum number of faces that share one of its edges. We limit that vertices neighbors to the ones whose edge is shared by that maximum number of faces. This process smooths non-manifold junctions well, as shown in the example of two intersecting spheres shown in Fig. 5(right).

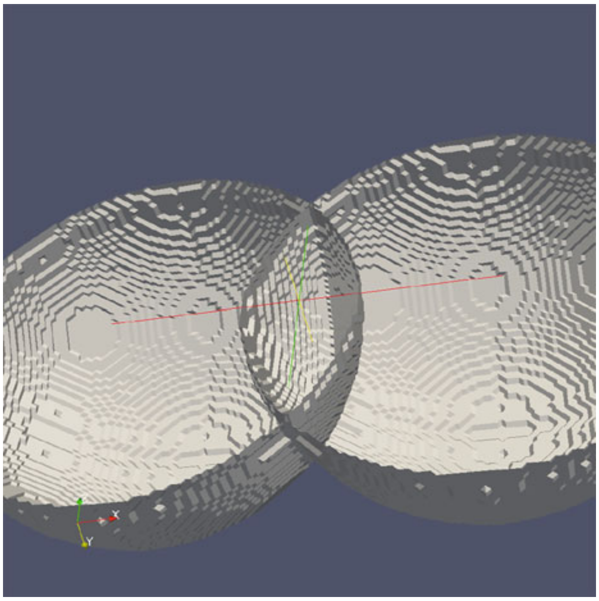


Fig. 4 Blocky artifacts created from multiple region marching cubes for voxels that only indicate region number

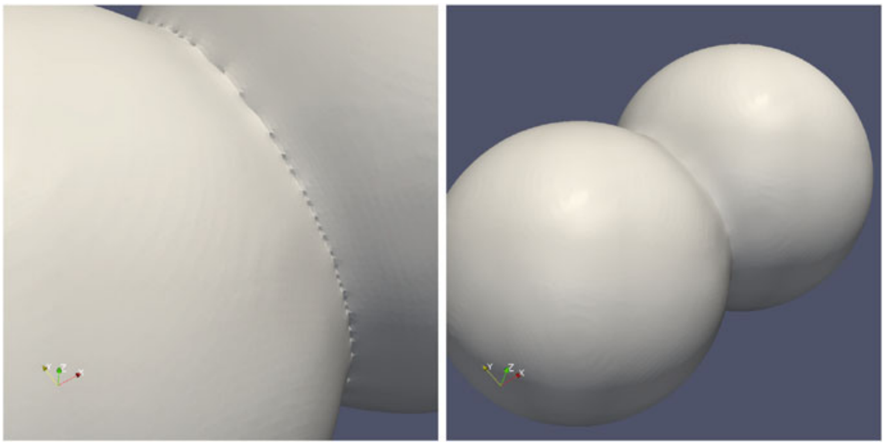


Fig. 5 Ordinary Laplacian smoothing of non-manifold surfaces creates unsmooth results (*left*) which are fixed by limiting the neighborhoods used for Laplacian averaging (*right*)

This Laplacian smoothing approach differs from the one used for multiple material marching cubes (M3C) [23]. Our approach smooths vertices even when they are shared by more than two surfaces, whereas M3C smoothing leaves such vertices stationary. Our approach also does not require additional information that M3C uses, such as which materials are adjacent to a given vertex.

5 Results

We compared the watershed approach to marching ridges on two datasets. The first is an Arnold-Beltrami-Childress (ABC) flow, shown in Fig. 6. The ABC flow dataset yields an FTLE field over a 201^3 voxel array with values ranging from 0.0643 to 0.512.

Figure 7 compares the Lagrangian coherent structures extracted from the FTLE field of the ABC Flow dataset. The marching ridges example follows the recommended noise filtering steps [15], including (1) scalar thresholding (remove ridges with FTLE less than 0.3), (2) least eigenvalue thresholding (remove "flat" ridges with eigenvalue greater than -1.0), and (3) a threshold on the size of connected components (removing disjoint components with less than 500K vertices). The displayed denoised marching ridge result consists of one connected component of 677K vertices, but even with filtering, some noise persists.

The V-S watershed approach yields 797 watershed regions at the lowest level of the watershed hierarchy, which we merge by removing low persistence ridges to 103 regions. The resulting mesh, after smoothing, consists of 1.275M faces and 622K vertices. Figure 7 also shows some smoothed stairstep artifacts that reveal some issues with the merging of watershed regions as discussed further at the end of the section.

The second dataset we used to compare the watershed approach to marching ridges is the abdominal aortic aneurysm (AAA) dataset, shown as an FTLE field in Fig. 8. The AAA dataset is constructed from a pulsatile bloodflow simulation of a lower aorta, reconstructed as a 4.4M tetrahedral mesh. The FTLE field is a $206 \times 231 \times 261$ voxel array, ranging from 0 to 5.29812, using the value -1 indicates the outside of the aorta.

Figure 8 compares the Lagrangian coherent structures extracted from the FTLE field of the AAA dataset. The marching ridges example filtered out ridges smaller than 3.0 (which was the highest setting that prevented holes from forming in the main connected structures), set an eigenvalue threshold of zero (negative values did not improve the result) and filtered out all but the largest connected component. This yielded a mesh of 2.27M faces and 1.29M vertices. As before, the structure is evident but noise is clearly visible.

The V-S watershed algorithm yields 663 regions, which we merge into 199 regions when the height between neighboring regions differs by 0.05 or less. The resulting smoothed mesh consists of 1.80M faces and 854K faces.

The time required for the watershed transform, polygonization and smoothing for the ABC flow and the AAA data is shown in Table 1. The watershed method produces a voxel region classification on the FTLE field. The AAA FTLE field is 53% larger than that of the ABC flow, and its watershed transform takes 49% longer to compute. These watershed voxel regions polygonized to produce the number of vertices listed which follow similar proportions, but the time of the polygonization was not a significant portion of the total time and so is not listed. The smoothing time represents a total of 20 smoothing iterations, but takes 68% longer for the larger

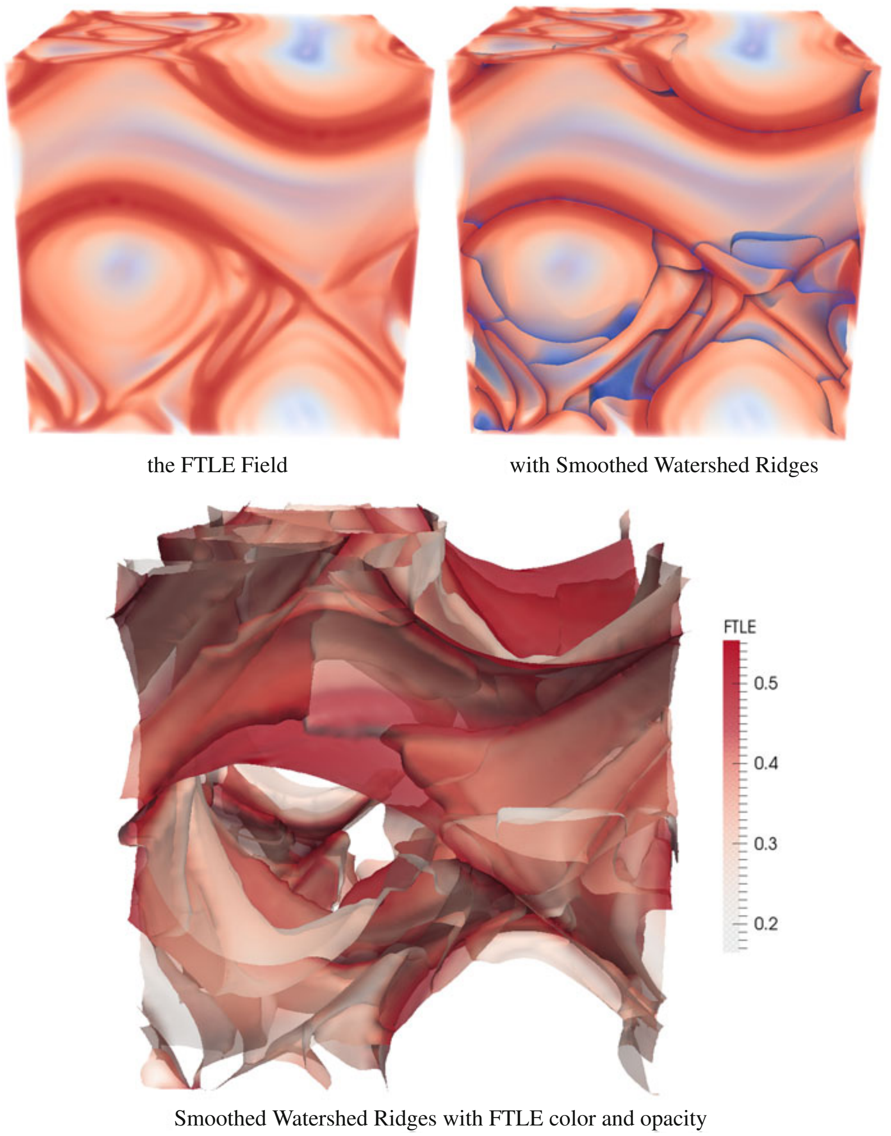

the FTLE Field
with Smoothed Watershed Ridges
FTLE
0.5
0.4
0.3
0.2
Smoothed Watershed Ridges with FTLE color and opacity

Fig. 6 The ABC flow, displayed as the FTLE field data (*upper left*) along with the embedded ridges extracted by the watershed method (*upper right*) and the watershed ridges themselves (*lower center*)

AAA data, because the polygonized ridges are more complex and their vertices have a greater number of neighbors.

One of the most exciting aspects of the global watershed approach is that the regions can be used to coherently color the ridge surfaces, as shown in Fig. 9. The

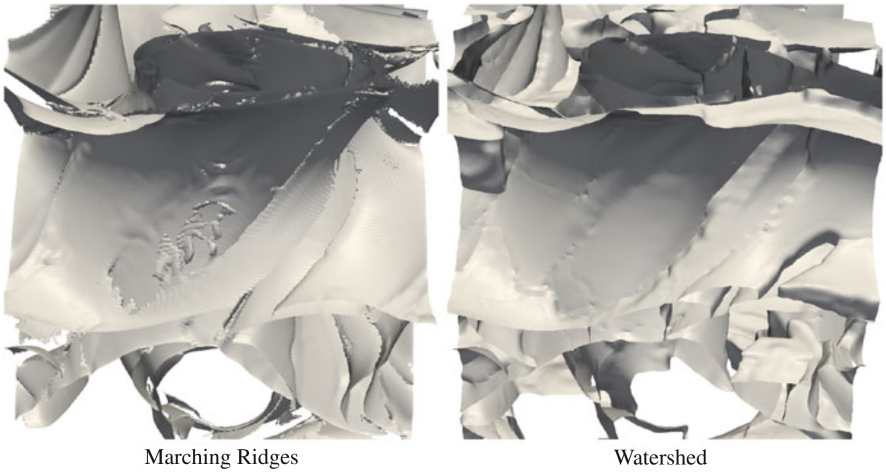


Fig. 7 Lagrangian coherent structures extracted from FTLE of the ABC Flow dataset using the marching ridges method v. the watershed method

choice of color can be arbitrary, but is useful to differentiate the LCS surfaces from each other as they undulate through the flow domain. Such colorings are enabled by two-sided surface shading, but are unavailable for local ridge definitions (e.g. marching ridges) that lack identification of these regions.

The main drawback of the watershed approach is that the initial application of the watershed transformation (before merging) creates significant over-segmentation of the datasets, resulting in many small watersheds. These small watersheds are merged when separated by shallow ridges, but sometimes the shallowness, which we use as the persistence of the ridge, not properly eliminate some shallow ridges even though this persistence based approach works well for most spurious ridges such as are shown in Fig. 1.

As shown in Fig. 10, the ridges detected by the hierarchical watershed approach are largely at the mercy of the fluctuations of the FTLE field. In this example, such an FTLE fluctuation causes the hierarchical watershed merging to follow the wrong shorter ridge instead of keeping the correct longer ridge. This wrong shorter ridge is indeed part of the FTLE field and in fact forms a better defined ridge according to persistence than does the correct ridge, so further work beyond the persistence measure used by the hierarchical watershed method is needed to eliminate these last few pathological cases.

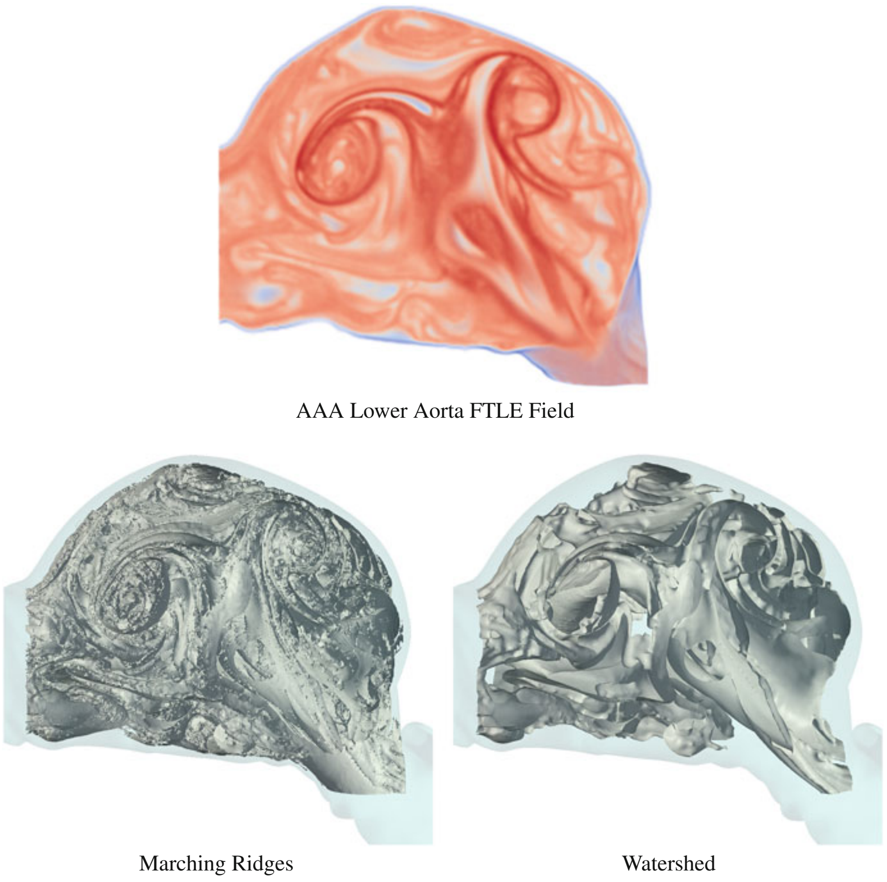


Fig. 8 Lagrangian coherent structures extracted from FTLE (*top*) of the AAA dataset using the marching ridges method (*lower left*) v. the watershed method (*lower right*)

Table 1 Performance of the watershed method for LCS extraction

Dataset	FTLE resolution	Watershed	Mesh vertices	Smoothing
ABC flow	201 × 201 × 201	34.21 s	1,225,558	36.4 s
Patent 96	206 × 231 × 261	50.87 s	1,800,230	61.4 s

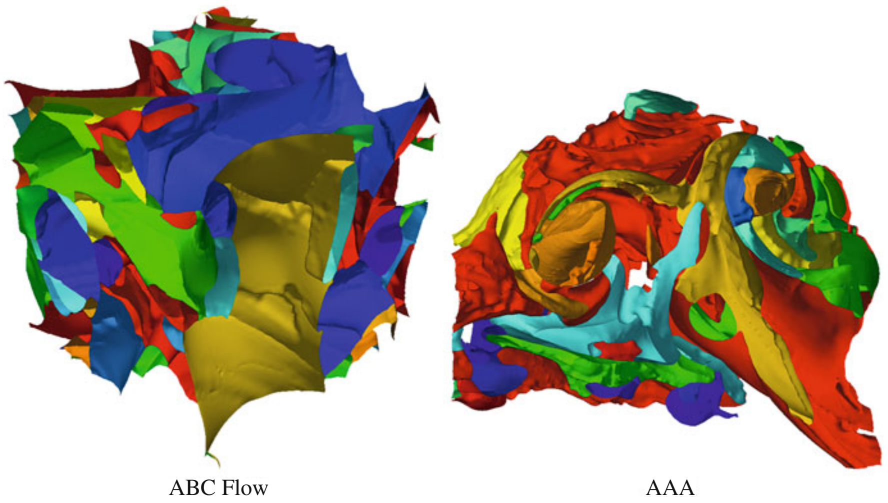


Fig. 9 Lagrangian coherent structures for the ABC flow and Patent 96, displayed using *random colors* assigned by regions, as extracted using our hierarchical watershed approach

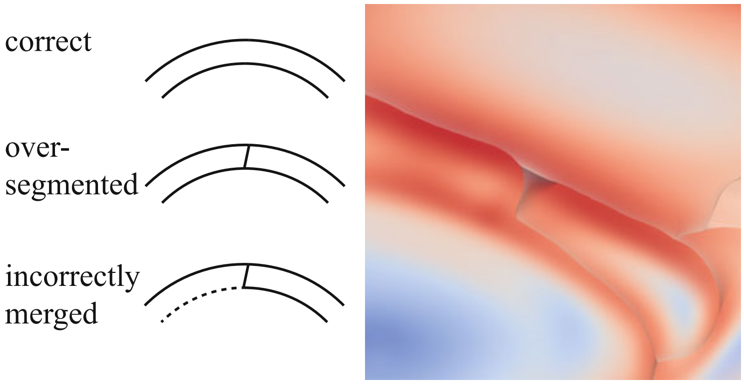


Fig. 10 Hierarchical watershed merging can sometimes merge the wrong regions

6 Conclusions and Further Research

While watershed methods commonly appear in the summaries of ridge extraction methods for Lagrangian coherent structures, they are often dismissed in favor of local methods such as marching ridges. We have shown that their results are often much smoother and less noisy than such local approaches, and should be considered further.

One of the main shortcomings of the watershed approach is that it does not detect ridges that end at a minimum. Such an example might be a ridge descending from the rim to the bottom of a crater. We plan to address such cases with a combination of local and global combinations, using the local ridge definition evaluated on the current sea-level coastline front of the V-S watershed method. This combination of local and global ridge methods could yield the best of both worlds.

We have used a persistence measure as the criterion for hierarchical watershed method to merge watershed regions based on ridge shallowness. This criterion works well in many but not all cases, as illustrated in Fig. 10. Further analysis and experimentation will be needed to explore new merging criteria to better preserve the important ridges for LCS visualization.

We also plan to work on high performance streaming implementations of the V-S and other watershed algorithms, updating earlier such work [14], as we as their applications to larger, out-of-core datasets.

Acknowledgements This work was supported in part by NSF Grants OCI-1047963 and OCI-1047764. We thank Siavash Ameli for his help with this project.

References

1. Beucher, S.: Watershed, hierarchical segmentation and waterfall algorithm. In: Mathematical Morphology and Its Applications to Image Processing, Computational Imaging and Vision, vol. 2, pp. 69–76. Springer, New York (1994)
2. Eberly, D.: Ridges in Image and Data Analysis. Kluwer Academic Publishers, Dordrecht (1996)
3. Edelsbrunner, H., Harer, J.: Persistent homology — a survey. In: Surveys on Discrete and Computational Geometry, Contemporary Mathematics, vol. 453, pp. 257–282. American Mathematical Society, Providence (2006)
4. Furst, J.D., Pizer, S.M.: Marching ridges. In: SIP, pp. 22–26 (2001)
5. Haller, G.: Distinguished material surfaces and coherent structures in three-dimensional fluid flows. Physica D **149**(4), 248–277 (2001)
6. Haller, G.: A variational theory of hyperbolic Lagrangian coherent structures. Physica D **240**(7), 574–598 (2011)
7. Hege, H.C., Stalling, D., Seebass, M., Zöckler, M.: A generalized marching cubes algorithm based on non-binary classifications. Technical Report SC-97-05, Konrad-Zuse-Zentrum (ZIB) (1997)
8. Hucko, M., Sramek, M.: Interactive segmentation of volume data using watershed hierarchies. Winter School Comput. Graph. **20**(3), 217–222 (2012)
9. Lindeberg, T.: Feature detection with automatic scale selection. Int. J. Comput. Vis. **30**(2), 79–116 (1998)
10. Lorensen, W.E., Cline, H.E.: Marching cubes: a high resolution 3D surface construction algorithm. Proc. SIGGRAPH **21**(4), 163–169 (1987)
11. Peacock, T., Haller, G.: Lagrangian coherent structures: the hidden skeleton of fluid flows. Phys. Today **66**(2), 41 (2013)
12. Peikert, R., Sadlo, F.: Height ridge computation and filtering for visualization. In: IEEE Pacific Visualization Symposium, pp. 119–126. IEEE, New York (2008)
13. Reitinger, B., Bornik, A., Beichel, R.: Consistent mesh generation for non-binary medical datasets. In: Bildverarbeitung für die Medizin, pp. 183–187. Springer, New York (2005)

14. Roerdink, J.B., Meijster, A.: The watershed transform: definitions, algorithms and parallelization strategies. Fundam. Inf. **41**(1), 187–228 (2000)
15. Sadlo, F., Peikert, R.: Efficient visualization of lagrangian coherent structures by filtered AMR ridge extraction. IEEE Trans. Vis. Comput. Graph. **13**(6), 1456–1463 (2007)
16. Sahner, J., Weinkauf, T., Teuber, N., Hege, H.C.: Vortex and strain skeletons in Eulerian and Lagrangian frames. IEEE Trans. Vis. Comput. Graph. **13**(5), 980–990 (2007)
17. Schindler, B., Fuchs, R., Barp, S., Waser, J., Pobitzer, A., Carnecky, R., Matkovic, K., Peikert, R.: Lagrangian coherent structures for design analysis of revolving doors. IEEE Trans. Vis. Comput. Graph. **18**(12), 2159–2168 (2012)
18. Schindler, B., Peikert, R., Fuchs, R., Theisel, H.: Ridge concepts for the visualization of Lagrangian coherent structures. In: Topological Methods in Data Analysis and Visualization II, pp. 221–235. Springer, Berlin (2012)
19. Schultz, T., Theisel, H., Seidel, H.P.: Crease surfaces: from theory to extraction and application to diffusion tensor MRI. IEEE Trans. Vis. Comput. Graph. **16**(1), 109–119 (2010)
20. Shadden, S.C.: Lagrangian coherent structures. In: R. Grigoriev (ed.) Transport and Mixing in Laminar Flows: From Microfluidics to Oceanic Currents, Chap. 3, pp. 59–89. Wiley, Weinheim (2012)
21. Shadden, S.C., Lekien, F., Marsden, J.E.: Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. Physica D **212**(3-4), 271–304 (2005)
22. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. Pattern Anal. Mach. Intell. **13**(6), 583–598 (1991)
23. Wu, Z., Sullivan, J.: Multiple material marching cubes algorithm. Int. J. Numer. Methods Eng. **58**(2), 189–207 (2003)

Finite Time Steady 2D Vector Field Topology

Anke Friederici, Christian Rössl, and Holger Theisel

Abstract Vector Field Topology describes the asymptotic behavior of a flow in a vector field, i.e., the behavior for an integration time converging to infinity. For some applications, a segmentation of the flow in areas of similar behavior for a finite integration time is desired. We introduce an approach for a finite-time segmentation of a steady 2D vector field which avoids the systematic evaluation of the flow map in the whole flow domain. Instead, we consider the separatrices of the topological skeleton and provide them with additional information on how the separation evolves at each point with ongoing integration time. We analyze this behavior and its distribution along a separatrix, and we provide a visual encoding for it. The result is an augmented topological skeleton. We demonstrate the approach on several artificial and simulated vector fields.

1 Introduction

Vector Field Topology has been established as one of the standard approaches to visualizing steady vector fields. Its main idea is simple and appealing: separate the field into regions of similar asymptotic flow behavior. This way, even complex flow structures can be represented by a low number of graphical primitives. In addition to this separation, Vector Field Topology has an attractive property in terms of computation: to get the whole segmentation, it is not necessary to consider every point in the domain. Instead, only a few points in the domain have to be touched (critical points, boundary switch points), and a few special stream lines starting from these points (separatrices) have to be computed.

Several approaches have been proposed to extend Vector Field Topology to unsteady fields. The main problem for this is that an asymptotic behavior cannot be analyzed any more: unsteady fields usually allow an integration over a finite time only. Lagrangian Coherent Structures (LCS) provide such a segmentation of the field after a finite integration time. The perhaps most prominent example for LCS are

A. Friederici (✉) • C. Rössl • H. Theisel
University of Magdeburg, Magdeburg, Germany
e-mail: anke.friederici@st.ovgu.de

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*,
Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_15

ridge structures in Finite Time Lyapunov Exponents (FTLE) fields. In general, LCS computation requires a dense computation of the flow map in the whole domain.

In recent years there are approaches to compute LCS of unsteady fields by using steady Vector Field Topology [1, 42]. The main idea is to subtract a certain background flow field (or consider a certain reference frame) and reduce this way the computation of LCS of an unsteady field to the computation of steady Vector Field Topology. While these approaches are appealing, they have a fundamental problem: a segmentation for a finite integration time is computed by considering the asymptotic behavior of another flow, i.e., by considering an integration time converging to infinity. In general, integrating a (modified) field until infinity should not be considered for unsteady fields because it works with information that is not present in the data.

This paper solves the problem mentioned above: we present an approach to a finite time flow segmentation in a steady 2D field where we do not have to evaluate the flow map in the whole domain. (By applying a flow map evaluation everywhere, the potential advantage of the approaches in [1, 42] is lost; in this case one could do an LCS analysis of the original field directly without subtracting a certain flow.) We start with the assumption that the relevant separation takes place along the separatrices even for a finite integration time. For them, we compute the separation perpendicular to the flow either in a local or in a discretized global way. The results are characteristic functions (here called *separation functions*) which provide information about the separation along a separatrix. After evaluating these functions for a finite time and setting them in relation to their behavior when integrating towards infinity, we provide a simple visual encoding for them. In summary, we keep the benefits of steady vector field topology, which is stable and well-defined, while adding a scalar separation quantity.

2 Related Work

Topological methods for 2D vector fields have been introduced to the visualization community in [10]. Later they were extended to higher order critical points [27], boundary switch points [2], and closed separatrices [43]. In addition, topological methods have been applied to simplify [2, 3, 34, 35], smooth [41], compress [16, 17, 32] and construct [31, 39] vector fields. 3D topological feature are considered in [6, 11, 18, 19, 33, 38]. State-of-the-Art-Reports on topological methods for flow visualization can be found in [13, 20].

Topological methods can be applied only to steady vector fields because they require an integration until infinity. For unsteady fields, Lagrangian Coherent Structures (LCS) have been established to find regions of homogeneous flow behavior. One of the most prominent approaches for this is the computation of ridge structures in FTLE fields, as introduced by Haller [7, 9]. To consider spatial separation only, Pobitzer et al. [21] weighted FTLE values by their angle to the separation direction. FTLE ridges were proposed for a variety of applications [8, 14, 29, 40]. Shadden et al. [28] showed that ridges of FTLE are approximate material structures, i.e., they converge to material structures for increasing integration times. This fact was used in [25, 36] to extract topological structures and in [15] to accelerate the FTLE computation in 2D flows. Also in the visualization community, different approaches have been proposed to increase performance, accuracy and usefulness of FTLE as a visualization tool [4, 5, 23, 24, 26].

In recent years approaches have evolved that aim at finding suitable moving frames of the underlying coordinate system to study the flow [1, 42]. This way, finite-time studies of time-dependent fields is lead back to a topological analysis of a derived steady field. This paper targets towards these approaches: by being able to analyze the finite-time behavior of steady fields without a dense sampling of the flow map, we make steady topology an appropriate tool also for a finite-time analysis.

3 The Approach

We start with an argumentation why we restrict the search for separating structures in steady flows to separatrices, i.e., the separating structures for integration times converging to infinity.

While it is not formally proven yet that separatrices and separation structures of LCS methods coincide [22], they behave similarly in general. The FTLE field and topological skeleton of our most complex dataset is shown in Fig. 1. As can be seen, the FTLE ridges and separatrices overlap.

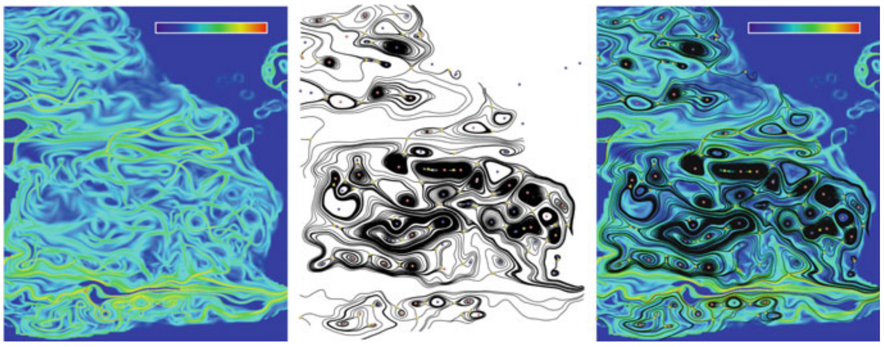

(a) FTLE (b) Topological skeleton (c) Combined image

Fig. 1 FTLE field and topological skeleton of the ocean dataset. (**a**) Maximum of forward and backward integrated FTLE. (**b**) Topological skeleton integrated starting from the saddles. (**c**) Topological skeleton superimposed on FTLE image. The separatrices coincide with the FTLE ridges

Notation

In the following, we consider a 2D steady vector field $\mathbf{v}$. Let $\mathbf{J}$ be its Jacobian. We assume that $\mathbf{J}$ is bounded, i.e., $\|\mathbf{J}\|$ does not exceed a certain fixed maximal value in the whole domain. Furthermore, let $\phi(\mathbf{x}, \tau) : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^2$ denote the flow map of $\mathbf{v}$. Then a stream line is a parametric curve $\phi(\mathbf{x}, \tau)$ starting from $\mathbf{x}$. The gradient of the flow map is denoted $\nabla\phi$. Furthermore we use the normalized perpendicular vector field

$$\mathbf{w}(\mathbf{x}) = \frac{1}{||\mathbf{v}||} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \cdot \mathbf{v}(\mathbf{x}) \,. \tag{1}$$

Note that $\mathbf{w}$ has unit length and is defined for non-critical points only. This is not a serious restriction because stream lines starting in critical points do not leave them and are not considered.

The Separation Function for Stream Lines

Given a stream line, we analyze the separation along it. For this we focus on a separation perpendicular to the flow while removing the separation along the flow. To consider the separation along the stream line $\phi(\mathbf{x}, \tau)$, we integrate a second stream line $\phi(\mathbf{x}_1, \tau)$ with the starting point

$$\mathbf{x}_1 = \mathbf{x} \ + \ \varepsilon_1 \, \mathbf{w}(\mathbf{x})$$

Then we consider $\varepsilon(\tau)$ as the distance of $\phi(\mathbf{x}_1, \tau)$ to the straight line $\phi(\mathbf{x}, \tau) + \lambda \, \mathbf{v}(\phi(\mathbf{x}, \tau))$ for $\lambda \in \mathbb{R}$:

$$\varepsilon(\tau) = \mathbf{w}(\phi(\mathbf{x}, \tau))^T (\phi(\mathbf{x}_1, \tau) - \phi(\mathbf{x}, \tau)).$$

Figure 2 illustrates this. We define the *separation function* of the stream line $\phi(\mathbf{x}, \tau)$ as

$$s(\mathbf{x}, \tau) = \lim_{\varepsilon_1 \to 0} \ln \frac{\varepsilon(\tau)}{\varepsilon_1} \,. \tag{2}$$

Keeping in mind that

$$\lim_{\varepsilon_1 \to 0} \frac{\phi(\mathbf{x}_1, \tau) - \phi(\mathbf{x}, \tau)}{\varepsilon_1} = \nabla\phi(\mathbf{x}, \tau) \cdot \mathbf{w}(\mathbf{x}) \,,$$

(2) can be written as

$$s(\mathbf{x}, \tau) = \ln \left(\mathbf{w}(\phi(\mathbf{x}, \tau))^T \nabla\phi(\mathbf{x}, \tau) \cdot \mathbf{w}(\mathbf{x}) \right) \,. \tag{3}$$

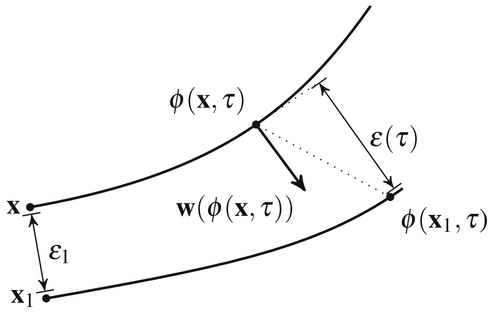


Fig. 2 Configuration for defining $s(\mathbf{x}, \tau)$

Equation (2) already gives a way to numerically compute the separation function: choosing a sufficiently small ε_1 for estimating the directional derivative of ϕ, the term $\ln \frac{\varepsilon(\tau)}{\varepsilon_1}$ is an approximation of s. However, its computation depends strongly on the choice of ε_1. If ε_1 is too small, (2) may run into numerical problems. If ε_1 is too large, $\mathbf{x}_1$ tends to move out of the linear neighborhood of $\mathbf{x}$. Fortunately, there is a localized version of s which avoids a discretization of the directional derivative of the flow map:

$$s(\mathbf{x}, \tau) = \int_0^\tau \mathbf{w}(\phi)^T \mathbf{J}(\phi)\, \mathbf{w}(\phi)\; dr \tag{4}$$

with $\phi = \phi(\mathbf{x}, r)$. In order to prove that (3) and (4) are identical, we have to show that

$$\frac{\partial}{\partial \tau}\left(\ln\left(\mathbf{w}(\phi)^T \nabla\phi\, \mathbf{w}(\mathbf{x})\right)\right) = (\mathbf{w}(\phi))^T \cdot \mathbf{J}(\phi)\mathbf{w}(\phi)$$

with $\phi = \phi(\mathbf{x}, \tau)$. This can be shown by a straightforward application of elementary differentiation rules.

Note that (4) has some similarities to the local FTLE computation in [12]. In fact, [12] integrates the gradient of the flow map by a repeated matrix multiplication, leading to the fact that integrated measures can exponentially grow/shrink with increasing integration time. Contrary, (4) is a repeated addition of scalar values for the numerical integration.

Also note that s does not necessarily capture the maximal distortion in the neighborhood of a particle. Instead, it describes the distortion into one particular direction: perpendicular to the flow direction. For the evaluation of s on separatrices only, this derivative would generally give a close to maximal distortion.

Properties of the Separation Function We list some properties of s. They can be shown by considering either (2)–(4).

- $s(\mathbf{x}, 0) = 0$ (follows from (2) and $\varepsilon(0) = \varepsilon_1$).
- $s(\mathbf{x}, \tau)$ grows at most linearly with increasing τ. (To show this, we have to show that $\frac{ds}{d\tau}$ is bounded. This follows directly from (4) and the boundedness of $\mathbf{J}$.)

- s is additive: $s(\mathbf{x}, \tau_1 + \tau_2) = s(\mathbf{x}, \tau_1) + s(\phi(\mathbf{x}, \tau_1), \tau_2)$ [follows from (4)].
- For two points $\mathbf{x}, \mathbf{y}$ on the same stream line, their separation functions differ only by a translation: let $\mathbf{y} = \phi(\mathbf{x}, \tau_{\mathbf{y}})$. Then $s(\mathbf{y}, \tau) = s(\mathbf{x}, \tau + \tau_{\mathbf{y}}) - s(\mathbf{x}, \tau_{\mathbf{y}})$ (follows from the point above).
- s is inverted under backward integration: $s(\mathbf{x}, \tau) = -s(\phi(\mathbf{x}, \tau), -\tau)$ [follows from (4)].

The Separation Function for Separatrices

Up to now, the separation function was defined for an arbitrary streamline. In this section, we show that the separation function has a special behavior for separatrices induced by a saddle point. Let $\mathbf{x}$ be on a separatrix, i.e., $\phi(\mathbf{x}, \tau)$ converges under forward integration to a saddle point $\mathbf{c}$ for $\tau \to \infty$. Then $s(\mathbf{x}, \tau)$ converges to a linear function for $\tau \to \infty$:

$$\lim_{\tau \to \infty} s(\mathbf{x}, \tau) = a\, \tau + b(\mathbf{x}) \tag{5}$$

where the slope a is determined by the Jacobian $\mathbf{J_c}$ of $\mathbf{v}$ in the saddle point $\mathbf{c}$. Let $\lambda_1 < 0 < \lambda_2$ be the eigenvalues of $\mathbf{J_c}$ and $\mathbf{e}_1, \mathbf{e}_2$ be the corresponding normalized eigenvectors. Then

$$a = (\mathbf{e}_1^{\perp})^T\, \mathbf{J_c}\, \mathbf{e}_1^{\perp} \tag{6}$$

where $\mathbf{e}_1^{\perp}$ is the vector $\mathbf{e}_1$ rotated by $\frac{\pi}{2}$ that is perpendicular to $\mathbf{e}_1$.

Figure 3 gives an illustration. The proofs of (5) and (6) are obtained by in the following way: since we are interested in the asymptotic behavior around a saddle, we can consider a linear vector field having a saddle with the desired Jacobian at the desired locations. Then the flow map and its derivatives can be written in a closed form, allowing to show (5) and (6) by simple computations.

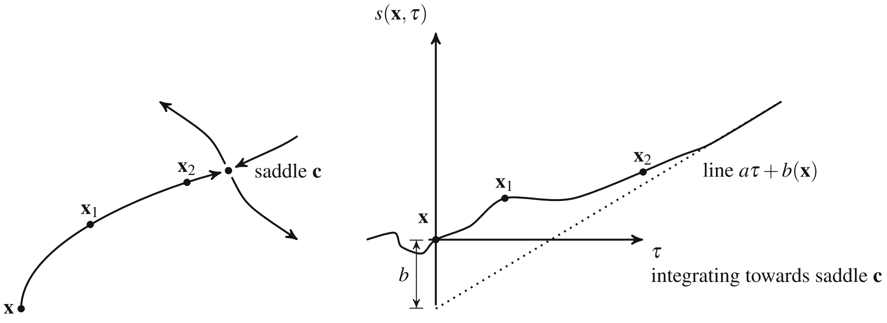


Fig. 3 Behavior of $s(\mathbf{x}, \tau)$ for a separatrix

The parameter $b(\mathbf{x})$ can be considered as a measure on how far s is away from its final linear behavior as $\tau \to \infty$. We will use this parameter for characterizing the separation. Note that for a particular point $\mathbf{x}$ on the separatrix, the computation of $b(\mathbf{x})$ is cumbersome because it requires an accurate integration towards the saddle. In order to get a more stable computation of $b(\mathbf{x})$, we formulate:

Theorem 1 *Given are two points* $\mathbf{x}$, $\mathbf{y}$ *on the same separatrix such that* $\mathbf{y} = \phi(\mathbf{x}, \tau_\mathbf{y})$. *Then*

$$b(\mathbf{y}) = b(\mathbf{x}) + s(\mathbf{x}, \tau_\mathbf{y}) - a\ \tau_\mathbf{y}.$$

The sketch of the proof is in Fig. 4. Theorem 1 provides a way to compute b along a whole separatrix: given a saddle point $\mathbf{c}$, we start the integration of the separatrix at a point $\mathbf{x}_0 = \mathbf{c} + \varepsilon_0\ \mathbf{e}_1$. Note that ε_0 has to be chosen small enough such that $\mathbf{x}_0$ can be assumed to be in the linear neighborhood of $\mathbf{c}$. This assumption gives $b(\mathbf{x}_0) = 0$. From this we compute the separatrix by backward integration, i.e., we consider $\phi(\mathbf{x}_0, \tau)$ for $\tau \in]-\infty, 0]$. This way we get

$$b(\phi(\mathbf{x}_0, \tau)) = \begin{cases} 0 & \text{for } \tau \geq 0 \\ s(\mathbf{x}_0, \tau) - a\ \tau & \text{for } \tau < 0 \end{cases}$$

where a is computed as (6). Note that $b(\mathbf{x})$ does not depend on the location of $\mathbf{x}$ as long as $\mathbf{x}_0$ is in the linear neighborhood of the saddle.

Properties and Visualization of b

The number $b(\mathbf{x})$ for a point $\mathbf{x}$ on a separatrix is a measure of how the separation behaves when $\mathbf{x}$ is integrated towards the saddle. It gives a measure of how far

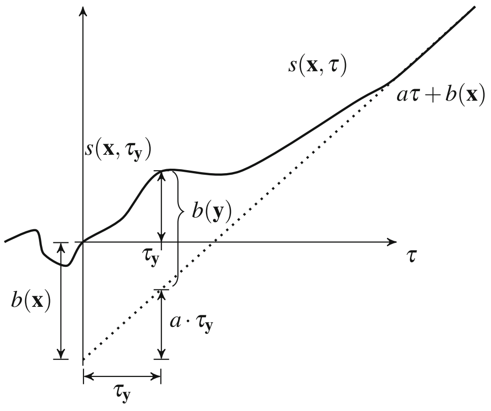


Fig. 4 Sketch for proof of Theorem 1

the separation is away from the final asymptotic separation after integrating only a finite time. In a linear neighborhood of the saddle we have $b(\mathbf{x}) = 0$: starting an integration there leads to an exponential separation that is determined by the eigenvalues of the Jacobian of the saddle. For $\mathbf{x}$ in a certain distance of the saddle, $b(\mathbf{x})$ is a measure on how far $\mathbf{x}$ is from the final exponential separation for shorter integration time. Note that $b(\mathbf{x})$ is *not* a local measure: instead it contains the essence of the separation behavior for a finite integration time when starting at $\mathbf{x}$.

In order to visualize b, we use the visual metaphor of a wall: instead of a 2D separating line, we render an extruded 3D surface where its height is connected to the strength of separation. For this, we introduce as height of the wall:

$$h(\mathbf{x}) = a\, e^{k\, b(\mathbf{x})}$$

where $k > 0$ is a degree of freedom for the visualization. This way we have $h(\mathbf{x}) = a$ in a linear neighborhood of a saddle, denoting the strength of the local separation at the saddle. The parameter k indicates how strong the height diminishment of the wall is when $b(\mathbf{x})$ deviates from 0. A small k gives that the height of the wall reflects strong deviation of b, a larger k brings the focus on small deviations of b from 0. We show examples of different choices of k in Sect. 4.

Details and Implementation

We determine critical points and classify saddles in a preprocess using the standard method [37] implemented in the Amira software [30]. They determine the starting points of separatrices. The computation of $s(\mathbf{x}, \tau)$ requires the numerical solution of an initial value problem.

We apply a standard Runge-Kutta method in two passes. The first step is a standard streamline integration that yields a discrete curve representation of $\phi(\mathbf{x}, \tau)$ for starting at a point $\mathbf{x}_0$ near a saddle in the initial direction given by some eigenvector of the Jacobian at the critical point. We use a fourth-order Runge-Kutta method with step size adaptation that provides the streamline ϕ as a cubic C^1-continuous spline. The second step integrates the projection of the directional derivative, i.e., we apply the same Runge-Kutta method for the numerical integration of a scalar field. Finally, $b(\mathbf{y})$, and thus the height function $h(\mathbf{y})$, is computed by evaluating the separation function $s(\mathbf{x}_0, \tau)$ for $\mathbf{x}_0$ and $\tau_{\mathbf{y}}$. Note that the evaluation is backwards in time (see discussion of Theorem 1), i.e., there is no need for a reparametrization of ϕ or s as both were in fact computed by backward integration.

For visualization, we use standard line integral convolution to provide a global overview of the vector field $\mathbf{v}$ an underlying image. The separatrices are planar curves in the image. We lift each separatrix by interpreting the values $h(\mathbf{x})$ as height; this gives a second curve. We render all curves as tubes and connect the separatrix and its lifted counterpart by semitransparent surfaces. The critical points are the start and end points of curves and are emphasized by cylindrical structures with height h. Figure 6 shows an example.

4 Results

We compute and visualize the separation function for a number of vector fields.

Simple Example The first vector field is a piecewise linear function on a small 12×3 grid. The construction of the dataset is shown in Fig. 5. For each vertex of the grid, we prescribe a vector such that bilinear interpolation in cells forms source (left) and saddle (right), and the center region yields a converging flow. The next Fig. 6 visualizes the separation function as described in the previous section. The white tubes show the separatrix in the plane and lifted by the height function h, and all pairs of curve are connected by semitransparent walls. Figure 7a shows the graph of the separation function $s(\mathbf{x}, \tau)$ for the separatrix from the source to the saddle. It shows, from left to right, that the flow is diverging from the source in the beginning. Then there is region without separation followed by a region with converging flow. As the separatrix approaches the saddle, s shows asymptotic behavior of a line. The graph of h is parametrized over integration time. To visualize the distortion of the space-time map $\tau_{\mathbf{x}}$, Fig. 5 shows positions of equally spaced samples $\mathbf{x}_0, \ldots, \mathbf{x}_3$ of the separatrix, and Fig. 7a shows the associated times τ_{vx_i}.

Figure 7b compares the graphs of the height function h for different values of the user parameter k.

Random Grid Figure 8b shows results for a vector field generated from bilinear interpolation of vectors that were randomly chosen at the vertices of a 5×5 grid.

Slice of Rayleigh-Bénard Convection Cells Figure 8a visualizes separation for a vertical slice through a Rayleigh-Bénard convection. The selected region of interest shows 16 critical points. The vector field is given as samples on a regular 64×64 grid, which are interpolated bilinearly.

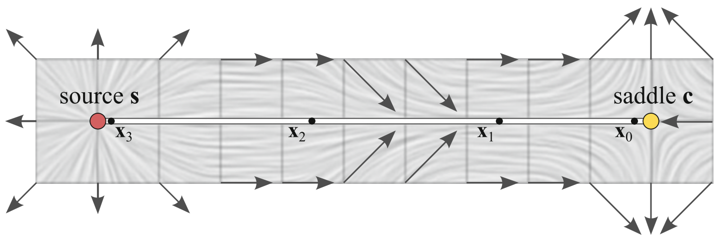


Fig. 5 Construction of a simple vector field on a 12×3 grid with a source and a saddle and a region of converging flow between. The velocity at the vertices are interpolated bilinearly within grid cells

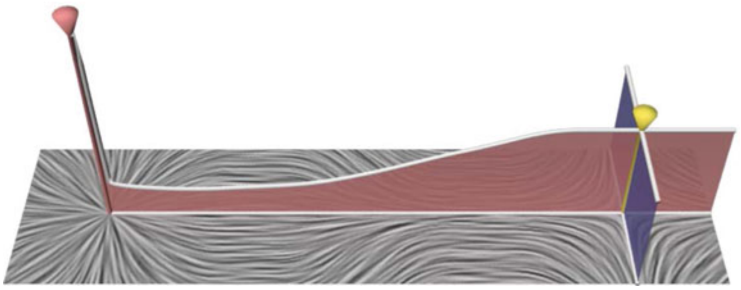
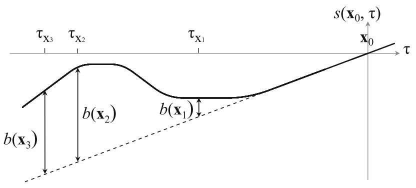
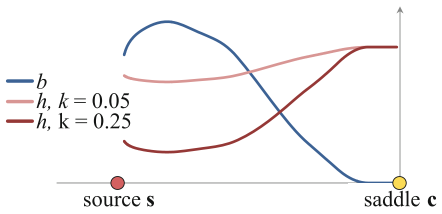


Fig. 6 Visualization of separation function as "height" h. The planar and lifted separatrices are connected by semitransparent walls

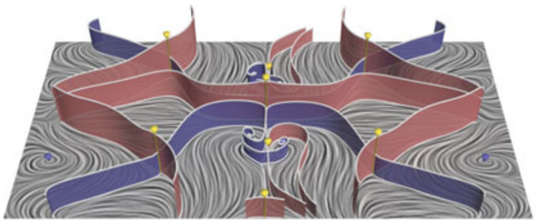


(a) Graph of $s(\mathbf{x}, \tau)$

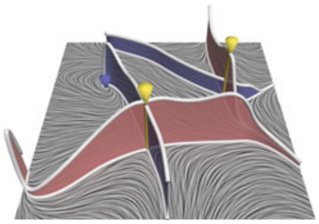


(b) Graphs of b and h for varying k

Fig. 7 Behavior of s and h. (**a**) Graph of separation function s along the separatrix. (**b**) Graphs of two height function h for different values of k. Both heights h are a exponential function of b, which is also shown



(a) Rayleigh-Bénard convection cell dataset



(b) Random vector field

Fig. 8 Visualization of two datasets. (**a**) Vertical slice through a Rayleigh-Bénard convection cell dataset with 16 critical points. (**b**) Separation walls for a random vector field interpolated on a 5×5 grid

Slice of an Ocean Simulation Figure 9 shows one slice from a simulation of the south pacific ocean. One hundred and fifty critical points are present, from which 277 separatrices have been integrated. This number of walls being shown allows for an overall image of the dataset as well as the analysis of single separating structures. In the front of Fig. 9a the walls are significantly higher than in the back, while the separation in the regions further away is so small that no wall is visible. This way, the highlight is set to the relevant structures with high separation.

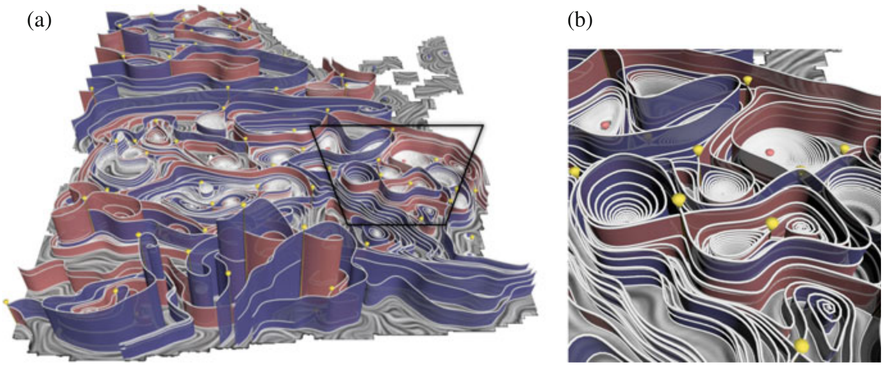


Fig. 9 Simulated ocean dataset with 150 critical points. (**a**) Visualization of the full dataset. (**b**) Close-up on swirling structures

Figure 9b shows a close-up on the walls of some swirling regions. Their behavior when moving near the critical point varies: some stay on a constant level, while others show diminishing separation.

5 Discussion and Limitations

The separation function s depends on the choice of ϵ_0, which tells how far $\mathbf{x}_0 = \mathbf{c} + \epsilon_0 \mathbf{e}_1$ is located from the saddle, which in turn determines, how far the separatrix is integrated forward in time starting from the source. For $\epsilon_0 \to 0$ we have $\tau_{\mathbf{x}_0} \to \infty$. At the same time we constructed s such that it converges to a linear function as $\phi(\mathbf{x}, \tau)$ approaches $\mathbf{c}$. The slope of this line decreases for decreasing ϵ_0. However, the exact slope is not of particular interest: All we require is arriving in the region of asymptotic behavior of s, which is typically achieved for small ϵ_0. This is because we are mainly interested in the behavior of $b(\mathbf{x})$, which is *invariant* to the slope of s (and hence ϵ_0) and *outside* the region of asymptotically linear s.

We don't visualize $b(\mathbf{x})$ directly but use an exponential scaling to obtain the height function $h(\mathbf{x})$. This scaling introduces the additional parameter k. It steers the exponential fall-off, which puts emphasis on different ranges of b.

The examples in the previous section range from simple constructed vector field to one with moderately complex topology. While the proposed method would work also for more and fairly complex vector fields, the visualization will not "scale" well. There are two reasons for this. First, the rendering of walls as semitransparent surface leads to problems with occlusion. This could partially be alleviated by advanced rendering techniques, e.g., steering transparency, or simply by displaying height walls only for selected separatrices. The second reason is more fundamental: Generally, all topology-based visualization methods are known to work well as long as the topological structure of the data is not too complex. This is the case, e.g., for noisy or turbulent vector fields. They typically show a vast number of critical points and a complex network of separatrices that is not suited for direct visualization. Our method shares this limitation with other topology-based visualization methods.

5.1 Future Research

As future research, the approach can be extended to 2D steady divergence-free fields where a separatrix from a saddle re-enters the saddle again. Also, the finite-time separation of other separating structures such as closed stream lines and boundary switch curves should be analyzed. The extension to 3D is a challenging problem. Here, the finite time behavior of separating surfaces has to be studied. By replacing $\mathbf{w}$ in (1) by the normalized surface normal field of the separating surface, all following computations hold for 3D as well. The actual challenge is the visual representation of $h(\mathbf{x})$ since a wall-metaphor is not appropriate in 3D.

Acknowledgements We wish to thank Niklas Röber and Michael Böttinger from the DKRZ for providing the South Pacific ocean dataset, as well as Tobias Günther for his general support.

References

1. Bhatia, H., Pascucci, V., Kirby, R.M., Bremer, P.: Extracting features from time-dependent vector fields using internal reference frames. Comput. Graphics Forum **33**(3), 21–30 (2014)
2. de Leeuw, W.C., van Liere, R.: Collapsing flow topology using area metrics. In: IEEE Visualization, pp. 349–354. IEEE, New York (1999)
3. de Leeuw, W.C., van Liere, R.: Visualization of global flow structures using multiple levels of topology. In: VisSym, pp. 45–52 (1999)
4. Garth, C., Gerhardt, F., Tricoche, X., Hagen, H.: Efficient computation and visualization of coherent structures in fluid flow applications. IEEE Trans. Vis. Comput. Graph. **13**(6), 1464–1471 (2007)
5. Garth, C., Li, G.S., Tricoche, X., Hansen, C.D., Hagen, H.: Visualization of coherent structures in transient 2d flows. In: Hege, H.C., Polthier, K., Scheuermann, G. (eds.) Topology-Based Methods in Visualization II, pp. 1–13. Springer, Berlin, Heidelberg (2009)
6. Globus, A., Levit, C., Lasinski, T.: A tool for visualizing the topology of three-dimensional vector fields. In: IEEE Visualization, pp. 33–41. IEEE, New York (1991)
7. Haller, G.: Distinguished material surfaces and coherent structures in three-dimensional fluid flows. Phys. D **149**, 248–277 (2001)
8. Haller, G.: Lagrangian coherent structures from approximate velocity data. Phys. Fluids **14**(6), 1851–1861 (2002)
9. Haller, G., Yuan, G.: Lagrangian coherent structures and mixing in two-dimensional turbulence. Phys. Fluids **147**(3–4), 352–370 (2000)
10. Helman, J., Hesselink, L.: Representation and display of vector field topology in fluid flow data sets. IEEE Comput. **22**(8), 27–36 (1989)
11. Helman, J., Hesselink, L.: Visualizing vector field topology in fluid flows. IEEE Comput. Graph. Appl. **11**(3), 36–46 (1991)

12. Kasten, J., Petz, C., Hotz, I., Noack, B.R., Hege, H.: Localized finite-time Lyapunov exponent for unsteady flow analysis. In: Proceedings of the Vision, Modeling, and Visualization Workshop 2009, November 16–18, 2009, Braunschweig, pp. 265–276 (2009)
13. Laramee, R.S., Hauser, H., Zhao, L., Post, F.H.: Topology-based flow visualization, the state of the art. In: Hauser, H., Hagen, H., Theisel, H. (eds.) Topology-Based Methods in Visualization, pp. 1–19. Springer, Berlin (2007)
14. Lekien, F., Coulliette, C., Mariano, A.J., Ryan, E.H., Shay, L.K., Haller, G., Marsden, J.E.: Pollution release tied to invariant manifolds: a case study for the coast of Florida. Phys. D **210**(1–2), 1–20 (2005)
15. Lipinski, D., Mohensi, K.: A ridge tracking algorithm and error estimate for efficient computation of Lagrangian coherent structures. Chaos **20**(1), 017504 (2010)
16. Lodha, S.K., Renteria, J.C., Roskin, K.M.: Topology preserving compression of 2d vector fields. In: IEEE Visualization, pp. 343–350. IEEE, New York (2000)
17. Lodha, S.K., Faaland, N.M., Renteria, J.C.: Topology preserving top-down compression of 2d vector fields using bintree and triangular quadtrees. IEEE Trans. Vis. Comput. Graph. **9**(4), 433–442 (2003)
18. Mahrous, K., Bennett, J., Hamann, B., Joy, K.I.: Improving topological segmentation of three-dimensional vector fields. In: VisSym - Symposium on Visualization, pp. 203–212 (2003)
19. Mahrous, K., Bennett, J., Scheuermann, G., Hamann, B., Joy, K.I.: Topological segmentation in three-dimensional vector fields. IEEE Trans. Vis. Comput. Graph. **10**(2), 198–205 (2004)
20. Pobitzer, A., Peikert, R., Fuchs, R., Schindler, B., Kuhn, A., Theisel, H., Matkovic, K., Hauser, H.: The state of the art in topology-based visualization of unsteady flow. Comput. Graphics Forum **30**(6), 1789–1811 (2011)
21. Pobitzer, A., Peikert, R., Fuchs, R., Theisel, H., Hauser, H.: Filtering of FTLE for visualizing spatial separation in unsteady 3d flow. In: Peikert, R., Hauser, H., Carr, H., Fuchs, R. (eds.) Topological Methods in Data Analysis and Visualization II: Theory, Algorithms, and Applications, pp. 237–253. Springer, Berlin, Heidelberg (2012)
22. Sadlo, F.: Lyapunov time for 2D Lagrangian visualization. In: Bennett, J., Vivodtzev, F., Pascucci, V. (eds.) Topological and Statistical Methods for Complex Data: Tackling Large-Scale, High-Dimensional, and Multivariate Data Spaces, pp. 167–181. Springer, Berlin, Heidelberg (2015)
23. Sadlo, F., Peikert, R.: Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. IEEE Trans. Vis. Comput. Graph. **13**(6), 1456–1463 (2007)
24. Sadlo, F., Peikert, R.: Visualizing Lagrangian coherent structures and comparison to vector field topology. In: Hege, H.C., Polthier, K., Scheuermann, G. (eds.) Topology-Based Methods in Visualization II, pp. 15–29. Springer, Berlin, Heidelberg (2009)
25. Sadlo, F., Weiskopf, D.: Time-dependent 2D vector field topology: An approach inspired by Lagrangian coherent structures. Comput. Graphics Forum **29**(1), 88–100 (2010)
26. Sadlo, F., Rigazzi, A., Peikert, R.: Time-dependent visualization of Lagrangian coherent structures by grid advection. In: Pascucci, V., Tricoche, X., Hagen, H., Tierny, J. (eds.) Topological Methods in Data Analysis and Visualization: Theory, Algorithms, and Applications, pp. 151–165. Springer, Berlin, Heidelberg (2011)
27. Scheuermann, G., Krüger, H., Menzel, M., Rockwood, A.P.: Visualizing nonlinear vector field topology. IEEE Trans. Vis. Comput. Graph. **4**(2), 109–116 (1998)
28. Shadden, S.C., Lekien, F., Marsden, J.E.: Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. Phys. D **212**, 271–304 (2005)
29. Shadden, S.C., Lekien, F., Paduan, J.D., Chavez, F.P., Marsden, J.E.: The correlation between surface drifters and coherent structures based on high-frequency radar data in monterey bay. Deep-Sea Res. II Top. Stud. Oceanogr. **56**(3–5), 161–172 (2009)
30. Stalling, D., Westerhoff, M., Hege, H.: Amira: a highly interactive system for visual data analysis. In: The Visualization Handbook, pp. 749–767. Elsevier, Amsterdam (2005)
31. Theisel, H.: Designing 2d vector fields of arbitrary topology. Comput. Graphics Forum **21**(3), 595–604 (2002)

32. Theisel, H., Rössl, C., Seidel, H.: Compression of 2d vector fields under guaranteed topology preservation. Comput. Graphics Forum **22**(3), 333–342 (2003)
33. Theisel, H., Weinkauf, T., Hege, H., Seidel, H.: Saddle connectors - an approach to visualizing the topological skeleton of complex 3d vector fields. In: IEEE Visualization, pp. 225–232. IEEE Computer Society, Washington, DC (2003)
34. Tricoche, X., Scheuermann, G., Hagen, H.: A topology simplification method for 2d vector fields. In: IEEE Visualization, pp. 359–366. IEEE, New York (2000)
35. Tricoche, X., Scheuermann, G., Hagen, H.: Continuous topology simplification of planar vector fields. In: IEEE Visualization, pp. 159–166. IEEE Computer Society, Washington, DC (2001)
36. Üffinger, M., Sadlo, F., Ertl, T.: A time-dependent vector field topology based on streak surfaces. IEEE Trans. Vis. Comput. Graph. **19**(3), 379–392 (2013)
37. Weinkauf, T.: Extraction of topological structures in 2d and 3d vector fields. Ph.D. thesis, Otto von Guericke University Magdeburg (2008)
38. Weinkauf, T., Theisel, H., Hege, H., Seidel, H.: Boundary switch connectors for topological visualization of complex 3d vector fields. In: VisSym, pp. 183–192. Eurographics Association, Goslar (2004)
39. Weinkauf, T., Theisel, H., Hege, H., Seidel, H.: Topological construction and visualization of higher order 3d vector fields. Comput. Graphics Forum **23**(3), 469–478 (2004)
40. Weldon, M., Peacock, T., Jacobs, G.B., Helu, M., Haller, G.: Experimental and numerical investigation of the kinematic theory of unsteady separation. J. Fluid Mech. **611**, 1–11 (2008)
41. Westermann, R., Johnson, C.R., Ertl, T.: Topology-preserving smoothing of vector fields. IEEE Trans. Vis. Comput. Graph. **7**(3), 222–229 (2001)
42. Wiebel, A., Garth, C., Scheuermann, G.: Computation of localized flow for steady and unsteady vector fields and its applications. IEEE Trans. Vis. Comput. Graph. **13**(4), 641–651 (2007)
43. Wischgoll, T., Scheuermann, G.: Detection and visualization of closed streamlines in planar flows. IEEE Trans. Vis. Comput. Graph. **7**(2), 165–172 (2001)

Comparing Finite-Time Lyapunov Exponents in Approximated Vector Fields

Stefan Koch, Sebastian Volke, Gerik Scheuermann, Hans Hagen, and Mario Hlawitschka

Abstract In the context of fluid mechanics, larger and larger flow fields arise. The analysis of such fields on current work stations is heavily restricted by memory. Approximation limits this problem. In this paper, we discuss the impact of vector field approximation on visualization techniques on the example of Finite-Time Lyapunov Exponent (*FTLE*) computations. Thereby, we consider the results of three different vector field compression approaches and analyze the reliability of integration results as well as their impact on two different FTLE variants.

1 Introduction

Visualization is one of the most important tools for the investigation of complex flow fields. Many different visualization approaches have been developed. These include techniques ranging from the extraction and depiction of specific flow features such as vortices [11, 12], to the computation of the topological structure of vector fields [6, 20]. There also exist many methods to cluster vector fields [3, 18] or compute simplified representations [1, 18]. Their main concern is to allow users to get a good overview of the overall flow behavior. A detailed summary of the most common visualization approaches can be found in these state-of-the-art reports [9, 13, 15].

In order to handle increasingly large and complex simulation results, compression approaches have been introduced in the past [2, 8, 10, 19]. These algorithms aim at a dataset size reduction to allow an easier transmission of datasets via networks and a fast evaluation even on low-end computers. While small-scale flow features could be lost during compression, large-scale features, as well as the global flow characteristic, are preserved in most cases.

S. Koch • S. Volke (✉) • G. Scheuermann • M. Hlawitschka
Leipzig University, Augustusplatz 10, 04109 Leipzig, Germany
e-mail: stefan.koch@informatik.uni-leipzig.de; volke@informatik.uni-leipzig.de; scheuermann@informatik.uni-leipzig.de; hlawit@informatik.uni-leipzig.de

H. Hagen
University of Kaiserslautern, Erwin-Schrödinger-Str., 67663 Kaiserslautern, Germany
e-mail: hagen@informatik.uni-kl.de

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_16

Although the previous works provide an evaluation of the quality of their compressed vector fields, they often approach their evaluations differently (cf. Sect. 2.1). This makes a comparison of the advantages and disadvantages of the different algorithms difficult. Furthermore, a comprehensive investigation of how vector field compressions, respectively their corresponding approximation errors, influence common integration-based visualization techniques has not been conducted, yet.

In this paper, we present a quantitative evaluation of the influence of the approximation errors on streamline integration. *Line Integral Convolution* (*LIC*) [17] images are used to contrast the quality of the approximated flow behavior and to analyze the spatial distribution of the integration errors. The obtained results are used to qualitatively discuss the applicability of a common integration-based visualization methods on compressed vector fields on the example of *Finite-Time Lyapunov Exponent* (*FTLE*). The underlying comparison approach can be used to characterize the error distribution of the integration results between different compression techniques.

2 Related Work

The main focus of our work is the comparison of the results of two FTLE variants on vector fields that were compressed using different techniques. In the following, we briefly review related work on vector field compression, as well as foundations of the FTLE computation.

2.1 Vector Field Compression

Most vector field clusterings in literature aim at the computation of simplified representations of a vector field, rather than to compress its dataset size. Lodha et al. [10] first extended the clustering approach of Telea and Wijk [18] in order to compute an error-bounded vector field compression that preserves the main characteristics of the stationary points.

Later, Theisel et al. [19] introduced a vector field compression algorithm based on an equivalence relation between topologies: two topological skeletons are equivalent if they (1) have an identical set of stationary points, including the first derivative at their positions; (2) have the same set of boundary switch points; and if (3) all separatrices start or end in the same stationary point, respectively enter or leave the domain in the same in- or outflow region. They compress the vector field iteratively by simplifying the underlying grid using edge collapses. A simplification is allowed only if the topology stays equivalent throughout the change. Although the topology-aware compression algorithm of Theisel et al. [19] achieves very high compression rates, it does not provide any error threshold that limits the rate of change of the local flow behavior. This can lead to large distortions in the compressed vector field.

Platis and Theoharis [14] introduced a compression method based on iterative vertex removal. For this, different metrics are evaluated to limit the domain and field error, as well as to improve the mesh quality. Dey et al. [2] presented a similar approach that operates on a Delaunay triangulation and limits the local relative approximation error in the vector field. In contrast to Theisel et al., their algorithm does not directly guarantee the preservation of the vector fields topology.

In order to provide an error-bound that retains the main characteristic of the local flow and additionally preserves the vector field topology as one important flow feature, Koch et al. [8] presented a compression algorithm that is based on a region-wise linear approximation of the input field. The maximal approximation error within each locally linearized region is bounded by an error threshold E_{max}.

All of these methods have been evaluated both visually and quantitatively. The visual evaluations include a flow comparison by computing hedgehogs [10, 14], streamlines [2], LIC [8, 19] and the topological skeleton [2, 8, 10, 19] for the compressed and original fields. Also the compressed and uncompressed grids are compared [8, 19] in order to study their quality. The quantitative evaluations involve compression rates [2, 8, 10, 14, 19] and the distribution of the local approximation errors in the resulting fields [2, 8, 10, 14]. There are also some brief comparisons against each other: [19] compares against [10], [2] against [14], and [8] against [19]. However, there is no quantitative evaluation of the impact of the compression on integration based visualization methods and also no comprehensive comparison between multiple general approaches. In this paper, we study the integration error and its visual impact on the example of FTLE images.

2.2 Variants of Finite-Time Lyapunov Exponents

A common visualization approach, which has been investigated in detail in the past years, is FTLE as presented by Haller [5]. A FTLE field shows the rate of separation, respectively convergence, of closely neighbored particles over the time. Since its introduction, the basic FTLE computation was extended, for instances, to compute the separation of flow from surfaces [4] up to the efficient extraction of separation surfaces in three-dimensional fields [16].

Besides the common *flow-map based FTLE* (*F-FTLE*) computation, we will use another variant: The *Localized Finite-time Lyapunov Exponent* (*L-FTLE*) of Kasten et al. [7]. It uses a computation along pathlines that only depends on the first derivative. An important advantage of this FTLE variant is that it is more robust with respect to noise and, thus, might be well suited for compressed fields.

3 Data Acquisition

We use different types of compression algorithms for our comparison. We focus on topology-preserving and error-bounded compression techniques and want study the influence of these properties on visualizations of the compressed fields. Thereby, we compare the differently compressed fields against each other as well as against the original, uncompressed vector field.

Two approaches are applied for this comparison: a quantitative statistical analysis of the accuracy of streamline integration and a qualitative visual analysis of visualization results. Both approaches are described in the following.

3.1 Quantitative Analysis of Streamline Integration

The goal of the quantitative analysis is to assess the impact of the compression on the quality of integrations in the compressed field. This facilitates the comparison to the original vector field and allows to obtain a quality measure for the compression technique.

For this evaluation, we compare the deviation of streamlines in the compressed vector field from streamlines in the original one. A natural concept that allows to derive the deviation values is the *flow map*. The flow map stores the end point after a certain integration time for every position in the field. To assure comparability between the original and the compressed field, we use the grid points of the original field as seed positions for both fields. The resulting flow map is stored on the grid of the original vector field. So, the deviation of streamlines can then be derived as a scalar field that contains the Euclidean distance of the particle end points between the two flow maps. The resulting field shows how much the flow behavior differs locally between the compressed and the uncompressed vector field.

As we are interested in the minimal and maximal deviation, as well as the distribution of integration errors, we use box plots [21] and histograms to visualize the flow map differences. Thereby, we can juxtapose the deviation distribution for multiple integration times. In our examples, we focus on integration times that are suitable for FTLE computations.

3.2 Qualitative Visual Analysis

For a qualitative comparison of different compression techniques, we compute LIC images to visually compare the characteristic, topological flow patterns of the original and compressed fields. Furthermore, to evaluate how well convergent and divergent flow behavior is preserved, we also compare different FTLE images of the compressed fields. From the previously computed flow maps, we obtain F-FTLE fields in a first step. Here, streamline integration instead of pathline integration is used, to apply FTLE on steady fields. As a second FTLE variant, we compute the L-FTLE fields. In contrast to the original implementation of Kasten et al. [7], we do not use a precomputed and interpolated Jacobian field. We use the constant derivative that is given for each triangle cell, respectively for each linearly approximated region.

4 Results

We now discuss the results of the flow map and FTLE computations described in Sect. 3 on the basis of two real-world datasets: the Kármán vortex street and a two-dimensional simulation of a jet stream. In the following, we describe the used dataset and then present the results of (1) a topology-preserving non-error-bounded, (2) an error-bounded non-topology-preserving, and (3) a topology-preserving and error-bounded compression algorithm.

4.1 Datasets

4.1.1 Kármán Vortex Street

The Kármán vortex street is a well-known flow pattern that shows the flow separation from an obstacle. In this dataset, the flow goes around a bar, which is rotated by 45° towards the inflow. After passing the obstacle, the flow shows a characteristic periodic swirling. The original dataset is given on a triangulated grid with 156,842 cells. Figure 1a, b show a LIC image and the F-FTLE image of this vector field.

4.1.2 Jet Stream

The Jet Stream dataset shows a flow that enters the domain through a narrow opening on the left side at a velocity of Mach 0.1. This produces an expansion of the flow which results in a complex flow behavior with many vortices and small turbulent flow structures. The original vector field is given on a triangulated grid with 2,922,912 cells. Figure 1c, d show a LIC image and the F-FTLE image of this vector field respectively.

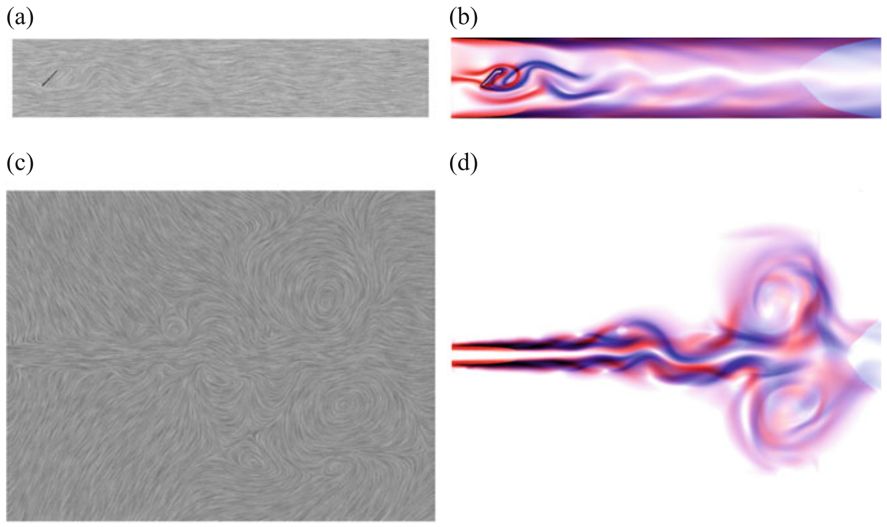


Fig. 1 LIC and F-FTLE images of the uncompressed, original Kármán vortex street (*upper row*) and the Jet Stream dataset (*bottom row*). (**a**) LIC image of the Kármán vortex street. (**b**) F-FTLE ($dt = 3.0$). (**c**) LIC image of the Jet Stream. (**d**) F-FTLE ($dt = 4.0$)

4.2 Topology-Preserving Compression Without Error-Bounds

As an example for a topology-preserving compression algorithm, we consider the algorithm of Theisel et al. [19] and achieve very high compression rates of 99.9% for the Kármán vortex street and 99.6% for the Jet Stream dataset.

First, we quantitatively analyze the integration errors. Figure 2a shows box plots of the flow map differences for various integration times for the Kármán vortex street. The errors are distributed mostly uniformly, as can be seen from the underlying histograms. With increasing integration times the average as well as the maximal error increases nearly linearly. Given the dimensions of the vector field of 20 by 4, we have a deviation of about 10% of the domain diameter at an integration time of 3.0. Figure 3a shows the results of the quantitative integration error analysis on the Jet Stream dataset. This field has a dimension of 25 by 20. Therefore, the maximal error at an integration time of 4.0 is about 12% of the domain diameter. However, the overall distribution of the errors is not uniform but concentrated near the minimum. Three quarters of the data points have an error of below 0.9.

For a qualitative analysis of the compression results, we consider the visual quality of LIC and F-FTLE images. Despite the high compression rate, both LIC images in Figs. 2e and 3b show that overall flow patterns are preserved very well. Especially in the Jet Stream dataset, the singularities, their region of influence as well as their interaction is clearly visible and the image is very similar to the LIC image of the uncompressed field (Fig. 1). Only the inflow area in the Jet Stream dataset is obviously distorted.

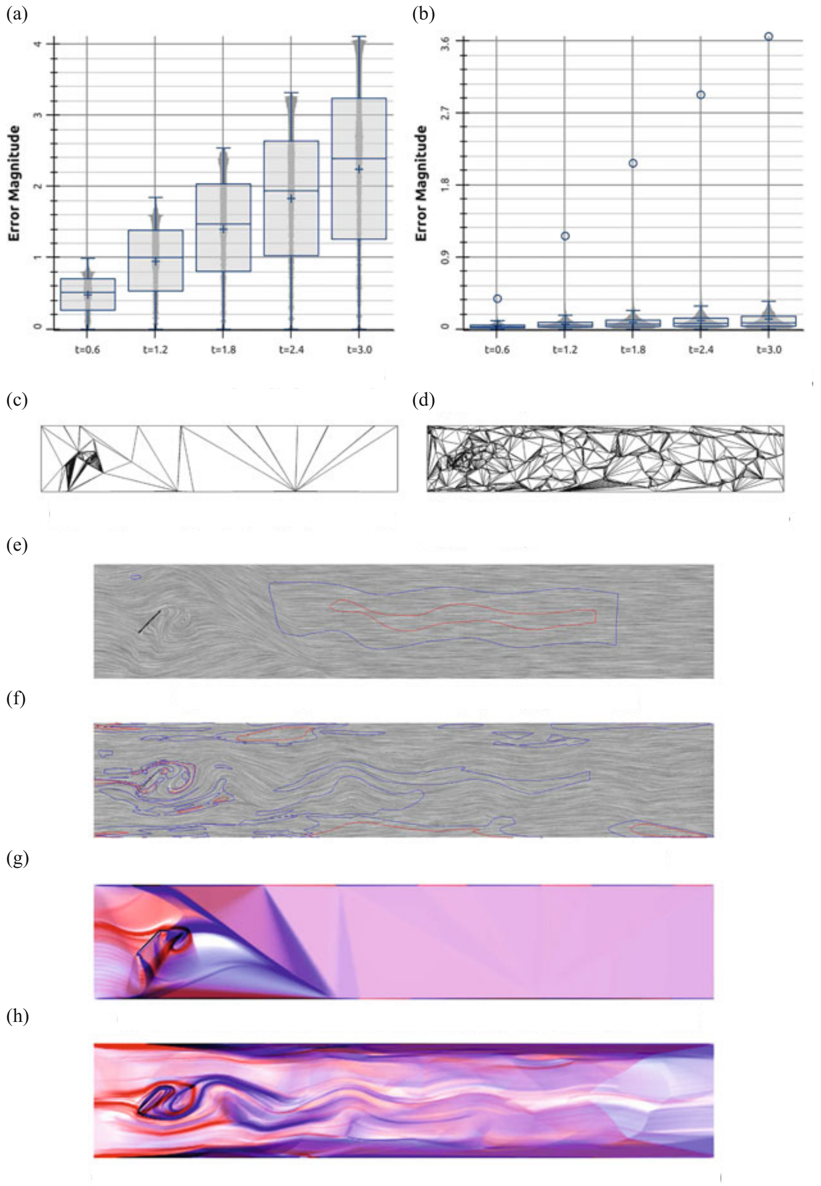
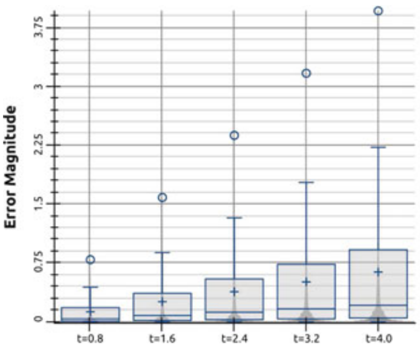
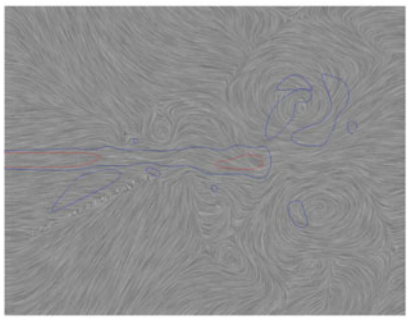


Fig. 2 In order to compare the integration results of the first two compression techniques, we applied them to the Kármán vortex street. (**a**), (**b**) show the histogram of the error values in flow map differences for multiple integration durations. The *box plots* mark their quartiles, the plus their average and the *circle* their maximal value. (**e**)–(**h**) show LIC images and F-FTLE images of the compressed vector fields. To analyze the distribution of the integration error in the compressed field for an integration time of 3.0, we added iso-lines to emphasize the upper 25% (*blue*) and the upper 5% (*red*) of all error values in the domain. Please note the different scale of the ordinate of the shown error plots. Although the areas of the higher approximation error seem to cover large parts of the domain, the overall error in (**f**) is much smaller compared to (**e**)

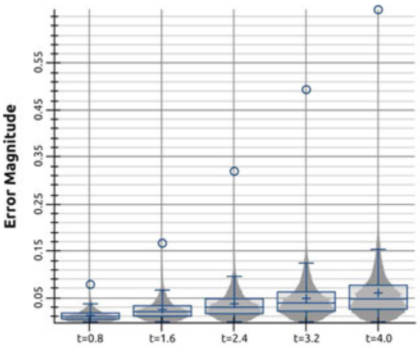
(a)



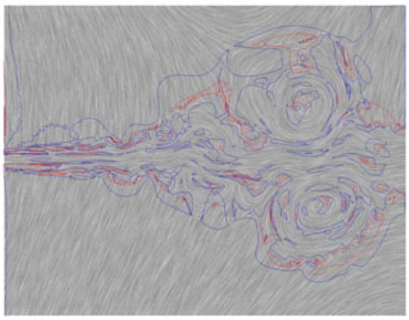
(b)



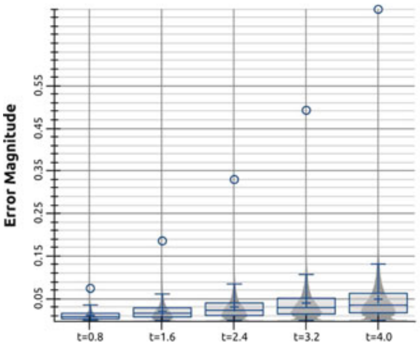
(c)



(d)



(e)



(f)

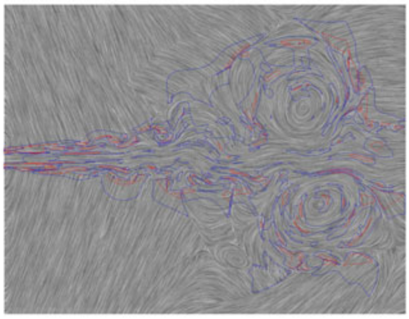


Fig. 3 In order to compare the integration results of the used compression techniques, we applied them to the Jet Stream dataset. (**a**), (**c**), (**e**) show the histogram of the flow map differences for multiple integration times. The *box plots* mark their quartiles, the plus their average and the *circle* their maximal value. (**b**), (**d**), (**f**) show the LIC images the compressed vector fields. To analyze the distribution of the integration error in the compressed field for an integration time of 4.0, we added iso-lines to emphasize the upper 25% (*blue*) and the upper 5% (*red*) of all error values in the domain

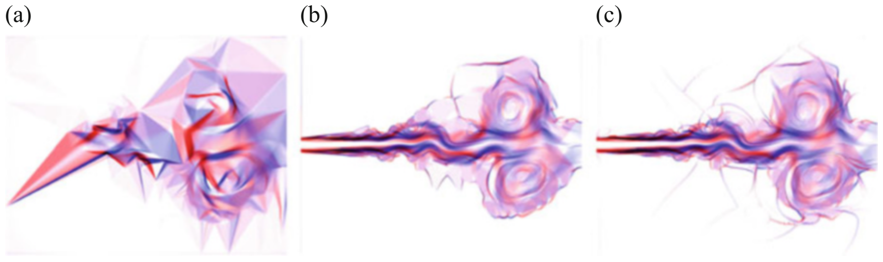


Fig. 4 These three images show the F-FTLE results for the Jet Stream dataset. The results were computed for all compression techniques that are used in this work. An integration time of 4.0 and an error-bound of $E_{max} = 0.1$ was used. (**a**) Topology-preserving. (**b**) Error-bounded. (**c**) Topology-preserving and error-bounded

We see the reason for these good results in the preservation of the topological skeleton. When interpreting LIC images, the viewer focuses on well known flow patterns, e.g., laminar or characteristic linear flow in the vicinity of stationary points. Therefore, the perception of a LIC image is mainly sensitive to the number and location of the stationary points. As these do not change during compression because of the algorithm design, the overall quality of the LIC images remains very good.

Figures 2g and 4a show the F-FTLE images of the Kármán vortex street and the Jet Stream respectively. In both cases, we see strong artifacts from the coarse underlying grid. Because Theisel et al. only aim at the construction of a topologically equivalent compressed vector field, they remove features that are not represented by the topological skeleton. Thus, the typical flow separation and convergence of the Kármán vortex street has been removed as it can not be seen in Fig. 2g. Only close to the bar and the topological skeleton, some features are still visible, but a distortion is also noticeable in those areas. The F-FTLE results of the Jet Stream dataset suffers from similar problems. By comparing Figs. 1c and 3b, one can see that the compression causes a shift of the inlet towards the lower left domain boundary. Therefore, the F-FTLE image is not only too coarse to see small features but also shows high FTLE values in regions of nearly stagnating flow in the original field.

FTLE images show divergence and convergence in the flow and thus are highly sensitive to deviations in direction and magnitude of the vector field. Topology-preservation only guarantees a preservation of the overall flow behavior in the individual basins, but it does not necessarily preserve the course of the streamlines. High deviations in the flow are still possible without altering the topology. Therefore, the high compression rate has a significant impact on the quality of FTLE images.

4.3 Error-Bounded Compression Without Topology-Preservation

Since FTLE computation on topology-preserved vector fields is affected by large artifacts from the underlying coarsened grid, we study the results of the error-bounded vector fields approximation algorithm that is presented by Koch et al. in [8]. In order to study only the influence of the error-bound E_{max} on the vector field approximation, we removed the topology-preservation from the algorithm. This approach achieves compression rates of 97.8% for the vortex street and 99.2% for the Jet Stream dataset that we used in Figs. 2 and 3. An error-bound of 0.1 is used for the Kármán vortex street, respectively 0.4 for the Jet stream dataset.

Again, we first quantitatively analyze the integration error-based on the flow map differences in the original and the approximated vector fields. The results in Figs. 2b and 3c show strong differences to the results of the previous section. On both fields, the maximal error is only about 1% of the field dimensions. On both fields, we see an over-linear growth of the maximal error with integration time. However, the average error grows much slower. When considering the distribution of the error values, we see that the range of possible errors is dominated by outliers and the majority of errors have a small magnitude. The associated LIC images (cf. Figs. 2f and 3d) show that these outliers are concentrated around the main vortices and between the inflow and the neighbored nearly stagnating flow. We expect only small integration errors even with higher integration times, though we know that some artifacts will be present due to the outliers.

The LIC images in Figs. 2f and 3d verify our assumptions. In case of the Kármán vortex street, the typically swirling flow behind the rotated bar is better preserved then in Fig. 2e. This is due to the fact that the error-bounded algorithm preserves a higher number of cells (cf. Fig. 2c, d, which allows a more detailed approximation of the original field while decreasing the compression rate. In contrast to the simple flow characteristic of the vortex street, which is well preserved, the approximation of the Jet Stream shows strong differences. Figure 3d shows that the inflow as well as the main vortices are correctly approximated. But since the topology is very sensitive against vector field perturbations, one can clearly see changes of the vector fields topology. Especially in the lower part of the field, the weaker vortices begin to vanish at an error threshold of 0.1. So, depending on the chosen error threshold, the topology can change significantly and LIC images can no longer give reliable information of the main global flow behavior.

The F-FTLE results in Figs. 2h and 4b are very similar to the corresponding images of the original fields in Fig. 1. All main features of the original F-FTLE fields are preserved. Especially in case of the Kármán vortex street, the characteristic flow separation and attachment behind the obstacle are very well perceptible. This is due to the fact that the local error threshold limits the allowed magnitude and angle changes that are introduced by the compression approach. These findings are also indicated by the small error magnitudes that we find in the flow map differences (Figs. 2b and 3c). We did not expect this result because vanished saddle and center points should result in much higher integration errors theoretically. But there are also clearly visible artifacts, especially in the case of the Jet Stream. These are probably caused by the outliers that are visible in the error plots. The position of the artifacts correspond to the areas of maximal error in the LIC images (Figs. 2f and 3d). These areas also correspond to the boundaries of the linearly approximated regions that are used by the algorithm [8], which, by the design of the algorithm, tend to have the largest errors.

4.4 Error-Bounded Topology-Preserving Compression

As a synthesis from the previous sections, we apply the original approximation algorithm of [8] to the datasets. This algorithm combines the topology-preservation approach of Theisel et al. with an error-bound. We achieve compression rates between 92.3% (for $E_{max} = 0.1$) to 94.8% (for $E_{max} = 0.4$) for the Kármán vortex street and compression rates between 99.5% (for $E_{max} = 0.02$) to 96.0% (for $E_{max} = 0.1$) for the Jet Stream dataset.

Similar to the error-bounded approach without the topology-preservation, the error distributions in Fig. 3e show fast increasing error outliers and a slow increasing average error over integration time. Also the location and the extent of the regions with the highest error are similar. Thus, the topology-preservation has no further influence on the error distribution in this case. However, the LIC images clearly show that all features of Fig. 1c could be preserved. The inflow could be correctly approximated as well as all topological features. Also the F-FTLE fields in Fig. 4b, c look nearly the same. This is due to the fact that the same error measure was used, which locally limits the magnitude and angle differences between the approximated the original field.

We use this compression method to study FTLE on compressed fields in more detail. First, we investigate the influence of the integration times. Figure 5 shows a sequence of F-FTLE results of the Kármán vortex street for different integration times. Counterintuitively, longer integration times lead to smoother F-FTLE images. Whereas the highest integration time theoretically leads to the most error-prone image, these images also contain nearly no visual artifacts of the underlying coarse region-wise linearly approximated vector field. Compared to that, the smaller integration times are more precise and lead to a coarser looking result, since shorter integration times emphasize the transitions between to neighbored linearizations. Given the overall small integration error, we can still consider the results with high integration times reliable.

Second, we applied the L-FTLE algorithm to the compressed fields. L-FTLE is expected to compensate discontinuities at cell or linearly approximated region boundaries. The results for the Jet Stream dataset are shown in Fig. 6. While on the original field, L-FTLE leads to sharper images (cf. Figs. 1d and 6c), on the compressed fields it appears to emphasize artifacts. These outweigh the flow features, so that the images cannot be interpreted correctly anymore. We assume

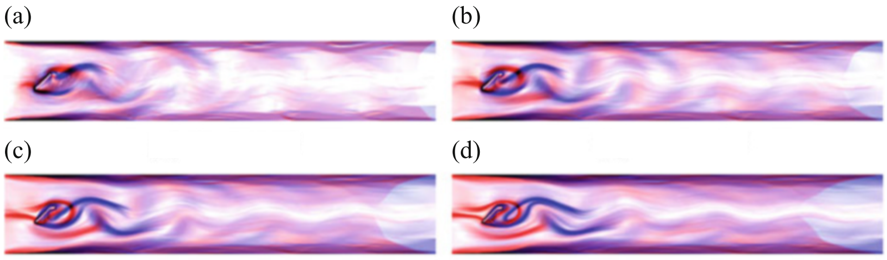


Fig. 5 F-FTLE images of the error-bounded, compressed Kármán vortex street (the topology-preserving and error-bounded algorithm was used with $E_{max} = 0.1$) for different integration lengths. The main flow features can clearly be seen in all four images. However, the result appears to be much smoother with longer integration times. (**a**) $dt = 0.75$. (**b**) $dt = 1.50$. (**c**) $dt = 2.25$. (**d**) $dt = 3.00$

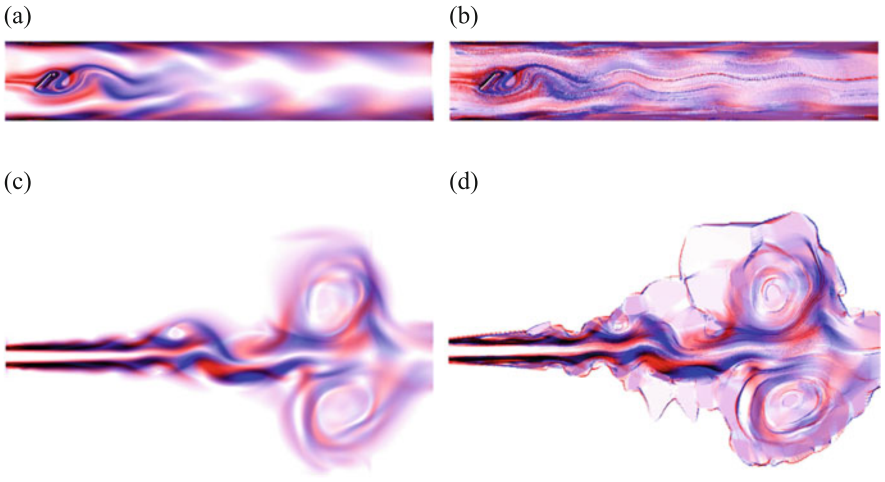


Fig. 6 These two images show a comparison of the L-FTLE results computed on the uncompressed Jet Stream dataset (*left*) and its error-bound compression (*right*). The result of the compressed field clearly shows artifacts of the region-wise approximation. (**a**) $dt = 3.0$. (**b**) $E_{max} = 0.4, dt = 3.0$. (**c**) $dt = 4.0$. (**d**) $E_{max} = 0.1, dt = 4.0$

that this is a consequence of the coarse, piece-wise linear vector field approximation, especially the piece-wise constant Jacobian.

5 Discussion

We found that the two criteria, preservation of topology and a bounded approximation error, have a different impact on the resulting compressed field with respect to the achieved compression rate as well as on the preserved flow features. The best compression rates can be achieved when no error-bound is used. This is shown by the results of Theisel et al. [19]. The introduction of an error-bound has a large negative impact on the compression rate. The worst compression rates were achieved when fulfilling both criteria.

Depending on the flow features that should be preserved by the compression, the different compression algorithms have their advantages and disadvantages. In order to preserve the topology, either the topology has to be preserved explicitly, or an appropriately small approximation error has to be used (cf. Dey et al. [2]). When the compressed fields are mainly used for the generation of LIC images, the preservation of topology seems to be the more important compression criterion. For non-topological flow features, such as convergence and divergence, a suitable error-measure should be used, e.g., one that limits the deviation of flow direction.

The integration error that results from the compression not only depends on the vector field itself but on the design of the particular compression algorithm. Error-bounded algorithms show a characteristic small number of high error outliers, which are located at the transitions between the linearly approximated regions.

In every case, the integration error increases with longer integration times, because the error accumulates along the streamlines. On the other hand, longer integration times lead to smoother FTLE images with less visual artifacts in our examples. Since the overall error is higher in these images, they are also less reliable. Therefore, the integration time is not only one of the most important parameters of the FTLE computation, here it also controls the trade-off between the visual appeal of the images and their correctness. As long as the integration error in the flow maps remains in the scale of the cell resolution, a meaningful preview of the original dataset can be obtained, as shown in the examples of Sect. 4.4.

6 Conclusion and Future Work

In this work we compared three different vector field compression algorithms with respect to their applicability in vector field visualization. One method with an unbounded approximation error results in very high compression rates while preserving the vector fields topology. The others limit the local vector field deviation—and also the compression rate—by using a user-defined error threshold. In particular, we compared the quality of the flow map computation in uncompressed and compressed vector fields as well as the results of FTLE computations.

We showed, that topology-preservation is not sufficient to preserve flow features. Additionally, a mechanism to control the overall approximation error is needed. The quality of the integration results as well as the computed FTLE images of compressed vector fields largely depends on the used error-bounds, respectively the compression rates. Our results suggest that error-bound and topology-preservation are orthogonal concepts and establish a design space for compression methods.

For future work, more datasets and vector field compression techniques could be investigated to extend the comparison of the different algorithm approaches and further investigate the design space: An interesting question is, whether there is a characteristic development of the error distribution with increasing integration times for a particular compression method and which properties of the method are causing it. Such research appears to be possible, since we have seen characteristic error distributions for the individual algorithms. Further, it could be possible to develop a guideline that helps to decide, which compression techniques can be used to facilitate certain visualizations or analysis tasks on compressed vector fields.

Acknowledgements We thank Markus Rütten, Guillaume Daviller, and Bernd Noack for providing the simulation datasets. Special thanks go to the FAnToM development group for providing the visualization software. We also thank Jens Kasten for the fruitful discussions. This work was partially supported by the European Social Fund (Appl. No. 100098251).

References

1. Auer, C., Kasten, J., Kratz, A., Zhang, E., Hotz, I.: Automatic, tensor-guided illustrative vector field visualization. In: IEEE Pacific Visualization Symposium, pp. 265–272 (2013)
2. Dey, T., Levine, J., Wenger, R.: A delaunay simplification algorithm for vector fields. In: Pacific Conference on Computer Graphics and Applications, pp. 281–290 (2007)
3. Garcke, H., Preußer, T., Rumpf, M., Telea, A., Weikard, U., Van Wijk, J.: A continuous clustering method for vector fields. In: Proceedings IEEE Visualization 2000, pp. 351–358 (2000)
4. Garth, C., Wiebel, A., Tricoche, X., Joy, K., Scheuermann, G.: Lagrangian visualization of flow-embedded surface structures. Comput. Graph. Forum **27**(3), 1007–1014 (2008)
5. Haller, G.: Lagrangian structures and the rate of strain in a partition of two-dimensional turbulence. Phys. Fluids **13**(11), 3365–3385 (2001)
6. Helman, J.L., Hesselink, L.: Representation and display of vector field topology in fluid flow data sets. IEEE Comput. **22**(8), 27–36 (1989)
7. Kasten, J., Petz, C., Hotz, I., Noack, B.R., Hege, H.C.: Localized finite-time Lyapunov exponent for unsteady flow analysis. In: Proceedings Vision, Modeling and Visualization, pp. 265–274 (2009)
8. Koch, S., Kasten, J., Wiebel, A., Scheuermann, G., Hlawitschka, M.: 2D vector field approximation using linear neighborhoods. The Visual Computer, pp. 1–16 (2015). doi:10.1007/s00371-015-1140-9
9. Laramee, R., Hauser, H., Zhao, L., Post, F.: Topology-based flow visualization, the state of the art. In: Topology-based Methods in Visualization, Mathematics and Visualization, pp. 1–19. Springer, Berlin/Heidelberg (2007)
10. Lodha, S.K., Renteria, J.C., Roskin, K.M.: Topology preserving compression of 2D vector fields. In: Proceedings IEEE Visualization 2000, pp. 343–350 (2000)
11. Lu, K., Chaudhuri, A., Lee, T.Y., Shen, H.W., Wong, P.C.: Exploring vector fields with distribution-based streamline analysis. In: IEEE Pacific Visualization Symposium, pp. 257–264 (2013)
12. Marchesin, S., Chen, C.K., Ho, C., Ma, K.L.: View-dependent streamlines for 3D vector fields. IEEE Trans. Vis. Comput. Graph. **16**(6), 1578–1586 (2010)
13. McLoughlin, T., Laramee, R.S., Peikert, R., Post, F.H., Chen, M.: Over two decades of integration-based, geometric flow visualization. Comput. Graph. Forum **29**(6), 1807–1829 (2010)
14. Platis, N., Theoharis, T.: Simplification of vector fields over tetrahedral meshes. In: Proceedings IEEE Computer Graphics International, pp. 174–181 (2004)

15. Post, F.H., Vrolijk, B., Hauser, H., Laramee, R.S., Doleisch, H.: The state of the art in flow visualization: feature extraction and tracking. Comput. Graph. Forum **22**(4), 775–792 (2003)
16. Sadlo, F., Peikert, R.: Efficient visualization of Lagrangian coherent structures by filtered AMR ridge extraction. IEEE Trans. Vis. Comput. Graph. **13**(6), 1456–1463 (2007)
17. Stalling, D., Hege, H.C.: Fast and resolution-independent line integral convolution. In: Proceedings of SIGGRAPH, pp. 249–256 (1995)
18. Telea, A., Van Wijk, J.: Simplified representation of vector fields. In: Proceedings IEEE Visualization, pp. 35–507 (1999)
19. Theisel, H., Rössl, C., Seidel, H.P.: Compression of 2D vector fields under guaranteed topology preservation. Comput. Graph. Forum **22**(3), 333–342 (2003)
20. Tricoche, X., Wischgoll, T., Scheuermann, G., Hagen, H.: Topology tracking for the visualization of time-dependent two-dimensional flows. Comput. Graph. **26**(2), 249–257 (2002)
21. Tukey, J.: Exploratory Data Analysis. Addison-Wesley Series in Behavioral Sciences. Addison-Wesley Publishing Company, Reading, MA (1977)

Transfer Operator-Based Extraction of Coherent Features on Surfaces

Kathrin Padberg-Gehle, Sebastian Reuther, Simon Praetorius, and Axel Voigt

Abstract Transfer operator-based approaches have been successfully applied to the extraction of coherent features in flows. Transfer operators describe the evolution of densities under the action of the flow. They can be efficiently approximated within a set-oriented numerical framework and spectral properties of the resulting stochastic matrices are used to extract finite-time coherent sets. Also finite-time entropy, a density-based stretching quantity similar to finite-time Lyapunov exponents, is conveniently approximated by means of the discretized transfer operator. Transfer operator-based computational methods are purely probabilistic and derivative-free. Therefore, they can also be applied in settings where derivatives of the flow map are hardly accessible. In this paper, we summarize the theory and numerics behind the transfer operator approach and then introduce a straightforward extension, which allows us to study coherent structures in complex flows on triangulated surfaces. We illustrate our general computational framework with the well-known periodically driven double-gyre flow. To demonstrate the applicability of the approach for complex flows, we consider an approximation of the surface ocean flow, obtained by a numerical solution of the incompressible surface Navier-Stokes equation in a complicated geometry on the sphere.

1 Introduction

Transfer operator-based methods within a set-oriented numerical framework have only recently been recognized as powerful computational tools for analyzing and quantifying coherent structures and transport processes in time-dependent dynamical systems.

Of key interest are regions in the phase space of a nonautonomous dynamical system that remain coherent under the action of the flow. Almost-invariant sets [2, 6, 8] are spatially fixed regions, while coherent sets [7, 10, 11] move about with minimal dispersion. Almost-invariant and coherent sets can be efficiently identified

K. Padberg-Gehle (✉) • S. Reuther • S. Praetorius • A. Voigt
Institut für Wissenschaftliches Rechnen, TU Dresden, 01062 Dresden, Germany
e-mail: kathrin.padberg@tu-dresden.de; sebastian.reuther@tu-dresden.de; simon.praetorius@tu-dresden.de; axel.voigt@tu-dresden.de

via Perron-Frobenius operators (transfer operators). Recently, a unified functional analytic setting for optimal almost-invariant and coherent set constructions has been developed [10]. Transfer operators are linear Markov operators that can be approximated within a set-oriented framework [2]. Leading eigenvectors or singular vectors of the resulting stochastic matrices are heuristically used to determine the phase space structures of interest.

Transfer operators can also be employed to estimate finite-time expansive behavior along trajectories in autonomous and nonautonomous dynamical systems. Finite-time entropy (FTE) captures nonlinear stretching directly from the entropy growth experienced by a small localized density evolved by the transfer operator. In other words, FTE measures the increase in uncertainty of a small perturbation in the initial condition under the action of the flow and thus gives similar results to finite-time Lyapunov exponent calculations. Within the set-oriented approach an approximation of the FTE-field is obtained very efficiently and without relying on derivatives of the flow map. The FTE-concept has been introduced in [9], see also [18] for related previous work.

In this paper, we will sketch the theory and numerics behind the transfer operator approach. In addition, we will demonstrate how such probabilistic methods can be extended to highlight coherent features in complex flows approximated on unstructured grids. We begin by briefly reviewing the transfer operator framework in Sect. 2. In Sect. 3 the numerical approximation of such operators within a set-oriented approach is described and as well as the extension to flows on triangulated surfaces. The discretized transfer operator is the fundamental tool for the extraction of coherent sets and transport barriers and we will introduce the respective approaches. In Sect. 4 we will illustrate these methods in two example systems. First, we will apply our computational framework to the well-known periodically driven double-gyre flow. Second, we consider an approximation of the global ocean surface flow obtained by the numerical solution of the incompressible surface Navier-Stokes equation in a complex geometry on the sphere [20] and highlight transport barriers and coherent sets. We will conclude with a short discussion and outlook in Sect. 5.

2 Nonautonomous Dynamics, Transfer Operators and Transport

We consider a nonautonomous differential equation

$$\dot{\mathbf{x}} = \mathbf{u}(\mathbf{x}, t) \tag{1}$$

with state $\mathbf{x} \in M \subset \mathbb{R}^d$, time $t \in \mathbb{R}$ and sufficiently smooth right-hand side $\mathbf{u}$ such that the flow map $\Phi(\mathbf{x}, t; \tau) : M \times \mathbb{R} \times \mathbb{R} \to M$, $M \subset \mathbb{R}^d$ exists; here τ denotes the flow time and t the initial time. We are interested in extracting and visualizing coherent subsets of M, i.e. mobile regions in M that minimally mix with the surrounding phase space. Coherent structures and their boundaries provide a time-dependent skeleton of the dynamics, similar to invariant manifolds of hyperbolic fixed points in autonomous dynamical systems.

Many approaches are based on finite-time Lyapunov exponents

$$\Lambda_\tau(\mathbf{x},t) = \frac{1}{2|\tau|}\log\left(\lambda_{max}[D_\mathbf{x}\Phi(\mathbf{x},t;\tau)^\top D_\mathbf{x}\Phi(\mathbf{x},t;\tau)]\right). \tag{2}$$

Ridges in these scalar fields are frequently used to approximate transport barriers in the flow [14, 22]. This often heuristically applied concept has been recently set on a sound mathematical basis [15]. Finite-time Lyapunov exponents have also been successfully used for the Lagrangian visualization of flow structures [12, 13].

Probabilistic approaches aim at a direct extraction of finite-time coherent sets by solving an optimization problem. To be more precise, one seeks to find pairs of sets $(A_t, A_{t+\tau})$ such that

$$\rho(A_t, A_{t+\tau}) = \frac{\mu(A_t \cap \Phi(A_{t+\tau}, t+\tau; -\tau))}{\mu(A_t)} \tag{3}$$

is maximal under some constraints (conservation of mass, robustness with respect to perturbations). Such constraints are necessary to regularize the essentially ill-posed optimization problem [7]. Here μ is a reference probability measure on M at time t. Equation (3) measures the proportion of the set A_t at time t that is mapped to the set $A_{t+\tau}$ at time $t+\tau$. One seeks to find sets such that $A_{t+\tau} \approx \Phi(A_t, t; \tau)$. This problem can be solved by considering the Perron-Frobenius operator $\mathbf{P}_{t,\tau} : L^1(M,m) \to L^1(M,m)$ associated with the flow map Φ, where m denotes Lebesgue measure. The transfer operator is defined by

$$\mathbf{P}_{t,\tau} f(\mathbf{x}) = \frac{f(\Phi(\mathbf{x}, t+\tau, -\tau))}{|\det D\Phi(\Phi(\mathbf{x}, t+\tau, -\tau), t, \tau)|} \tag{4}$$

The interpretation is that if f is a density and $f(\mathbf{x})$ the density value in $\mathbf{x}$ at time t, then $\mathbf{P}_{t,\tau} f(\mathbf{x})$ describes the density value in $\Phi(\mathbf{x}, t; \tau)$ at time $t+\tau$ induced by the flow map. In [7, 10] it was shown that maximizing ρ can be described in the framework of optimizing an inner product involving a compact self-adjoint operator obtained from $\mathbf{P}_{t,\tau}$. In order to avoid the technical functional analytic description underlying [7, 10], we will briefly recall the concept of finite-time coherent sets in the finitary setting in Sect. 3.2. This will be based on a finite-rank approximation of $\mathbf{P}_{t,\tau}$ in terms of a stochastic matrix, which will be introduced in Sect. 3.1 (see also [11]).

$\mathbf{P}_{t,\tau}$ may also be used to derive a stretching measure, termed finite-time entropy (FTE). It is similar to finite-time Lyapunov exponents and based on the concept of differential entropy $h(f) = -\int_\Omega f \log f \, dm$, where Ω is the support of the density f. Note that on a given domain Ω, the uniform density maximizes h.

For a given initial condition $\mathbf{x}_0$, let $f_{\epsilon,\mathbf{x}_0} := (1/m(B_\epsilon(\mathbf{x}_0)))\mathbf{1}_{B_\epsilon(\mathbf{x}_0)}$ denote a uniform density supported on $B_\epsilon(\mathbf{x}_0)$, a ball of radius ϵ about $\mathbf{x}_0$. An ϵ-smoothing operator is defined by $\mathbf{A}_\epsilon f(\mathbf{x}) := (1/m(B_\epsilon(\mathbf{x}))) \int_{B_\epsilon(\mathbf{x})} f\, dm$.

The rate of increase in entropy experienced in the ϵ-neighborhood of $\mathbf{x}_0$ over the time span $[t, t+\tau]$ of the ϵ-perturbed dynamics can now be described by

$$FTE_\epsilon(\mathbf{x}_0, t; \tau) := \frac{1}{|\tau|}[h(\mathbf{A}_\epsilon \mathbf{P}_{t,\tau} f_{\epsilon,\mathbf{x}_0}) - h(f_{\epsilon,\mathbf{x}_0})]. \tag{5}$$

Note that FTE_ϵ takes high values when the evolved density $\mathbf{A}_\epsilon \mathbf{P}_{t,\tau} f_{\epsilon,\mathbf{x}_0}$ is very spread out. To be more precise, FTE_ϵ measures nonlinear stretching and the resulting quantity is very much determined by the sum of the positive finite-time Lyapunov exponents w.r.t $\mathbf{x}_0$. Thus, especially in the relevant setting of two-dimensional divergence-free flows, it can be well compared with finite-time Lyapunov exponents. In [9] several properties of FTE_ϵ and its deterministic limit $\lim_{\epsilon\to 0} FTE_\epsilon$ have been derived. In Sect. 3.3 we will outline a very efficient set-oriented approximation of the FTE-field.

3 Set-Oriented Numerical Framework

We now describe a set-oriented numerical framework for the approximation of the nonautonomous Perron-Frobenius operator in terms of a transition matrix of a finite-state Markov chain. The discretized transfer operator is the basis for extracting coherent sets (Sect. 3.2) and for the computation of FTE-fields (Sect. 3.3). In Sect. 3.4 we discuss how these approaches easily extend to the case of flows on a triangulated surface.

3.1 Approximation of Transfer Operator

Following [11], we consider some compact subset $X \subset M$ and a small neighborhood Y of $\Phi(X, t; \tau)$. Let $\{B_1, \ldots, B_k\}$ be a partition of X, $\{C_1, \ldots, C_n\}$ a partition of Y. The partition elements are typically generalized rectangles, but other settings are possible. Applying Ulam's method [23], a finite-rank approximation of $\mathbf{P}_{t,\tau} : L^1(X, m) \to L^1(Y, m)$ is given via the transition matrix

$$P_{ij} = \frac{m(B_i \cap \Phi(C_j, t+\tau; -\tau))}{m(B_i)},\ i = 1, \ldots, k,\ j = 1, \ldots, n \tag{6}$$

where we drop the t and τ-dependence of P for brevity. In practice the entries P_{ij} of the transition matrix P are estimated via

$$P_{ij} \approx \frac{\#\{r : \Phi(\mathbf{z}_{i,r}, t; \tau) \in C_j\}}{R}, \tag{7}$$

with uniformly distributed sample points $\mathbf{z}_{i,r}, r = 1, \ldots, R$ chosen in each partition element B_i, $i = 1, \ldots, k$. So P_{ij} describes the conditional probability that a point chosen at random in B_i will be mapped to C_j under the action of $\Phi(\cdot, t; \tau)$. Note that P is a sparse, row-stochastic matrix and thus all its eigenvalues are contained in the unit circle.

The interpretation of the P-induced dynamics is that if $\mathbf{p} \geq 0$ (component-wise) is a probability vector ($\sum_i p_i = 1$), then $\mathbf{p}' = \mathbf{p}P$ is the push-forward of $\mathbf{p}$ under the discretized action of $\Phi(\cdot, t; \tau)$. Note that the numerical scheme introduces diffusion—which is also theoretically needed for robust results [7, 10].

3.2 *Extracting Finite-Time Coherent Sets*

Consider a reference probability measure μ on X at time t, which is discretely represented as a probability vector $\mathbf{p}$ with $p_i = \mu(B_i)$, $i = 1, \ldots, k$. The image probability vector on Y at time $t + \tau$ is then simply computed as $\mathbf{q} = \mathbf{p}P$. We assume both $\mathbf{p} > 0$ and $\mathbf{q} > 0$ (component-wise) and form a normalized matrix L via

$$L_{ij} = \frac{p_i P_{ij}}{q_j}. \tag{8}$$

This matrix has the property that $\mathbf{1}_{\mathbb{R}^k} L = \mathbf{1}_{\mathbb{R}^n}$. In [7, 11] it was shown that (under some technical assumptions) the problem of finding optimal coherent sets can be approximated by considering the left eigenvectors $\mathbf{w}_2 \in \mathbb{R}^k$ of LL^* and $\hat{\mathbf{w}}_2 \in \mathbb{R}^n$ of L^*L to the second largest eigenvalue $\lambda_2 < 1$. Here $L^* = P^\top$. Note that these two eigenvalue problems can be turned into the task of finding leading singular values and corresponding left and right singular vectors of a sparse matrix (see [11] for the exact construction), which can be very efficiently computed by iterative schemes (e.g. `svds` in MATLAB). The signed vector entries of $\mathbf{w}_2$ and $\hat{\mathbf{w}}_2$ can be interpreted as relaxations of indicator functions of the sets A_t and $A_{t+\tau}$ and their complements. Thus the vector $\mathbf{w}_2$ defines fuzzy coherent sets on X, whereas $\hat{\mathbf{w}}_2$ represents their image on Y. Optimal partitions of X and Y into finite-time coherent pairs can be approximated via a line search in $\mathbf{w}_2$ and $\hat{\mathbf{w}}_2$ [10, 11], but plotting these vectors will already give a good visual impression of the location of coherent sets.

3.3 Set-Oriented Computation of FTE

In the discrete context, densities (which are central to the FTE-construction in Eq. (5)) are now represented by discrete probability measures μ. Thus, the entropy of a probability vector $\mathbf{p}$ with $p_i = \mu(B_i)$, $i = 1, \dots, k$, is simply $H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log p_i$. Under the assumption that all partition elements $\{B_1, \dots, B_k\}$ are of equal volume, let δ_i be an k-vector with 1 in i^{th} position and 0 elsewhere. Then the discrete FTE of a partition set B_i is given by

$$FTE(B_i, t; \tau) = \frac{1}{|\tau|} H(\delta_i P) = -\frac{1}{|\tau|} \sum_{j=1}^{n} P_{ij} \log P_{ij}. \tag{9}$$

In case of differing box volumes the equation reads

$$FTE(B_i, t; \tau) = -\frac{1}{|\tau|} \left(\sum_{j=1}^{n} P_{ij} \log \frac{P_{ij}}{m(C_j)} + \log m(B_i) \right). \tag{10}$$

In addition, one may obtain stretching rates for the backward-time dynamics (i.e. from $t + \tau$ to t) directly from the forward-time computation. For this consider $\tilde{P} \approx \mathbf{P}_{t+\tau,-\tau}$ that approximates the backward-time evolution where

$$\tilde{P}_{ji} = \frac{m(B_i \cap \Phi(C_j, t + \tau; -\tau))}{m(\Phi(C_j, t + \tau; -\tau))} = \frac{m(B_i) P_{ij}}{\sum_{i=1}^{k} m(B_i) P_{ij}}, \tag{11}$$

i.e. $\tilde{P}$ is computable directly from P and $m(B_i)$, $i = 1, \dots, k$. We denote the FTE-field obtained from $\tilde{P}$ by $\widetilde{FTE}$.

Note that once the transition matrix P has been computed, FTE-fields (5) can be very quickly approximated by application of Eqs. (9) or (10). In particular, we do not require to explicitly push forward probability densities with P. This is in contrast to the related uncertainty estimation proposed in [19].

3.4 Extension to Flows on Triangulated Surfaces

In order to calculate coherent sets for flows on closed oriented 2-manifolds Γ, i.e. surfaces of co-dimension one embedded in $\mathbb{R}^3$, we consider a partition $\mathscr{S}_h(\Gamma)$ of Γ that approximates the surface. We assume that $\mathscr{S}_h(\Gamma)$ is a conforming triangulation consisting of triangles, such that

$$\Gamma_h := \bigcup_{S \in \mathscr{S}_h(\Gamma)} S \subset \mathbb{R}^3$$

forms a polytope, with $\mathbf{v} \in \Gamma$ for all vertices $\mathbf{v} \in \mathscr{S}_h(\Gamma)$. Additionally, we assume the surface Γ to be fixed, so in the construction of the approximate transfer operator outlined in Sect. 3.1 one only needs to consider a single partition.

The transfer operator $\mathbf{P}_{t,\tau} : L^1(\Gamma, m) \to L^1(\Gamma, m)$ is approximated via

$$P_{ij} = \frac{m(S_i \cap \Phi(S_j, t+\tau; -\tau))}{m(S_i)}$$

where S_i denotes the i-th triangle in the triangulation $\mathscr{S}_h(\Gamma)$ and m Lebesgue measure on Γ_h.

The surface Γ is assumed to be oriented, thus also its approximation Γ_h. This orientation is given by the local numbering of the vertices $\mathbf{v}^S$ of the triangles S in $\mathscr{S}_h$. So one can calculate the outward normal $\mathbf{n}_S$ to each triangle $S \in \mathscr{S}_h(\Gamma)$, by

$$\hat{\mathbf{n}}_S := (\mathbf{v}_1^S - \mathbf{v}_0^S) \times (\mathbf{v}_2^S - \mathbf{v}_0^S), \quad \mathbf{n}_S = \frac{\hat{\mathbf{n}}_S}{\|\hat{\mathbf{n}}_S\|_2},$$

where the area of the triangle is given by $m(S) = \frac{1}{2}\|\hat{\mathbf{n}}_S\|_2$.

A simple approach to define a normal in each vertex of Γ_h that approximates the normal of Γ in this point is a weighted averaging over normals of adjacent triangles. Let $\mathscr{S}(\mathbf{v}) := \{S \in \mathscr{S}_h(\Gamma) : \mathbf{v} \in S\}$, where $\mathbf{v} \in S$ means that $\mathbf{v}$ is a vertex of S. The vertex normal $\mathbf{n}(\mathbf{v})$ is then given by

$$\mathbf{n}(\mathbf{v}) := \frac{\sum_{S \in \mathscr{S}(\mathbf{v})} \frac{\mathbf{n}_S}{m(S)}}{\sum_{S \in \mathscr{S}(v)} \frac{1}{m(S)}}.$$

In order to approximate the flow map $\Phi(S, \cdot, \cdot)$ that maps points in S to points in another triangle $S' \in \mathscr{S}_h$, we first map a point $\mathbf{x} \in \Gamma_h$ to a point $\hat{\mathbf{x}} \notin \Gamma_h$ and project $\hat{\mathbf{x}}$ back to the polytope.

Let $\hat{S} := \operatorname{argmin}\,(\operatorname{dist}(\hat{\mathbf{x}}, S) \;:\; S \in \mathscr{S}_h)$ be the triangle closest to $\hat{\mathbf{x}}$ and let

$$\hat{\mathbf{y}} := \operatorname{argmax}\Big(\|\mathbf{y}_i\|_2 \;:\; \mathbf{y}_i = \hat{\mathbf{x}} - \mathbf{v}_i^{\hat{S}},\ i = 0, 1, 2\Big)$$

be the longest edge connecting $\hat{\mathbf{x}}$ to the vertices of $\hat{S}$. Then we define the projection $\pi_h : \mathscr{U} \to \Gamma_h$, where $\mathscr{U}$ is a neighborhood of Γ_h, by

$$\pi_h(\hat{\mathbf{x}}) := \hat{\mathbf{x}} - \mathbf{n}_{\hat{S}}(\mathbf{n}_{\hat{S}} \cdot \hat{\mathbf{y}}) \in \Gamma_h.$$

In order to obtain an approximation $\Phi_h : \Gamma_h \times \mathbb{R} \times \mathbb{R} \to \Gamma_h$ for the flow map $\Phi : \Gamma \times \mathbb{R} \times \mathbb{R} \to \Gamma$ using the triangulated surface Γ_h and the projection operator π_h, we have to construct a time integration scheme on Γ_h. With stepsize Δt one step of the Euler scheme followed by projection is given by

$$\hat{\Phi}_h(\mathbf{x}_t, t; \Delta t) = \pi_h\left(\mathbf{x}_t + \Delta t\mathbf{u}(\mathbf{x}_t, t)\right), \tag{12}$$

for the nonautonomous system $\dot{\mathbf{x}} = \mathbf{u}(\mathbf{x}, t)$, with a given velocity field $\mathbf{u}$ and initial condition $\mathbf{x}(t) = \mathbf{x}_t$. We note that the error in the projection step is negligible compared to the local truncation error of the Euler scheme. The approximate flow map $\Phi_h(\cdot, t, t+\tau)$ is then obtained in terms of the respective sequence of Euler steps $\hat{\Phi}_h$. Let $\mathbf{u}$ be only represented by discrete values at the vertices of Γ_h, then the evaluation of $\mathbf{u}(\mathbf{x}, t)$ is given by a linear combination of the vertex values. To be more precise, let $\mathbf{x} = \lambda_0\,\mathbf{v}_0^S + \lambda_1\,\mathbf{v}_1^S + \lambda_2\,\mathbf{v}_2^S$ be the representation of $\mathbf{x} \in \Gamma_h$ in barycentric coordinates (with $\lambda_0 + \lambda_1 + \lambda_2 = 1$), then

$$\mathbf{u}(\mathbf{x}, t) = \lambda_0\,\mathbf{u}(\mathbf{v}_0^S, t) + \lambda_1\,\mathbf{u}(\mathbf{v}_1^S, t) + \lambda_2\,\mathbf{u}(\mathbf{v}_2^S, t).$$

The Euler-based integration scheme described here is only of order one and should therefore only be seen as a simple means to compute an approximate flow on a triangulated surface. This will be sufficient for the purpose of demonstrating the application of the transfer operator approach in Sect. 4.2, where only a modest flow time is needed. For studies that require higher numerical accuracy and longer flow times more sophisticated integration schemes should be used.

4 Examples

4.1 Double-Gyre Flow

As a benchmark problem for analysing flow structures, we consider the time-dependent system of differential equations [8, 22]

$$\begin{aligned}\dot{x} &= -\pi A \sin(\pi f(x, t)) \cos(\pi y) \\ \dot{y} &= \pi A \cos(\pi f(x, t)) \sin(\pi y) \frac{df}{dx}(x, t),\end{aligned} \tag{13}$$

where $f(x, t) = \delta \sin(\omega t)x^2 + (1 - 2\delta \sin(\omega t))x$. We choose $A = 1$, $\delta = 0.25$, $\omega = 2\pi$ and obtain a one-periodic flow. The FTLE fields at $t = 0$ for flow times $\tau = 1$ and $\tau = 2$ are shown in Fig. 1, highlighting parts of the stable manifold of a hyperbolic periodic orbit on the x-axis. This invariant manifold serves as a major transport barrier.

For the transfer operator approach we partition the domain $M = [0, 2] \times [0, 1]$ by $n = 32768 = 2^{15}$ square boxes. We form matrices P_i, $i = 1, \ldots, 5$ by integrating with a fourth-order Runge Kutta scheme with constant stepsize $h = 0.01$ over the

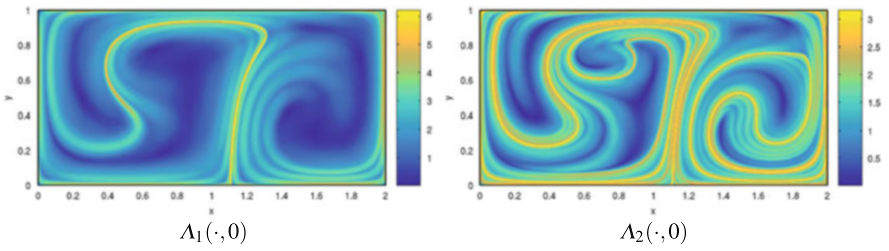


Fig. 1 FTLE fields on $[0, 1]$ and on $[0, 2]$ highlight parts of the stable manifold of a hyperbolic periodic orbit

time intervals $I_i = [(i-1)\tau, i\tau]$, $i = 1, \ldots, 5$, where $\tau = 0.2$, using $K = 100$ uniformly distributed test points per box (inner grid points). The corresponding algorithms are implemented in the software package GAIO [3]. The product of matrices $Q = P_1P_2P_3P_4P_5$ approximates the transfer operator on the time interval $[0, 1]$. As the flow is periodic, Q^2 does so on $[0, 2]$. Note that we could also have approximated the transfer operator by forming a matrix directly over the interval $[0, 1]$. However, again due to the periodicity, the concatenation of matrices allows us to easily change the intervals under consideration without any additional matrix approximations (which is the expensive part of the approach). Different FTE-fields extracted from the transition matrices are shown in Fig. 2 and they compare well with the FTLE fields in Fig. 1. We obtain the time-reversed FTE-fields ($\widetilde{FTE}$) by simple manipulation of the transition matrix according to Eq. (11). These pick up the unstable manifold emanating from the periodic orbit located on the upper boundary of M, see second row in Fig. 2.

We can also extract coherent sets via the vectors $\mathbf{w}_2$ and $\hat{\mathbf{w}}_2$ as described in Sect. 3.2. Lebesgue measure on M is taken as the reference measure. The corresponding coherent pairs are shown in Fig. 3. Here, negative and positive entries indicate the location of the two coherent pairs. Note that they align well with the transport barriers highlighted by FTLE or FTE.

Finally, we consider the stochastic differential equation (SDE)

$$d\mathbf{X}_t = \mathbf{u}(\mathbf{X}_t, t)dt + \delta\mathbf{h}(\mathbf{X}_t, t)d\mathbf{W}_t, \tag{14}$$

with $\mathbf{u}$ being the right-hand side of (13), $\delta = 0.001$, $\mathbf{h} \equiv (1, 0)^\top$, and $\mathbf{W}$ denoting a Wiener process. We impose periodic boundary conditions and integrate the SDE on $[0, 1]$ with an Euler-Maruyama scheme with stepsize $h = 0.01$. The resulting paths are then used to set up the transition matrix as before—without any postprocessing. Again large FTE values are found in the vicinity of the major transport barrier (Fig. 4).

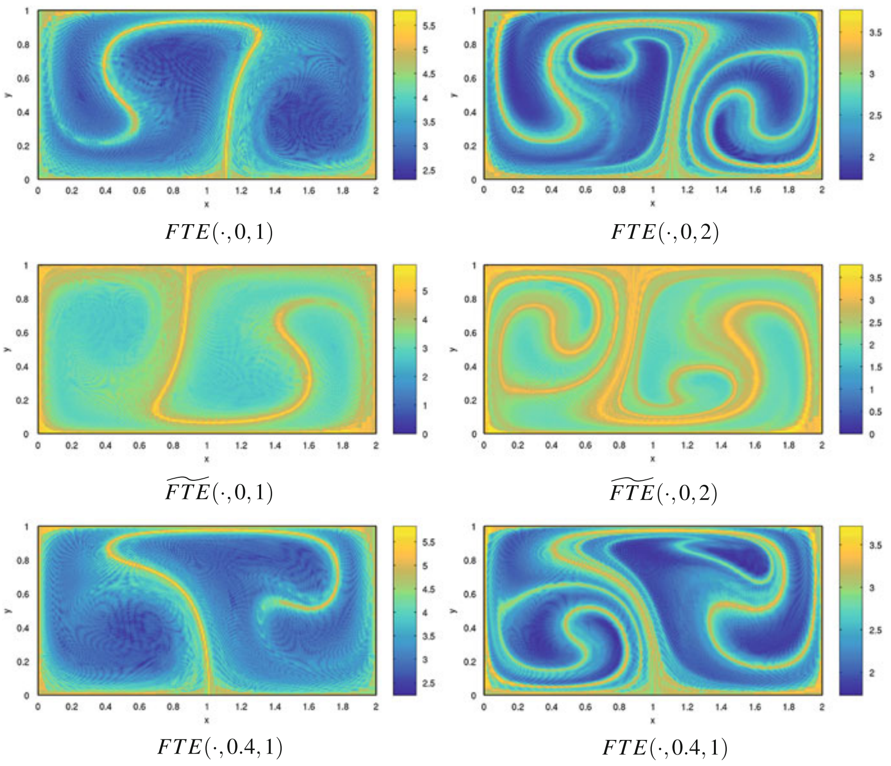


Fig. 2 Finite-time entropy fields computed on different intervals take high values in vicinity of major transport barriers

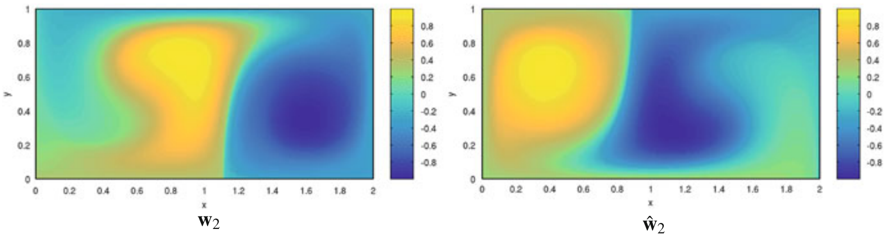


Fig. 3 Finite-time coherent sets on $[0, 1]$ are highlighted by the *left* and *right* second singular vectors $\mathbf{w}_2$ and $\hat{\mathbf{w}}_2$ of a reweighted transition matrix as described in Sect. 3.2

4.2 *Ocean Flow*

In [20] we have derived a simple two-dimensional model for fluid motion in the surface ocean, which is just driven by the Coriolis force. The basis is an incompressible surface Navier-Stokes equation with no-slip boundary conditions along

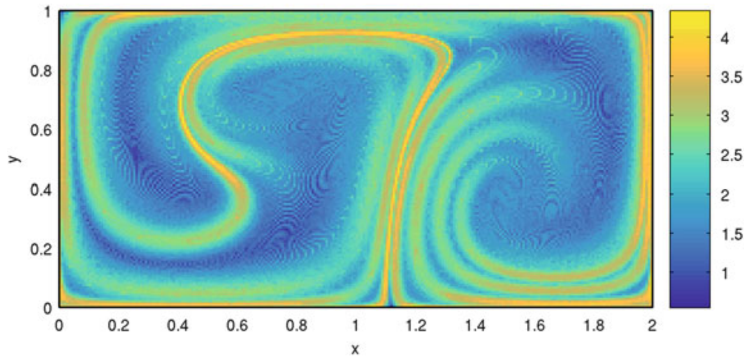


Fig. 4 Finite-time entropy field $FTE(\cdot, 0, 1)$ for the stochastic double gyre flow (14)

the continental borders. A diffuse domain approach [16] is used to describe the complex geometry on the whole spherical surface. The no-slip boundary condition is incorporated through a penalty like term. The equations now read

$$\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla_\Gamma \mathbf{u} + \gamma \mathbf{n} \times \mathbf{u} + \frac{\mu}{\epsilon^3}(1-\xi)\mathbf{u} = -\nabla_\Gamma p + \mu \left(\Delta_\Gamma^R \mathbf{u} + 2K\mathbf{u} \right) \quad \text{on } \Gamma$$

$$\nabla_\Gamma \cdot \mathbf{u} = 0 \quad \text{on } \Gamma$$

with Γ a two-dimensional surface, $\mathbf{u}$ the velocity vector, p the pressure, K the Gaussian curvature, ∇_Γ the surface gradient, Δ_Γ^R the Laplace-de Rham operator or Hodge-de Rham Laplacian, $\gamma = 2\omega \sin\theta$ the Coriolis parameter, ω the angular frequency, θ the latitude, $\mathbf{n}$ the surface normal, μ the viscosity, ϵ a small parameter, and ξ a phase-field variable representing the continental and oceanic phases. For more details—also on the choice of parameters—we refer to e.g. [16, 17, 20]. Following [17], we can derive a convenient vorticity-stream function formulation, which reads

$$\partial_t \phi + J(\psi, \phi + \gamma) - \frac{\mu}{\epsilon^3} \nabla_\Gamma \cdot ((1-\xi)\nabla_\Gamma \psi) = \mu(\Delta_\Gamma \phi + 2\nabla_\Gamma \cdot (K \nabla_\Gamma \psi)) \quad \text{on } \Gamma \tag{15}$$

$$\Delta_\Gamma \psi = \phi \quad \text{on } \Gamma$$

where ψ denotes the surface stream function, ϕ the surface vorticity, $J(\psi, \phi) = (\mathbf{n} \times \nabla_\Gamma \psi) \cdot \nabla_\Gamma \phi$ the Jacobian and Δ_Γ the Laplace-Beltrami operator. In the special case of the two-dimensional x-y-plane as the underlying geometry, the model agrees very well with common benchmark computations [21] and computations in multiply connected domains [1]. Note that the velocity field $\mathbf{u}$ can be retrieved from the stream function ψ by the relation $\mathbf{u} = \nabla_\Gamma \psi \times \mathbf{n}$.

The system (15) is numerically solved using adaptive parametric finite elements [4, 5, 24]. Within the finite element toolbox AMDiS [24], we use a domain decomposition approach and a parallel iterative solver BiCGStab(ell). For all simulations the unit sphere $\mathbb{S}_2 \subset \mathbb{R}^3$ serves as a simple representation of the earth. Moreover, an initial condition is chosen that includes the typical flow direction in the oceans to the west near the equator and to the east near the polar circles. The respective analytical form is given by $\psi_0(\mathbf{x}) = \frac{1}{10}x_3(1 - x_3^2)$ with $\mathbf{x} = (x_1, x_2, x_3)^T \in \Gamma = \mathbb{S}_2 \subset \mathbb{R}^3$, which ignores the appearance of the continental borders. We therefore need an initialization phase to fulfill the internal boundary conditions.

The stream function that we obtain numerically as a solution of Eq. (15) turns out to be nearly stationary. For the sake of simplicity, we consider a stationary velocity field derived from a time average of the stream function for our further investigations. Figure 5 (top) shows the time averaged stream function in terms of streamlines as well as the respective flow direction. The major flow structures—especially in the Indian, Pacific, Atlantic as well as in the Southern Ocean—can be observed and agree qualitatively with the known global circulation patterns.

To further visualize these major circulation patterns, we make use of the transfer operator-based approaches introduced in Sects. 2 and 3. As we have an autonomous velocity field, the coherent structures extracted by means of these methods should be aligned with streamlines and therefore our simplified setting is particularly suited for providing a proof of concept. We approximate the flow map numerically using the explicit Euler scheme (see Eq. (12)) with step size $\Delta t = 0.1$ and flow time $\tau = 50$ with respect to the approximated velocity field on the triangulated surface. The matrix entries P_{ij} are estimated using 105 uniformly distributed test points per triangle $S_i \in \mathscr{S}_h$ and counting how many of them are mapped to $S_j \in \mathscr{S}_h$, $i, j = 1, \dots, k$ with $k = 50802$. Note that the complete triangulation of Γ consists of 445812 triangles, but most of them are irrelevant for our probabilistic study (e.g. continents, highly resolved coastlines). We plot both $FTE(\cdot, t; \tau)$ and $\widetilde{FTE}(\cdot, t; \tau)$ see Fig. 5 (bottom). In fact, the FTE-fields nicely highlight the boundaries of the major oceans and also the circulation patterns of the Antarctic Circumpolar Current, which are also clearly visible in the respective plots of the time averaged stream function (Fig. 5 (top)).

Using the same matrix P from the FTE-study, we can also visualize finite-time coherent sets. For this, we approximate the leading left eigenvectors $\mathbf{w}_i$, $i = 2, 3, \dots$ of LL^* to eigenvalues very close to one, where we take normalized Lebesgue measure on Γ_h as the reference probability measure and construct L from P as described in Sect. 3.2.

$\mathbf{w}_2$ indicates that the Northern and Southern Pacific make up two coherent structures (left panel of Fig. 6), $\mathbf{w}_3$ looks similar (not shown), whereas $\mathbf{w}_4$ (center panel) suggests a partition of the Atlantic Ocean into two coherent sets with the sharp boundary apparently nicely aligned both with streamlines and also with FTE-ridges when compared with Fig. 5 (center). Finally $\mathbf{w}_5$ (Fig. 6, right) highlights a coherent feature in the Indian Ocean, which is also clearly delineated by the separatrix visible in Fig. 5 (right).

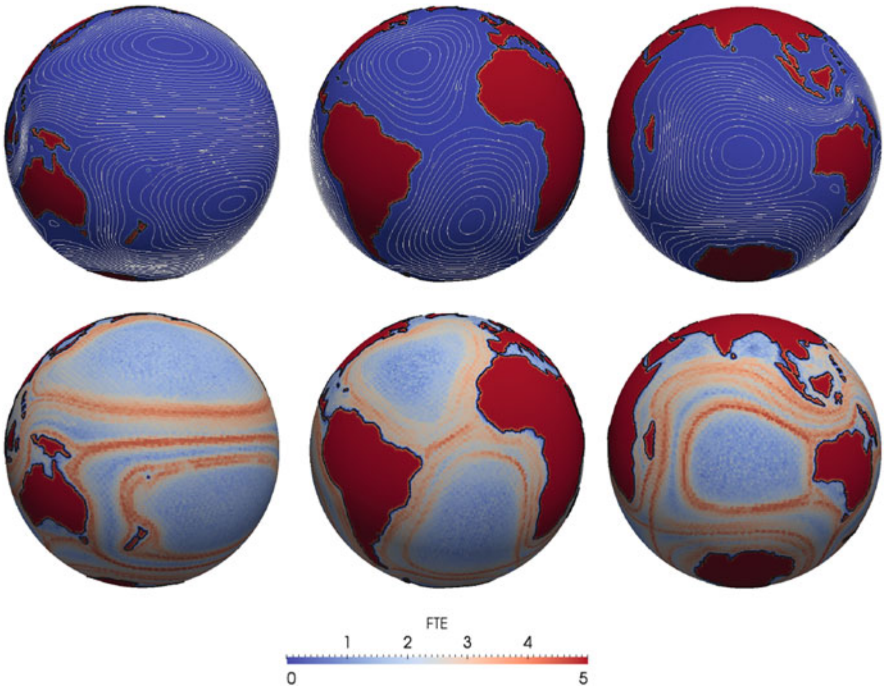


Fig. 5 *Top*: streamlines of the time averaged stream function indicate the major circulation patterns in the simulated ocean flow (from *left* to *right*: Pacific, Atlantic and Indian Ocean). *Bottom*: FTE is used to visualize the dynamical skeleton of the global ocean. Dark colors mark regions of high stretching indicating transport barriers. The results show the double-gyre structure in the Pacific Ocean delineated by regions of high FTE values (*left*), the circulation in the Indian Ocean (*right*), and the two major gyres in the Atlantic Ocean separated by a transport barrier (*center*)

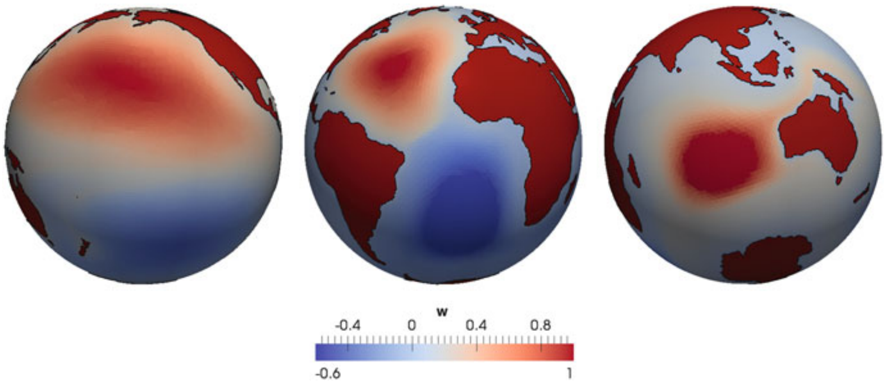


Fig. 6 Coherent structures in the simulated ocean flow are highlighted by the negative and positive entries of leading left eigenvectors $\mathbf{w}_i$ of a stochastic matrix derived from the transition matrix approximating the transfer operator of the flow map. *Left*: $\mathbf{w}_2$ defines coherent sets in the Pacific Ocean; *center*: $\mathbf{w}_4$ indicates a clear separation of the Northern and Southern Atlantic Ocean; *right*: $\mathbf{w}_5$ highlights a coherent feature in the Indian Ocean

5 Conclusion

In this contribution we have briefly reviewed the transfer operator-based framework for the analysis of transport phenomena in time-dependent dynamical systems and the numerical approximation of transfer operators, finite-time coherent sets and transport barriers. Unlike the frequently used geometric approaches for the extraction of Lagrangian coherent structures [15], our framework is purely probabilistic and thus relies neither on smoothness properties of the flow map nor on any distance measure that might be difficult to approximate in a complex domain. Therefore, the transfer operator-based computational methods are easily applicable for studying the dynamical skeleton in flows confined to complicated geometries. In addition to a numerical study of the well-known double gyre flow, we have illustrated the strength of our approach in an example of a two-dimensional surface flow on a sphere $\mathbb{S}_2 \subset \mathbb{R}^3$. With our methods we could highlight major circulation patterns of the global ocean. The spherical surface together with the continental borders give rise to a complex flow domain. Based on a transition matrix defined with respect to the triangulation of the surface and a stationary (numerically approximated) velocity field, we have visualized transport barriers and finite-time coherent sets. As expected, the extracted structures align well with equilines of the stream function in the considered autonomous dynamics, demonstrating that our approach returns meaningful results.

Future work will include the treatment of nonautonomous dynamics on evolving surfaces as well as the analysis of active fluid flows.

Acknowledgements KPG acknowledges funding from EU Marie-Skłodowska-Curie ITN Critical Transitions in Complex Systems (H2020-MSCA-2014-ITN 643073 CRITICS). SR and AV acknowledge support from DFG through Vo 899/6 and Vo 899/11. We further acknowledge the provided computing resources at ATLAS at ZIH at TU Dresden. Part of these results were obtained while AV was guest researcher at HIM at University Bonn.

References

1. Calhoun, D.: A Cartesian grid method for solving the two-dimensional streamfunction-vorticity equations in irregular regions. J. Comput. Phys. **176**(2), 231–275 (2002)
2. Dellnitz, M., Junge, O.: On the approximation of complicated dynamical behavior. SIAM J. Numer. Anal. **36**(2), 491–515 (1999)
3. Dellnitz, M., Froyland, G., Junge, O.: The algorithms behind GAIO – set oriented numerical methods for dynamical systems. In: Fiedler, B. (ed.) Ergodic Theory, Analysis, and Efficient Simulation of Dynamical Systems. Springer, Berlin (2009)
4. Dziuk, G., Elliott, C.M.: Finite elements on evolving surfaces. IMA J. Numer. Anal. **27**(2), 231–275 (2007)
5. Dziuk, G., Elliott, C.M.: Surface finite elements for parabolic equations. J. Comput. Math. **25**(4), 385 (2007)
6. Froyland, G.: Statistically optimal almost-invariant sets. Phys. D **200**, 205–219 (2005)

7. Froyland, G.: An analytic framework for identifying finite-time coherent sets in time-dependent dynamical systems. Phys. D **250**, 1–19 (2010)
8. Froyland, G., Padberg, K.: Almost-invariant sets and invariant manifolds – connecting probabilistic and geometric descriptions of coherent structures in flows. Phys. D **238**, 1507–1523 (2009)
9. Froyland, G., Padberg-Gehle, K.: Finite-time entropy: a probabilistic approach for measuring nonlinear stretching. Phys. D **241**, 1612–1628 (2012)
10. Froyland, G., Padberg-Gehle, K.: Almost-invariant and finite-time coherent sets: directionality, duration, and diffusion. In: Bahsoun, G.F.W., Bose, C. (eds.) Ergodic Theory, Open Dynamics, and Coherent Structures, vol. 70, pp. 171–216. Springer, Berlin (2014)
11. Froyland, G., Santitissadeekorn, N., Monahan, A.: Transport in time-dependent dynamical systems: finite-time coherent sets. Chaos **20**, 043,116 (2010)
12. Garth, C., Gerhardt, F., Tricoche, X., Hagen, H.: Efficient computation and visualization of coherent structures in fluid flow applications. IEEE Trans. Vis. Comput. Graph. **13**(6), 1464–1471 (2007)
13. Garth, C., Wiebel, A., Tricoche, X., Joy, K.I., Scheuermann, G.: Lagrangian visualization of flow-embedded surface structures. Comput. Graph. Forum **27**(3), 1007–1014 (2008)
14. Haller, G.: Distinguished material surfaces and coherent structures in three-dimensional fluid flows. Phys. D **149**, 248–277 (2001)
15. Haller, G.: A variational theory of hyperbolic Lagrangian coherent structures. Phys. D **240**, 574–598 (2011)
16. Li, X., Lowengrub, J., Rätz, A., Voigt, A.: Solving PDEs in complex geometries: a diffuse domain approach. Commun. Math. Sci. **7**, 81–107 (2009)
17. Nitschke, I., Voigt, A., Wensch, J.: A finite element approach to incompressible two-phase flow on manifolds. J. Fluid Mech. **708**, 418–438 (2012)
18. Padberg, K., Thiere, B., Preis, R., Dellnitz, M.: Local expansion concepts for detecting transport barriers in dynamical systems. Commun. Nonlinear Sci. Numer. Simul. **14**(12), 4176–4190 (2009)
19. Reich, W., Scheuermann, G.: Analysis of streamline separation at infinity using time-discrete Markov chains. IEEE Trans. Vis. Comput. Graph. **18**(12), 2140–2148 (2012)
20. Reuther, S.: Modellierung und Simulation von Oberflächenzirkulationen in Ozeanen (diploma thesis). Technical Report, Technische Universität Dresden (2012)
21. Schäfer, M., Turek, J.: Benchmark computations of laminar flow around a cylinder. In: Hirschel, B. (ed.) Flow Simulation with High-Performance Computers II. Springer, Berlin (1996)
22. Shadden, S.C., Lekien, F., Marsden, J.E.: Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. Phys. D **212**, 271–304 (2005)
23. Ulam, S.: Problems in Modern Mathematics. Interscience, New York (1964)
24. Vey, S., Voigt, A.: AMDiS: adaptive multidimensional simulations. Comput. Vis. Sci. **10**(1), 57–67 (2007)

Part VI
Software and Algorithms

ADAPT: Adaptive Thresholds for Feature Extraction

Peer-Timo Bremer

Abstract Threshold-based feature definitions remain one of the most widely used and most intuitive choice in a wide range of scientific areas. However, it is well known that in many applications selecting a single optimal threshold is difficult or even impossible. Common examples of this are quantities with locally exponential behavior like the scalar dissipation rate in turbulent combustion simulations or indicator functions, such as, vorticity or delta popular in defining vortices. In these cases, local thresholds defined, for example, using contour trees can provide significantly better results, but typically require a data analysis expert to create and use. We present the ADAPT framework a new open source code that transforms a given scalar field such that global thresholds in the resulting field correspond to a variety of local thresholds in the original data. Consequently, the resulting field can be easily processed using any of the existing tool chains and provides scientists easy access to a wide variety of more advanced feature definitions. Currently, the package provides two techniques to define local thresholds: The relevance transform originally developed for combustion analysis and a model based fit initially designed to extract eddies in global ocean simulations. Furthermore, it provides an extendable API to easily add other transforms.

1 Introduction

One of the oldest and most common methods to define features of interest is using thresholds, for example, to define burning regions of a flame [3] as regions of high fuel consumption, or vortices in a turbulent flow through vorticity or other indicators [1, 9, 12, 18]. However, in many areas finding a single global threshold that performs adequately throughout a given dataset is difficult if not impossible. A typical example of this is vortex detection as shown in Fig. 1. It shows vortices extracted from a jet in crossflow [11] using the Q-criterion [6] at various thresholds. Since the strength of the vortices differs significantly between the shear layer

P.-T. Bremer (✉)
Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA, USA
e-mail: bremer5@llnl.gov

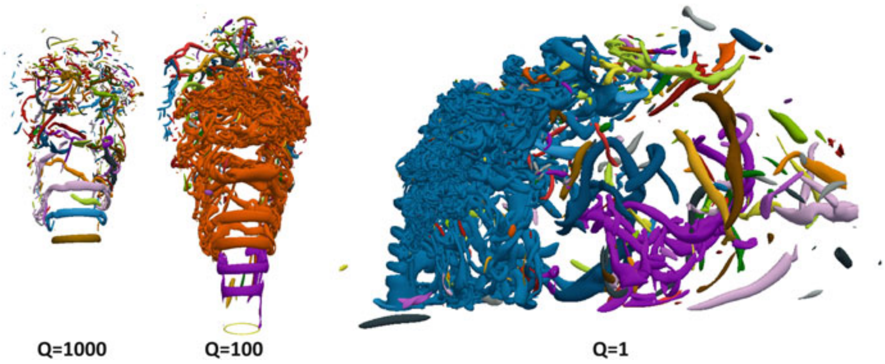


Fig. 1 Vortices in a turbulent combustion simulation identified using the Q criterion at different thresholds. Each image shows an isosurface of Q at the given threshold colored by component using on of 27 random colors. Clearly there exists no threshold able to extract both the strong vortices shown on the *left* as well as the weak ones on the *right*

vortices in front of the flame and the wake vortices behind, any threshold either misses the weak vortices entirely or drastically under-segments the strong vortices. Nevertheless, both types of vortices are equally important to understand the nature of the flow and the sensitivity of the solution to the choice of threshold has been a long standing point of critique [12].

Instead, more recently solutions have been proposed that rather than determining a single threshold allow one to compute thresholds on a per feature, i.e. per local neighborhood, basis [2, 13]. In particular, using merge trees to encode threshold based features [2, 3, 13] provides a simple and intuitive framework to define local thresholds as valid cuts through the tree. This follows a long standing trend of using topological information to provide more fine grained control over either the visualization or extraction of features. One of the earliest examples is the use of contour trees to accelerate the computation of iso-surfaces [19] which exploits the fact that "cutting" any branch of the contour tree by an isovalue is guaranteed to result in a single connected component of an iso-surface. While most early work still focuses on a single threshold and traditional iso-surfaces extraction, Carr et al. [5] propose to extract only specifically selected contours to create more sophisticated visualizations, for example, of medical scans. Weber et al. [20] use very similar ideas to construct localized transfer functions for volume rendering. Schneider et al. [16] use the locally largest contour to compare scalar fields and later multi-variate scalar fields [17]. One of the first applications to automatic feature extraction is the definition of the relevance metric by Mascarenhas et al. [13] which scales a threshold locally according to the highest local maximum. However, while well known in the data analysis and visualization community, these techniques, as currently described in the literature, require not only special tools to compute topological structures but also special tools to process and/or visualize the results. This has effectively prevented these techniques from being used more widely even though substantial benefits have been demonstrated in various application areas.

This paper presents the ADAPT—ADAPtive Thresholds framework (http://github.com/scalability-llnl/ADAPT), a light weight, extendable Open Source library to compute localized thresholds for arbitrary scalar fields. ADAPT exploits the well known yet rarely used fact that computing various metrics on, for example, a merge tree can be interpreted as transforming the tree and by extension the original function values. As will be discussed in more detail in Sect. 2 global thresholds in the transformed function correspond to local thresholds in the original one, yet can be extracted and processed with any off-the-shelf analysis software. In particular, ADAPT provides a fast in-memory merge tree computation for regular grids and two different examples of transformations: First, an implementation of the relevance metric [2, 13] as an example for a functional transformation; and Second, a model based fit using the R^2 criterion initially proposed to asses eddies in global ocean simulations [21]. In both cases the result is a transformed volume in which traditional iso-surfaces now correspond to local thresholds, i.e. relevance values, or goodness-of-fit in case of the R^2 criterion. Furthermore, the code is kept highly flexible and could easily be extended to include other transforms and models as well as other mesh types. We demonstrate the power of this approach using two examples: the jet in cross flow simulation created using S3D [7] and a global ocean simulation create by the POP ocean model [14].

2 Threshold-Based Features Using Merge Trees

As mentioned above, ADAPT exploits the fact that any valid cut through a merge tree corresponds to a segmentation using local thresholds defined as the function values at which branches of the tree are cut. Here we briefly reiterate the necessary concepts from scalar topology, define the concept of a *valid* cut through the tree, and finally introduce two approaches to define valid cuts.

2.1 Merge Trees

Given a simply connected domain $\mathbb{M}$ and a smooth function $f : \mathbb{M} \to \mathbb{R}$ the region $Sup_f(c) = \{p \in \mathbb{M} | f(p) \geq c\}$ of $\mathbb{M}$ with function value greater c is called the *super-level set* of c. We call a connected component of a super-level set a *super-contour*. The merge tree of f encodes the evolution of the super-contours as c is swept from ∞ to $-\infty$. Each time c passes a local maxima of f a new super-contour is created (Fig. 2a) and super-contours merge at selected saddles sometimes referred to as *merge-saddles* (Fig. 2b, c). Collectively, this structure is typically represented as a tree with local maxima forming the leaves, merge saddles the internal nodes, and the global minimum of f the root (see Fig. 2d) and traditionally the tree is drawn

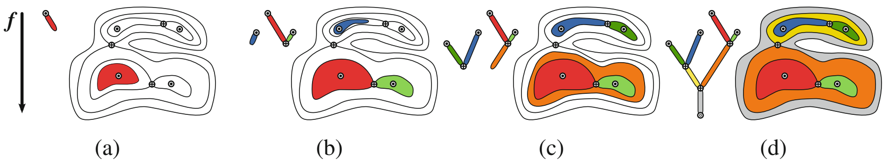


Fig. 2 Merge Trees encode how super-level sets merge as threshold value is swept through the range *top* to *bottom* (**a–d**). Each time the threshold passes a local maximum a new component on the super-level set is created which corresponds to a new leaf branch i the corresponding tree (a, b). As the threshold is lowered the corresponding super-contours grow and merge which corresponds to new valence three saddles in the tree. By construction, the branches of the tree correspond to regions in the domain making merge trees ideal to encode threshold based segmentations

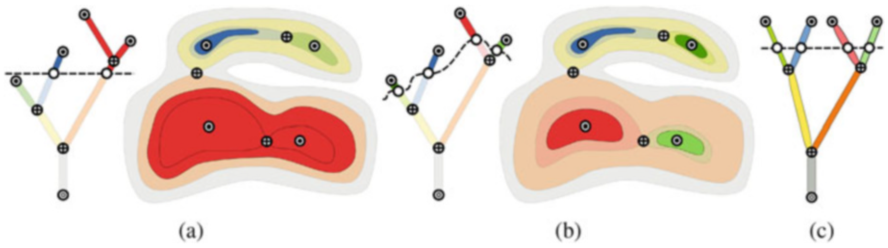


Fig. 3 (**a**) A traditional global threshold corresponds to a horizontal cut in the tree with each subtree above the threshold representing on feature. (**b**) Any valid cut through the tree defines a set of local thresholds and the corresponding segmentation. (**c**) Given an arbitrary cut, the tree can be transformed to make this cut a horizontal. The corresponding transformation of the original function results in a new scalar field in which global iso-surfaces correspond to local thresholds

such that the y-axis represents the function value. As indicated by the colors in the figures the branches of the tree directly correspond to regions of space which makes the tree very useful to define segmentations.

In particular, a traditional threshold based segmentation is defined as a horizontal cut in the tree. More specifically, as shown in Fig. 3a, to define features at any threshold one *cuts* the tree at the desired value and then collects all regions of the resulting subtrees as individual features. However, horizontal cuts are somewhat limiting. For example, in situations where the length of a branch indicates its importance, the dark green feature on the left of the tree in Fig. 2d is more significant than the light green one on the right. However, using horizontal cuts there exists no threshold to select the dark green feature before the light green one and in fact the former only appears once the latter has been absorbed by the red feature. Fortunately, there is no reason to restrict the technique to horizontal cuts. Instead, it is easy to see that any *valid* cut will result in a segmentation. In this context valid is defined as a cut that intersects each path from a leaf to the root at most once. For example, Fig. 3b shows a such a cut that enables one to simultaneously select individual features related to each local maximum. Effectively, the cut defines *local thresholds* within each branch used to define the local feature or alternatively a cut creates a set of subtrees corresponding to features. Furthermore, it is easy to see how approaches such as persistence-based simplification [5] could be used to remove noise and artifacts from the tree before defining a cut.

Defining features in terms of cuts through the merge tree now provides a significant flexibility in adapting segmentations to a particular application. ADAPT currently provides examples for two classes of cuts: *Monotone transforms*; and *Model-based fits*. The former is typically a functional expression in terms of local function values while the latter selects subtrees based on how well they represent a given model of a feature.

2.2 Monotone Transforms

One of the easiest ways to define useful cuts is by using other monotone properties of super-contours. Monotone transforms in general are very useful as they effectively define a new tree (see Fig. 3c) in which horizontal cuts correspond to adaptive cuts in the original tree. Furthermore, the transform of the tree also implies a transformation of the original function such that iso-surfaces in the transformed volume correspond to adaptive thresholds according to whatever metric used. For example, some of the metrics proposed in [5] such as volume or hypervolume naturally define new functions that select super-contours based on size. More abstract examples are indicators such as the height of a subtree, i.e. the maximal number of nodes from the root to any leaf. For example, defining the *height* of a point p with respect to its merge tree branch $\overline{a,b}$ can be defined as:

$$height(p) = \frac{f(b) - f(p)}{f(b) - f(a)} + height(b),$$

where the *height* of a maximum is defined as 0 and each saddle uses the *height* of is larger subtree . A cut at $height = 1 - \epsilon$ produces the largest super-contours as defined in [16] and other heights may produce interesting generalizations. Another option is to re-scale the threshold values. For example, computing the *relevance* of a feature has shown to be very effective in a number of scientific application. The relevance of any point p on the merge tree is defined as

$$relevance(p) = 1 - \frac{localMax(p) - f(p)}{localMax(p) - globalMin},$$

where the local maximum of p is the function value of the highest maximum in p's subtree. Relevance ranks points according to their relative distance in function space from the most dominant point in their neighborhood. Note that, this is a monotone transform in the sense that if two points p and q are on the same path to the root of the tree then $f(p) < f(q)$ implies $relevance(p) < relevance(q)$. Relevance has been used successfully in large scale turbulent combustion to analyzed local extinction regions [2, 13] and to define vortices [10].

2.3 Model-Based Fits

Monotone transforms especially those defined through closed form solutions like relevance are somewhat limited. They must preserve a strong connection to the original function values which may not contain sufficient information to accurately define features. Instead, one can directly consider all possible features that could be defined through super-contours, i.e. through local thresholds. As mentioned above, each subtree defines a potential feature. Therefore, assuming some metric to evaluate the quality of a feature one could iterate through all subtrees and find all features above a given quality threshold. Figure 4 shows an example for a "sphericalness" metric that uses the ration of principle axis of a least-squares ellipsoid fit to define how circular a contour is. Note that this selection may contain nested features as not all combination of subtrees define a valid cut as the selected subtrees may not be disjunct (Fig. 4b). To avoid this problem it is often useful to compute the quality of features and then either *inflate* or *deflate* the values to create a monotone transform (Fig. 4c).

Inflating a metric will assign each node in the merge tree the maximum of its own quality metric and that of its parent while deflating it assigns it the minimum between its own metric and that of its children. Intuitively, deflating a metric is a conservative choice and will select features for which all nested features are at least as good. Since many, especially model based, quality metrics can be unreliable for small features inflating a metric is often a better choice since it will select the locally largest feature of a certain quality. Both inflated and deflated metrics by definition are monotone transforms and thus can be translated into a transformation of the original function. As discussed in more detail in Sect. 4 model based fits have recently been used for eddy surveys in global ocean simulations [14, 21] to rank eddies according to an idea model.

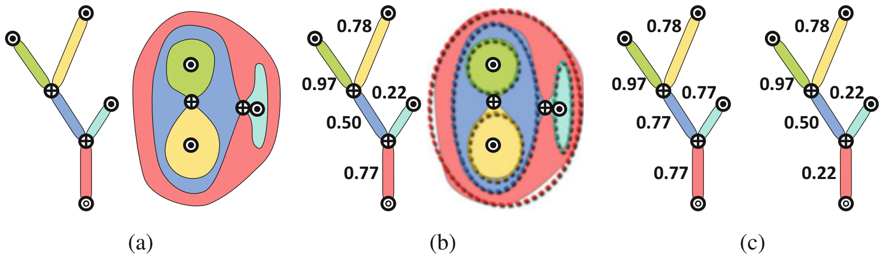


Fig. 4 A model based transformation according to how circular contours are. (**a**) The original tree and corresponding contour plot. (**b**) The least-squares ellipsoidal fits for each contour and their "cyclicity" values defined as the rations of the principle axis. (**c**) The cyclicity metric inflated (*left*) and deflated (*right*) to create a monotone transformation

3 Software

This section discusses some of the implementation details and design decisions. Note that the implementation is flexible and expected to change as new features are added. Therefore, we refer the reader to the code documentation for the most up-to-date details. ADAPT consists of three interfaces for the input mesh, the merge tree structure, and the metric used to define adaptive thresholds. Furthermore, it contains a prototypical merge tree implementation using a variant of the algorithm proposed by Carr et al. [4].

3.1 Input Mesh

Assuming a monotone interpolation scheme along edges the merge tree computation relies solely on whether two vertices are connected by a super-contour. Consequently, only the one-skeleton of any mesh is required. Therefore, the input mesh is simply described as an array of function values of vertices and an iterator that for each vertex provides the list of neighbors. Currently, ADAPT implements regular, but potentially curved, grids using a fully connected neighborhood of 8 and 26 neighbors for 2D and 3D respectively.

3.2 Merge Tree Interface

ADAPT implements an index based tree structure in which each node stores a single pointer to its *parent* (the next node towards the root) and a single pointer to one of its *children* as well as a pointer to a sibling to form a circular list of siblings. This structure encodes both up (towards the leafs) and down (towards the root) pointer with a constant memory footprint. Furthermore, each node optionally stores the list of vertices corresponding to its lower arc. While this can be memory intensive it is often necessary to efficiently compute model based fits as otherwise extracting the collection of vertices corresponding to a subtree would require a potentially costly mesh traversal. Furthermore, for convenience each node stores the index of its *representative* the highest maximum within its corresponding subtree. Representatives are useful, for example, to compute relevance values without a tree traversal or as a stable means to assign colors to the features as thresholds are changed.

3.3 Metric Interface

Currently, there exist two type of metrics: First, functional transforms that require only the structure of the tree and the local function values; and Model based fits that require fully populated arcs and are inflated by default. Functional metrics are evaluated on a per-vertex bases. However, for efficiency reasons model based fits are typically only computed for nodes of the merge tree and all vertices on an arc inherit the metric of the upper critical point. One potential problem of evaluating model based metrics only on nodes in the merge tree are that it introduces potentially substantial sampling artifacts especially for long arcs. To address the problem ADAPT provides the ability to refine the merge tree by splitting arcs either according to their range in function value or the number of vertices to ensure a sufficient resolution.

3.4 Application

The primary goal of ADAPT is to provide non-experts a simple tool to create transformed volumes for further processing. Therefore, we provide a command line tool that accepts a regular grid of up to three dimensions as input and produces a transformed volume as output. As mentioned above the various command line options are expected to change and we refer the reader to the online help for the latest information. As of the submission of this manuscript all options shown in Table 1 are enabled

The threshold option allows users to simply ignore values below/above a given value which can provide a substantial speed-up as often large amounts of a volume

Table 1 Command-line options for the adaptive_threshold executable

Option	Description
- -input <string>	Name of the input file
- -output <string>	Name of the output file (defaults to stdout)
- -dim <int><int><int>	Dimension of the input grid
- -tree-type <int>	[0 (default) | 1] to compute merge or split trees respectively
- -threshold <float>	Minimal (merge trees) or maximal (split-trees) function value considered relevant
- -split-type <string>	[length (default) | size] the metric used to split arcs of the tree until all arcs are shorter, for length, or contain fewer vertices, for size, than the given threshold
- -split <float>	The maximal length or size allowed for an arc
- -metric <string>	[relevance (default) | R2] the metric used for the transformation

are uninteresting. As discussed above the user can decided to split arcs either to limit their range or the number of vertices the contain to increase the fidelity of the R^2 criterion.

4 Examples

To demonstrate the utility of ADAPT we present two use cases: Vortex finding in a turbulent flow; and Model based eddy detection in an ocean simulation.

Figure 6 shows the same simulation of Fig. 1 at a relevance threshold of 0.8. To this end we transformed the original Q field for the entire $1399 \times 1080 \times 1100$ grid, loaded the result into Paraview [8] and colored the 0.8 isosurfaces by component using 27 randomly assigned colors. Clearly, the relevance is able to extract a vastly greater number of well formed vortices than the global thresholds including the extremely weak vortices on the side and below the flame as well as the very strong vortices in front of it. Note that, in practice, not all the extracted features are necessarily useful as the transformation also enhances very minor local fluctuations. However, these are typically filtered, for example, by size or by setting an appropriate global cut-off or are simply ignored as part of the expected noise and artifacts in the statistics, i.e. see Fig. 9a in [2]. Even considering these artifacts we have found ADAPT to provide a significant advantage as dealing with a small number of undesired features tends to be preferable over the inability to extract all features of interest. Figure 5 shows the results on a subsampled subset of this data which is included in the source distribution as example. The results of Fig. 5d were produced using the following command line:

```
adaptive_threshold  --i jicf_Q_400x350x200.raw
                    --dim 400 350 200
                    --threshold 1
                    --metric relevance
```

Figure 7 shows isolines at various global thresholds of the Okubo-Weiss criterion of a global ocean simulation. The Okubo-Weiss criterion is the default indicator for eddies in ocean simulations and habitually used, for example, to compute surveys of all eddies. Note that low values of Okubo-Weiss indicate eddies with the theoretically ideal threshold of 0. Similar to the vortex case, low thresholds produce few very strong eddies but miss a large number of weaker ones. However, higher thresholds produce very large numbers of misshapen eddies caused by shear and other artifacts that are false positive detections. Furthermore, individual eddies start merging into larger false structures. Instead, we compute the (inflated) R^2 goodness-of-fit of all potential eddies with respect to a well respected model of an ideal eddy [21]. Figure 8 shows the example data provided with the distribution

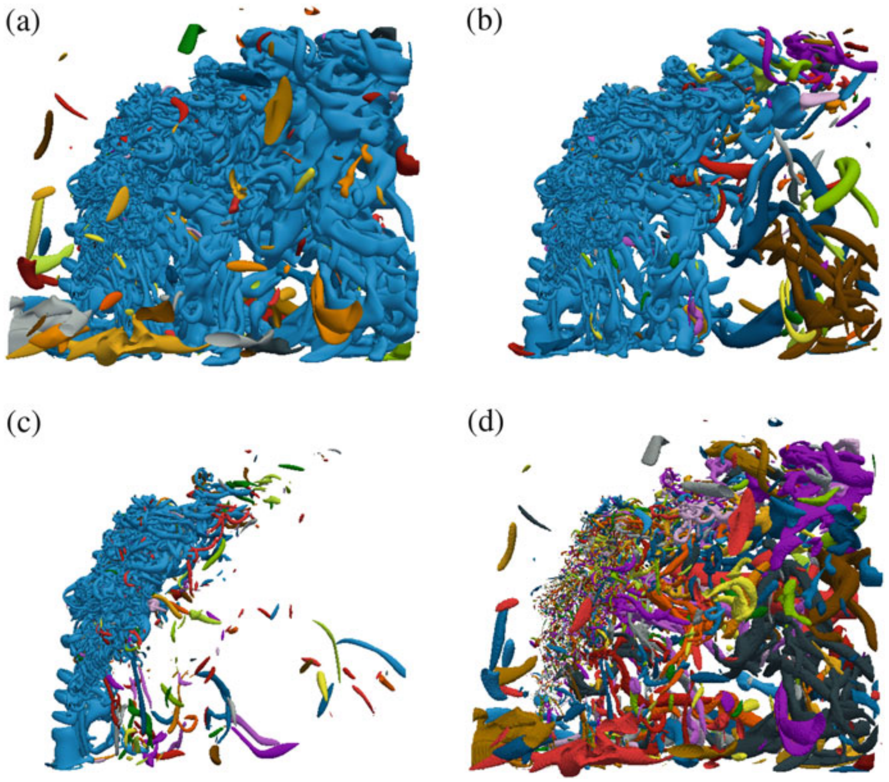


Fig. 5 Vortices in a subsampled subset of the turbulent combustion simulation of Figs. 1 and 6 for Q threshold 0.1 (**a**), 1 (**b**), and 10 (**c**) respectively and for a relevance of 0.8 (**d**)

which is the surface slice of a large scale POP simulation performed at Los Alamos National Laboratory [14, 15]. The results of Fig. 8 were produced using the following command line:

```
adaptive_threshold  --i ../../data/OkuboWeiss_03_01.raw
                    --dim 3600 2400 1
                    --threshold -0.001
                    --metric R2
                    --split-type size
                    --split 50
                    --tree-type 1
```

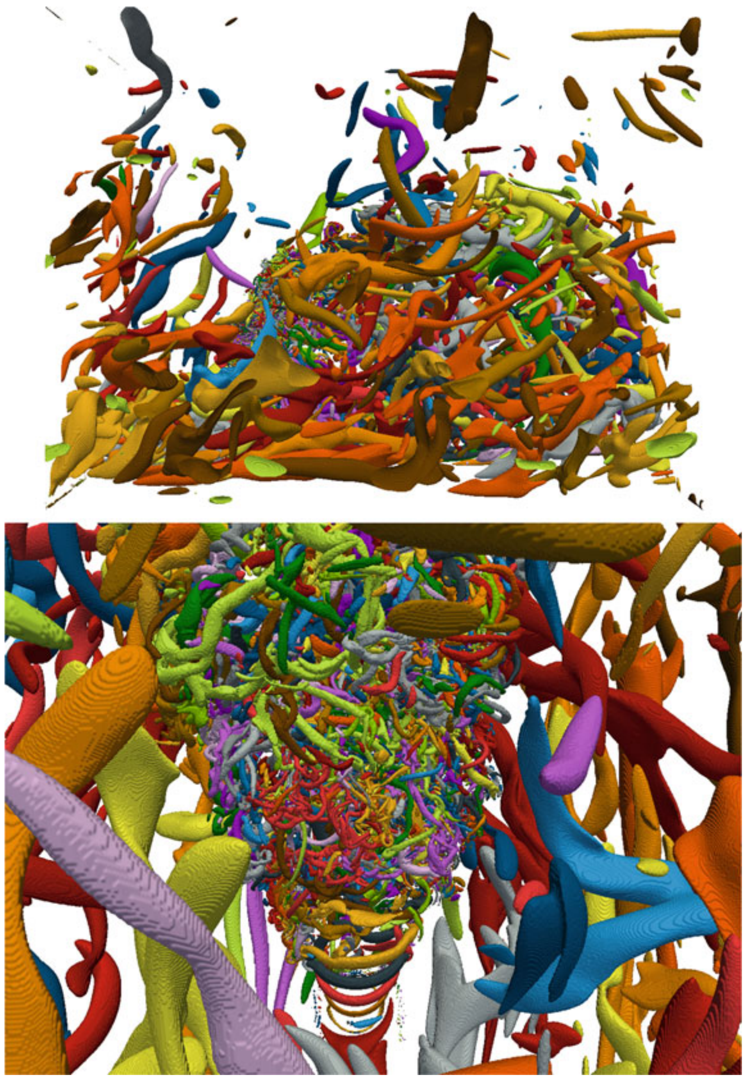


Fig. 6 Vortices in the turbulent combustion simulation of Fig. 1 at relevance 0.8 from the side (*top*) and a zoomed in view of the front (*bottom*). Both images show isosurfaces at 0.8 colored by component and randomly assigned one of 27 colors

Figure 7a–c shows various Okubo-Weiss thresholds with similar artifacts to the combustion simulation above. Figure 8 shows a model based threshold of 0.9, i.e. all local eddies that fit the model with a coefficient of determination of 0.9 or greater. As expected, the model based threshold is able of extracting significantly more well-formed eddies as compared to a global Okubo-Weiss threshold.

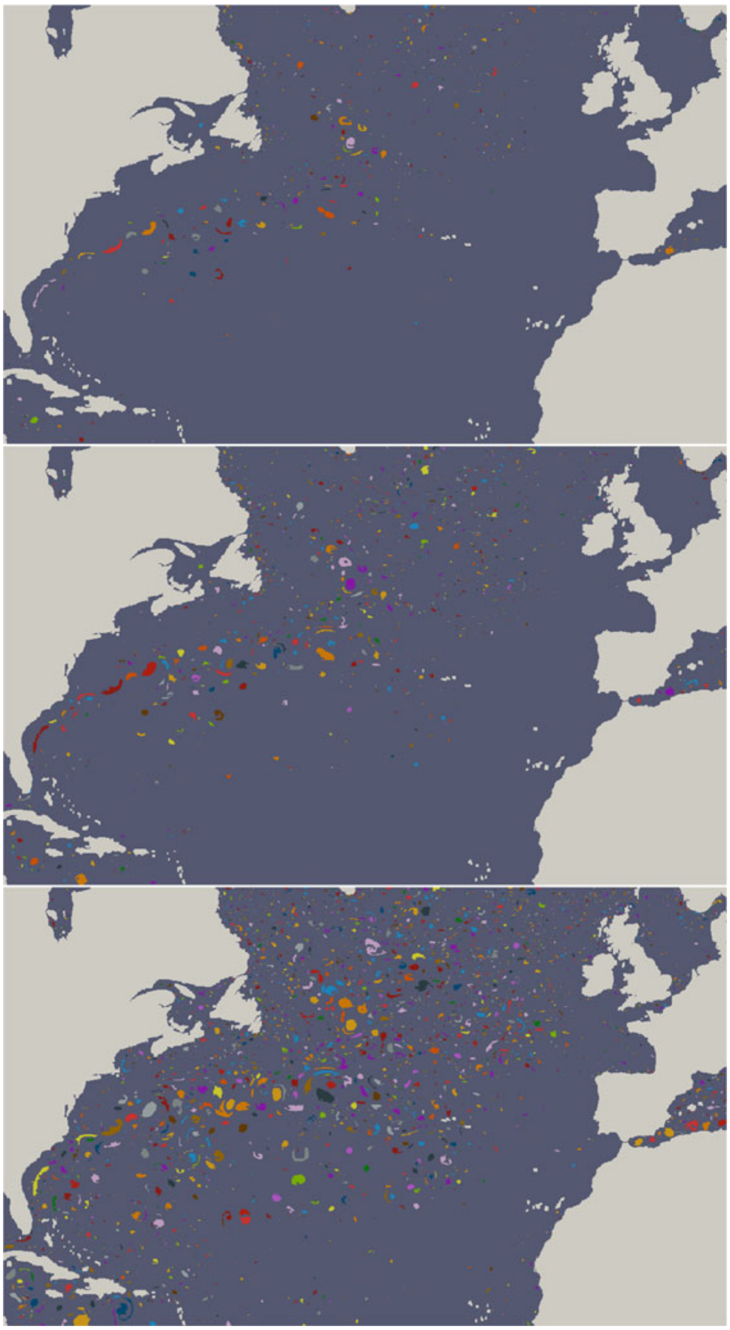


Fig. 7 Eddies on the surface of a global ocean simulation. The images show connected components of Okubo-Weiss thresholds randomly colored using 27 colors at -1 (*top*), -0.5 (*middle*), and -0.1 (*bottom*)

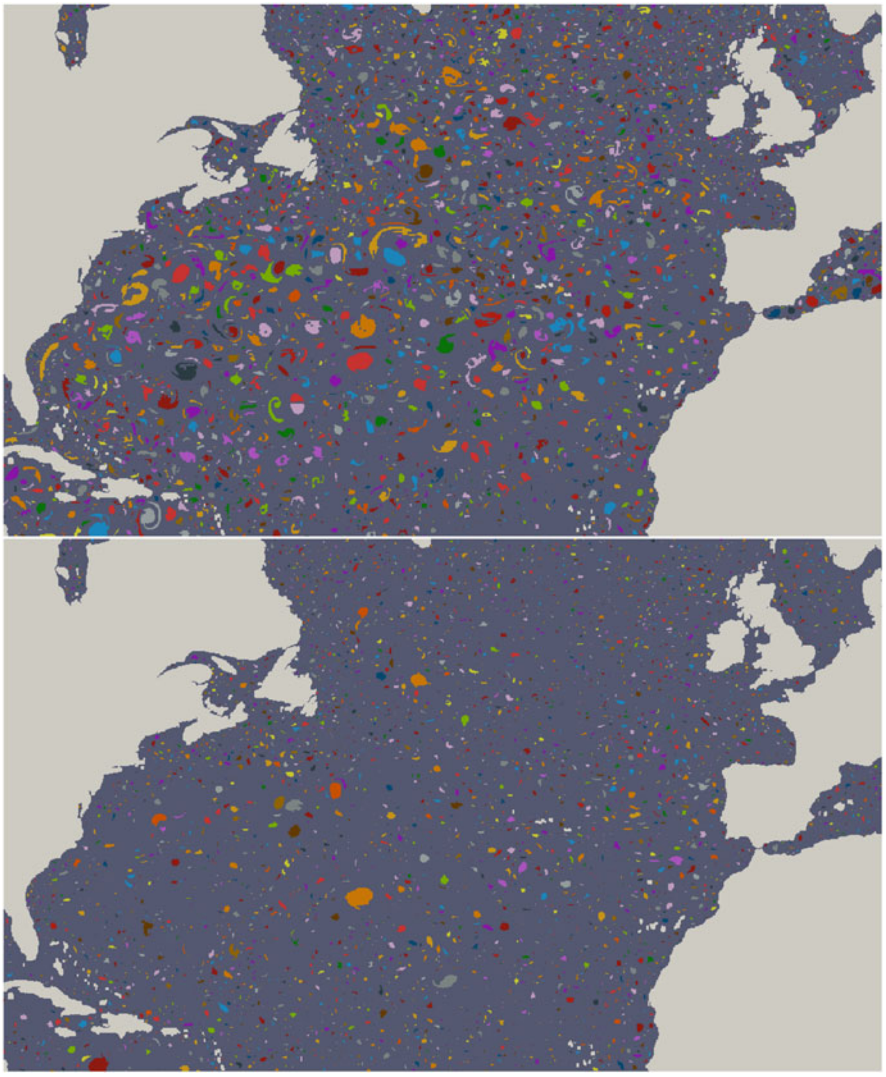


Fig. 8 Eddies on the surface of a global ocean simulation at Okubo-Weiss threshold of -0.01 (*top*) and extracted at a coefficient of determination threshold of 0.9 (*bottom*). The well-formed eddies from the *top* figure are all found including both the strong eddies in the Gulfstream as well as much weaker ones further south yet few of the erroneous ones are flagged

5 Summary

We have presented the ADAPT—ADAPtive Thresholds framework to allow non-experts easy access to some of the more advanced feature definitions developed by the data analysis and visualization community. In particular, ADAPT provides a simple and efficient tool to transform a given scalar field according to various metrics such that the resulting transformed scalar represents a potentially highly advanced feature definition (e.g. a model based fit) yet can be easily processed using common visualization tools. In the future, we plan to extent the framework to include additional meshes as well as new metrics and other features.

Acknowledgements This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) program. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. UCRL LLNL-CONF-681469.

References

1. Ashurst, W.T., Kerstein, A., Kerr, R., Gibson, C.: Alignment of vorticity and scalar gradient with strain rate in simulated Navier-Stokes turbulence. Phys. Fluids **30**(8), 2343– (1987)
2. Bennett, J., Krishnamurthy, V., Liu, S., Pascucci, V., Grout, R., Chen, J., Bremer, P.T.: Feature-based statistical analysis of combustion simulation data. IEEE Trans. Vis. Comput. Graph. **17**(12), 1822–1831 (2011)
3. Bremer, P.T., Weber, G., Tierny, J., Pascucci, V., Day, M., Bell, J.B.: Interactive exploration and analysis of large scale simulations using topology-based data segmentation. IEEE Trans. Vis. Comput. Graph. **17**(9), 1307–1324 (2011)
4. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. Theory Appl.**24**(3), 75–94 (2003)
5. Carr, H., Snoeyink, J., van de Panne, M.: Simplifying flexible isosurfaces using local geometric measures. In: Proceedings of IEEE Visualization, pp. 497–504. IEEE Computer Society (2004)
6. Chakraborty, P., Balachandar, S., Adrian, R.J.: On the relationships between local vortex identification schemes. J. Fluid Mech. **535**, 189–214 (2005)
7. Chen, J.H., Choudhary, A., deSupinski, B., DeVries, M., Hawkes, E.R., Klasky, S., Liao, W.K., Ma, K.L., Mellor-Crummey, J., Podhorszki, N., Sankaran, R., Shende, S., Yoo, C.S.: Terascale direct numerical simulations of turbulent combustion using s3d. Comput. Sci. Discovery **2**, 015001 (2009)
8. Fabian, N., Moreland, K., Thompson, D., Bauer, A., Marion, P., Gevecik, B., Rasquin, M., Jansen, K.: The paraview coprocessing library: a scalable, general purpose in situ visualization library. In: Proceedings of IEEE Symposium on Large Data Analysis and Visualization (LDAV), pp. 89–96 (2011)
9. Gibson, C.H.: Isoenstrophy points and surfaces in turbulent flow and mixing. Fluid Dyn. Res. **3**(1–4), 331–336 (1988)
10. Grout, R., Gruber, A., Kolla, H., Bremer, P.T., Bennett, J., Gyulassy, A., Chen, J.: Heat release and turbulence statistics from a DNS of reacting jet in cross- flow parameterized in a jet natural coordinate system developed from scalar quantities. In: Proceedings of 3th International Conference on Numerical Combustion (2011)
11. Grout, R., Gruber, A., Kolla, H., Bremer, P.T., Bennett, J., Gyulassy, A., Chen, J.: A direct numerical simulation study of turbulence and flame structure in a round jet in cross-flow. J. Fluid Mech.**706**, 351–383 (2012)
12. Jeong, J., Hussain, F.: On the identification of a vortex. J. Fluid Mech. **285**, 69–94 (1995)
13. Mascarenhas, A., Grout, R.W., Bremer, P.T., Hawkes, E.R., Pascucci, V., Chen, J.: Topological feature extraction for comparison of terascale combustion simulation data. In: Pascucci, V., Tricoche, X., Hagen, H., Tierny, J. (eds.) Topological Methods in Data Analysis and

Visualization: Theory, Algorithms, and Applications. Mathematics and Visualization, pp. 229–240. Springer, Berlin (2011)

14. Petersen, M., Williams, S., Maltrud, M., Hecht, M., Hamann, B.: A three-dimensional eddy census of a high-resolution global ocean simulation. J. Geophys. Res. **118**, 1759–1774 (2013)
15. Petersen, M., Williams, S., Maltrud, M., Hecht, M., Hamann, B.: A three-dimensional eddy census of a high-resolution global ocean simulation – supplementary information (2015). http://onlinelibrary.wiley.com/doi/10.1002/jgrc.20155/suppinfo
16. Schneider, D., Wiebel, A., Carr, H., Hlawitschka, M., Scheuermann, G.: Interactive comparison of scalar fields based on largest contours with applications to flow visualization. IEEE Trans. Vis. Comput. Graph. **14**, 1475–1482 (2008)
17. Schneider, D., Heine, C., Carr, H., Scheuermann, G.: Interactive comparison of multifield scalar data based on largest contours. Comput. Aided Geom. Des. **30**(6), 521–528 (2013)
18. Tanahashi, M., Miyauchi, T., Ikeda, J.: Scaling law of coherent fine scale structure in homogeneous isotropic turbulence. In: Proceedings of 11th Symposium on Turbulent Shear Flows, Grenoble, France, pp. 4–17 (1997)
19. van Kreveld, M.J., van Oostrum, R., Bajaj, C.L., Pascucci, V., Schikore, D.: Contour trees and small seed sets for isosurface traversal. In: Symposium on Computational Geometry, pp. 212–220 (1997)
20. Weber, G., Dillard, S., Carr, H., Pascucci, V., Hamann, B.: Topology-controlled volume rendering. IEEE Trans. Vis. Comput. Graph. **13**(2), 330–341 (2007)
21. Williams, S., Petersen, M., Bremer, P.T., Hecht, M., Pascucci, V., Ahrens, J., Hlawitschka, M., Hamann, B.: Adaptive extraction and quantification of atmospheric and oceanic vortices. IEEE Trans. Vis. Comput. Graph. **17**(12), 2088–2095 (2011)

Efficient Software for Programmable Visual Analysis Using Morse-Smale Complexes

Nithin Shivashankar and Vijay Natarajan

Abstract The Morse-Smale complex is a topological data structure that represents the behavior of the gradient of an input scalar field. Recent years have witnessed a significant number of applications that use this data structure for visualization and analysis of data from various scientific domains. However, these applications have required significant expertise in the implementation of algorithms. This potentially makes such analysis inaccessible to a large audience. In this paper we present open source software modules for the computation, analysis, and visualization of scientific data using the Morse-Smale complex. The modules, named *pymstri* and *pyms3d*, are intended for domains represented using 2D triangle meshes and 3D structured grids respectively. The software is designed to significantly reduce the effort required to use Morse-Smale complex based analysis. Also, the software leverages modern multi-core CPU and GPU architectures for computational efficiency. We demonstrate the usefulness via a case study to visually analyze and interactively segment the eye of the Hurricane Isabel simulation dataset. In particular, we highlight the ability to couple the visual analysis and the computation with ParaView, a popular general purpose visualization tool. The code is available at the project website http://vgl.csa.iisc.ac.in/mscomplex.

1 Introduction

In recent years, topological methods have gained wide popularity for scientific data analysis. In particular, gradient based analysis and the Morse-Smale complexes have been extensively applied for a multitude of data-analysis and visualization tasks [8, 15, 16, 20, 28, 30, 40]. A key reason for the success of Morse-Smale complex based analysis is that it enables a translation from scalar function data into topology and gradient based features. It is therefore no surprise that a lot of

N. Shivashankar (✉) • V. Natarajan
Department of Computer Science and Automation, Indian Institute of Science, CV Raman Road, Bangalore-12, India
e-mail: nithin@csa.iisc.ernet.in; nithin19484@gmail.com; vijayn@csa.iisc.ernet.in

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_19

recent work has focused on various aspects of the Morse-Smale complex such as its efficient computation [14, 21, 32, 34, 37, 38], feature directed visualization [22], user defined feature preservation [23, 24], etc. In most of the above applications of the Morse-Smale complex, the software implementations are not open source. Also, the software is often developed as standalone a executable that does not easily interface with other large software tools.

Related Work Morse-Smale complexes were first introduced for the analysis of dynamical systems by Smale [39]. The first computational algorithms were described by Edelsbrunner et al. [12] and Bremer et al. [6], where the definition of the 2D Morse-Smale was extended to triangulated domains resulting in the *Quasi-Morse-Smale* complex. Other methods that develop this notion for 3D are available in the literature [11, 17–19]. In recent years, many algorithms based on Forman's [13] discrete Morse theory have become popular [14, 21, 34, 37, 38]. A primary reason for this is the combinatorial robustness of these algorithms, which greatly simplifies implementation effort. However, to the best of our knowledge, source code implementation of these methods are only available for Sousbie et al. [40]. A crucial aspect in Morse-Smale complex based analysis is in its topological simplification and subsequent analysis. Edelsbrunner et al. describe the notion of topological persistence [10] and its application to Morse-Smale complex simplification in 2D [12]. Most of the implementations described in the above literature describe some form of simplification using persistence. However, in many applications [15, 20, 40] successful feature identification requires simplification using other sources of data as well as domain specific criteria. In these cases the effort needed to apply the analysis is dependent on the ease with which one interfaces with the Morse-Smale complex data-structure. More generally, source code implementation for Reeb graphs [9] contour trees [7] and persistent homology [2, 3] have become available over the years. These packages are most readily usable as standalone packages, though most offer access to their internal data structures only in their native programming language. Python [41], being particularly suited for high level scripting operations, is very often available as the de-facto scripting interface in a many large software tools. A primary reason for this is that Python is a mature interpreted language which allows for easy runtime loading/unloading of modules. A few examples of tools that offer Python interfaces include Pymol [35], VMD [26], Chimera [33], ParaView [25], VTK [36], Blender [4].

Contributions In this paper, we describe a Python-scriptable Morse-Smale complex computation and analysis package. The package consists of two modules, named *pyms3d* and *pymstri*, which contain implementations for 3D structured grids and 2D triangle meshes. The modules implement efficient algorithms for Morse-Smale complex computation [34, 37, 38] and hierarchical feature analysis [6, 22]. Furthermore, the implementations leverage the OpenCL and OpenMP frameworks for parallel computation. Access to the combinatorial structure and geometric elements of the Morse-Smale complex are granted through a Python interface. We demonstrate the usefulness of the software via a case study. We interactively segment the Hurricane Isabel Dataset [42] to highlight the interactivity and the ease of use. The *pyms3d* module is used to develop a programmable filter which is loadable in ParaView [25]. The extracted features may be changed and updated from within the ParaView runtime environment to drive interactive feature based visual analysis. This case study is demonstrated via a video hosted at https://youtu.be/UX1q9gI2DEk.

2 Background

In this section, we introduce the necessary background relevant to the definition of the Morse-Smale (MS) complex.

Morse Theory and the MS Complex Morse theory studies critical points of smooth scalar functions defined on manifolds [29]. Given a smooth scalar function $f : \mathbb{R}^n \to \mathbb{R}$, its *critical points* are points where the *gradient*, the vector of first order partial derivatives $\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \frac{\partial f}{\partial x_2}(x), \ldots, \frac{\partial f}{\partial x_n}(x)\right)$, is identical to zero. A critical point is non-degenerate if the *Hessian*, the matrix of second order partial derivatives, is non-singular. The function f is said to be *Morse* if all its critical points are non-degenerate. The *index* of a critical point is the number of negative eigenvalues of the Hessian matrix. An *integral line* is a maximal curve in $\mathbb{R}^n$ whose tangent at every point is collinear with the gradient of f at that point. The limit points of integral lines, $t \to \pm\infty$, are the critical points of f.

The set of all integral lines that share a common source (destination) p, is called the *ascending manifold* (*descending manifold*) of p. The *Morse-Smale (MS) complex* is a partition of the domain into cells formed by the collection of integral lines that share a common source and a common destination. The combinatorial structure is a graph, where the nodes are critical points and edges are arcs between them if there is an integral line that connects them and their indices differ by one.

Discrete Morse Theory Forman [13] introduced discrete Morse theory to study the topology of cell complexes. A d-cell α^d is a topological space homeomorphic to a d-ball $B^d = \{x \in \mathbb{R}^d : |x| \leq 1\}$. Lower dimensional d-cells include vertices, edges, triangles/quads, and tetrahera/cubes. A *cell complex* K is a collection of cells where the set of cells incident on the boundary of a cell are also in K, and two cells intersect only along a single common boundary cell. Examples of cell complexes include 2D triangle meshes and 3D cubical complexes (see Fig. 2a, b). A function $f : K \to \mathbb{R}$ is said to be a *discrete Morse function* if for all d-cells α in K, there exists no more than one incident higher dimensional cell β so that $f(\alpha) \leq f(\beta)$, and no more than one incident lower dimensional cell γ exists so that $f(\gamma) \leq f(\alpha)$. A pairing between two incident d-$d+1$ cells, α-β, so that $f(\alpha) \geq f(\beta)$ is called a *discrete vector*. A *V-path* is a sequence of unique d-$d+1$ discrete vectors, $\alpha_0^d, \beta_0^{d+1}, \alpha_1^d, \beta_1^{d+1}, \ldots, \alpha_r^d, \beta_r^{d+1}, \alpha_{r+1}^d$, so that every $d+1$ cell is incident on the next d-cell. A *discrete gradient field* is a collection of V-paths without any non-trivial loops. Acyclic V-paths correspond to the notion of integral lines of Morse functions. A cell that is not paired is called a *critical cell* and is analogous to the notion of a critical point. Ascending/descending manifolds and the combinatorial structure are similarly defined for discrete Morse functions. Figure 2c shows an example of the combinatorial structure of the Morse-Smale complex defined on a cell complex using discrete Morse theory.

Topological Cancellation and the Hierarchical MS Complex A topological cancellation is a process of removal of a pair of index $i - i + 1$ critical points p-q that are singularly connected in the combinatorial structure [12]. The combinatorial structure is modified so that all other index $i + 1$ critical points connected to p are connected to all other index i critical points connected to q. The ascending (descending) manifold of p (q) is merged with the ascending (descending) manifolds of all other i ($i + 1$) critical points connected to q (p). *Topological persistence* [12] measures the importance of a pair of critical points. Pairs of critical points are canceled in increasing order of its *persistence*. As one iteratively applies the above operations to simplify the MS complex, each application results in a new combinatorial and geometric version of the MS complex. This sequence of MS complex versions is referred to as the hierarchical MS complex, where each version is indexed by its position in the sequence. Selecting appropriate versions for feature analysis is often challenging and thus it is desirable to try multiple versions before selecting one. Figure 1 shows an example of a Morse-Smale complex along with two cancellations applied to it to generate a hierarchical MS complex with three versions in the sequence.

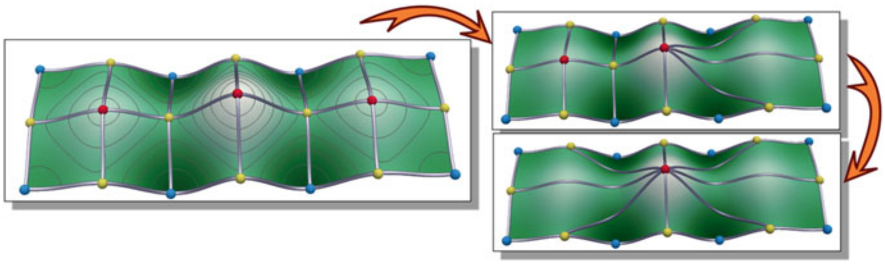


Fig. 1 *(left)* The combinatorial structure of the Morse-Smale complex of a function with three maxima. The critical points are shown as *spheres*, *blue* for minima, *yellow* for saddles, and *red* for maxima. The arcs are shown as gray tubes. *(right)* Two cancellation operations applied to the Morse-Smale complex eliminate two maximum saddle pairs connected to the central maximum. Each cancellation operations re-routes arcs from the maxima connected to the canceled saddle to the saddles connected to the canceled maximum. The two cancellations result in two successive versions of the Morse-Smale. This sequence of versions is referred to as the hierarchical Morse-Smale complex

3 Data Representation and Algorithms

We now describe the data structures and algorithms that are implemented in the software package. We first describe how the data over 2D surfaces and 3D grids are represented. Next, we describe the data structure used for the Morse-Smale complex. Finally, we briefly describe the discrete Morse-Smale complex computation algorithm and the construction of the Hierarchical Morse-Smale complexes.

3.1 Data Representation

Cells of the underlying domain are represented using unique identifiers. This is relevant when querying the geometry of the Morse-Smale complex. The data representation of the Morse-Smale complex is common to both *pymstri* and *pyms3d*. This is relevant for analysis of the combinatorial structure of the Morse-Smale complex.

2D Surfaces in Pymstri In the implementation of pymstri, a triangle mesh representing the surface is stored using the edge-facet data structure [27]. Triangle meshes are assumed to be available as a set of vertices and a set of triangles where each triangle specifies three indices into the list of vertices.

3D Structured Grids The structured 3D grid domain is interpreted as a cubical cell complex whose cells are vertices, edges, quads, and cubes. The cells of the domain are implicitly represented using the Cartesian coordinates of their centroids as identifiers. Each cell is uniquely identified using a tuple with three integers. We scale the coordinates by two so that the interleaving cells, namely edges, faces, and cubes, also obtain integral coordinate values at their centroids. Queries for facets/cofacets are therefore implicitly computed taking into consideration the boundary conditions imposed by the grid. These queries use integer arithmetic instead of floating point arithmetic as a consequence of the scaling described above. The center panel in Fig. 2b shows a simple structured grid interpreted as a cubical complex.

Morse-Smale Complex Representation The Morse-Smale complex is represented as a graph whose nodes represent critical points of the Morse-Smale complex and edges represent arcs. Let M denote the Morse-Smale complex. Each critical point is identified by a unique integer identifier. M stores per-vertex information such as the function and index of the critical point. The adjacencies between critical points are maintained in a pair of lists, one for the ascending and one for the descending adjacencies. In 2D, each list consists of a list critical point identifiers. In 2D, each critical point may be incident upon another via at most two arcs. In 3D, however, there may be arbitrarily more. Hence, each entry in the list comprises of a tuple, where the first value identifies the incidence relation, and the second value

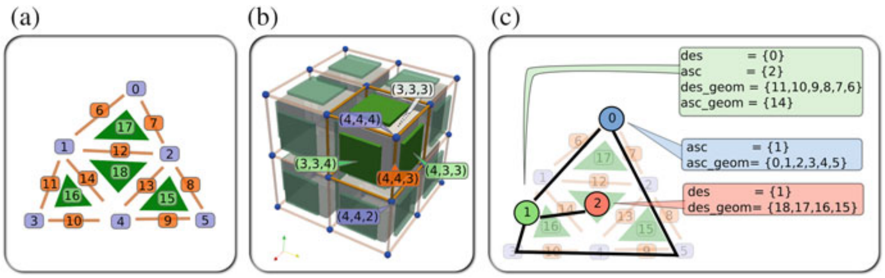


Fig. 2 Data representation. (**a**) In 2D triangle complexes, each cell in the complex is identified by a unique integer ID. (**b**) A 3D structured grid is interpreted as a cubical complex. Each cell in the complex is uniquely identified by the centroid of its Cartesian coordinate scaled by two. (**c**) The Morse-Smale complex is represented as a graph whose nodes are critical points and arcs are edges between them. Each critical point is given a unique integer identifier. For each critical point, the lists *asc* and *des* contain the ID's of the ascending and descending critical points connected to it. Similarly, for each critical point, the lists *asc_geom* and *des_geom* contain the cell identifiers of ascending and descending cells from the underlying domain, [shown in (**a**)]

identifies the multiplicity. The ascending and descending geometry of each critical point is maintained in a pair of lists. Each list consists of a list of cell identifiers which identify cells of the underlying domain. The right panel in Fig. 2c shows a simple Morse-Smale complex representation with the adjacency and geometry data.

3.2 Algorithms

For computing the discrete gradient field, we use the algorithm by Robins et al. for *pymstri* as it results in the fewest spurious critical points. For *pyms3d*, we use the algorithm by Shivashankar et al. [37] as it is implementable in massively parallel architectures using GPUs. Though it results in more spurious critical points, the trade-off is acceptable in terms of the efficiency offered by GPUs. We use the algorithms described by Shivashankar et al. [37] for efficient traversal of the gradient field on the CPU and GPU. A full performance evaluation of the above algorithm for 2D and 3D structured grids is available by Shivashankar et al. [37, 38]. For the hierarchical Morse-Smale complex, we directly implement the cancellation and anti-cancellation operations described by Bremer et al. [6]. Cancellations may be scheduled based on topological persistence [10]. Alternatively, cancellations may be specified explicitly and performed sequentially via the Python interface. To traverse the combinatorial structure of the hierarchical Morse-Smale complex, the list of cancellation pairs are stored and the cancellation / anti-cancellation operations are repeatedly applied using this list to obtain the desired level in the hierarchy. We employ the *merge_dag* [22] data structure for efficient geometry queries from the hierarchical Morse-Smale complex.

4 Design and Implementation

In this section, we briefly discuss design and implementation issues of the above data structures. The above described algorithms are implemented using C++. The code is extensively parallelized to exploit multi-core CPUs when available. This is done using the OpenMP framework, which has the advantage of requiring only compiler directives to concisely indicate shared memory parallel loops and sections. The discrete gradient algorithm as well as gradient field traversals for extrema in *pyms3d* are implemented using the OpenCL framework. The implementation selects the GPU when available. If a GPU is unavailable, it runs the OpenCL code on the CPU.

The C++ implementations are made available as Python modules using the Boost Python framework [1]. Both modules expose a Python class representing the Morse-Smale complex. The implementation makes extensive use of the NumPy [31] module to enable efficient passing of data arrays from C++ runtime objects to the Python environment. This module is easily available in most Python installations and its C++ bindings are made available by the Boost NumPy project [5]. Table 1 shows a subset of methods that are exposed to the Python Interface along with a brief description.

The implementations allow for computation of the Morse-Smale complex, querying for the combinatorial structure, querying for the ascending and descending manifolds of the critical points, generating a hierarchy (either by persistence or by a user defined sequence using the *cancel_pair* call), and querying the combinatorial and geometric structures at different levels of the hierarchy.

We now discuss the functions listed in Table 1. The interface can be broadly classified into three groups, namely computation functions, simplification, and query functions. The first two groups primarily alter the state of the Morse-Smale complex. Therefore, they involve minimal overhead in terms of interfacing with Python as they simply reroute the calls to C++ implementation. The third group generally involves data copying overheads via the Boost NumPy [31] array API. The first group comprises of the computation functions that compute the Morse-Smale complex and collect the geometry associated with its critical points. The calls to *compute_bin*, *compute_arr* and *compute_off*, detailed in Table 1, compute the combinatorial Morse-Smale complex. In many applications, either the combinatorial structure suffices or the geometry is desired after some pre-simplification. The call to *collect_geom* collects the Morse-Smale complex geometry at the current hierarchical version. Optionally, the user may choose to only collect the ascending or descending geometry of critical points with a given index.

The second group of functions may be used to simplify the Morse-Smale complex. The primary function here is the *cancel_pair* routine which cancels a given pair of critical points as long as the cancellation is permissible. Each cancellation results in a new hierarchical version of the Morse-Smale complex. Due

Table 1 A subset of the methods available to an **mscomplex** object created by the modules *pyms3d* and *pymstri*

Method	Brief description
compute_off(f)	Computes the mscomplex of a triangle mesh and scalar function given in a file f
compute_bin(f,s)	Computes the mscomplex of a structured grid. Scalar function is given as 32-bit floating point binary file f in fortran order. Grid size is given in the tuple s
compute_arr(a)	Compute the mscomplex of a structured grid. Scalar function is given as 3d NumPy array a. Internally, data is converted to single precision (32-bit) floating point values
collect_geom([d,dir])	Collects the [($dir =$) Ascending/Descending] geometry of all [d-] critical points in the current hierarchical version
get_hversion(n) / set_hversion(n)	Gets/Sets the hierarchical version of the Morse-Smale complex
cancel_pair(p,q)	Cancels a pair of singularly connected unpaired critical points p and q
simplify_pers(t)	Simplifies the Morse-Smale complex upto persistence threshold t
cps([d])	Returns a list of ids of active critical points [with index d]
des(i) /asc(i)	Returns a list of descending/ascending critical points connected to i
des_geom(i,[n]) / asc_geom(i,[n])	Returns descending/ascending manifold of the critical point i [in the nth hierarchical version]
get_primal_points(), get_dual_points()	Returns the coordinates of 0-dimensional cells of the primal/dual cell complex
cp_func(i), cp_index(i), *etc.*	Returns the per-critical point information such as the function value, pair
get_hversion_pers(t)	Returns hierarchy version number where pairs with persistence t are eliminated
save(f)/load(f)	Save/load all data to/from file f

Optional arguments to the methods are shown indicated in square braces ([. . .])

to the popularity of the persistence hierarchy, the call to *simplify_pers* is provided to generate a persistence hierarchy so that arcs with a specified threshold are simplified. This call may also be supplied with a number of extrema that are desired to be retained. After some simplification, one may obtain the earlier versions of the Morse-Smale complex using *set_hversion*. For instance, one may obtain the Morse-Smale complex prior to two simplifications by using the call *msc.set_hversion(msc. get_hversion()-2)*.

```
import pyms3d, numpy

X,Y,Z     = (500,500,100)
msc       = pyms3d.mscomplex()            # Create the object.
msc.compute_bin("Speed03.bin",(X,Y,Z)) # Compute from bin
                                          file.
msc.simplify_pers(thresh=0.05)            # Simplify.
msc.collect_geom(1,0)                     # Collect the asc geom
                                          # of 0 index cps.
ids = numpy.empty([X*Y*Z],numpy.int32) # Array to hold ids.
for m in msc.cps(0):                      # For each minima m,
  ids[msc.asc_geom(m)] = m                # set id's of vert's
                                          # in asc of m to m.
setattr(pyms3d,"msc",msc)                 # cache the object.

# pass the ids array to ParaView (Code omitted for brevity).
```

Listing 1 Code for segmenting the Isabel dataset using the *pyms3d* module

The third group of functions involve providing data pertaining to the Morse-Smale complex via the NumPy [31] array module. This is designed as a copy only mechanism for two reasons. First, NumPy is a very popular scientific computation API which eases further computations and interfacing with other tools, such as ParaView's Python API (see Listing 1 for example). Second, the memory management is eased by relinquishing the returned array objects to the Python runtime as opposed to maintaining the same in the C++ runtime. The list of critical points in the current hierarchical version, optionally of a given index d, may be obtained as an array of integer identifiers via the call to *cps([d])*. The list of ascending/descending critical points connected to a given critical point i may be obtained by the *asc(i)/des(i)* calls. In 3D, each entry in the returned list is a pair, where the first is the ID and the second is the multiplicity of the connection. In 2D, multiple entries are simply repeated as the multiplicity is at most two. The ascending/descending manifold of a given critical point i may be obtained using the *asc_geom(i)/des_geom(i)* calls. For a given critical point of index d, these methods return an array containing d-cells of gradient pairs that originate/terminate at i. By default, this set is constructed for the current hierarchical version of the Morse-Smale complex object. The ascending/descending manifold may also be obtained at a given hierarchical version n without altering the current hierarchical version by using the *merge_dag* data-structure. Note that non-empty geometry can only be returned if a prior call to *collect_geom* was made at a hierarchical version lower than n. For *pyms3d*, each cell is represented by a tuple of indices. In the case of ascending manifolds of 1-saddles and 2-saddles, the indices index into a list of vertex coordinates, which may be obtained by the call to *get_primal_points*. Analogously, for ascending manifolds of saddles, the indices index into a list of cube coordinates obtained via the call to *get_dual_points*. For minima, as their ascending manifolds form a partition on the set of vertices, the indices index into the list of vertex coordinates. Analogously, for maxima, the indices index into a list of cube coordinate, i.e. the cube centroids. For *pymstri*, an analogous output form is supported. Additionally, the indices discussed above respect the ordering of triangles and vertices given as the input to the computation. Other utility functions to query per-critical point information, such as the index and function value of critical points, as well as save/load the computed data from/to the file-system are also available.

5 Case Study: Interactive Visual Feature Analysis

In this section, we present a case study, where the Morse-Smale complex based analysis is coupled with the ParaView [25] visualization package. ParaView offers a Python programmable interface to its visualization pipeline. We apply the Morse-Smale complex to perform hierarchical segmentation of a simulation of the hurricane Isabel. Hurricane Isabel was a hurricane that struck the coast of Florida, USA, in September 2003. The simulation dataset was made available by Wang et al., for the 2004 IEEE Visalization contest [42]. The simulation is available as 32-bit floating point values on a 500 × 500 × 100 3D structured grid. This dataset is well understood in visualization literature and therefore helps illustrate the ease of feature analysis and visualization using the software. We begin by simply computing the Morse-Smale complex of the wind-speed field. Listing 1 shows code to generate the segmentation data using *pyms3d*. Figure 3 shows the volume segmentation using a persistence threshold of 0.05.

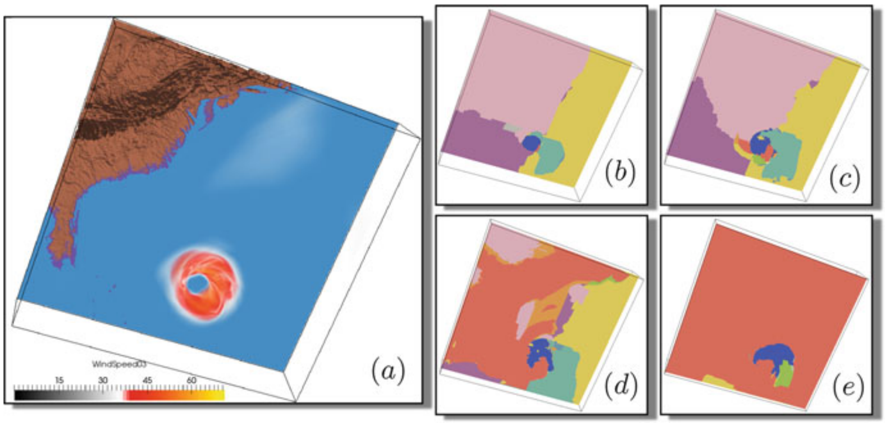


Fig. 3 Segmenting the wind speed field in 3rd time-step of the Isabel simulation. (**a**) A volume visualization of the wind speed field. The eye is distinctly discernible as a low wind speed region enveloped by high wind speed regions. The height field representing the land and sea regions with appropriate colors is shown. (**b**), (**c**), (**d**), and (**e**) Segmentation of the scalar field at four equally spaced z-slices using a persistence threshold of 0.05, generated using Listing 1. The distinctive structure of the eye is retained in the lower z-slices (**b**) and (**c**), which is less discernible in the higher slices (**d**) and (**e**)

```
import pyms3d, numpy

msc = getattr(pyms3d,"msc")    # get the cached msc object.

if not hasattr(pyms3d,"eye"): # if 'eye' is not in cache
  msc.simplify_pers(nmin=2)    # simplify until 2 minima
                               survive
  m1,m2 = msc.cps(0)           # get the 2 minima
  msc.simplifiy_pers(nmin=1)   # simplify the penultimate
                               minimum
  m,    = msc.cps(0)           # get the surviving minimum
  e = (m1 if m == m2 else m2) # the required minimum is the
                               other
  setattr(pyms3d,"eye",e)      # cache the crit. pt. id of eye.

e = getattr(pyms3d,"eye")      # get crit. pt. id of eye

v = msc.get_hversion_pers(thresh=0.018)  # change hier.
                                         version
msc.set_hversion(v)

ag = msc.asc_geom(e)           # id's of verts in e's asc mfold

fns = numpy.fromfile("Speed03.bin",numpy.float32) #read
                                                  scalars

ag_fn   = fns[ag]              # save func. values at eye
fns[:]  = -1                   # set value everywhere to -1
fns[ag] = ag_fn                # set scalar value only at eye.

# pass the fns array to ParaView (Code omitted for brevity).
```

Listing 2 Code for extracting the id of the minimum representing the eye of the hurricane from the Morse-Smale complex object computed and cached in Listing 1. The corresponding ascending manifold region is segmented and the scalar values within this region is volume rendered in Fig. 3

Next, we identify the highest finite-persistent minimum and segment its corresponding ascending manifold. This is done by simplifying using persistence till only two minima remain. Further simplification eliminates the desired minimum. Since the object is cached during runtime, it may be retrieved without re-computation for these operations as shown in Listing 2. Using the minimum representing the eye, the function value restricted to the ascending manifold of the desired minimum is generated.

The above listings are demonstrated in a video hosted at https://youtu.be/UX1q9gI2DEk, where they are plugged into ParaView's programmable filter. In particular, changes to the persistence threshold to visualize the eye at different hierarchical versions is demonstrated. These modifications are made at runtime and the visualization updates occur interactively. The video demonstrates the execution of the above listings on a HP xw8600 workstation with 8 CPU cores, 8GB RAM, and Nvidia 260 GPU that has 895MB VRAM. The time taken to compute the Morse-Smale complex and generate the visualization using Listing 1 is approximately 30 s. The time to update the visualizations using Listing 2 is under 1 second. A more detailed study of the performance of the efficient computation algorithms is presented by Shivashankar et al. [37, 38].

Due to the efficient computation, as well as the interactive visual analysis, Listing 2 was used for multiple time steps. Figure 4 shows the wind-speed restricted to the eye using this simultaneous visualization setup. The transfer function is chosen to highlight the low speeds within the eye.

We observe the performance of a few of the crucial methods over 50 executions of Listing 1. The mean execution time of *compute_bin* with OpenCL on the GPU is 12.99 s (std[1]=0.1 s). The same method deployed on the 8-core CPU has a mean execution time of 52.31 s (std=0.916 s). For both invocations time taken for the call to transition from Python to the C++ runtime had a mean of 2.28×10^{-3} s (std=1.17×10^{-4} s). The transition time for other function calls had similar timings and hence we do not consider it to be a performance issue. The call to *collect_geom* has a mean time of 1.45 s (std=0.082 s) with the GPU deployment and 3.06 (std=0.052 s) with the CPU deployment. The total time taken for all calls to *asc_geom* in Listing 1 had a mean of 0.46 s (std=0.013 s) over 50 executions. The method has three steps. First, the geometry representing the ascending manifold at a given hierarchy is computed using the *merge_dag*. Second, the data is converted from cell identifiers to vertex indices. Third, a NumPy array is allocated, the data is copied to it, and returned. The mean (and variance) for each of the above steps is 0.34 s (std=0.0027), 0.12 s

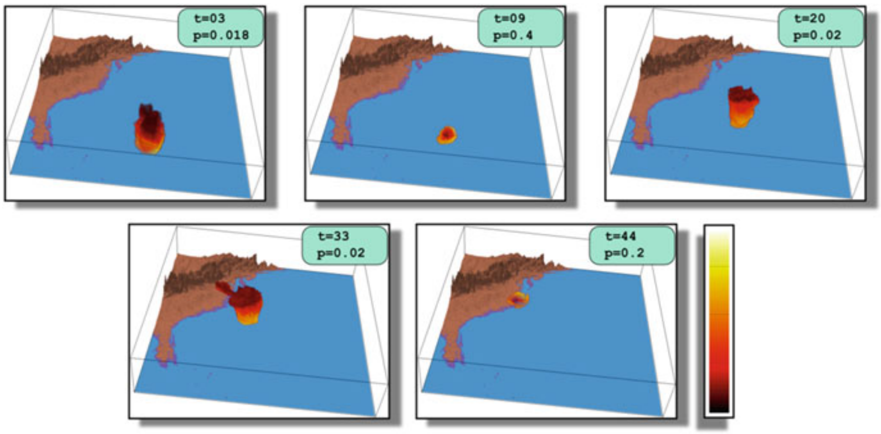


Fig. 4 Visualizations of multiple time-steps of the eye of the Isabel simulation segmented using the Listing 2. Due to the efficient implementation of computation and filtering, the above Listings can be used for multiple time-steps in the same instance of ParaView for interactive visual identification of appropriate persistence thresholds

[1]The standard deviation is abbreviated as *std*.

(std=0.0113), and 0.0013 s (std= 4.65×10^{-5}). From the above timings, we conclude that both the Python call transfer overhead as well as data copy overheads are negligible compared to the method timings. Also, the comparison of the CPU and GPU timings reaffirm the earlier experimental results [37, 38].

We also profile the memory usage of Listing 1 using both the CPU and GPU deployments. We super-sample the dataset using bilinear interpolation to generate a sequence of datasets, where the number of points along each dimension are increased by 10%, 20%, upto 100%. In the CPU deployment on the machine described above, we observe that the maximum physical memory used is 6 GB for a $1000 \times 1000 \times 200$ sized version of the dataset. In the GPU deployment, as there is no virtual memory provision, we observe failures in memory allocation for a dataset sized $550 \times 550 \times 110$. For this dataset, we require a byte buffer of size $1099 \times 1099 \times 219$ (252MB) to store the gradient information of all cells. The GPU device used does not allow individual buffer sizes beyond 224MB, even though the GPU has a 895MB VRAM. A detailed analysis of this experiment is available in the project website.

6 Conclusions

In this paper, we have presented an efficient implementation of Morse-Smale complex based algorithms for visual analysis. We demonstrated its versatility as a computation and a visual analysis tool by plugging into ParaView to generate visualizations of the Hurricane Isabel dataset. This implementation includes state of the art algorithms for computation of Morse-Smale complexes as well as implementations of algorithms for hierarchical analysis.

Acknowledgements This work was partially supported by the Department of Science and Technology, India, under Grant SR/S3/EECE/0086/2012 and by the Robert Bosch Centre for Cyber Physical Systems, Indian Institute of Science.

References

1. Abrahams, D., Grosse-Kunstleve, R.W.: Building hybrid systems with boost. python (2003)
2. Bauer, U., Kerber, M., Reininghaus, J.: Distributed computation of persistent homology. In: Proceedings of Algorithm Engineering and Experiments (ALENEX), pp. 31–38 (2014)
3. Bauer, U., Kerber, M., Reininghaus, J., Wagner, H.: Phat - persistent homology algorithms toolbox. In: Hong, H., Yap, C. (eds.) Mathematical Software - ICMS 2014, Lecture Notes in Computer Science, vol. 8592, pp. 137–143. Springer, Berlin (2014)
4. Blender Online Community: Blender - a 3D modelling and rendering package. Blender Foundation, Blender Institute, Amsterdam (2014)
5. Boost Python interface for NumPy (2016). https://github.com/ndarray/Boost.NumPy
6. Bremer, P.T., Edelsbrunner, H., Hamann, B., Pascucci, V.: A topological hierarchy for functions on triangulated surfaces. IEEE Trans. Vis. Comput. Graph. **10**(4), 385–396 (2004)

7. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. **24**(2), 75–94 (2003)
8. Cazals, F., Chazal, F., Lewiner, T.: Molecular shape analysis based upon the Morse-Smale complex and the Connolly function. In: Proceedings of 19th Annual ACM Symposium on Computational Geometry, pp. 351–360 (2003)
9. Doraiswamy, H., Natarajan, V.: Computing Reeb graphs as a union of contour trees. IEEE Trans. Vis. Comput. Graph. **19**(2), 249–262 (2013)
10. Edelsbrunner, H., Letscher, D., Zomorodian., A.: Topological persistence and simplification. Discrete Comput. Geom. **28**(4), 511–533 (2002)
11. Edelsbrunner, H., Harer, J., Natarajan, V., Pascucci., V.: Morse-Smale complexes for piecewise linear 3-manifolds. In: Proceedings of 19th Annual ACM Symposium on Computational Geometry, pp. 361–370 (2003)
12. Edelsbrunner, H., Harer, J., Zomorodian., A.: Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. Discrete Comput. Geom. **30**(1), 87–107 (2003)
13. Forman, R.: A user's guide to discrete Morse theory. Séminaire Lotharingien de Combinatoire **48** (2002)
14. Günther, D., Reininghaus, J., Wagner, H., Hotz, I.: Memory efficient computation of persistent homology for 3D image data using discrete Morse theory. In Proceedings of Conference on Graphics, Patterns and Images, 24 (SIBGRAPI), pp. 25–32 (2011)
15. Günther, D., Boto, R.A., Contreras-Garcia, J., Piquemal, J.P., Tierny, J.: Characterizing molecular interactions in chemical systems. IEEE Trans. Vis. Comput. Graph. **20**(12), 2476–2485 (2014)
16. Günther, D., Jacobson, A., Reininghaus, J., Seidel, H.P., Sorkine-Hornung, O., Weinkauf, T.: Fast and memory-efficient topological denoising of 2D and 3D scalar fields. IEEE Trans. Vis. Comput. Graph. **20**(12), 2585–2594 (2014)
17. Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P.T., Hamann, B.: Topology-based simplification for feature extraction from 3D scalar fields. In: Proceedings of IEEE Visualization, pp. 535–542 (2005)
18. Gyulassy, A., Natarajan, V., Pascucci, V., Bremer, P.T., Hamann, B.: A topological approach to simplification of three-dimensional scalar fields. IEEE Trans. Vis. Comput. Graph. **12**(4), 474–484 (2006)
19. Gyulassy, A., Natarajan, V., Pascucci, V., Hamann, B.: Efficient computation of Morse-Smale complexes for three-dimensional scalar functions. IEEE Trans. Vis. Comput. Graph. **13**(6), 1440–1447 (2007)
20. Gyulassy, A., Duchaineau, M., Natarajan, V., Pascucci, V., Bringa, E., Higginbotham, A., Hamann, B.: Topologically clean distance fields. IEEE Trans. Vis. Comput. Graph. **13**(6), 1432–1439 (2007)
21. Gyulassy, A., Bremer, P.T., Pascucci, V., Hamann, B.: A practical approach to Morse-Smale complex computation: Scalability and generality. IEEE Trans. Vis. Comput. Graph. **14**(6), 1619–1626 (2008)
22. Gyulassy, A., Kotava, N., Kim, M., Hansen, C., Hagen, H., Bremer, P.T.: Direct feature visualization using Morse-Smale complexes. IEEE Trans. Vis. Comput. Graph. **18**(9), 1549–1562 (2012)
23. Gyulassy, A., Bremer, P., Pascucci, V.: Computing Morse-Smale complexes with accurate geometry. IEEE Trans. Vis. Comput. Graph. **18**(12), 2014–2022 (2012)
24. Gyulassy, A., Günther, D., Levine, J.A., Tierny, J., Pascucci, V.: Conforming Morse-Smale complexes. IEEE Trans. Vis. Comput. Graph. **20**(12), 2595–2603 (2014)
25. Henderson, A.: ParaView Guide, A Parallel Visualization Application. Kitware Inc. (2007)
26. Humphrey, W., Dalke, A., Schulten, K.: VMD – visual molecular dynamics. J. Mol. Graph. **14**, 33–38 (1996)
27. Kettner, L.: CGAL HalfEdge Data-Structure: User (2016). http://doc.cgal.org/latest/HalfedgeDS/index.html

28. Laney, D., Bremer, P.T., Mascarenhas, A., Miller, P., Pascucci, V.: Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. IEEE Trans. Vis. Comput. Graph. **12**(5), 1053–1060 (2006)
29. Matsumoto, Y.: An Introduction to Morse Theory, vol. 208. American Mathematical Society, Providence, RI (2002). Translated from Japanese by K. Hudson and M. Saito
30. Natarajan, V., Wang, Y., Bremer, P.T., Pascucci, V., Hamann, B.: Segmenting molecular surfaces. Comput. Aided Geom. Des. **23**(6), 495–409 (2006)
31. NumPy & SciPy: Scientific computing in python (2016). http://www.numpy.org/
32. Peterka, T., Ross, R., Gyulassy, A., Pascucci, V., Kendall, W., Shen, H.W., Lee, T.Y., Chaudhuri, A.: Scalable parallel building blocks for custom data analysis. In: Proceedings of IEEE Symposium on Large-Scale Data Analysis and Visualization (LDAV), pp. 105–112. IEEE, New York (2011)
33. Pettersen, E.F., et al.: UCSF Chimera – a visualization system for exploratory research and analysis. J. Comput. Chem. **25**(13), 1605–1612 (2004)
34. Robins, V., Wood, P.J., Sheppard, A.P.: Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. IEEE Trans. Pattern Anal. Mach. Intell. **33**(9), 1646–1658 (2011)
35. Schrödinger, LLC: The PyMOL molecular graphics system, version 1.3r1 (2010)
36. Schroeder, W., Lorenson, B., Martin, K.: The Visualization Toolkit, 3rd edition. Kitware (2003)
37. Shivashankar, N., Natarajan, V.: Parallel computation of 3D Morse-Smale complexes. Comput. Graphics Forum **31**(3pt1), 965–974 (2012)
38. Shivashankar, N., Senthilnathan, M., Natarajan, V.: Parallel computation of 2D Morse-Smale complexes. IEEE Trans. Vis. Comput. Graph. **18**(10), 1757–1770 (2012)
39. Smale, S.: On gradient dynamical systems. Ann. Math. **74**(1), 199–206 (1961)
40. Sousbie, T.: The persistent cosmic web and its filamentary structure - I. Theory and implementation. Mon. Not. R. Astron. Soc. **414**(1), 350–383 (2011)
41. van Rossum, G.: Python tutorial. Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam (1995)
42. Wang, W., Bruyere, C., Kuo, B.: Competition data set and description in 2004 IEEE visualization design contest. http://vis.computer.org/vis2004contest/data.html (2004)

Notes on the Distributed Computation of Merge Trees on *CW*-Complexes

Aaditya G. Landge, Peer-Timo Bremer, Attila Gyulassy, and Valerio Pascucci

Abstract Merge trees are topological structures that record changes in super-level set topology of a scalar function. They encapsulate a wide range of threshold based features which can be extracted for analysis and visualization. Several distributed and parallel algorithms for computing merge trees have been proposed in the past, but they are restricted to simplicial complexes or regular grids. In this paper, we present an algorithm for the distributed computation of merge trees on *CW*-complexes. The conditions on the *CW*-complex required for the computation of the merge tree are discussed alongside a proof of correctness.

1 Introduction

Analysis and visualization are crucial components in gaining scientific insight from scientific simulations. In this regard, topological techniques have been successful at extracting features of interest from scientific datasets [2, 15]. Topological structures, such as merge trees, are combinatorial in nature and encode level-set based features of a scalar function. They enable threshold-based feature extraction and can be represented compactly, making them suitable for post process exploratory analysis. The continuous increase in computational resources available to scientists performing simulations of complex scientific phenomenon is leading to a corresponding increase in the size and complexity of data being generated. Concurrently, compute architectures are gradually moving towards multi-core and large scale distributed environments. In this scenario, its important to design and develop parallel topological analysis algorithms and techniques that can harness the parallelism provided by these massively parallel resources.

A.G. Landge (✉) • A. Gyulassy • V. Pascucci
Scientific Computing and Imaging (SCI) Institute, University of Utah, Salt Lake City, UT, USA
e-mail: aaditya@sci.utah.edu; jediati@sci.utah.edu; pascucci@sci.utah.edu

P.-T. Bremer
Lawrence Livermore National Laboratory, Livermore, CA, USA
e-mail: bremer6@llnl.gov

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*,
Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_20

Previously, several efficient serial [3, 5], and streaming algorithms [2, 12] have been presented for topological constructs. The first parallel algorithm for computing contour trees has been proposed by Pascucci et al. [11]. Subsequently, [6, 7, 9, 10] introduced techniques to compute contour or merge trees more efficiently at scale. All the above algorithms have been developed for simplicial complexes or rectilinear 2 or 3-d grids. But there are several scientific phenomenon [4, 13] that are modelled using meshes like structured/unstructured curvilinear meshes, finite element zoo meshes [14], adaptive mesh refinement(AMR) meshes [1], etc. For these types of meshes, its non-trivial and potentially costly to convert a mesh into a simplicial complex. However, the majority of such meshes can be represent as regular CW-complexes. Hence, there is a strong motivation to extend the topological analysis algorithms to handle CW-complexes. In this work, we present a distributed algorithm for computing the merge tree on a regular CW-complex. Furthermore, we present a proof of correctness for this approach, and apply our result to validate the previous algorithms.

2 Background

We briefly review some basic concepts from algebraic topology, and refer the reader to Massey [8] for further reading. Let there be a space $\mathbb{M} \subset \mathbb{R}^d$, which is represented using a d-dimensional, finite, regular CW-complex , $\mathcal{K}^d$, such that a k-cell is an open k-ball, $0 \leq k \leq d$. In computational science, a continuous Morse function $F : \mathbb{M} \to \mathbb{R}$, is typically discretized by assigning values to 0-cells(vertices) and interpolating the function within every k-cell to obtain a function $f : \mathcal{K} \to \mathbb{R}$, such that each vertex is associated with a distinct function value and the interpolation scheme ensures f is C^0 on $\mathcal{K}^d$ with simple critical points. For example, in the cases when $\mathcal{K}$ is a simplicial complex or a $3d$-rectilinear grid, the interpolation schemes that could be used are linear and trilinear respectively.

Let there be a k-dimensional cell, $\alpha^k \in \mathcal{K}$, $k \leq d$. Then the closure of α, denoted as $\overline{\alpha}$, is the cell α and the limit points of α. Thus, the *boundary* of α, denoted as $\partial\alpha$, is given by $\partial\alpha = \overline{\alpha} \backslash \alpha$. If another cell, $\beta^m \in \mathcal{K}, k < m \leq d$ such that $\alpha \subset \overline{\beta}$, then α^k is the *face* of β^m, denoted as $\alpha \ll \beta$.

Definition 1 The **level set**, l_c, of f at a value $c \in \mathbb{R}$ is the set of points in the domain of f such that $l_c = f^{-1}(c)$. A connected component of the level set is called a **contour**.

Definition 2 The **super-level set**, L_c, of f is the set of points in the domain of f with value in f greater than $c \in \mathbb{R}$ and is given as $L_c = f^{-1}[c, \infty)$.

Definition 3 Given a super-level set, L_c, of f having n connected components denoted as $\{C_1, C_2, \ldots, C_n\}$, then $L_c = \cup C_i$ for $i = 1 \ldots n$ and $C_i \cap C_j = \emptyset$ $(i \neq j)$. We define an equivalence relation '$\sim$' on $\mathcal{K}$ such that two points $x, y \in \mathcal{K}$ are related if $f(x) = f(y) = c$ and $x, y \in C_i$ i.e. belong to the same level-set and the same super-level component of L_c. Then the quotient space, $\mathcal{K}/\sim$, is the **merge tree** of f on $\mathcal{K}$, denoted as $MT(\mathcal{K})$. The many-to-one map, $\phi : \mathcal{K} \to \mathcal{K}/\sim$, maps points from $\mathcal{K}$ onto a point in the merge tree.

Intuitively, the merge tree encodes the evolution of the frontiers of connected components of the super-level set of f on $\mathcal{K}$. The merge tree is composed of *nodes* and *arcs*. The nodes represent the critical points that create, merge, or destroy super-level set components which are the *maxima* - leaves in the tree, *saddles* - interior nodes and the *global minimum* - the root of the tree respectively. Sometimes, the merge tree is augmented with valence-2 nodes, which are non-critical nodes that do not correspond to any change in the super-level set topology. We call these nodes *regular* nodes. In this paper, we assume without loss of generality that $\mathcal{K}$ is simply-connected. In the case when $\mathcal{K}$ is not connected, we would obtain a forest of merge trees where each tree corresponds to a connected component of the domain.

3 Computing Merge Trees on CW-Complexes

Our technique is based on the divide and conquer strategy where a merge tree is computed for each partition of the domain. These trees are then joined to form the merge tree of the domain. In this section, we first describe the domain decomposition used by the divide and conquer strategy. The strategy is then expressed as a recursive algorithm followed by the necessary modifications to adapt it to a distributed setting.

3.1 Domain Decomposition

In order to apply a divide and conquer strategy we partition the domain $\mathcal{K}$ into a finite number of *patches*.

Definition 4 A **patch**, P, is a d-dimensional sub-complex of $\mathcal{K}$, where P is composed of a set of d-cells along with their boundaries, i.e. let $P \subseteq \mathcal{K}$ such that if $\alpha^d \in P$, then $\beta \in P$ iff $\beta \subset \overline{\alpha^d}$.

Definition 5 The **patch-boundary**, ∂P, of P is the intersection of P with the closure of $\mathcal{K} \backslash P$. Thus $\partial P = \overline{(\mathcal{K} \backslash P)} \cap P$.

Definition 6 The **domain-decomposition** of $\mathcal{K}$ is given as a union of finite number of patches given as $\mathcal{K} = \cup P_i, 0 < i \leq n$ such that for any two patches $P_i, P_j \in \mathcal{K}$, $(P_i \backslash \partial P_i) \cap (P_j \backslash \partial P_j) = \emptyset$, $i \neq j$.

Definition 7 Let there be patches, $P, Q \in \mathcal{K}$. Then P and Q are **neighbors** if $P \cap Q \neq \emptyset$. The **boundary components**, $\partial_i P$, of P are the intersections of P with each of its neighbors, Q_i. Thus, the boundary components of P are $\partial_i P = P \cap Q_i$.

We denote the set of all $\partial_i P$ as $\widehat{\partial}P$. Since P and Q_i are neighbors, each $P \cap Q_i$ is also a boundary component of each neighbor Q_i.

Definition 8 A **patch hierarchy** consists of *levels*, $h = 0, \ldots, h_{max}$, where $h = 0$ is the finest level and $h = h_{max}$ is the coarsest level. Each *level* is a complex of patches such that patch at level h, denoted as P^h is the union of patches at level $h - 1$. Thus, $P^h = \cup P_i^{h-1}, 0 < i \leq n$. At the finest level, $h = 0$, a patch, P^0 is a d-dimensional cell, $\alpha^d \in \mathcal{K}^d$ along with its *boundary*, $\partial\alpha^d$. Thus, $P^0 = \alpha^d \cup \partial\alpha^d = \overline{\alpha^d}$.

One may consider the hierarchy as a tree where each node is a patch. The leaves are patches composed of a single d-dimensional cell and the interior nodes are patches composed of the union of the children. We do not enforce any restriction on the neighborhoods of the children.

3.2 Joining Merge Trees

Given two neighboring patches, $P, Q \in \mathcal{K}$, we can compute their respective merge trees $MT(P)$ and $MT(Q)$ and glue them in a specific way to obtain the merge tree of the union, $P \cup Q$. We call this the *join* operation. But to perform the join, we have to preserve the connectivity of the super-level set components that expand into the neighboring patch. This information is provided by points on the shared boundary $P \cap Q$ and one can achieve the join by adding all these points into the merge trees of the patches as noncritical, *regular* nodes. These can then be used in the join to form the merge tree of $P \cup Q$. But as there are infinitely many points on the boundary this approach is not feasible. Instead, we can restrict the number of noncritical points from the boundary by only adding the *boundary maxima* as regular nodes to the merge trees.

Definition 9 The maxima of f, when f is restricted to every boundary component of a patch, P, are known as the **boundary maxima** of P.

Once we have the merge tree along with the boundary maxima, we can now *join* trees from neighboring patches along these nodes. Before we look at the details of the join, let us define the inputs to this operator.

Definition 10 Let $P \in \mathcal{K}$ be a patch and let $\widehat{\partial}S$ be a set of boundary components. The **augmented merge tree**, $AMT(P, \widehat{\partial}S)$, is the merge tree of f restricted to the patch, P, augmented with the boundary maxima of all boundary components, $\widehat{\partial}S$.

The *JoinMT*() routine described in Algorithm 1 performs this operation on a set of merge trees. It assembles the resulting tree from the arcs and nodes of input trees by introducing each arc, along with its end nodes, in a descending order based on the function value of the lower node. Each time an arc, (u, v), bounded by nodes u and $v, f(u) > f(v)$, is introduced, it gives rise to one of the following cases in the tree being constructed, T:

1. $u, v \notin T$ then arc (u, v) does not attach to any of the existing arcs but gets added as a disjoint arc in T—this represents the creation of a new super-level set component
2. $u \in T,\ v \notin T$ and u has no descendants, then we attach (u, v) to u as a descendant—this represents the growth of an existing super-level set component by addition of the region represented by (u, v)
3. $u \in T,\ v \notin T$ and u has descendants with the lowest descendant being u', then we attach v as a descendant of u' by introducing an arc (u', v)—this means that the super-level set component containing u has already grown till u', so grow it further till v.
4. $v \in T,\ u \notin T$, then we attach (u, v) to v making v a saddle—this represents creation of a super-level set component that merges with another super-level set component at v. Note that since we are adding arcs in the descending order based on the function value of the lower node, v will not have any descendants present in T.
5. $u, v \in T$ and v is not a descendant of u, then we add the arc between the lowest descendant of u, given by u', and v—this represents merging of super-level components
6. $u, v \in T$ and v is a descendant of u, then we discard the arc (u, v)—this mean that the region of the super-level set component represented by (u, v) is already present in the domain.

The above discussed process can be achieved by a union-find like traversal of the sorted nodes as seen in Algorithm 1. The sorting of the nodes is performed in linear time as the input AMTs would have their nodes in sorted order. The $Find(u, AMT)$ routine returns the lowest descendant of u in AMT and adds u if it is not present in AMT. This is implemented as a union-find data structure and has amortized constant running time. The number of edges for a given node is constant and hence the Algorithm 1 has a linear run time complexity given by the number of nodes in the input *AMT*s (Fig. 1).

Algorithm 1: Join(AMT(...))

```
Data: Array of AMTs for individual patches
Result: AMT of the union of patches
AMT = [];
Nodes[] = Sorted list of the union of vertices from input AMTs in decreasing order of
function value;
for All nodes $v$ in Nodes do
    for Arcs $(u, v)$ in input AMTs do
        $u' = Find(u, AMT)$;        // If $u$ not in AMT, add and return $u$
        $v' = Find(v, AMT)$;        // If $v$ not in AMT, add and return $v$
        if $u' \neq v'$ then
            AddArc($u'$, $v'$);
return AMT;
```

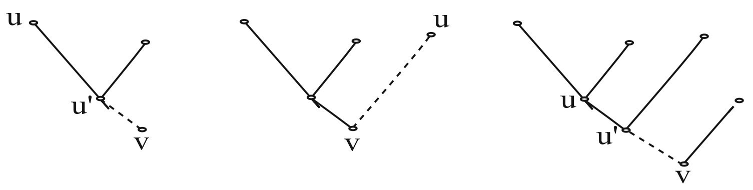


Fig. 1 Examples for case 3 (*left*), case 4 (*middle*), and case 5 (*right*) from the above description

Definition 11 Given two augmented merge trees, $AMT(P, \widehat{\partial P})$ and $AMT(Q, \widehat{\partial Q})$, the **join** operator, $(+)$, joins them along the boundary maxima of $P \cap Q$ to form $AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$.

Theorem 1 $AMT(P, \widehat{\partial P}) + AMT(Q, \widehat{\partial Q}) = AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$.

Proof From Definition 3 of a merge tree, a point $x \in P \cup Q$ is mapped to a point on $AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$ under the map ϕ. If x lies on the arc (u, v), where $f(u) > f(v)$, we say that x has the label u. We now have to show that the join operator on $AMT(P, \widehat{\partial P})$ and $AMT(Q, \widehat{\partial Q})$ produces the same set of labels for points in $P \cup Q$ as produced by $AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$. If the points have the correct labels, we have the correct connectivity of arcs under the join operation.

Let us assume that the correct label for a point $x \in P \cup Q$ is u. Then x lies on an arc $(u, v) \in AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$. Now, there are two cases,

1. $x, u \in P$. In this case, u must be the label of x in $AMT(P, \widehat{\partial P})$. Let us assume that the join operation assigns a label w to x. This is possible only if $w \in Q \backslash P$. Now, if w is to be the label of x, we should have $f(u) > f(w) > f(x)$ and they must lie on the same super-level set component. This implies that the u and w must have a common ancestor. In this case, the join operation ensures that u, w and x lie on the same path to the root. Thus, w should be the label of x, but that contradicts our assumption that u is the correct label, hence, our assumption that the join operation assigns w as the label is false.
2. $x \in P$ and $u \in Q \backslash P$. Then before the join lets assume that x had a label w in $AMT(P, \widehat{\partial P})$. Now, if u is the label of x, then $f(w) > f(u) > f(x)$ and u and x must be on the same super-level set component. But as x has a label w in $AMT(P, \widehat{\partial P})$ implies that x and w are also on the same super-level set component. This implies that w and u have a common ancestor. This is possible only if all them lie on the same path from the ancestor to the root. If u, w are on the same path the algorithm will assign the label u to x since $f(w) > f(u)$ thereby assigning the correct label.

Thus, the join operator always maintains the correct labels and hence generates the correct $AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$. □

The resulting tree from the join contains valence two nodes from $P \cap Q$ that are no longer required. On the other hand, we need to retain the boundary maxima of the boundary components of the union of patches, $P \cup Q$. The boundary maxima of the components of $\partial(P \cup Q)$ are already present in the joined tree. The following lemma proves this claim.

Lemma 1 *Given two neighboring patches $P, Q \in \mathcal{K}$ and their augmented merge trees $AMT(P, \widehat{\partial}P)$ and $AMT(Q, \widehat{\partial}Q)$, then the join of these trees contains the boundary maxima of $\widehat{\partial}(P \cup Q)$.*

Proof $AMT(P, \widehat{\partial}P)$ and $AMT(Q, \widehat{\partial}Q)$ contain the boundary maxima of $\widehat{\partial}P$ and $\widehat{\partial}Q$ respectively and the join operation does not remove any nodes from the participating trees while creating the resulting tree. Thus, $AMT(P, \widehat{\partial}P) + AMT(Q, \widehat{\partial}Q) = AMT(P \cup Q, \widehat{\partial}P \cup \widehat{\partial}Q)$ contains the boundary maxima of $\widehat{\partial}(P \cup Q)$. □

3.3 Obtaining the Boundary Maxima and Pruning the Merge Tree

Definition 12 The $\mathcal{B}_{max}(\widehat{\partial}P)$ operator takes a patch and returns the boundary maxima for each boundary component, $\widehat{\partial}P$, of P.

The function *GetBoundaryMax*(P, h) returns the list of boundary maxima for the patch P at level h. This call takes help of the *BuildMT*() routine that generates a merge tree for a patch. We shall define this routine in detail later in this section. As the merge tree by definition preserves all the maxima of a function on a given domain, we obtain the boundary maxima by computing the merge tree of each boundary component of P and extract the maxima from the respective trees. Since, we do not know the function interpolation scheme on $\mathcal{K}$, we take help of an oracle as defined below to obtain the merge tree of a cell in the *CW*-complex.

Definition 13 Let $\alpha^k \in \mathcal{K}^d$, $0 \leq k \leq d$, be a k-dimensional cell. The **oracle**, given as *OracleMT*(f, α), returns the merge tree, denoted as $MT(\alpha)$ of the cell and its boundary i.e. $\alpha \cup \partial\alpha$.

A similar approach of using an oracle was taken in [11], but the authors did not include the boundary maxima of the cell in their computation which can result in an incorrect join.

As we have seen in Theorem 1 that the boundary maxima are required in order to perform the correct join operation so we must include them in the merge tree returned by the oracle. Since, $\mathcal{K}$ is a regular *CW*-complex, the boundary components, $\widehat{\partial}\alpha$, are $(d-k)$-dimension sub-complexes where $1 \leq k \leq d$. To obtain the boundary maxima of $\widehat{\partial}\alpha$, we make use of the oracle to give us the merge trees of individual cells in $\widehat{\partial}\alpha$. We can then extract the maxima from these trees.

The *GetMaxFromMT*() is a trivial routine that returns the maxima nodes from a merge tree. Algorithm 2 describes the *GetBoundaryMax*(P, h) function. As long as the *BuildMT*() and the *OracleMT* generate the correct merge tree this routine shall identify the correct boundary maxima.

Algorithm 2: GetBoundaryMax(P, h)

```
Data: Patch, $P$, and level, $h$, in the hierarchy
Result: Array with references to maxima, max[]
for all patch-boundary components $\partial_i P^h \in P$ do
    if $h > h_0$ then
        MT = BuildMT($\partial_i P^h$, $h$);
        max[...] = GetMaxFromMT(MT);          // Returns maxima from MT
    else
        for all cells $\alpha_j$ in $\partial_i P^h$ do
            MT = OracleMT($\alpha_j$);
            max[...] = GetMaxFromMT(MT);      // Returns maxima from MT
return max[...];
```

The max need to be relabeled in $AMT(P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$. The following operator performs this operation.

Definition 14 The **mark boundary max** operator, $\mathcal{M}(AMT(P, \widehat{\partial S}), \mathcal{B}_{max}(\widehat{\partial R}))$, such that $\widehat{\partial R} \subseteq \widehat{\partial S}$, returns the $AMT(P, \widehat{\partial R})$, with the boundary max of all components of $\widehat{\partial R}$ marked as boundary.

The *MarkBoundary*() routine performs the above operation. It first traverses the tree and unmarks all the nodes. It then finds the boundary maxima in the tree and marks them as boundary. This results in a tree that has only the boundary maxima marked as boundary.

Finally, we need to remove the redundant valence two or regular nodes that are not boundary. These are no longer required as they are not critical and do not lie on the boundary. We remove them from tree using the following operator.

Definition 15 The **prune** operator, $\mathcal{P}(AMT)$, removes the regular nodes that are not on the boundary from the AMT.

This is a trivial operation and can be performed by simply traversing the tree and deleting the regular nodes that are not boundary.

3.4 Recursive Computation

Now that the above operations have been defined, the merge tree for the entire domain $\mathcal{K}$ can be built in a recursive fashion. We compute the merge tree for every patch at the finest level in the patch hierarchy, $h = 0$, and *join* them using the join operator, find the boundary maxima of the union of the patches using the $\mathcal{B}_{max}$ operator, mark them using the $\mathcal{M}$ operator and finally prune the tree using the $\mathcal{P}$ operator to form the merge tree of a patch at the next coarser level, $h = 1$. We perform this operation recursively till we have obtained the merge tree of the final level, $h = h_{max}$. The recursive solution is given in the following theorem.

Theorem 2 *Given a patch, $P^h \in \mathcal{K}^d$ at level h, the $AMT(P^h, \widehat{\partial} P^h)$ is given by,*

$$AMT(P^h, \widehat{\partial} P^h) = \mathcal{P}\{\mathcal{M}[(\sum_{i=0}^{n} AMT(P_i^{h-1}, \widehat{\partial} P_i^{h-1})), \mathcal{B}_{max}(\widehat{\partial} P^h)]\}, \textit{ where}$$

$$\sum_{i=0}^{n} AMT(P_i^{h-1}, \widehat{\partial} P_i^{h-1}) = AMT(P_0^{h-1}, \widehat{\partial} P_0^{h-1}) + \cdots + AMT(P_n^{h-1}, \widehat{\partial} P_n^{h-1})$$

i.e. the join of all the merge trees of patches at level, $h-1$, within the patch P^h.

Proof This can be proved using induction. At $h = 0$, the patch, $P^0 = \alpha^d \cup \partial\alpha^d$, is a d-cell and its boundary. We make use of the oracle to obtain $MT(\alpha)$. By using the $\mathcal{B}_{max}$ operator on $\widehat{\partial}\alpha$ we can obtain the boundary maxima, which can be added to $MT(\alpha)$ to give us $AMT(P^0, \widehat{\partial} P^0)$.

At $h = k$, let $P^k = \cup P_i^{k-1}$, $0 \leq i \leq n$. Lets assume that

$$AMT(P^k, \widehat{\partial} P^k) = \mathcal{P}\{\mathcal{M}[(\sum_{i=0}^{n} AMT(P_i^{k-1}, \widehat{\partial} P_i^{k-1})), \mathcal{B}_{max}(\widehat{\partial} P^k)]\} \tag{1}$$

At $h = k + 1$, let $P^{k+1} = \cup P_j^k$, $0 \leq j \leq m$. Now, since we can compute $AMT(P_j^k, \widehat{\partial} P_j^k)$ using Eq. (1) and join them to get,

$$\sum_{j=0}^{n} AMT(P_j^k, \widehat{\partial} P_j^k) = AMT(\cup P_j^k, \cup \widehat{\partial} P_j^k) = AMT(P^{k+1}, \cup \widehat{\partial} P_j^k) \tag{2}$$

From (2), we have obtained the *AMT* of the union of patches, $\cup P_i^h = P^{k+1}$, that contains the boundary maxima of the union of the boundaries, $\cup (\widehat{\partial} P_i^k)$. But we need the boundary maxima of components of the boundary of the union i.e. $\widehat{\partial}(\cup P_j^k) = \widehat{\partial}(P^{k+1})$, which need to be marked in $AMT(P^{k+1}, \cup \widehat{\partial} P_j^k)$. The boundary max is obtained from $\mathcal{B}_{max}(\widehat{\partial} P^k)$. These can then be marked by using the $\mathcal{M}$ operator followed by a prune giving,

$$\mathcal{P}\{\mathcal{M}[AMT(P^k, \cup \widehat{\partial} P_i^{k-1}), \mathcal{B}_{max}(\widehat{\partial} P^k)]\} = AMT(P^{k+1}, \widehat{\partial} P^{k+1}) \tag{3}$$

$$\textit{Thus}, \ AMT(P^{k+1}, \widehat{\partial} P^{k+1}) = \mathcal{P}\{\mathcal{M}[(\sum_{j=0}^{n} AMT(P_j^k, \widehat{\partial} P_j^k)), \mathcal{B}_{max}(\widehat{\partial} P^{k+1})]\} \tag{4}$$

□

The function *BuildMT*(P, h), described in Algorithm 3, where P is a patch or patch-boundary component and h is the level in the hierarchy shows the recursive construction. The merge tree for all patches within a higher level patch are computed and joined. The boundary maxima are computed and marked followed by pruning the tree. This is done recursively. Note that at the base level of the recursion the patch is a single d-cell. The oracle returns the merge tree of the cell but we still need to explicitly add the boundary maxima for the cell. Hence, we make the extra call to *GetBoundaryMax*() and *MarkBoundary*() within the *else* after we have invoked the oracle. At the end of the recursive computation, the *BuildMT*() routine computes the $AMT(\mathcal{K}, \widehat{\partial\mathcal{K}})$. We can easily delete the boundary nodes to obtain the $MT(\mathcal{K})$.

Algorithm 3: BuildMT(P, h)

```
Data: Patch, $P$, and level, $h$, in the hierarchy
Result: $MT(P)$
for all patches $P_i^{h-1} \in P$ do
    if $h > 0$ then
        MT[i] = BuildMT($P_i^{h-1}$, $h-1$);
    else
        MT[i] = OracleMT($P_i^h$);
        max[...] = GetBoundaryMax($P_i^h$, $h$);
        MarkBoundary(MT[i], max[...]);
if $h > 0$ then
    MT = Join(MT[...]);
max[] = GetBoundaryMax($P$, $h$);
MarkBoundary(MT, max[...]);
Prune(MT);
return $MT(P)$;
```

3.5 Distributed Computation

In the above section, we have shown that we can compute the merge tree of a domain $\mathcal{K}$ by decomposing $\mathcal{K}$ into a hierarchy of patch levels and recursively computing the tree on each level. In the distributed scenario, we unroll the recursion so as to perform the computation of every patch on to an individual compute resource. This results into a patch containing multiple d-cells being allocated to every compute resource. The information between patches is exchanged using a message passing interface. The merge tree of $\mathcal{K}$ can be computed by joining the merge trees from the distributed patches in a successive join hierarchy corresponding to the patch hierarchy, until the merge tree of the whole domain, denoted as the *global tree*, is computed.

The computation of the *global tree*, by simply unrolling the recursive algorithm is not an efficient distributed solution as it involves communicating entire intermediate merge trees by every patch incurring heavy communication costs. At the same time, the computation is highly load imbalanced as fewer compute resources are in use as one approaches higher levels in the hierarchy. This technique has been used by Pascucci et al. [11] for computing merge trees of 3d regular grids. A more efficient strategy is used by Morozov and Weber [9], Landge et al. [6] where the global tree is distributed where each patch maintaining only the part of the global tree pertaining to that patch, referred as *local merge tree*. Here we extend the technique from [6] to d-dimensional, finite, regular CW-complexes and present its proof of correctness. This proof can be extended to all other existing parallel merge tree computation approaches like [9, 11].

Definition 16 Let P and S be d-dimensional sub-complexes of $\mathcal{K}$. The **local merge tree**, $LT(\mathcal{K}, P, \widehat{\partial S})$, of f on the domain $\mathcal{K}$, local to the patch P with respect to the boundary of $S \subseteq \mathcal{K}$ is given by:

- the arcs and/or nodes of $MT(\mathcal{K})$ that contain at least one point corresponding to the image of point/s in P under the map ϕ;
- the upper and lower nodes of the above arcs;
- these arcs and/or nodes are augmented with the maxima of f when f is restricted to individual *boundary-components*, $\widehat{\partial S}$, of S.

Given a domain decomposition of $\mathcal{K}$ into patches P_i, our goal is to compute the $LT(\mathcal{K}, P_i, \widehat{\partial \mathcal{K}})$ for each $P_i \in \mathcal{K}$. By Definition 16, $MT(\mathcal{K})$ can be easily obtained once we have the individual $LT(\mathcal{K}, P_i, \widehat{\partial \mathcal{K}})$.

In order to compute the $LT(\mathcal{K}, P_i, \widehat{\partial \mathcal{K}})$, we start by computing the $LT(P_i, P_i, \widehat{\partial P_i})$, for each patch. We compute these using the recursive algorithm from Sect. 3.4 by invoking the $BuildMT(P_i, h = 1)$ for each of the patch. Now instead of joining these trees with neighboring patches, we can modify the algorithm to reduce the communication cost of the distributed computation.

Definition 17 Given an arc with end nodes (u, v) in a merge tree, we say that u is the **parent** of v if $f(u) > f(v)$. Given a patch $P \subseteq S \subseteq R \in \mathcal{K}$ and having $LT(R, P, \widehat{\partial S})$, the **boundary merge tree**, denoted as $BT(R, P, \widehat{\partial S})$, is the set of nodes and arcs that lie on the monotonic descending paths from the parents of the boundary maxima of components of $\widehat{\partial S}$ to the root in $LT(R, P, \widehat{\partial S})$. Thus, $BT(R, P, \widehat{\partial S}) \subseteq LT(S, P, \widehat{\partial S})$.

Theorem 3 *Given patches $P, Q \in \mathcal{K}$ such that $P \cap Q \neq \emptyset$. The nodes and arcs of $LT(P, P, \widehat{\partial P})$ that are not part of $BT(P, P, \widehat{\partial P})$ remain unchanged in*

$LT(P \cup Q, P \cup Q, \widehat{\partial}(P \cup Q))$.

Proof We refer the reader to [9] for the proof. As the components of the super-level sets that reside entirely in $P \backslash Q$ and are not connected to the boundary, they cannot be affected by the join operation. □

The following operator extracts the boundary merge tree from a given merge tree augmented with boundary maxima of boundary components.

Definition 18 Given a local merge tree, $LT(P, P, \widehat{\partial S})$ or a boundary merge tree, $BT(P, P, \widehat{\partial S})$, the **extract boundary** operator, $\mathbb{E}$, with respect to $\widehat{\partial R}$, extracts the boundary merge tree, $BT(P, P, \widehat{\partial R})$, where $\widehat{\partial R} \subseteq \widehat{\partial S}$ from $LT(P, P, \widehat{\partial S})$ or $BT(P, P, \widehat{\partial S})$. Thus, $\mathbb{E}(LT(P, P, \widehat{\partial S}), \widehat{\partial R}) = BT(P, P, \widehat{\partial R})$.

The boundary merge trees can be easily extracted by traversing the tree from the leaves, identifying the boundary maxima and selecting the nodes and arcs from the parents of these maxima till the root of the tree.

3.5.1 Join Hierarchy of Boundary Trees

As the nodes and arcs that lie in the interior of a patch do not change after a join, this implies that only the *BT* is affected by the join. We exploit this property by joining only the boundary trees of patches to form boundary trees of patches at the next level. These in turn are joined successively creating a hierarchy of join stages.

Theorem 4 *Given the boundary merge trees of two patches P and Q as $BT(P, P, \widehat{\partial P})$ and $BT(Q, Q, \widehat{\partial Q})$ then,*

$$BT(P, P, \widehat{\partial P}) + BT(Q, Q, \widehat{\partial Q}) = BT(P \cup Q, P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$$

Proof Let $R_P = LT(P, P, \widehat{\partial P}) \setminus BT(P, P, \widehat{\partial P})$,
$R_Q = LT(Q, Q, \widehat{\partial Q}) \setminus BT(Q, Q, \widehat{\partial Q})$,
$R = LT(P \cup Q, P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q}) \setminus BT(P \cup Q, P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$

From Theorem 3,

R_P and R_Q remain unchanged in the join $LT(P, P, \widehat{\partial P}) + LT(Q, Q, \widehat{\partial Q})$. Thus, $LT(P, P, \widehat{\partial P}) + LT(Q, Q, \widehat{\partial Q}) = R_P \cup R_Q \cup (BT(P, P, \widehat{\partial P}) + BT(Q, Q, \widehat{\partial Q}))$

Also, $LT(P, P, \widehat{\partial P}) + LT(Q, Q, \widehat{\partial Q}) = LT(P \cup Q, P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$

The boundary maxima in the above join remain unchanged as the join operation does not give rise to new maxima or does not alter the existing boundary maxima. Thus, $BT(P \cup Q, P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$ contains all boundary maxima from $\widehat{\partial P}$ and $\widehat{\partial Q}$. This implies, $R = R_P \cup R_Q$. Hence, $BT(P, P, \widehat{\partial P}) + BT(Q, Q, \widehat{\partial Q}) = BT(P \cup Q, P \cup Q, \widehat{\partial P} \cup \widehat{\partial Q})$ □

After every join stage, the resulting tree is used by the patches to update their *LT*s with respect to the grown boundary. We refer to this as the *localization* described in the next section. After this the *BT*s are pruned and the *BT* for the next level patches are extracted. These are used by the next stage of the merge. As we consider $\mathcal{K}$ to have no boundary, $BT(\mathcal{K}, \mathcal{K}, \widehat{\partial \mathcal{K}})$ is empty. An example of this process is shown in Fig. 2.

3.5.2 Localization of Merge Trees to Patches

After every join stage of the *BT*s, the resulting boundary trees are joined with corresponding patches to obtain the *LT*s. This way a patch can obtain the connectivity information of its super-level set components that grow into neighboring patches by using the boundary trees of its neighbors.

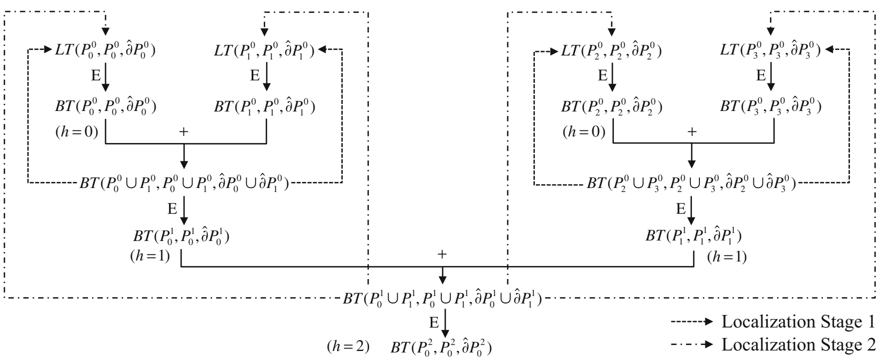

$LT(P_0^0,P_0^0,\widehat{\partial}P_0^0)$
$LT(P_1^0,P_1^0,\widehat{\partial}P_1^0)$
E
$BT(P_0^0,P_0^0,\widehat{\partial}P_0^0)$
$BT(P_1^0,P_1^0,\widehat{\partial}P_1^0)$
$(h=0)$
+
$BT(P_0^0\cup P_1^0,P_0^0\cup P_1^0,\widehat{\partial}P_0^0\cup\widehat{\partial}P_1^0)$
$BT(P_0^1,P_0^1,\widehat{\partial}P_0^1)$
$(h=1)$
$LT(P_2^0,P_2^0,\widehat{\partial}P_2^0)$
$LT(P_3^0,P_3^0,\widehat{\partial}P_3^0)$
$BT(P_2^0,P_2^0,\widehat{\partial}P_2^0)$
$BT(P_3^0,P_3^0,\widehat{\partial}P_3^0)$
$BT(P_2^0\cup P_3^0,P_2^0\cup P_3^0,\widehat{\partial}P_2^0\cup\widehat{\partial}P_3^0)$
$BT(P_1^1,P_1^1,\widehat{\partial}P_1^1)$
$BT(P_0^1\cup P_1^1,P_0^1\cup P_1^1,\widehat{\partial}P_0^1\cup\widehat{\partial}P_1^1)$
$(h=2)$ $BT(P_0^2,P_0^2,\widehat{\partial}P_0^2)$
Localization Stage 1
Localization Stage 2

Fig. 2 An overview of the distributed computation for a domain consisting of four patches at level $h = 0$, two patches at level $h = 1$, and the whole domain at $h = 2$

Theorem 5 *Given neighboring patches* $P, Q \in \mathcal{K}$, *then*
$LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q)) \ \subseteq\ LT(P,\ P,\ \widehat{\partial}P) + BT(Q,\ Q,\ \widehat{\partial}Q)$.

Proof We know from Definition 11 that
$LT(P,\ P,\ \widehat{\partial}P) + LT(Q,\ Q,\ \widehat{\partial}Q) = LT(P \cup Q,\ P \cup Q,\ \widehat{\partial}P \cup \widehat{\partial}Q)$.
We also know from Theorem 3 that the arcs that lie entirely in the interior of Q do not get affected by the join. So, only $BDT(Q,\ Q,\ \widehat{\partial}Q)$ which contains all the connectivity information participates in the join. Thus, $LT(P,\ P,\ \partial P) + BT(Q,\ Q,\ \widehat{\partial}Q) \subseteq LT(P \cup Q,\ P \cup Q,\ \widehat{\partial}P \cup \widehat{\partial}Q)$.
Now, by definition of LT, $LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q)) \subseteq LT(P \cup Q,\ P \cup Q,\ \widehat{\partial}P \cup \widehat{\partial}Q))$ and contains all arcs local to P along with the correct connectivity with the arcs from $BT(Q,\ Q,\ \widehat{\partial}Q)$.
Thus, $LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q)) \subseteq LT(P,\ P,\ \widehat{\partial}P) + BT(Q,\ Q,\ \widehat{\partial}Q)$. □

From the resulting tree of $LT(P,\ P,\ \widehat{\partial}P) + BT(Q,\ Q,\ \widehat{\partial}Q)$, we have to mark the boundary maxima of the components $\widehat{\partial}(P \cup Q)$ and prune the regular nodes. Let the tree obtained after marking the boundary maxima and prune operations be T. But $T \neq LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q))$ as T might contain arcs from Q that do not correspond to any point in P and hence cannot be in $LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q))$. To obtain the $LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q))$ from T we restrict it to P in the following way

- Let X is the set of nodes in T that correspond to the image of points in P under the map ϕ. Let $m \in X$ be the node with the minimum value in X. Let Y be all nodes and arcs that lie on monotonic descending paths from X to m
- We now add arcs, $(u, v), f(u) > f(v)$, along with the end nodes to Y such that $u \notin Y,\ v \in Y$.
- Lastly, we add an arc, $(m, v), f(m) > f(v)$ to Y, if it exists, such that $m \in X$ is the node with the minimum function value in X and $v \notin Y$.

The tree Y is the $LT(P \cup Q,\ P,\ \widehat{\partial}(P \cup Q))$. After every join stage we carry out this localization to obtain $LT(\mathcal{K},\ P,\ \widehat{\partial}\mathcal{K})$.

3.5.3 Time Complexity Analysis of the Distributed Merge Tree Computation

Let us assume that $\mathcal{K}$ has n cells distributed over p processors. Also, let us assume the patch hierarchy to be a k-way hierarchy, where k low level patches combine to form a higher level patch. Assume the oracle of a cell with v vertices on average generates a tree with m nodes in t time. We expect the output trees to be sorted so t is at least $O(v \cdot log(v))$. However, if v is bounded, i.e. for regular grids, t will be constant. Thus, generating the merge trees for all of the CW-cells on each processor is $O(t \cdot n/p)$ and assuming a linear merge of the trees the time to construct the level 0 trees is $O((t+m) \cdot n/p)$. The additional steps to identify boundary maxima, extract the boundary tree, etc. are all linear in the size of the tree and thus do not add to the overall complexity. For known mesh types one could substitute any of the existing algorithms.

The expected size of a boundary tree is proportional to the size of the boundary. Assuming we merge spatially coherent patches, i.e. blocks of a regular grid or groups creates from a mesh partitioning scheme, the boundary trees of level 0 are expected to be of size $O((n/p)^{\frac{2}{3}})$. The merge on level 1 will thus take $O((n/p)^{\frac{2}{3}} \cdot k \cdot \log(k))$ with the additional $\log k$ factor corresponding to the priority queue needed during the merge. The resulting tree will be of size $O((k \cdot n/p)^{\frac{2}{3}})$. Thus the expected total time for all merges of all levels will be

$$O\left(k \cdot \log(k) \left(\frac{n}{p}\right)^{\frac{2}{3}} \cdot \sum_{i=0}^{\log_k p-1} k^{\frac{2i}{3}}\right) = O\left(k \cdot \log(k) \left(\frac{n}{p}\right)^{\frac{2}{3}} \cdot \frac{(p^{\frac{2}{3}}-1)}{(k^{\frac{2}{3}}-1)}\right)$$

$$= O\left(\frac{k \cdot \log(k)(n^{\frac{2}{3}})}{(k^{\frac{2}{3}}-1)}\right).$$

The localization step, which is derived from the join operator, occurs after every join operation in the hierarchy and is linear in the number nodes of the participating local tree and boundary tree. Furthermore, it is performed in parallel by each of the processor and hence does not add to the time complexity.

Complexity-wise the behavior for increasing mesh sizes is therefore dominated by the potentially $O((n/p)^2)$ behavior of the initial local compute, in case m is of order $O(n/p)$. In practice, this cost turns out to be negligible and the behavior is dominated by the number of merges and the increasing size of the boundary trees. Note, that in this respect the size of the boundary tree is a conservative estimate as the global domain boundaries actually do not contribute to the size of the boundary trees.

4 Conclusion

Topological analysis techniques have been extensively used to analyze and visualize data generated by scientific simulations. But the growing compute power and enormity of data has created a need for scalable distributed algorithms. Furthermore, scientific simulations are moving towards using more sophisticated meshes in place of regular grids. For example, adaptive mesh refinement meshes are being adopted by various large scale scientific simulations.

In this paper, we have presented a distributed algorithm for computing merge trees on regular *CW*-complexes, which provides a theoretical foundation for computing merge trees for various types of meshes. We rely on an oracle to provide the merge tree of a single cell within a mesh and state the conditions and requirements from the oracle. Hence, as long as the oracle satisfies those conditions, the merge tree can be computed using the presented algorithm for any type of meshes that are regular *CW*-complexes.

Acknowledgements This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-CONF-681469).

References

1. Berger, M.J., Colella, P.: Local adaptive mesh refinement for shock hydrodynamics. J. Comput. Phys. **82**, 64–84 (1989)
2. Bremer, P., Weber, G.H., Tierny, J., Pascucci, V., Day, M.S., Bell, J.B.: Interactive exploration and analysis of large-scale simulations using topology-based data segmentation. IEEE Trans. Vis. Comput. Graph. **17**(9), 1307–1324 (2011)
3. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. Comput. Geom. **24**(2), 75–94 (2003)
4. Chen, J.H.: Petascale direct numerical simulation of turbulent combustion - fundamental insights towards predictive models. Proc. Combust. Inst. **33**(1), 99–123 (2011)
5. Chiang, Y., Lenz, T., Lu, X., Rote, G.: Simple and optimal output-sensitive construction of contour trees using monotone paths. Comput. Geom. **30**(2), 165–195 (2005)
6. Landge, A.G., Pascucci, V., Gyulassy, A., Bennett, J., Kolla, H., Chen, J., Bremer, P.: In-situ feature extraction of large scale combustion simulations using segmented merge trees. In: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1020–1031. IEEE, New York (2014)
7. Maadasamy, S., Doraiswamy, H., Natarajan, V.: A hybrid parallel algorithm for computing and tracking level set topology. In: 19th International Conference on High Performance Computing, pp. 1–10. IEEE Computer Society, New York (2012)
8. Massey, W.S.: A Basic Course in Algebraic Topology. Springer Science & Business Media, New York (1991)
9. Morozov, D., Weber, G.H.: Distributed merge trees. In: Nicolau, A., Shen, X., Amarasinghe, S.P., Vuduc, R.W. (eds.) ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP '13, Shenzhen, Feb 23–27, 2013, pp. 93–102. ACM, New York (2013)
10. Morozov, D., Weber, G.H.: Distributed contour trees. In: Bremer, P., Hotz, I., Pascucci, V., Peikert, R. (eds.) Topological Methods in Data Analysis and Visualization III, Theory, Algorithms, and Applications, pp. 89–102. Springer, Berlin (2014)

11. Pascucci, V., Cole-McLaughlin, K.: Parallel computation of the topology of level sets. Algorithmica **38**(1), 249–268 (2003)
12. Pascucci, V., Scorzelli, G., Bremer, P., Mascarenhas, A.: Robust on-line computation of Reeb graphs: simplicity and speed. ACM Trans. Graph. **26**(3), 58 (2007)
13. Rathkopf, J.A., Miller, D.S., Owen, J., Stuart, L., Zika, M., Eltgroth, P., Madsen, N., McCandless, K., Nowak, P., Nemanic, M., Gentile, N., Keen, N., Palmer, T.S.: KULL: LLNL's ASCI inertial confinement fusion simulation code. In: Physor 2000, ANS Topical Meeting on Advances in Reactor Physics and Mathematics and Computation into the Next Millennium (2000)
14. Tautges, T.J., Ernst, C., Stimpson, C., Meyers, R.J., Merkley, K.: MOAB: a mesh-oriented database. Technical Report SAND2004-1592, Sandia National Laboratories (2004)
15. Williams, S., Petersen, M., Bremer, P., Hecht, M., Pascucci, V., Ahrens, J.P., Hlawitschka, M., Hamann, B.: Adaptive extraction and quantification of geophysical vortices. IEEE Trans. Vis. Comput. Graph. **17**(12), 2088–2095 (2011)

Computing Invariants of Knotted Graphs Given by Sequences of Points in 3-Dimensional Space

Vitaliy Kurlin

Abstract We design a fast algorithm for computing the fundamental group of the complement to any knotted polygonal graph in 3-space. A polygonal graph consists of straight segments and is given by sequences of vertices along edge-paths. This polygonal model is motivated by protein backbones described in the Protein Data Bank by 3D positions of atoms. Our KGG algorithm simplifies a knotted graph and computes a short presentation of the Knotted Graph Group containing powerful invariants for classifying graphs up to isotopy. We use only a reduced plane diagram without building a large complex representing the complement of a graph in 3-space.

1 Introduction: Our Motivations, Key Concepts and Problems

This research is on the interface between knot theory, algebraic topology, homological algebra and computational geometry. Our main motivation is the application of topological and algebraic methods to recognizing knotted structures in 3-dimensional geometric graphs of long molecules such as protein backbones.

Backbones of Proteins are Polygonal Curves in 3-Space A *protein* is a large molecule containing a big number of amino acid residues. The primary structure or the *backbone* of a protein is the linear sequence of its amino acids. More than 100K proteins have been tabulated in the Protein Data Bank http://www.rcsb.org/pdb, which is a large database of pdb files. The *pdb file* of a single protein contains noisy coordinates (x, y, z) of all atoms that are linearly ordered in the backbone.

A natural way to model a protein is to assume that each atom is a point in 3-dimensional Euclidean space $\mathbb{R}^3$, while every chemical bond between atoms is a straight line segment between corresponding points. In general, a *polygonal* curve with vertices $p_1, \dots, p_m \in \mathbb{R}^3$ is the union of line segments connecting each point p_{i-1} with p_i for $i = 2, \dots, m$. In addition, if $p_0 = p_m$, we get a closed curve in $\mathbb{R}^3$.

V. Kurlin (✉)
Department of Computer Science, University of Liverpool, Ashton Building, Ashton Street, Liverpool L69 3BX, UK
e-mail: vitaliy.kurlin@gmail.com

H. Carr et al. (eds.), *Topological Methods in Data Analysis and Visualization IV*, Mathematics and Visualization, DOI 10.1007/978-3-319-44684-4_21

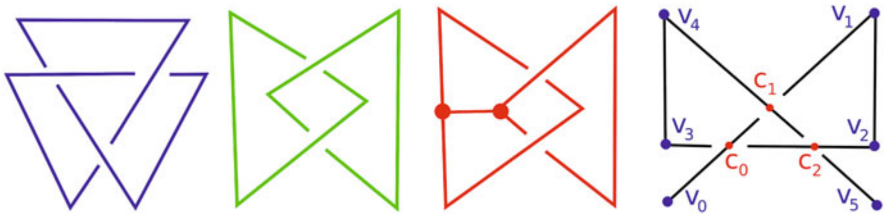


Fig. 1 Knotted polygonal graphs (trefoil, Hopf link, Hopf graph) and open trefoil with vertices $v_0 = (-2,-2,1)$, $v_1 = (2,2,-1)$, $v_2 = (2,-1,0)$, $v_3 = (-2,-1,0)$, $v_4 = (-2,2,2)$, $v_5 = (2,-2,-1)$ in $\mathbb{R}^3$ and crossings $c_0 = (-1,-1)$, $c_1 = (0,0)$, $c_2 = (1,-1)$ in the (x,y)-plane $\mathbb{R}^2$

Definition 1 A *polygonal knotted* graph is any embedded graph $K \subset \mathbb{R}^3$ consisting of finitely many straight line segments with pairwisely disjoint interiors. The number n of line segments in K is called the *length* of the polygonal graph $K \subset \mathbb{R}^3$.

The *degree* of a vertex v in a graph K is the number $\deg v$ of edges attached to v, and a loop is counted twice. Vertices with $\deg \neq 2$ are *essential*. An *edge-path* of K is a polygonal chain with essential vertices at two endpoints and only *non-essential* vertices of degree 2 between them. The open trefoil in Fig. 1 is the edge-path with four non-essential vertices v_1, v_2, v_3, v_4 between two essential vertices v_0, v_5 of degree 1. In practice, a polygonal graph K in 3-space is represented in a computer memory as

- an unordered list of points (x, y, z) corresponding to all essential vertices of K;
- a sequence of points (x, y, z) at non-essential vertices along every edge-path of K. The edge-path of the trefoil K in Fig. 1 is represented by the sequence v_1, v_2, v_3, v_4.

If the graph K is a circle, then $K \subset \mathbb{R}^3$ is a *knot*. If K is a disjoint union of several circles, then $K \subset \mathbb{R}^3$ is a *link*. Knotted graphs are usually studied up to *isotopy* that is a continuous deformation of $\mathbb{R}^3$ moving one graph to another, see Definition 3.

Recognition Problem for Protein Backbones and Knotted Graphs in 3-Space To distinguish different knots or graphs $K \subset \mathbb{R}^3$ up to isotopy, mathematicians construct *knot invariants* that should take the same value on all knots isotopic to each other. If such an invariant has different values on two knots, these knots are different.

The simplest non-trivial invariant is the number of connected components of a graph $K \subset \mathbb{R}^3$, which is preserved under any continuous deformation of $\mathbb{R}^3$. Hence a knot is not equivalent to a link consisting of at least two circles. However, this simple invariant can not distinguish any knots, so more powerful invariants are needed. A knot invariant can be called *complete* if it distinguishes all knots up to isotopy.

The complement $\mathbb{R}^3 - K$ of a knotted graph is 3-dimensional and contains more information about the isotopy class of K in the ambient space $\mathbb{R}^3$ than the 1-dimensional graph K itself. The oldest invariant of a knot $K \subset \mathbb{R}^3$ is the *fundamental group* of the knot complement $\mathbb{R}^3 - K$. Briefly, this group describes algebraic properties of closed loops that go around K in $\mathbb{R}^3$ and can be continuously deformed without intersecting K, see Definition 6. The *Alexander polynomial* of K is a simpler invariant that can be extracted from the fundamental group [2]. We highlight the advantages of the fundamental group over combinatorial invariants of knots.

- The group $\pi_1(\mathbb{R}^3 - K)$ is defined for any graph $K \subset \mathbb{R}^3$, not only for knots, and is an almost complete invariant of the isotopy class of K, see Theorems 7–9.
- Many invariants of knots $K \subset \mathbb{R}^3$ are introduced in terms of a *plane diagram*, which is a projection of K to $\mathbb{R}^2$ with only *double crossings*. These invariants are often computed in time exponential with respect to the number of crossings.
- Despite the group $\pi_1(\mathbb{R}^3 - K)$ is non-abelian, it leads to numerous abelian invariants that distinguish all prime knots with up to 11 crossings, see Theorem 12, using practically efficient algorithms from the HAP package of GAP [3].

Contributions of the Current Work to Recognizing Knotted Graphs Our input is any knotted polygonal graph K, which is motivated by real-life knotted structures. Our preferred invariant is the fundamental group $\pi_1(\mathbb{R}^3 - K)$ and is justified above. Our main result (Theorem 2 below) is a robust algorithm for a guaranteed fast computation of this almost complete invariant for arbitrary knotted graphs $K \subset \mathbb{R}^3$.

Theorem 2 *Given any polygonal graph $K \subset \mathbb{R}^3$ of a length n, our KGG algorithm first simplifies K to a small diagram with c crossings in time $O(n^2)$ and then writes a short presentation of the Knotted Graph Group $\pi_1(\mathbb{R}^3 - K)$ in time $O(c)$.*

The KGG algorithm and a proof of Theorem 2 are presented in Sect. 4. We highlight the improvements over the related past work, see more details in Sect. 3.

- We work with a Gauss code of a knotted graph $K \subset \mathbb{R}^3$ without modelling the complement $\mathbb{R}^3 - K$ by a cubical complex at a fixed resolution as in [1] and speed up the running time from seconds to milliseconds on a similar laptop, see Table 3.
- The fundamental group $\pi_1(\mathbb{R}^3 - K)$ is more powerful than the Alexander polynomial, which was used for recognising knotted proteins in the KnotProt [6].
- We substantially extend the KMT algorithm [9, 18], which smooths polygonal curves, to a simplification of any polygonal graph $K \subset \mathbb{R}^3$. Our implementation handles round-off errors much better than the state-of-the-art version in [8].

The KGG algorithm can fit well in a future version of the Homological Algebra Programming package (HAP) of GAP: Groups, Algorithms, Programming [3]. Moreover, the KGG algorithm can be used for connecting the Rosetta software (predicting protein structures as geometric graphs in 3-space) with the state-of-the-art recognition algorithm of trivial knots at http://www.javaview.de/services/knots.

This knot recognition is based on 3-page embeddings whose full theory was already extended to knotted graphs in $\mathbb{R}^3$ [10]. Gauss codes of knotted proteins produced by the KGG algorithm in this paper can be the input for the linear time algorithm [12] drawing 3-page embeddings of graphs. Hence we can visualize knotted proteins in a *3-page book* (a union of three half-planes with the same boundary line).

2 Background on Topological Invariants of Knotted Graphs

Knot Theory: Equivalences and Plane Diagrams of Knotted Graphs A *homeomorphism* is a bijection $f : X \to Y$ such that both f, f^{-1} are continuous. It is convenient to consider knots and graphs in the compact sphere S^3, which is obtained from $\mathbb{R}^3$ by adding a point at infinity, so $S^3 - \{\text{any point}\} \approx \mathbb{R}^3$ are homeomorphic.

Definition 3 Two knotted graphs $K, K' \subset S^3$ are called *equivalent* if there is a homeomorphism $f : S^3 \to S^3$ taking K to K', so $f(K) = K'$. The graphs $K, K' \subset S^3$ are *ambiently isotopic* if the above homeomorphism also preserves an orientation of S^3 or, equivalently, there is an *ambient isotopy* that is a continuous family of homeomorphisms $f_t : S^3 \to S^3$, $t \in [0, 1]$, such that $f_0 = \text{id}$ on S^3 and $f_1(K) = K'$.

The two mirror images of a trefoil are equivalent, but not isotopic, see a short proof in [4]. A knot $K \subset S^3$ is *trivial* (or the *unknot*) if K is isotopic to a round circle. The main problem in knot theory is to classify knots and more general knotted graphs up to equivalence or ambient isotopy from Definition 3. A *plane diagram* of a knotted graph $K \subset S^3$ is the image of K under a projection to a horizontal plane $\mathbb{R}^2$ in a general position having only transversal intersections (*double crossings*).

At each crossing we specify a short arc that crosses over another arc, see Fig. 1. The natural visual complexity of the isotopy class of a knotted graph $K \subset \mathbb{R}^3$ is the minimum number of crossings over all plane diagrams representing the graph K.

Knot Recognition: Reidemeister Moves and Gauss Codes of Graphs For the KGG algorithm in Sect. 4, we use the Reidemeister move R1 from generalized Reidemeister's Theorem 4 below saying that any isotopy of knotted graphs in $\mathbb{R}^3$ can be realized by a finite sequence of moves on plane diagrams in Fig. 2.

Theorem 4 ([7]) *Two plane diagrams represent isotopic knotted graphs in 3-space $\mathbb{R}^3$ if and only if the diagrams can be obtained from each other by an isotopy in (a continuous deformation of) $\mathbb{R}^2$ and finitely many Reidemeister moves in Fig. 2. (The move* R5 *is only for rigid graphs, the move* R5′ *is only for non-rigid graphs.)*

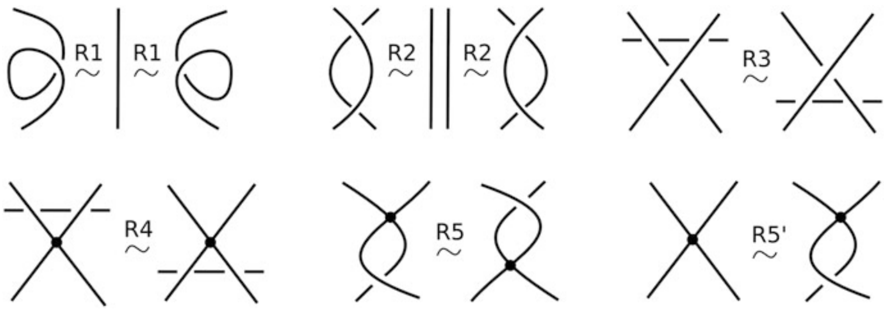


Fig. 2 Reidemeister moves on plane diagrams of knotted graphs, see Theorem 4

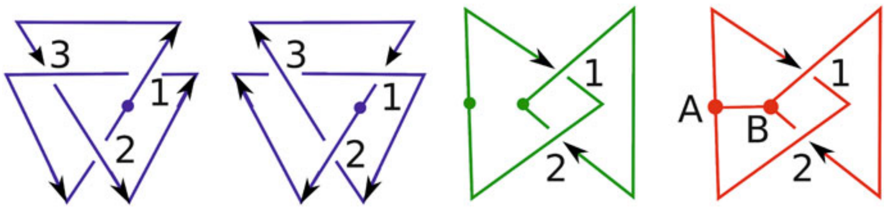


Fig. 3 Plane diagrams with directed edge-paths and labeled crossings illustrating Definition 5

The move R4 in Fig. 2 is for a vertex of degree 4 and similarly works for other degrees. The move R5 turns a small neighborhood of a vertex upside down. So a cyclic order of edges at vertices is preserved by R5. The move R5′ can reorder all edges at a vertex. Theorem 4 includes all symmetric images of moves in Fig. 2.

As described in [11], we shall encode a diagram of a knotted graph $K \subset \mathbb{R}^3$ by a simple Gauss code, which will be later converted into a presentation of the Knotted Graph Group $\pi_1(\mathbb{R}^3 - K)$. If a graph K contains a circle S^1 that is disjoint with the rest of the graph, then one of degree 2 vertices on S^1 will be *essential* so that the circle can be formally considered as an edge-path from this vertex to itself.

Definition 5 Let $D \subset \mathbb{R}^2$ be a plane diagram of a knotted graph K with only double crossings and essential vertices $A, B, C, \dots$ of degree not equal to 2. We fix directions of all edge-paths in K and arbitrarily label all crossings of D by $1, 2, \dots, l$. The *Gauss code* of D consists of all words W_{AB} associated to directed edge-paths AB of K from one essential vertex A to another essential vertex B as follows, see Fig. 3:

- W_{AB} starts with A, finishes with B and has the labels of all crossings in AB;
- if the edge-path AB goes under another edge-path of the graph K at a double crossing i, then we add the negative sign in front of the label i in the word W_{AB}.

The neighbors (vertices or crossings) of each vertex A are clockwisely ordered in $\mathbb{R}^2$, so the Gauss code specifies a cyclic order of all words starting or finishing at A.

The trefoils in Fig. 3 have codes $(1, -3, 2, -1, 3, -2)$ and $(2, -3, 1, -2, 3, -1)$, which are defined up to cyclic permutations. The Hopf link has the Gauss code consisting of two words $(1, -2)$ and $(-1, 2)$. The Hopf graph has the Gauss code consisting of three words corresponding to the three edges: (A, B), $(A, -1, 2, A)$, $(B, 1, -2, B)$.

The Fundamental Group and Abelian Invariants of a Graph Complement in S^3

Definition 6 Let $X \subset \mathbb{R}^3$ be a path-connected subset, so any two points in X can be connected by a continuous path within X. A closed *loop* at a base point $p \in X$ is a continuous map $f : [0, 1] \to X$ with $f(0) = p = f(1)$. Two such loops $f_0, f_1 : [0, 1] \to X$ are *path-homotopic* if they can be connected by a continuous

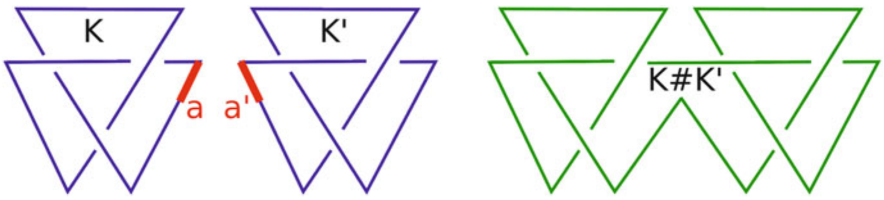


Fig. 4 A connected sum $K\#K'$ of two trefoils K and K' is well-defined up to ambient isotopy in $\mathbb{R}^3$

Table 1 Exact numbers of prime non-trivial knots from http://www.indiana.edu/~knotinfo

Number of crossings	≤ 6	7	8	9	10	11	12	≤ 12
Knot isotopy classes	7	7	21	49	165	552	2176	2977

family of loops $f_t : [0,1] \to X$, $t \in [0,1]$, always passing through the base point $p = f_t(0) = f_t(1)$ for $t \in [0,1]$. The *fundamental group* $\pi_1(X,p)$ is the group of all path-homotopy classes of closed loops in X. The product of two loops is obtained by going along the first loop (starting and finishing at the base point p), then along the second loop.

A *connected sum* $K\#K'$ of knots K, K' is obtained by removing two short open arcs $a \subset K$, $a' \subset K'$ and by joining the resulting four endpoints to form a larger knot $(K - a) \cup (K' - a')$, see Fig. 4. The isotopy class of $K\#K'$ depends only on the isotopy classes of K, K', not on a choice of a, a'. A knot not isotopic to a connected sum of non-trivial knots is called *prime*. Any knot uniquely decomposes into a connected sum of prime knots (up to permutations), hence only prime knots are classified (Table 1).

Theorems 7 and 8 imply that $\pi_1(S^3 - K)$ is a complete invariant for all prime knots. So $\pi_1(S^3 - K)$ and its abelian invariants can be used for recognizing knots. For a knotted graph $K \subset S^3$, let $N(K)$ be a small open neighborhood of the graph K. For instance, this neighbourhood can be the open ε-*offset* $K^\varepsilon = \cup_{p\in K} B(p;\varepsilon)$ consisting of open balls with a small radius $\varepsilon > 0$ and centers at all points $p \in K$. The complement $S^3 - N(K)$ is a compact 3-manifold whose boundary is $\partial N(K)$.

Theorem 7 ([5, Theorem 1]) *Two knots $K, K' \subset \mathbb{R}^3$ are equivalent* if and only if *there is a homeomorphism between their complements $S^3 - N(K) \approx S^3 - N(K')$. Two knots $K, L \subset \mathbb{R}^3$ are ambiently isotopic if and only if there is an orientation-preserving homeomorphism between their complements $S^3 - N(K) \approx S^3 - N(K')$.*

Theorem 8 ([20]) *If prime knots $K, K' \subset S^3$ have isomorphic groups $\pi_1(S^3 - K) \cong \pi_1(S^3 - K')$, then their complements are homeomorphic: $S^3 - K \approx S^3 - K'$.*

The Knotted Graph Group $\pi_1(S^3 - K)$ is almost a complete invariant in the sense that a *peripheral structure* of $\pi_1(S^3 - K)$ should be also preserved under a group isomorphism. Peripheral structures are completely characterised for links in [13].

We will assume that a knotted graph $K \subset S^3$ (if disconnected) is not *splittable*, namely K is not equivalent to a graph whose components are located in disjoint balls in S^3. The complement of any splittable graph K contains a sphere $S^2 \subset S^3 - N(K)$ separating components of K, so $S^3 - N(K)$ can be simplified by cutting S^2. The complement of any non-splittable graph can not be simplified in this way. In this case any $S^2 \subset S^3 - N(K)$ is called *incompressible* and $S^3 - N(K)$ is called *irreducible*.

A cycle $C \subset K$ in a knotted graph $K \subset S^3$ is *trivial* if the knot C in $C \cup (S^3 - K)$ is trivial, namely C bounds a topological disk D^2 in $S^3 - K$. If a knotted graph K has a trivial cycle C, we can compress the complement $S^3 - N(K)$ along the disk D^2 spanning C, so $S^3 - N(K)$ can be simplified by cutting D^2. The complement of K without trivial cycles can not be simplified in this way. In this case $\partial(S^3 - N(K))$ is called *incompressible* and $S^3 - N(K)$ is called *boundary-irreducible*.

Theorem 9 ([19, Corollary 6.5]) *For two non-splittable knotted graphs $K, K' \subset S^3$ without trivial cycles, let $\phi : \pi_1(S^3 - N(K)) \to \pi_1(S^3 - N(K'))$ be an isomorphism that descends to an isomorphism $\pi_1(\partial(S^3 - N(K))) \to \pi_1(\partial(S^3 - N(K')))$. Then there is a homeomorphism $S^3 - N(K) \approx S^3 - N(K')$ inducing the isomorphism ϕ.*

Theorems 7 and 9 imply that K, K' are equivalent. So the Knotted Graph Group $\pi_1(S^3 - K)$ is an almost complete invariant (complete with a peripheral structure).

Theorem 10 *Any finitely generated abelian group Z is isomorphic to a direct sum of cyclic groups $\mathbb{Z}^r \oplus \mathbb{Z}_{q_1} \oplus \cdots \oplus \mathbb{Z}_{q_l}$, where $r \geq 0$ is the* rank *and $q_1, \ldots, q_l$ are powers of primes. The numbers $r, q_1, \ldots, q_l$ are called the* abelian invariants *of the group Z and are uniquely determined by Z up to a permutation of indices $q_1, \ldots, q_l$.*

The above classification theorem says that any finitely generated abelian group can be completely described by its abelian invariants (a set of integers) and leads to numerous abelian invariants below that can be extracted from a non-abelian group G and efficiently computed by GAP if G has a short enough presentation [3].

Definition 11 The *index* of a subgroup H in a group G is the number of disjoint cosets $gH = \{gh \mid g \in G,\ h \in H\}$ that fill the group G. The *abelianization* of H is the quotient $H/[H, H]$ over the *commutator subgroup* $[H, H]$ generated by all $[a, b] = aba^{-1}b^{-1}$, $a, b \in H$. The *abelian invariants* of a non-abelian group G are the abelian invariants of $H/[H, H]$ over all subgroups $H \subset G$ up to a certain index.

3 Past Work on Computing Invariants of Knotted Proteins

Standard KMT Algorithm for Shortening a Knotted Protein Backbone The KMT algorithm is named after Koniaris and Muthukumar [9] and Taylor [18], though their methods are different. Taylor [18] actually suggested how to smooth

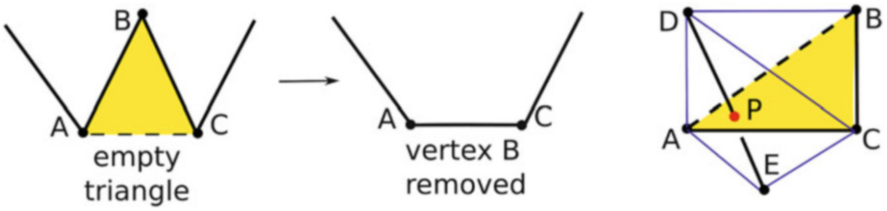


Fig. 5 *Left*: removing vertex B when $\triangle ABC$ is empty. *Right*: $ABCDE$ splits into three tetrahedra

a protein backbone. Namely, each vertex B with two neighbours A, C is iteratively replaced by the center of the triangle $\triangle ABC$, which visually smooths an original polygonal curve K. The standard KMT algorithm simply shortens K replacing the chain ABC by the single edge AC when the isotopy class of K is preserved.

We discuss the implementation of the KMT algorithm [8] used in the Rosetta program predicting structures of proteins at https://www.rosettacommons.org. One orders all degree 2 vertices $v_1, \dots, v_l$ according to the distance between their only neighbors A, C. Then the triangle $\triangle ABC$ based on a shortest segment AC is likely to be small and will probably not intersect any edge of K, see Fig. 5.

To check a potential intersection of $\triangle ABC$ with another edge DE, the plane ABC is intersected with the infinite line through DE. Finding an exact intersection point P requires divisions and leads to floating point errors, especially when DE is almost parallel to the plane ABC. Then three angles $\angle APB$, $\angle APC$, $\angle BPC$ are computed by using the arccos function, which also quickly accumulates computational errors.

Now the point P is inside the triangle $\triangle ABC$ if and only if the sum of three angles is $2\pi = \angle APB + \angle APC + \angle BPC$. In practice, for points P inside $\triangle ABC$, the above sum is only close to 2π, so the width of 3×10^{-4} is used to handle round off errors.

In Sect. 4 we extend KMT to the KGG (Knotted Graph Group) algorithm using only additions and multiplications without evaluations of complicated functions. We have checked that our algorithm correctly runs on similar protein backbones from the PDB database with the much smaller error of only 10^{-10}, see Sect. 5.

Alexander Polynomial of Knotted Proteins in KnotProt [6] The knot recognition of polygonal graphs $K \subset \mathbb{R}^3$ in the largest database KnotProt of knotted proteins is based on the Alexander polynomial [2, Sect. 8.3], which is a polynomial invariant of the fundamental group $\pi_1(\mathbb{R}^3 - K)$. Historically, there were no efficient algorithms to compare non-abelian groups up to isomorphism, hence a cubic computational time for the Alexander polynomial was acceptable. Moreover, the Alexander polynomial indeed classifies all knots with up to eight crossings.

However, the Alexander polynomial attains only 550 different values on 801 prime non-trivial knots (without mirror images) up to 11 crossings. So we feel that the time has come for more powerful invariants, especially due to the efficient algorithms in GAP [3]. The following experimental result by Brendel et al. [1]

demonstrates the power of the fundamental group for a practical classification of knots.

Theorem 12 ([1, Theorem 2]) *The abelianizations of subgroups with an index up to 6 in the fundamental group $\pi_1(\mathbb{R}^3 - K)$ distinguish all 801 prime non-trivial knots (up to mirror image) with plane diagrams having up to 11 crossings.*

Best methods for enumerating knots are based on triangulations of knot complements with a hyperbolic metric, which is not adapted yet for knotted graphs.

Discrete Morse Theory for Computing the Fundamental Group of a Complex Brendel et al. [1] suggested a general algorithm for computing the fundamental group of any regular cell complex. The algorithm uses a discrete Morse theory and is practically fast, though the theoretical complexity was hard to determine.

A protein backbone was modelled by a cubical knot $K \subset \mathbb{R}^3$, which is a union of small cubes at a fixed manually chosen resolution. For instance, the complement of the protein backbone $1V2X$ with joined endpoints in $\mathbb{R}^3$ was represented as a cubical complex C with 5,674,743 cells. This 3-dimensional complex C is deformed through several stages to a regular 2-dimensional complex C''' with 30,743 cells. The time for computing the knot group of $1V2X$ is about 35 s [1, Sect. 5], while our KGG algorithm takes 67 ms on a similar laptop, see Table 2.

Here are the key differences between our new approach and past work [1, 6, 8].

- The KMT algorithm only shortens a linear chain, while our KGG algorithm simplifies any knotted graph K and computes $\pi_1(\mathbb{R}^3 - K)$, which is more powerful than the Alexander polynomial used for recognizing knotted proteins in [6].
- Our KGG algorithm avoids evaluations of complicated functions and better handles floating point errors than KMT, also using the Reidemeister move R1 for extra reductions in the overall size of a knotted polygonal graph $K \subset \mathbb{R}^3$.

Table 2 Reduction in the number of vertices and crossings by the KMT and KGG algorithms

PDB code	Original #vertices	Original #crossings	Reduced #vertices	Reduced #crossings	Knot type	KMT time in seconds	KGG time in seconds
1yrl	1875	1144	37	43	4_1	0.82	0.81
4n2x	1788	1033	81	211	6_1	1.05	1.01
1qmg	2049	1455	44	71	4_1	1.03	1.02
3wj8	1788	972	79	180	6_1	1.08	1.07
4d67	6548	5485	97	301	?	14.45	14.12
4uwa	13,296	17,288	99	391	?	59.79	57.05
4ujc	11,938	10,180	217	731	?	61.77	59.91
4uwe	13,288	25,449	114	686	?	61.48	61.05
4ujd	11,938	10,565	212	755	?	72.42	70.27
4ug0	11,675	10,073	206	617	?	81.59	78.23

- We compute a simple presentation of the fundamental group $\pi_1(\mathbb{R}^3 - K)$ by working with only a given knotted polygonal graph $K \subset \mathbb{R}^3$ without modelling the complement $\mathbb{R}^3 - K$ as a large cubical complex at a fixed resolution as in [1].

4 KGG Algorithm for Computing the Knotted Graph Group

The input is a knotted polygonal graph $K \subset \mathbb{R}^3$ given by sequences of vertices along edge-paths. The output is a presentation of the Knotted Graph Group $\pi_1(\mathbb{R}^3 - K)$ with generators and relations. Here is a high-level description of all the stages.

1. In a given graph $K \subset \mathbb{R}^3$, identify all non-essential vertices that can be removed keeping the isotopy class of K after computing only five 3-by-3 determinants.
2. For a simplified graph $K' \subset \mathbb{R}^3$, find all crossings in a plane diagram of K'. Going along K', compute the Gauss code using the found crossings in the plane diagram of K'. Apply the Reidemeister move R1 for a further reduction if possible.
3. Turn a Gauss code into a presentation of the fundamental group $\pi_1(\mathbb{R}^3 - K)$ whose abelian invariants can be calculated using efficient algorithms of GAP.

Stage 1: Robust Algorithm for Shortening a Polygonal Graph Each degree 2 vertex B of a graph K has two neighbours, say A, C. We process all non-essential vertices B in the increasing order of $|AC|$. In comparison with the KMT algorithm, we much more robustly check if the interior of the triangle $\triangle ABC$ meets any edges of K.

For any edge DE with endpoints $D, E \notin \{A, B, C\}$, first we check if D, E are on different sides of the plane ABC. It is enough to compute the signed volumes of the tetrahedra $ABCD$ and $ABCE$, see Fig. 5. The volume V_{ABCD} is proportional (with factor $\frac{1}{6}$) to the 3-by-3 determinant whose columns are formed by the three coordinates of the three vectors $\overrightarrow{AB}, \overrightarrow{AC}, \overrightarrow{AD}$. The points D, E are on different sides of the plane ABC if and only if the signed volumes V_{ABCD} and V_{ABCE} have opposite signs.

If the edge DE intersects the plane ABC, we should check whether the intersection is inside the triangle $\triangle ABC$. Here is a simple geometric criterion: the edge DE meets the triangle $\triangle ABC$ if and only if the three tetrahedra $ABDE$, $ACDE$, $BCDE$ in Fig. 5 cover the union of the tetrahedra $ABCD$ and $ABCE$ *without any overlap*.

Such a geometric splitting is equivalent to the following algebraic identity between unsigned volumes: $|V_{ABCD}| + |V_{ABCE}| = |V_{ABDE}| + |V_{ACDE}| + |V_{BCDE}|$. Hence it remains to compute only three more 3-by-3 determinants for the quadruples $ABDE$, $ACDE$, $BCDE$. All computations involve only basic additions and multiplications.

Stage 2: Computing a Gauss Code of a Reduced Plane Diagram of K Let $K \subset \mathbb{R}^3$ be a simplified polygonal graph obtained by all possible shortenings at Stage 1 above. Now we simply project K' to the (x, y)-plane $\mathbb{R}^2$ finding all intersections between straight edges in the projection of K'. If the plane diagram of K' is not in a general position, we slightly perturb its vertices to guarantee that we have only double crossings, because we are interested only in the isotopy class of $K' \subset \mathbb{R}^3$.

This stage requires a quadratic time $O(m^2)$ in the length m of the simplified graph K', because we check all potential pairwise intersections of non-adjacent edges. In experiments on protein backbones in Sect. 5, the chain K' is much shorter than the original backbone K, hence this stage is fast enough in practice. For each edge $[v_i, v_{i+1}]$ of K', we build a list of crossings with other edges $[v_j, v_{j+1}]$. We keep this list of crossings in order from the vertex v_i to v_{i+1}. Apart from the actual coordinates (x, y) of a crossing, we also note the corresponding indices i, j and the heights z_i, z_j of the points in the intersecting edges above the crossing (x, y).

After completing these ordered lists over all edges, we can go along each edge-path of K' and assign a correct label to every crossing, because we can recognize if a crossing has been passed before. Since we kept actual heights z_i, z_j at each crossing, we can add negative signs to all undercrossings as needed by Definition 5.

Finally, if a Gauss code contains a consecutive pair of labels $(l, -l)$ or $(-l, l)$, the plane diagram contains a small loop that can be easily removed by the Reidemeister move R1 in Fig. 2. Assume that this crossing (x, y) in the move R1 is formed by (projections of) edges $[v_i, v_{i+1}]$ and $[v_j, v_{j+1}]$ for $i + 1 < j$. Then we can shorten these edges by continuously moving the endpoints v_{i+1} and v_j towards the points (at the heights z_i, z_j, respectively) that project exactly to the crossing (x, y) in $\mathbb{R}^2$.

The chain of edges from v_{i+1} to v_j does not cross any other edges by our choice of the crossing in the move R1 and can be replaced by the single vertical edge from (x, y, z_i) to (x, y, z_j). This extra simplification can potentially make a few triangles on three consecutive vertices empty. Hence we can check if the simplifications from Stage 1 are possible for a few triangles related to the vertices $v_i, v_{i+1}, v_j, v_{j+1}$.

Stage 3: Writing a Wirtinger Presentation for the Fundamental Group We remind how to write down a presentation of the group $\pi_1(\mathbb{R}^3 - K)$ by using a plane diagram D of a knotted graph $K \subset \mathbb{R}^3$, see more details in [2, Sect. 6.1].

We arbitrarily orient all edge-paths of K, though our choice will not affect $\pi_1(\mathbb{R}^3 - K)$. We fix a base point $p \in \mathbb{R}^3$ at infinity, say at the point $(0, 0, z)$ for a large coordinate $z > 0$. If we cut all essential vertices (of degree at least 3) and crossings (in lower edges), the diagram D splits into several oriented arcs $a_1, \dots, a_m$. In the 3rd picture of Fig. 6 these arcs in D contain the following vertices and crossings:

$$a_1 = [v_0, c_1, c_2], \quad a_2 = [c_2, v_1, v_2, c_3, c_1], \quad a_3 = [c_1, v_3, v_4, c_2, c_3], \quad a_4 = [c_3, v_5].$$

We associate to every resulting arc a_i a generator $x_i \in \pi_1(\mathbb{R}^3 - K)$. Each generator x_i can be represented by a closed loop $\tilde{x}_i$ that goes from the base point p to a point near the arc a_i along a path γ_i, makes a loop around the oriented arc a_i and then goes back to the base point p along γ_i in the opposite direction. In the 2nd and 3rd pictures of Fig. 6 we show each long loop $\tilde{x}_i$ only by a short arrow under a_i. Each short arrow is labelled by the generator $x_i \in \pi_1(\mathbb{R}^3 - K)$ represented by the loop $\tilde{x}_i$.

At each a crossing, two consecutive arcs a_j, a_{j+1} share the same endpoint c and another arc a_i crosses over c, see the 2nd picture of Fig. 6. To this crossing we associate the relation $x_i x_j x_i^{-1} = x_{j+1}$ saying that the loop $\tilde{x}_{j+1}$ around a_{j+1} can be obtained by going first along $\tilde{x}_i$, then along $\tilde{x}_j$ and along the reversed loop $\tilde{x}_i{}^{-1}$.

If we join two vertices of degree 1 away from the rest of the diagram, the initial and final generators are equal, e.g. $x_1 = x_4$ in the 3rd picture of Fig. 6. The crossings c_1, c_2, c_3 in the same picture have the associated relations $x_1 x_2 x_1^{-1} = x_3$ and $x_3^{-1} x_1 x_3 = x_2$ and $x_2 x_4 x_2^{-1} = x_3$, respectively. Together with $x_1 = x_4$, the 4 relations reduce to the short presentation $\langle x_1, x_2 \mid x_1 x_2 x_1 = x_2 x_1 x_2 \rangle$ of the trefoil group.

If a vertex v has attached arcs $a_1, \dots, a_l$, then write the relation $x_1^{\varepsilon_1} \dots x_l^{\varepsilon_l} = 1$, where $\varepsilon_i = +1$ for arcs a_i coming to v and $\varepsilon_i = -1$ for arcs a_i going out of v. The vertex v in the 1st picture of Fig. 6 has the associated relation $x_i x_j x_k^{-1} = 1$.

Any closed loop in the complement $\mathbb{R}^3 - K$ easily decomposes into a product of loops $\tilde{x}_i$ around arcs $a_1, \dots, a_m$. However, it is a non-trivial theorem that the simple relations above define the fundamental group $\pi_1(\mathbb{R}^3 - K)$, see [2, Sect. 6.3].

We can convert a Gauss code of a plane diagram of K into a Wirtinger presentation of $\pi_1(\mathbb{R}^3 - K)$ as follows. The above arcs a_i between successive undercrossings in the plane diagram D correspond to *subsequences* between vertices and negative labels in the Gauss code of D. For each negative label $(-j)$, we know two subsequences that meet at $(-j)$ and also we can find the ith subsequence containing the positive label j, so we can write the corresponding relation $x_i x_j x_i^{-1} = x_{j+1}$.

For each vertex v of deg ≥ 3, we can find subsequences in the code that start or finish with the symbol v and write the product of corresponding generators (if the subsequence starts with v) or their inverses (if the subsequence finishes with v).

Proof of Theorem 2. At Stage 1 of the KGG algorithm in Sect. 4, for each degree 2 vertex B of a polygonal graph $K \subset \mathbb{R}^3$, we compute the distance between the two neighbours A, C of B in total time $O(n)$, where n is the length of K. We sort all degree 2 vertices $B \in K$ by the increasing distances AC in time $O(n \log n)$.

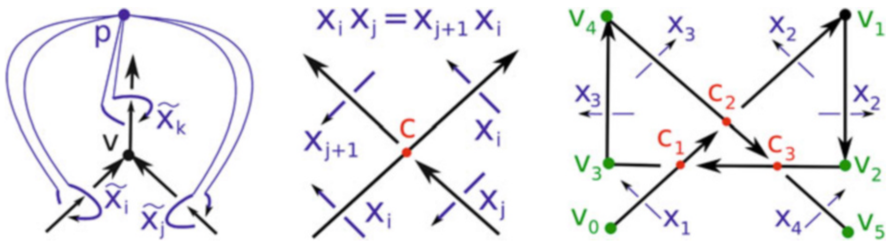


Fig. 6 *Left*: generators around a vertex and crossing. *Right*: four generators for four arcs in a diagram

Starting with a vertex B with a shortest segment AC, we check if the 3-chain ABC can be replaced by the single edge AC, which requires five 3-by-3 determinants for every other edge DE of K. If $\triangle ABC$ doesn't meet all edges DE, we remove B and update the sorted distances AC in time $O(\log n)$. The time of Stage 1 is $O(n^2)$.

At Stage 2 we check all pairwise intersections of $m \leq n$ projected edges in the simplified graph $K' \subset \mathbb{R}^3$, which requires $O(m^2)$ time. Stage 3 is linear in the length of a Gauss code which has $c = O(m^2)$ crossings. Hence the total time is $O(n^2)$. □

5 Experimental Results: Recognizing Knots in Protein Backbones

Table 2 shows how the numbers of vertices and crossings of a protein backbone K are reduced by Stage 1 of the KGG algorithm from Sect. 4. The knot types are 0 (unknot), 3_1 (trefoil knot), 4_1 (figure-eight knot) and 6_1 (Stevedore's knot).

The classical KMT algorithm for a polygonal chain of n edges has the running time $O(n^2)$. The time to compute the Alexander polynomial of a knotted graph with k crossings is $O(k^3)$, where $k = O(m^2)$ for a simplified graph of a length m.

Recall that the backbone of a protein is a polygonal chain of carbon atoms ordered as in a given PDB file. We linearly extend terminal edges of a backbone and join them away from all other vertices to get a closed knot. Table 2 shows the numbers of vertices and crossing after reductions by the KGG algorithm. In some cases the KMT algorithm outputs a few more crossings. because the Reidemeister move R1 wasn't used. In all cases the KGG algorithm is faster despite these extra moves.

The last six rows in Table 2 are for longest proteins from PDB. Even simplified backbones are too long and we hope to determine their knot types in the future.

The KGG algorithm can be extended to visualize knotted proteins using 3-page embeddings [12, 14] and to compute abelian invariants of the Knotted Graph Group using GAP [3, Sect. 47.15]. The C++ code is at author's website http://kurlin.org. Table 3 shows Gauss codes obtained at Stage 3 of the KGG algorithm.

Table 3 Knot types and Gauss codes of the reduced backbones of knotted proteins from PDB

PDB code	Original #crossings	Knot type and Gauss code after KGG	PDB code	Original #crossings	Knot type and Gauss code after reduction by KGG
1v2x	39	$\mathbf{3}_1$ (1 -2 3 -1 2 -3)	3nou	304	$\mathbf{4}_1$ (-1 2 3 -4 5 1 -2 -5 4 -3)
3oil	102	$\mathbf{3}_1$ (1 -2 3 -1 2 -3)	3not	300	$\mathbf{4}_1$ (-1 2 3 -4 5 1 -2 -5 4 -3)

6 Conclusions, Discussion and Open Problems for Future Work

We have designed a new easy-to-implement KGG algorithm in Sect. 4 to compute the Knotted Graph Group $\pi_1(\mathbb{R}^3 - K)$ for any polygonal graph $K \subset \mathbb{R}^3$ given by a sequence of points in $\mathbb{R}^3$. The experimental results in Sect. 5 confirm substantial reductions in the complexity of knotted backbones. Our approach strikes right in the middle of a wide range of topological objects. Namely, the KGG algorithm works for arbitrary knotted graphs, which are more general than knots or links and runs faster than memory expensive methods designed for regular 2D complexes [1].

A *theta-curve* is a knotted graph $\theta \subset \mathbb{R}^3$ with two vertices joined by three edges as in the Greek character θ. The enumeration of theta-curves with up to seven crossings was manually completed [17] by analyzing the Alexander polynomial and three knots obtained from $\theta \subset \mathbb{R}^3$ by removing one of three edges. That is why we believe that abelian invariants of the quickly computable Knotted Graph Group $\pi_1(\mathbb{R}^3 - K)$ can be enough for enumerating more complicated theta-curves and general graphs.

Our robust computation of the fundamental group $\pi_1(\mathbb{R}^3 - K)$ for any knotted graph $K \subset \mathbb{R}^3$ opens the following new possibilities for further research.

- Automatically enumerate all isotopy classes of knotted graphs $K \subset \mathbb{R}^3$ with a few essential vertices and up to a maximum possible number of crossings. A good starting point is to check the manual classification of theta-curves in [17].
- Build distributions for isotopy classes of large random knots modelled as in [16], when the Alexander polynomial can be too weak to distinguish different knots.
- Study the persistence and stability of abelian invariants similarly to the persistence of the group $\pi_1(\mathbb{R}^3 - K)$ in a filtration of knot neighborhoods from [15].

Acknowledgements This work was done during the EPSRC-funded secondment at Microsoft Research Cambridge, UK. We thank all reviewers for valuable comments and helpful suggestions.

References

1. Brendel, P., Dlotko, P., Ellis, G., Juda, M., Mrozek, M.: Computing fundamental groups from point clouds. Appl. Algebra Eng. Commun. Comput. **26**, 27–48 (2015)
2. Crowell, R., Fox, R.: Introduction to Knot Theory. Graduate Texts in Mathematics, vol. 57. Springer, New York (1963)
3. Ellis, G.: HAP — Homological Algebra Programming package for GAP. Version 1.10.13 (2013). Available for download at http://www.gap-systems.org/Packages/hap.html
4. Fenn, R.: Tackling the trefoils. J. Knot Theory Ramif. **21**, 1240004 (2012)
5. Gordon, C., Luecke, J.: Knots are determined by their complements. J. Am. Math. Soc. **2**, 371–415 (1989)
6. Jamroz, M., Niemyska, W., Rawdon, E., Stasiak, A., Millett, K., Sulkowski, P., Sulkowska, J.: KnotProt: a database of proteins with knots and slipknots. Nucleic Acids Res. **1**, 1–9 (2014)
7. Kauffman, L.: Invariants of graphs in three-space. Trans. AMS **311**, 697–710 (1989)

8. Khatib, F., Weirauch, M., Rohl, C.: Rapid knot detection and application to protein structure prediction. Bioinformatics **14**, 252–259 (2006)
9. Koniaris K., Muthukumar, M.: Self-entanglement in ring polymers. J. Chem. Phys. **95**, 2871–2881 (1991)
10. Kurlin, V.: Three-page encoding and complexity theory for spatial graphs. J. Knot Theory Ramif. **16**, 59–102 (2007)
11. Kurlin, V.: Gauss paragraphs of classical links and a characterization of virtual link groups. Math. Proc. Camb. Philos. Soc. **145**, 129–140 (2008)
12. Kurlin, V.: A linear time algorithm for visualizing knotted structures in 3 pages. In: Proceedings of IVAPP: Information Visualization Theory and Applications, Berlin, pp. 5–16 (2015)
13. Kurlin, V., Lines, D.: Peripherally specified homomorphs of link groups. J. Knot Theory Ramif. **16**, 719–740 (2007)
14. Kurlin, V., Smithers, C.: A linear time algorithm for embedding arbitrary knotted graphs into a 3-page book. In: Computer Vision, Imaging and Computer Graphics Theory and Applications, pp. 99–122. Springer, Berlin (2016). http://www.springer.com/gb/book/9783319299709
15. Letscher, D.: On persistent homotopy, knotting and the Alexander module. In: Proceedings of ITCS (2012)
16. Millett, K.C., Rawdon. E.J., Stasiak, A.: Linear random knots and their scaling behaviour. Macromolecules **38**, 601–606 (2005)
17. Moriuchi, H.: An enumeration of theta-curves with up to 7 crossings. In: Proceedings of the East Asian School of Knots, Links and Related Topics, Seoul (2004)
18. Taylor, W.: A deeply knotted protein structure and how it might fold. Nature **406**, 916–919 (2000)
19. Waldhausen, F.: On irreducible 3-manifolds which are sufficiently large. Ann. Math. (2) **87**, 56–88 (1968)
20. Whitten, W.: Knot complements and groups. Topology **26**, 41–44 (1987)