Branching and Circular Features in High Dimensional Data

Bei Wang, Brian Summa, Valerio Pascucci, *Member, IEEE*, and Mikael Vejdemo-Johansson

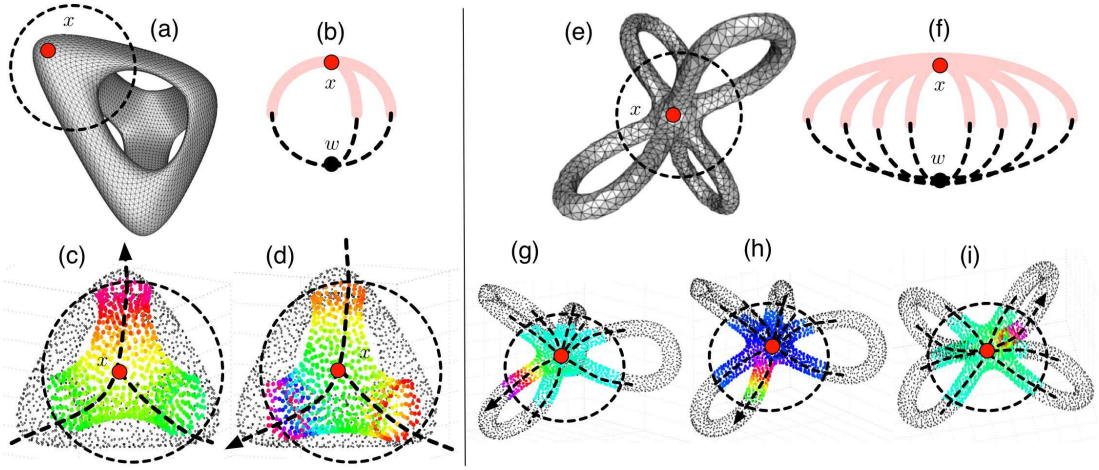

Fig. 1. Given point cloud data in high-dimensional space, we detect and visualize branching structures in a neighborhood surrounding a given point of interest. Here we give two simple examples with point clouds sampled from surfaces embedded in 3-dimensional space. In (a), given a genus-3 surface, we analyze the branching structure around one of its corners, x. We apply color-mapping transfer functions to local circle-valued coordinate functions to visualize the structure. Specifically, the color scale indicates the "direction" of the branches. As illustrated in (b), there is a local two-way branching around x, where the coordinate function of each branch is visualized in (c) and (d), respectively. In (e), given a genus-4 surface, we detect a seven-way branching around x (f), where three of the coordinate functions are shown in (g), (h) and (i), respectively.

Abstract—Large observations and simulations in scientific research give rise to high-dimensional data sets that present many challenges and opportunities in data analysis and visualization. Researchers in application domains such as engineering, computational biology, climate study, imaging and motion capture are faced with the problem of how to discover compact representations of high-dimensional data while preserving their intrinsic structure. In many applications, the original data is projected onto low-dimensional space via dimensionality reduction techniques prior to modeling. One problem with this approach is that the projection step in the process can fail to preserve structure in the data that is only apparent in high dimensions. Conversely, such techniques may create structural illusions in the projection, implying structure not present in the original high-dimensional data. Our solution is to utilize topological techniques to recover important structures in high-dimensional data that contains non-trivial topology. Specifically, we are interested in high-dimensional branching structures. We construct local circle-valued coordinate functions to represent such features. Subsequently, we perform dimensionality reduction on the data while ensuring such structures are visually preserved. Additionally, we study the effects of global circular structures on visualizations. Our results reveal never-before-seen structures on real-world data sets from a variety of applications.

Index Terms—Dimensionality reduction, circular coordinates, visualization, topological analysis.

1 Introduction

Many scientific investigations depend on exploratory data analysis and visualization of high-dimensional data sets that represent complex phenomena. Given a collection of high-dimensional data points, dimensionality reduction techniques are typically applied prior to modeling and feature detection. These techniques find a low-dimensional representation of the data with simple guarantees, by assuming that real-valued low-dimensional coordinates are sufficient to capture its underlying intrinsic structure.

In mathematical terms, given a collection of high-dimensional data points $X \in \mathbb{R}^d$, dimensionality reduction techniques obtain an embedding that maps a point $x = (x_1, x_2, ..., x_d) \in X$ to a point $y = (y_1, y_2, ..., y_m)$, where $m \ll d$, through a set of real-valued coordinate functions $\phi = (\phi_1, \phi_2, ...\phi_m) : X \to \mathbb{R}$, where $y_i = \phi_i(x)$, with the assumption that the data typically has the topological structure of a convex domain [15]. However, if the underlying space in high dimension contains nontrivial topology, either globally or locally, dimensionality reduction alone is no longer sufficient to preserve the topology.

In [15], the authors challenge the convex domain assumption in dimensionality reduction through topological analysis. They give a method for computing global circular coordinates and interpreting the results, and illustrate their methods on data sets containing circular structures, such as a circle, an annulus and a torus. In particular, the authors describe a topological procedure that enlarges the class of coordinate functions for dimensionality reduction to include global circle-valued functions, mapping the point cloud to a closed circle: $\theta : X \to \mathbb{S}^1$.

We extend this work by introducing *local circular coordinates* by computing *persistent local cohomology*. As a local coordinate, we give a procedure using *relative cohomology* to compute a function $\theta_U : U \to \mathbb{S}^1$ from a (small) neighborhood in the point cloud to the closed circle. We observe that these local coordinates produce a natu-

- *Bei Wang, Brian Summa, and Valerio Pascucci are with SCI Institute, University of Utah, E-mails: beiwang@sci.utah.edu, bsumma@sci.utah.edu, pascucci@sci.utah.edu.*
- *Mikael Vejdemo-Johansson is with Stanford University, E-mail: mik@math.stanford.edu.*

Manuscript received 31 March 2011; accepted 1 August 2011; posted online 23 October 2011; mailed on 14 October 2011.
For information on obtaining reprints of this article, please send email to: tvcg@computer.org.

ral interpretation as encoding branching behaviors. The acquired coordinates are visualized by applying a color map transfer function. In this paper, we will describe this extension, and also give examples from a number of datasets of both global and local circular coordinates, with interpretations of these coordinates in both the global and local realm. Even though [15] produced some examples of the global circular coordinate structures, our emphasis will be on less artificial datasets and on the interpretation of the resulting coordinates.

There are two advantages to using topologically motivated circle-valued coordinate functions. First, they enrich data representations by revealing branching and circular features in the data. Second, by reflecting topological properties in the high-dimensional embedding domain, they help differentiate intrinsic structure in the data from structural illusions.

For example, for a point cloud sampled from a torus embedded in 2D as shown in Figure 2 (a) and (b), dimensionality techniques alone can always visualize one of its essential loops (generators of the homology groups) represented by θ_1, while the same techniques fail to showcase the other essential loop as revealed by θ_2 without tearing or cutting. In Figure 2 (c), (d) and (e), we can see that global circle-valued coordinate functions differentiate a trefoil knot from two linked circles based upon seemingly similar projections. While this differentiation could arguably be performed with clustering techniques, the cohomological approach is more robust to knotting and interleaving of the data sets than a pure clustering technique would have been.

On the other hand, local circle-valued coordinate functions reveal topological features within a sub-region of the point cloud, as shown in Figure 3. It is able to capture the three-way branching structure surrounding the crossing point in figure eight (Figure 3 (e), (f) and (g)), while detecting structural illusion of a figure eight created by projecting a circle in a certain direction (Figure 3 (b) and (c)).

Our main contributions are as follows.

- We introduce local circle-valued coordinate functions that facilitate local structural analysis, especially the detection of branching features in data. We construct these functions in a local neighborhood through topological analysis of 1-dimensional cohomology. That is, we choose a subset of points U that are with close proximity of a given point, and construct coordinate functions $\theta_U : U \to \mathbb{S}^1$.
- On the technical level, we develop a local version of the persistent cohomology machinery through local cohomology computed on point cloud data. Persistence enables the detection of significant local features and separates features from noise within the data. That is, we obtain a parametrization of U through coordinate functions $\theta_1, \theta_2, ..., \theta_n : U \to \mathbb{S}^1$, where n indicates the number of significant local features.
- We present the first technique that approximates topological circular and branching structures in high-dimensional space to aid visualization in the low-dimensional projection.
- We present empirical evidence demonstrating that both the local and global circle-valued coordinate functions, for the first time, permit more precise analysis on real-world data sets. This extends the more artificial experiments in [15], where emphasis was on proof-of-concept using simulated data sets.

2 Related Work

A variety of nonlinear dimensionality reduction techniques, mostly spectral approaches, have been proposed in recent years. These spectral methods typically construct an adjacency graph from the point cloud data and compute a pair-wise distance matrix from which eigenvectors are extracted to represent the data in a low dimensional space [36]. Some of the popular methods include Isomap [46], locally linear embedding [43], Laplacian eigenmaps [4], and kernel PCA [44].

Several nonlinear dimensionality reduction techniques have been presented in recent years to analyze and visualize various topological features in high dimensions. Takahashi et al. extract approximate contour trees and Reeb graphs from high-dimensional point samples by introducing new distance metrics to manifold learning techniques such as Isomap [45]. They use the *k-nearest neighbor graph* to approximate point-wise proximity. Oesterling et al. visualize the topological structure of point clouds indirectly by visualizing the topology of their density distribution [40]. They first approximate the density function

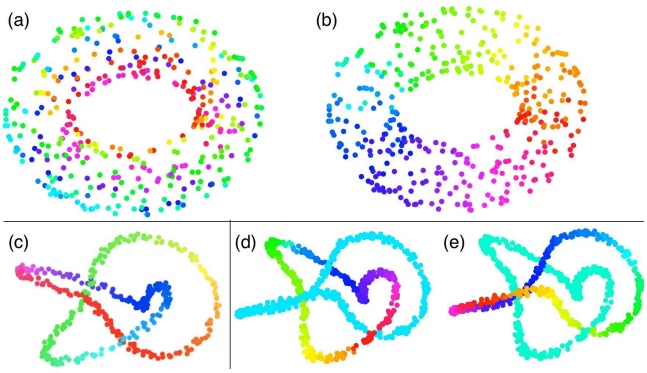

Fig. 2. Visualized global circular coordinate functions. (a) and (b): two global circle-valued coordinate functions for a point cloud sampled from a torus. (a): θ_1. (b): θ_2. (c), projection of a trefoil knot; (d) and (e), projection of two linked circles. Figures are reproductions from [15].

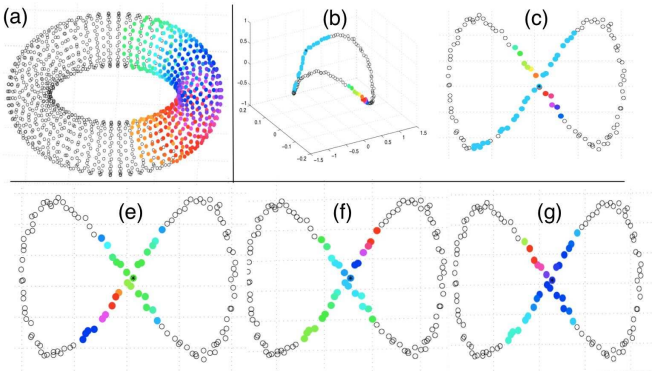

Fig. 3. Visualized local circle-valued coordinate function. (a): local circle-valued coordinate function for a point cloud sampled from a torus. (b) and (c): projection of a circle (b) on 2D (c) that gives an illusion of a figure eight, local circle-valued coordinate function indicates there is no local branching structure. (e), (f) and (g): projection of a figure eight on 2D, three circle-valued coordinate functions are visualized to describe the local branching structures.

by sampling on the point cloud's *Gabriel graph* [31], compute its join tree, generate the *topological landscape* [48] from the join tree, and finally argument the landscape by placing the original data points at chosen locations [40]. The technique has been extended and applied successfully in topology-based projection and visualization of high dimensional document point clouds [41]. [45] focuses on preserving topological transitions of level-sets, and [41] studies the structure of data in terms of dense regions, while our work emphasizes local branching and global circular structure detection in high dimensions that are not necessarily captured by the above techniques. In particular, our work introduces new localized mapping of high-dimensional point cloud data for the analysis of branching structures.

Various algorithms have been proposed to compute loops on surfaces, or homology generators that satisfy certain geometric optimality [29, 12, 51, 26]. [29] computes shortest set of homology generators for 2-manifolds. [19] uses topological persistence [25] to computes topologically correct loops on surfaces, that wrap around their "handles" and "tunnels". Given a weighted simplicial complex and a nontrivial cycle, [18] computes its homologous cycle with minimal weight. [20] approximates a shortest basis of the one dimensional homology group of a manifold in $\mathbb{R}^d$ from its point sample. Algorithms have also been developed to compute shortest cycles, minimum cuts, or maximum flow related to graphs embedded on surfaces [30, 28, 27].

In terms of revealing circular structures or essential loops within data, several approaches have been taken to find alternative representations. [21] studies cylindrical manifolds, data whose generative model includes a cyclic and a linear parameter, and tries to find embedding functions that map them onto a cylinder $\mathbb{S}^1 \times \mathbb{R}$. [37] projects data with non-trivial topology by destroying essential loops via tearing and cutting. [42] maps data to a pre-chosen non-flat target space, such as a

cylinder or a sphere, using multidimensional scaling.

In [15], the authors present a framework of *persistent cohomology*, and demonstrates how a correspondence from homotopy theory enables the construction of circle-valued coordinate functions from cohomology classes. In this work, persistence aids the construction by providing a quality measure for the cohomology classes and thus also for the corresponding circle-valued coordinates. From this framework emerges coordinate functions for dimensionality reduction that respect and reflect the original topology of the high-dimensional embedding, while highlighting such structures in the data that can be continuously mapped onto the circle.

Recent work in [8] defines a modified version of topological persistence (level persistence) for 1-cocycles, and shows that such 1-cocycles can be interpreted as a circle valued map. While both [15] and [8] discuss circle-valued functions, we notice that the papers differ fundamentally in approach and in the notion of persistence used.

Algorithms that focus on cohomology computation, especially persistent cohomology, have been proposed in recent years [15, 8]. [22] designs efficient algorithm to compute cohomology basis. [16] addresses duality in persistent homology and cohomology computation, while [13] compares efficiencies of these algorithms. Local persistent homology has been used in stratification learning [5, 7].

Compared to these bodies of previous work, our paper is the first that constructs local circle-valued coordinates on high-dimensional data sets using persistent local cohomology, and is the first to observe the connections to branching structures. We discover and visualize topological structures such as circles and branches on some data sets that have never been realized before.

3 Technical Background

Our algorithm has key ingredients from both algebra, topology, and algorithmics. We review necessary background on the topological concepts for non-specialists, providing intuitive and illustrative examples. Using this, we describe *local cohomology*, and how it connects to the analysis of branching structures. For a readable mathematical introduction to algebraic topology, and algorithm details for computing homology and cohomology groups, see [39, 34]. For an introduction to persistent homology and its related algorithms, see [23, 24].

3.1 Homology and cohomology

Homology. Homology deals with topological features such as "holes" or "cycles" ; 0-, 1- and 2- dimensional homology groups correspond to components, tunnels and voids in a topological space. Here, we discuss its simplest and most concrete definition, at the level of simplicial homology.

Consider the simplicial complex K pictured in Figure 4, which is a triangulation of an annulus. The 0-, 1- and 2-simplexes in K are the vertices, edges and triangles, denoted by the sets $K^0 = \{v_i\}$, $K^1 = \{e_i\}$ and $K^2 = \{\Delta_i\}$ respectively. We assume all simplices are oriented. We define the 0-chains, 1-chains and 2-chains as formal sums of 0-, 1- and 2-simplexes with integer coefficients, respectively,

$$\begin{aligned} C_0 &= C_0(K;\mathbb{Z}) = \{b = \sum g_i v_i \mid g_i \in \mathbb{Z}\}, \\ C_1 &= C_1(K;\mathbb{Z}) = \{a = \sum g_i e_i \mid g_i \in \mathbb{Z}\}, \\ C_2 &= C_2(K;\mathbb{Z}) = \{c = \sum g_i \Delta_i \mid g_i \in \mathbb{Z}\}. \end{aligned}$$

By abuse of notation, each 0-, 1- and 2-simplex in K corresponds to an *elementary chain* of the same dimension. Then 0-, 1- and 2-chains can be considered as sums of elementary chains. Here, the 0-chain b_1 is $v_1+v_2+v_3$ (solid green), the 1-chain a_1 is $e_1+e_2+e_3+e_4+e_5+e_6+e_7+e_8$ (bold red), and the 2-chain c_1 is $\Delta_1+\Delta_2$ (bold pink). We now define *boundary maps*, $\partial_2 : C_2 \to C_1$ and $\partial_1 : C_1 \to C_0$. Representing an oriented p-simplex by its vertices $[v_0,...,v_p]$, we have,

$$\begin{aligned} \partial_2([v_0,v_1,v_2]) &= [v_1,v_2]-[v_0,v_2]+[v_0,v_1]. \\ \partial_1([v_0,v_1]) &= v_1 - v_0. \end{aligned}$$

It is easy to verify that $\partial \circ \partial = 0$. For instance, $\partial \circ \partial(\Delta_1) = \partial(\partial([v_4,v_5,v_6])) = \partial([v_5,v_6]-[v_4,v_6]+[v_4,v_5]) = \partial([v_5,v_6]) - \partial([v_4,v_6])+\partial([v_4,v_5]) = v_6-v_5-(v_6-v_4)+v_5-v_4 = 0$. Let $a \in C_1$. a is a 1-*cycle* if $\partial(a) = 0$. It is a 1-*boundary* if it is the boundary of some 2-chain c, that is, $\partial(c) = a$. Let $\ker\partial_1$ denote the set of all 1-cycles and $\mathrm{im}\,\partial_2$ denote the set of all 1-boundaries. Since the 1-boundaries are always 1-cycles, $\mathrm{im}\,\partial_2 \subseteq \ker\partial_1$. The 1-*homology* of K is the quotient group, $\mathsf{H}_1 = \mathsf{H}_1(K;\mathbb{Z}) = \ker\partial_1/\mathrm{im}\,\partial_2$. For example, a_1 (bold red) is a 1-cycle since $\partial(a_1) = 0$. $a_2 = e_9+e_{10}+e_{11}+e_{12}$ (bold cyan) is a 1-boundary since it is the boundary of the 2-chain c_1. a_1 is a 1-cycle, but not a 1-boundary, which makes $[a_1]$ a non-trivial element of H_1. Therefore a_1 can be used as a representative of the homology class that generates the first homology group of K. Two elements $a, a' \in C_1$ are homologous iff $a - a' = \partial(c)$, for some 2-chain c, denoted as $a \sim a'$. In this case $[a] = [a']$. Here a_1 (bold red) $\sim a_3$ (bold orange).

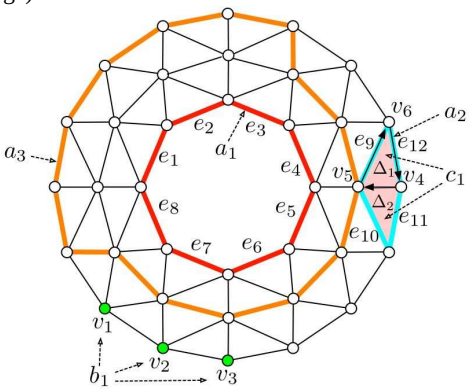

Fig. 4. The triangulation of an annulus. The 1-chain $a_1 = e_1+e_2+...+e_8$ (bold red) is a generator of H_1.

Consider the torus, in the triangulation K in Figure 6 top left, the 1-homology group is generated by the 1-chains a_1 (red) and a_2 (blue), that is, $a_1 = [a,b]+[b,c]+[c,a]$ and $a_2 = [a,d]+[d,e]+[e,a]$. We can verify that both a_1 and a_2 are 1-cycles, as $\partial(a_1) = \partial(a_2) = 0$. They are not 1-boundaries since neither $\partial(b) = a_1$ nor $\partial(b) = a_2$ admit a solution $b \in C_2$. In addition, a_1 and a_2 are not homologous.

Cohomology. Now we associate to K another sequence of groups called *cohomology groups*, whose origins lie in algebra rather than geometry [39]. In many ways, they are considered "dual" to homology groups, and are important in practice.

Consider our example in Figure 4, cohomology deals with functions on 0-, 1- and 2-chain groups. By abuse of notation, each 0-, 1- and 2-dimensional simplex in K corresponds to an *elementary cochain* of the same dimension. For example, the 1-simplex e has a corresponding elementary 1-cochain e^*, which is a function on 1-chain whose value is 1 on e and 0 on all other edges. In other words, $e^* : C_1 \to \mathbb{Z}$, where $e^*(e) = 1$ and $e^*(e') = 0$ for all $e' \in K^1, e' \neq e$. Similarly, we have elementary 0-cochains, v^* associated with the 0-simplices v; and elementary 2-cochains Δ^* associated with the 2-simplices Δ. 0-, 1- and 2-cochains can be considered as sums of elementary cochains, that is,

$$\begin{aligned} C^0 &= C^0(K;\mathbb{Z}) = \{\beta : C_0 \to \mathbb{Z}, \beta = \sum g_i v_i^* \mid g_i \in \mathbb{Z}\}, \\ C^1 &= C^1(K;\mathbb{Z}) = \{\alpha : C_1 \to \mathbb{Z}, \alpha = \sum g_i e_i^* \mid g_i \in \mathbb{Z}\}, \\ C^2 &= C^2(K;\mathbb{Z}) = \{\gamma : C_2 \to \mathbb{Z}, \gamma = \sum g_i \Delta_i^* \mid g_i \in \mathbb{Z}\}. \end{aligned}$$

The boundary maps from homology give us an accessible way to build higher-dimensional cochains from lower-dimensional ones. In order to find a value of a new k-cochain on a k-simplex σ we could compute the values of a known $(k-1)$-cochain on the boundary $\partial\sigma$ and accumulate these values.

This suggests a definition, dual to the boundary map, that we call the *coboundary map*, $\delta_0 : C^0 \to C^1$, $\delta_1 : C^1 \to C^2$,

$$\begin{aligned} (\delta_0\beta)([v_0,v_1]) &= \beta(\partial_1([v_0,v_1]) = \beta(v_1) - \beta(v_0), \\ (\delta_1\alpha)([v_0,v_1,v_2]) &= \alpha(\partial_2([v_0,v_1,v_2])) = \\ &= \alpha([v_1,v_2]) - \alpha([v_0,v_2]) + \alpha([v_0,v_1]). \end{aligned}$$

These notations are convenient in computing coboundaries. For example, If $\alpha = \sum g_i e_i^*$, then $\delta(\alpha) = \sum g_i(\delta e_i^*)$. To compute δe^* for each oriented simplex e, we have $\delta e^* = \sum \varepsilon_j \Delta_j^*$, where the summation

extends over all Δ_j having e as a face, and $\varepsilon_j = \pm 1$ is the coefficient with which e appears in the expression for $\partial\Delta_j$. A similar rule applies to computing δv^*.

In analogy to the treatment of homology above, for a cochain $\alpha \in C^1$, we call α a 1-*cocycle* if $\delta_1(\alpha) = 0$. We call α a 1-*coboundary* if there exists a cochain $\beta \in C^0$ such that $\delta_0(\beta) = 0$. It is easy to verify that $\delta \circ \delta = 0$. 1-coboundaries are always 1-cocycles, we have $\mathrm{im}(\delta_0) \subseteq \ker(\delta_1)$. We define the 1-*cohomology* of K to be the quotient group, $\mathsf{H}^1 = \mathsf{H}^1(K;\mathbb{Z}) = \ker(\delta_1)/\mathrm{im}(\delta_0)$. Two 1-cocycles α and α' are *cohomologous* if $\alpha - \alpha'$ is a coboundary.

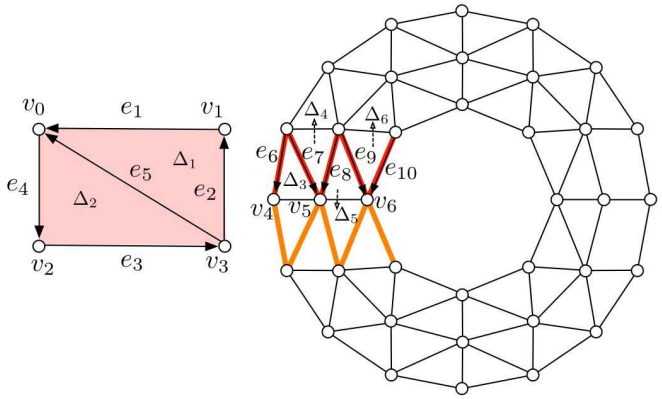

Fig. 5. Left: simple examples of cochains. Right: in this triangulation of an annulus, the 1-cochain $\alpha_1 = e_6^* + e_7^* + e_8^* + e_9^* + e_{10}^*$ is a generator of H^1.

In Figure 5 (left), assuming all triangles are oriented counterclockwise, we compute δe_5^*. $e_5^* : C_1 \to \mathbb{Z}$ has value 1 on e_5 and 0 on other edges. δe_5^* has value -1 on Δ_1 and 1 on Δ_2, because e_5 appears in $\partial\Delta_2$ and $\partial\Delta_1$ with signs $+1$ and -1, respectively. Therefore, $\delta e_5^* = \Delta_2^* - \Delta_1^*$. A similar remark shows that $\delta v_1^* = e_2^* - e_1^*$ and $\delta v_3^* = e_3^* - e_2^* - e_5^*$. The 1-cochain $\alpha = e_1^* + e_5^* - e_3^*$ is a 1-cocycle since $\delta(\alpha) = \delta(e_1^*) + \delta(e_5^*) - \delta(e_3^*) = (\Delta_1^*) + (\Delta_2^* - \Delta_1^*) - (\Delta_2^*) = 0$. Meanwhile, α is also a 1-coboundary since $\alpha = \delta(-v_1^* - v_3^*)$. In terms of generators, in Figure 5 (right), the 1-chain $\alpha_1 = e_6^* + e_7^* + e_8^* + e_9^* + e_{10}^*$ is a 1-cocycle, since $\delta(\alpha_1) = \delta(e_6^*) + .. + \delta(e_{10}^*) = \Delta_3^* + (\Delta_4^* - \Delta_3^*) + (\Delta_5^* - \Delta_4^*) + (\Delta_6^* - \Delta_5^*) - \Delta_6^* = 0$. It is not a 1-coboundary. Therefore, $[\alpha_1] \in \mathsf{H}^1$, and α_1 can be used as the representative of the 1st cohomology class. α_1 (bold red) is cohomologous to α_2 (bold orange), as we can check $\alpha_1 - \alpha_2 = \delta(v_4^* + v_5^* + v_6^*)$.

Consider the torus example in Figure 6 bottom, its 1-cohomology group is generated by the 1-cochains α_1 (red) and α_2 (blue). We can check that both α_1 and α_2 are 1-cocycles, not 1-coboundaries, and are not cohomologous. It is important to note the duality between cohomology and homology generators, which is slightly counter-intuitive. Here, $\alpha_1 \in \mathsf{H}^1$ (bold red) is dual to $a_1 \in \mathsf{H}_1$ (bold red), while $\alpha_2 \in \mathsf{H}^1$ (bold blue) is dual to $a_2 \in \mathsf{H}_1$ (bold blue).

3.2 Local cohomology

The notation $H^1(\mathbb{X},\mathbb{Y})$ is commonly referred to as *relative cohomology*, which is closely connected to the computation of the cohomology groups of the quotient space $\mathbb{X}/\mathbb{Y}$. Intuitively, when gluing all points in $\mathbb{Y}$ to a formally introduced dummy vertex w, any non-trivial topology within $\mathbb{Y}$ is destroyed, ensuring that $\mathsf{H}^1(\mathbb{X},\mathbb{Y})$ only cares about topological features that are in $\mathbb{X}$ and not in $\mathbb{Y}$.

Suppose we have a topological space $\mathbb{X}$. We may define the 1-*dimensional local homology group* of $\mathbb{X}$ at a point $x \in \mathbb{X}$ as $\mathsf{H}_1(\mathbb{X},\mathbb{X}-x)$ [39]. In analogy, the 1-*dimensional local cohomology group* of $\mathbb{X}$ is a corresponding relative cohomology group $\mathsf{H}^1(\mathbb{X},\mathbb{X}-x)$. Fixing some sufficiently small radius r, using the axiom of excision, we can define the above local homology group as $\mathsf{H}_1(\mathbb{X}\cap B_r(x), \mathbb{X}\cap\partial B_r(x))$, where $B_r(x)$ denotes the ball of radius r centered around x, and $\partial B_r(x)$ denotes its boundary. With the same analogy, we may define the 1-dimensional local cohomology group to be $\mathsf{H}^1(\mathbb{X}\cap B_r(x), \mathbb{X}\cap\partial B_r(x))$. This local cohomology group computes topological features of $\mathbb{X}$ within a local neighborhood $B_r(x)$, hence the term *local cohomology*.

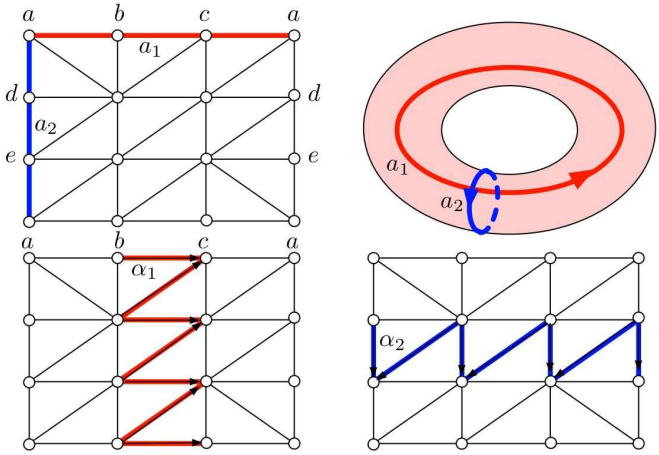

Fig. 6. The triangulation of a torus. Top: $a_1 = [a,b] + [b,c] + [c,a]$ (bold red) and $a_2 = [a,d] + [d,e] + [e,a]$ (bold blue) are the generators of H_1. Bottom: 1-cochains α_1 (bold red) and α_2 (bold blue) are generators of H^1.

In the discretized setting of simplicial complexes, these local cohomology groups are approximated in Section 5.

To put the above formal definition into context, see Figure 7 (left). The space $\mathbb{X}$ is an annulus. Given a point $x \in \mathbb{X}$ and a radius r, we draw a ball of radius r around x. The space that is inside the ball is $\mathbb{X}\cap B_r(x)$ (pink shaded region), and the space that is on the boundary is $\mathbb{X}\cap\partial B_r(x)$ (black). This allows us to compute $\mathsf{H}^1(\mathbb{X}\cap B_r(x), \mathbb{X}\cap\partial B_r(x))$.

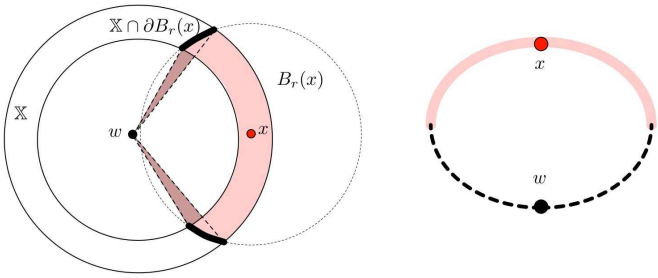

Fig. 7. A simple example of local homology and cohomology. Left: computing local (co)homology through the coning operation. Right: illustration of the coning operation.

3.3 Homotopy theory

As in [15], we rely on the following principle from homotopy theory, which relates circular coordinates with cohomology. Let $[X,\mathbb{S}^1]$ be the set of equivalence classes of continuous maps from space X to $\mathbb{S}^1$ under the homotopy relation. For topological spaces with the homotopy type of a cell complex, there is an isomorphism $\mathsf{H}^1(X;\mathbb{Z}) \cong [X,\mathbb{S}^1]$ [34]. This implies that *if X has a non-trivial 1-dimensional cohomology class $[\alpha] \in \mathsf{H}^1(X;\mathbb{Z})$, we can construct a continuous function $\theta : X \to \mathbb{S}^1$ from a representative α* (see [17] for a formal proof).

We shall assume that we can represent point cloud data X by a simplicial complex K containing vertices, edges and triangles. 1-dimensional cohomology classes consist of integer-valued functions on the edges of this complex. By the isomorphism above, non-trivial global 1-cohomology classes correspond to circular or cyclical structures while local 1-cohomology classes correspond to branching structure in the data underlying the point cloud.

4 Overview

With the technical tools described early, we now give an overview of our algorithm. We detect branching structures by computing local circular coordinates. Given a point cloud X and a point of interest $x \in X$, we choose a subset $U \subseteq X \subset \mathbb{R}^d$ in the neighborhood of x, and output local circular coordinate functions $\theta : U \to \mathbb{S}^1$, that give the

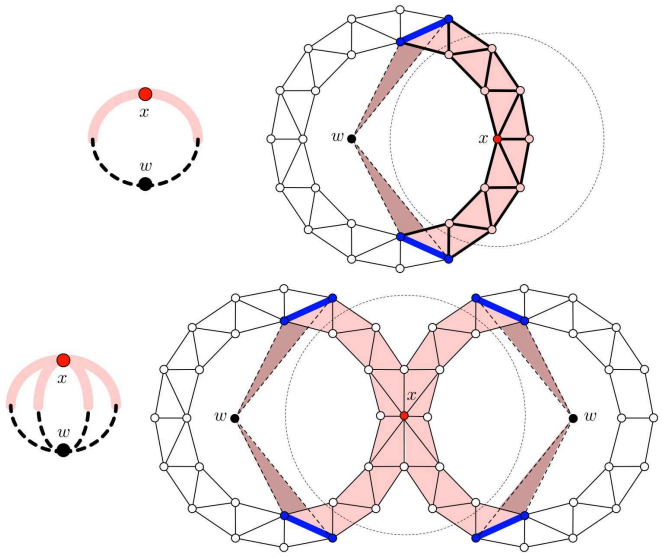

Fig. 8. Top: local cohomology computation in a simplicial setting for the annulus indicating an absence of branching. Bottom: local cohomology computation in a simplicial setting for the figure eight indicating 3-way branching.

values for points in the neighborhood of x. Our overall pipeline is as follows:

1. Compute a simplicial complex K from the point cloud data X in the local neighborhood of x.
2. Use our local persistent cohomology to detect a significant cohomology class $[\alpha_p]$ in the data: $[\alpha_p] \in \mathsf{H}^1(K, \mathbb{F}_p)$, where $\mathbb{F}_p$ is the field of integers modulo a fixed prime p.
3. Lift $[\alpha_p]$ to $[\alpha] \in \mathsf{H}^1(K, \mathbb{Z})$, smooth α to $\bar{\alpha} \in C^1(K, \mathbb{R})$, and integrate $\bar{\alpha}$ to a circle-valued function $\theta : U \to \mathbb{S}^1$.
4. Approximate topological circular and branching structures represented in $\bar{\alpha} \in C^1(K, \mathbb{R})$ in high dimension to aid visualization in a projection.
5. Encode each local circular coordinates with a color map transfer function to highlight true structures and rule out structural illusions.

Here, step 1 adapts and build upon previous work, and step 2 is our own development. Step 3 uses well-established procedures in [15] and step 4 introduces approximations of circular and branching structures to help visualization. We find it useful in practice as demonstrated in Section 6.

Branching behaviors. Branching is a phenomenon we observe in structures that are locally approximately 1-dimensional. We define a *branching point* to be a point at which any path leading in along some locally 1-dimensional structure has some number of different paths leading out again. Consider the example illustrated in Figure 8. The pink regions above and below both decompose into approximately 1-dimensional components. However, in the top figure, there is only one such component leading out, and in the bottom, there are three.

We call a point a *k-way branching point* if it is the point where $k+1$ such 1-dimensional components merge. The fundamental observation in this paper is that we can detect k-way branching by observing a local cohomology group of rank k, and that the resulting local circle-valued coordinates form a basis for the space spanned by the different possible branches. These basis elements often, though not always, immediately correspond to the distinct branches in the data set.

5 Algorithm Details

5.1 Data points to simplicial complexes

We describe step 1 of our algorithm in detail. **Coning operation**. Recall from Section 3.2, to compute local cohomology of a space $\mathbb{X}$ at a point x, we compute $\mathsf{H}^1(\mathbb{X} \cap B_r(x), \mathbb{X} \cap \partial B_r(x))$ for some appropriate fixed radius r. To approximate this situation simplicially, we introduce – for a simplicial complex K representing the point cloud – two new simplicial complexes K' and L'. K' consists of all simplices σ of K such that either some vertices of σ are contained in $B_r(x)$, or σ is a face of such a simplex. L' consists of all simplices σ of K' such that no vertices are contained in $B_r(x)$.

We note that if K is in fact a parametrized family of complexes $K(\varepsilon)$, then we define $K'(\varepsilon)$ and $L'(\varepsilon)$ correspondingly: by picking out appropriate simplices in $K(\varepsilon)$ for the definitions of $K'(\varepsilon)$ and $L'(\varepsilon)$ respectively.

For a simplicial complex K and a formally introduced dummy point w, we define the *cone on K with vertex w*, denoted CK, to be the simplicial complex whose simplices are of the form $[w, v_0, \ldots, v_p]$ or $[v_0, \ldots, v_p]$ for each (possibly empty) simplex $[v_0, \ldots, v_p]$ of K [33]. We call K the *base* of the cone.

Relative cohomology groups can be interpreted as the absolute cohomology groups of an associated simplicial complex [33]. By using the excision theorem, it follows that $\mathsf{H}^1(K', L') \cong \mathsf{H}^1(K' \cup CL')$. This complex $K' \cup CL'$ represents the local structure around x.

Consider the annulus, as triangulated in Figure 8 top right. L' contains all the blue vertices and edges. K' contains all the light shaded triangles, bold edges and solid vertices. The local structure around x can be represented by the simplicial complex $K' \cup CL'$ as indicated in the figure. Similarly, a more involved example is shown in Figure 8 bottom, with a triangulation of a figure eight.

Sequences of simplicial complexes. Given a collection of data points X with some distance metric, such as the L_2, Hamming or Edit distances, there are a number of popular ways to represent the data set as a simplicial complex. We can extract a representation of local information in each of these cases, by constructing $K' \cup CL'$ from K' and L' induced from an appropriate simplicial representation.

While there are ways to generate a single simplicial complex to represent a point cloud, the more useful approach has been to generate a nested family of simplicial complexes [14]. In our work, we use the Vietoris-Rips complex $\mathrm{Rips}(X, \varepsilon)$, which includes a p-simplex for every finite set of $p+1$ points in X with diameter at most ε. Since we are only interested in H^1, we can restrict our attention to the 2-skeleton of $\mathrm{Rips}(X, \varepsilon)$. For a sequence $\varepsilon_1 \le \varepsilon_2 \le \cdots \le \varepsilon_n$, we obtain a nested family of simplicial complexes $\mathrm{Rips}(X, \varepsilon_1) \subseteq \mathrm{Rips}(X, \varepsilon_2) \subseteq \cdots \subseteq \mathrm{Rips}(X, \varepsilon_n)$. For larger data sets, the *witness complex* construction may be more appropriate, as it tends to be smaller [14].

Naïvely we can now compute $K'(\varepsilon_i) \cup CL'(\varepsilon_i)$ induced from $\mathrm{Rips}(X, \varepsilon_i)$ by the conditions stated above. However, it is enough to compute $\mathrm{Rips}(X \cap B_{r+\varepsilon_i}(x), \varepsilon_i)$ instead, since all simplices of $K'(\varepsilon_i) \cup CL'(\varepsilon_i)$ are contained in a $r + \varepsilon_i$ neighborhood of x. The resulting $K(\varepsilon_i) = K'(\varepsilon_i) \cup CL'(\varepsilon_i)$ is a nested family of simplicial complexes that reflects local behavior around x.

5.2 Persistence cohomology in its local version

At step 2 of the algorithm, we are given a nested family of simplicial complexes that represent the local structure at different parameter values ε. We introduce the notion of *scale* for learning this local structure through the concept of *persistence*. Persistence studies the evolution of vectors in a sequence of vector spaces [11]. One main example of such a sequence comes from relating the cohomologies of simplicial complexes reflecting different scales. With persistence we can rank the significance of cohomology classes, and gain robustness in our proposed methods.

Formally, letting $K_i = K(\varepsilon_i)$ as constructed above, we are given a nested family of simplicial complexes connected by inclusions, $\mathscr{K} : K_1 \to K_2 \to \ldots \to K_n$. For $\varepsilon_i \le \varepsilon_j$, the inclusion of spaces $K_i \subseteq K_j$ induces a map between cohomology groups, $f : \mathsf{H}^1(K_j) \to \mathsf{H}^1(K_i)$, and we consider the sequence, $\mathsf{H}^1(K_1) \leftarrow \mathsf{H}^1(K_2) \leftarrow \ldots \leftarrow \mathsf{H}^1(K_n)$. A class $\alpha \in \mathsf{H}^1(K_a)$ is *born* at the time a if it appears for the first time as a cohomology class, and such a class *dies* entering $\mathsf{H}^1(K_b)$ when it disappears as a cohomology class. We call $\varepsilon_b - \varepsilon_a$ the *persistence* of α. We consider classes with high persistence as representing significant topological structure. It is important to note here that we compute $\mathsf{H}^1(K_1) = \mathsf{H}^1(K_1; \mathbb{F}_p)$ with coefficients in $\mathbb{F}_p$. As observed in [50], persistence as a characterization of topological features relies on the coefficients being a field. Computing over a finite field $\mathbb{F}_p$ allows for fast arithmetic, and any *free* cohomology classes over $\mathbb{Z}$ will still be classes over $\mathbb{F}_p$. As described in [15], if the computation detects

Fig. 9. The circular structure on the left has high persistence while the circular structure on the right is considered topological noise [15].

classes that do not correspond to free classes over $\mathbb{Z}$, the smoothing step will fail to converge. In this case, [15] recommend that the user re-run the computation with a different prime. Any classes that occur over both primes are guaranteed to be free.

Intuitively, persistence separates topological features from noise by measuring, in a sense, their size. An illustrative example is shown in Figure 9 where the global circle-valued coordinate function on the left corresponds to a persistent, or significant circular structure, while the circle-valued coordinate function on the right might be considered noise. We give topological methods to detect branching or circular structures in point clouds, and use persistence to detect their significance.

The algorithm that computes persistent cohomology of a sequence of simplicial complexes is a modified version of the persistent homology algorithm [25, 10], which in turn is a variation of the classic Smith normal form algorithm [39]. In a nutshell, it involves a specific ordering of matrix reduction steps on the coboundary matrices of the nested simplicial complexes. After reducing the matrix we obtain a collection of cocycles, each of which is represented as a set of edges together with their function values. For a detailed treatment and discussion of the persistent cohomology algorithm, see [16].

5.3 Lifting, smoothing and integration

For step 3 of our algorithm, we are given a collection of cocycles obtained from step 2. Each cocycle is represented as a collection of edges with coefficients in $\mathbb{F}_p$. We then lift the coefficients first to the integers ($\mathbb{Z}$) and then to the reals ($\mathbb{R}$). Once the cocycle is a real cochain, we can smooth the corresponding function, and then integrate to produce a function from vertices compatible with the given values on edges. These steps are detailed in [15]. Here we shall review some of their key ideas, for easier reference.

Lifting. Given α_p, we lift α_p to α, from $\mathbb{F}_p$ coefficient to integer coefficient. We lift $\alpha_p = \sum n_i e_i^*$ to $\alpha = \sum g_i e_i^*$, where g_i is the unique integer in $[-(p-1)/2, (p-1)/2]$ congruent to n_i.

Smoothing. Given α, we find the "smoothest" cocycle $\bar{\alpha} \in C^1(K;\mathbb{R})$ that is cohomologous to α. By smoothness we mean that $\bar{\alpha}$ has a small total variation defined as $||\bar{\alpha}||^2 = \sum_{e \in K^1} |\bar{\alpha}(e)|^2$. α and $\bar{\alpha}$ are cohomologous if there exists a $f \in C^0(K;\mathbb{R})$ such that $\bar{\alpha} - \alpha = \delta_0 f$. Therefore, we obtain $\bar{\alpha}$ by solving the following minimization problem: $\bar{\alpha} = \arg\min_{\tilde{\alpha}} \{||\bar{\alpha}||^2 \mid \bar{\alpha} - \alpha = \delta_0 f, \exists f \in C^0(K;\mathbb{R})\}$.

Integration. We can integrate $\bar{\alpha}$ brute force, by picking a point on each component (typically including the cone point) to have coordinate value 0, and then setting $\theta(b) = \theta(a) + \bar{\alpha}([a,b])$ to update a vertex b based on the value of a vertex a and an edge $[a,b]$. This defines a function θ on the vertices. Alternatively, as pointed out in [17], such a function θ comes out of the smoothing step above as the f minimizing $\bar{\alpha}$, modulo $\mathbb{Z}$.

5.4 Generator approximations

To aid visualization, we provide two methods to give a fast approximation to $\bar{\alpha}$, the cocycle generator cohomologous to α. Due to the high-dimensionality and complexity of the data, often many data points may map to the same value on $\mathbb{S}^1$, or close in parametrization by a small ε. Therefore tracing out all edges with non-zero coefficient in $\bar{\alpha}$ can lead to a messy visualization when projected onto a lower dimensional space.

One fast and simple approximation of $\bar{\alpha}$ computes a minimum weight cycle which spans bins of values. Here we assume a binning of parametrized values, where points lie in a common bin if their distance (difference in value) is less than a predetermined value. Bins are ordered according to their values with respect to $\mathbb{S}^1$. Between two neighboring bins, we assumes a complete graph where all pair-wise edges across bins are possible, with edge weights reflecting value differences at their end nodes. Our problem reduces to computing a minimum weight cycle across all bins. We demonstrate in Section 6 that even this simple approximation reveals much information on the structure of the parametrization.

On the other hand, we know that $\bar{\alpha}$ must operate on edges of the underlying Vietoris-Rips complex. Therefore we can augment the minimum weight cycle approximation to enforce this constraint.

5.5 Algorithm summary

The algorithm described above detects local 1-cocycles with the following highlights. First, computing local cohomology groups can be approximated by coning operations on Vietoris-Rips complexes. Second, persistent cohomology detects significant features from noise. Third, the above procedure leads to a local circular parametrization that emphasizes branching structure. Last but not least, generator approximations correspond to approximating circular and branching structures in high dimensions to aid visualization. It is important to note that these branching and circular structures are detected in the high dimensional space via cohomology computation. We only use dimensionality reduction techniques overlaying color-mapped coordinate functions to visualize them in their low dimensional projections. Furthermore, we can construct circle-valued coordinate functions locally even if topology is trivial globally.

6 Results

6.1 Software and data sets

The present results are obtained by our implementation of local cohomology computation on top of the C++ library Dionysus [38]. For classic dimension reduction techniques such as Isomap and Laplacian eigenmaps, we use a toolbox from [47].

We construct local and global circle-valued coordinate functions for a variety of synthetic data and real-world examples. Through these experiments, we demonstrate that both the global and local circular coordinates provide a detailed analysis on the intrinsic structure and are beneficial for many applications. Persistence parameters are chosen based on heuristics presented in [15]. Timing information is collected using Intel Xeon 2.67 GHz with 8GB memory.

6.2 Surfaces embedded in $\mathbb{R}^3$

We test our methods on several synthetic data sets with known branching structures. The first data set is a point cloud X sampled from a genus-3 surface as shown in Figure 1 (a). We focus on a point $x \in X$ from one of its four corners and construct local circle-valued coordinates in its neighborhood. Its two-way branching structure is illustrated in 1 (b). We construct their corresponding circle-valued coordinate functions from the point cloud, both of which are shown in Figure 1 (c) and (d).

The second data set is sampled from a genus-4 surface in Figure 1 (e), where seven-way branching exists in the neighborhood of x as indicated in Figure 1 (f). We construct seven corresponding circle-valued coordinate functions, three of which are shown in Figure 1, the rest are shown in Figure 10.

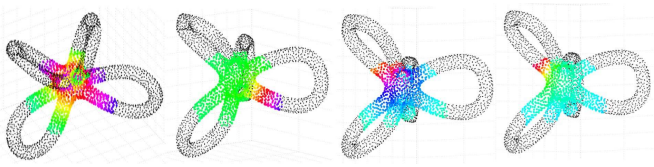

Fig. 10. Genus-4 surface data set, where four of its seven local circle-valued coordinate functions are shown.

6.3 Virus outbreak

We use the VAST 2010 mini challenge data set involving Drafa virus genetic sequences. 58 mutated genetic sequences form a collection of outbreak sequences rooted at the ancestor sequence named *Nigeria*

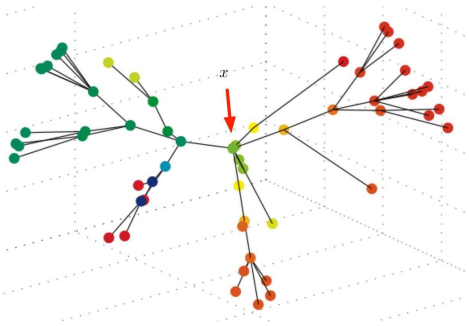

Fig. 11. Local structures among virus genetic sequences. Points are overlaid with a phylogenetic tree. The red arrow points to the ancestor sequence.

B (based on prior knowledge). Each genetic sequence contains 1045 nucleotides, and corresponds to a point in 1045 dimensional space. We focus on studying the local structure surrounding $x =$*Nigeria B*, using circle-valued coordinates and Hamming distance metric. We then embedded all 58 sequences into 3D space, with Multidimensional Scaling. As shown in Figure 11, local circle-valued coordinates reveal the branching structures surrounding *x*. To aid the visualization of the color-coded circle-valued coordinate function, we overlay the points with the phylogenetic tree among these sequences. This local cocycle took 0.08 seconds to compute (0.08 for Rips calculation, persistence calculation timing is negligible).

6.4 Motion capture data

For this example, we construct both global and local circular coordinates on a couple of motion capture data sets freely distributed online. Motion capture data is the recorded movement of a live actor over time. In the following we show that there are interesting features captured by our methods that are worth further investigation. For our testing, we have analyzed motion capture data saved in the Biovision BVH format. BVH is a hierarchical set of relative joint angles rooted at a node that is traditionally centered at the hips of the live actor. All translational motion of the actor in world space is also encoded at this node, to give a complete representation of the motion. Since the space for translational motion is small, finding circular structure due to this motion should not be difficult. A much more interesting problem is finding structure and correlation in the joint angles, therefore the translational motion is ignored in our testing.

Local illusions: walk, hop and walk. The first data set from OSU Motion Capture Lab data repository [35] involves a female actor walking, hopping over an obstacle and then walking again. It contains 189 frames with 66 joint angles per frame. For our tests, each frame is considered a point in 66 dimensional space with a L_2 distance metric. Figure 12 shows this point set embedded onto 3D Euclidean space using Laplacian eigenmaps. This 3D embedding appears to reveal some branching structure, denoted in the top image of this figure. Contrary to this, the local circle-valued coordinate functions computed in high dimension indicate that it is a visual illusion, shown in the rest of images. This example emphasizes the fact that traditional dimensionality reduction can introduce structure where none is present. Local circle-valued coordinates do not suffer from this flaw and therefore can help uncouple illusions from actual structure in visualizations. This local cocycle took 0.08 seconds to compute (0.08 for Rips calculation, persistence calculation timing is negligible).

Ballet dancer. The Ballet dancer data set, obtained from [1], involves a ballet dancer performing a traditional stretch. The dataset contains 471 frames of 54 joint angles. A sub-sampled (every 2nd frame) compilation of the frames is shown in Figure 13 top. In this figure, time flows in row-major order first from left to right then top to bottom by rows. We construct global circle-valued coordinate functions on the data. The parameterized value bins for the second largest 1-cocycle in terms of persistence is drawn over the frames of Figure 13 top. As this shows, the cycles coincide with the points of the motion data where there is hesitation or pauses before a point of inflection in the arm movement, see Figure 13 bottom right. Additionally, the global-circular coordinates give a binning of similar motions in this

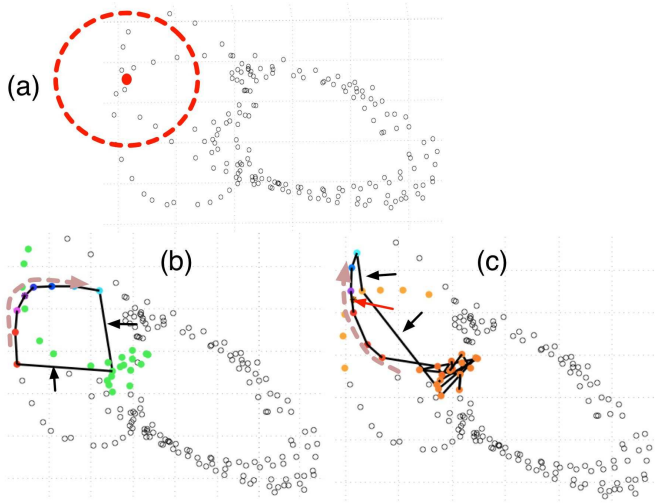

Fig. 12. Motion capture sequences from a female actor walking, hopping over an obstacle and then walking again. (a): Laplacian eigenmaps appears to reveal a branching structure in the local neighborhood of a point, marked in red. (b) and (c): when we visualize local-circle valued coordinate functions in the same neighborhood, we obtain two independent parametrizations. This indicates that no branching structures exist in that local neighborhood. We approximate the generators associated with the local cohomology classes to aid visualization of the circular structure. The approximated generator travels along a given branch where the color indicates its direction. Notice that there are extra edges pointed by arrows that are artifacts from the approximation. The red arrow in (c) shows that our sample point no longer has a distinct value in the parametrization.

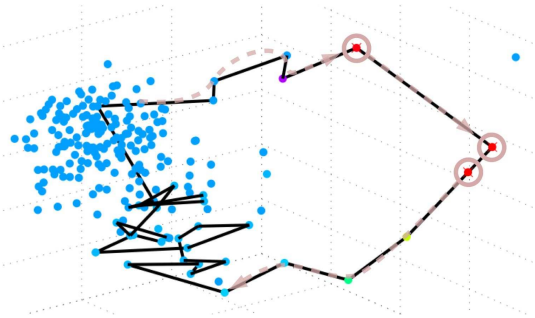

Fig. 14. 1995 voting data for the Democratic party affiliated members of the House of Representatives. We visualize the top global circle-valued coordinate function through a transfer function and with an approximated cocycle. The circled points indicate representatives who switched parties or resigned.

parametrization. The bins highlighted in blue in this figure share a common parametrization and, as the frames show, have very similar arm movement. Finally, Figure 13 bottom left shows the Laplacian eigenmaps and isomap projection of the joint angles on 3D Euclidean space, left and right respectively. This example emphasizes the fact that without the circular-coordinate parametrization, it would be impossible to infer this cycle from the embedded points alone. This cycle took 417.38 seconds to compute (363.67 for Rips calculation, 30.47 for persistence calculation).

6.5 Voting

Consider the House of Representatives of the US Congress. In a given session, a number of votes are being cast on proposed bills in the House, and a simple numeric scheme can be used to assign values $+1, -1, 0$ to votes of Yea, votes of Nay and absence of a vote. This yields a point cloud of representatives in a vector space of roll-calls. We study vote records extracted from `house.gov` that cover the voting behaviors of Democrats in the House of Representatives

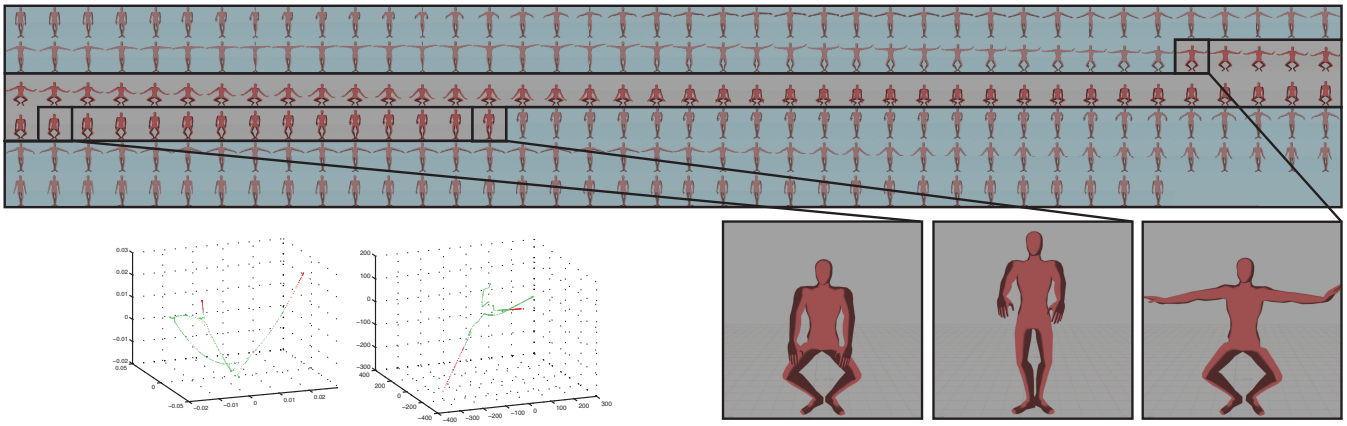

Fig. 13. Motion capture sequences from a Ballet dancer where global circle-valued coordinates reveal moments of hesitation or pauses before a point of inflection in the arm movement.

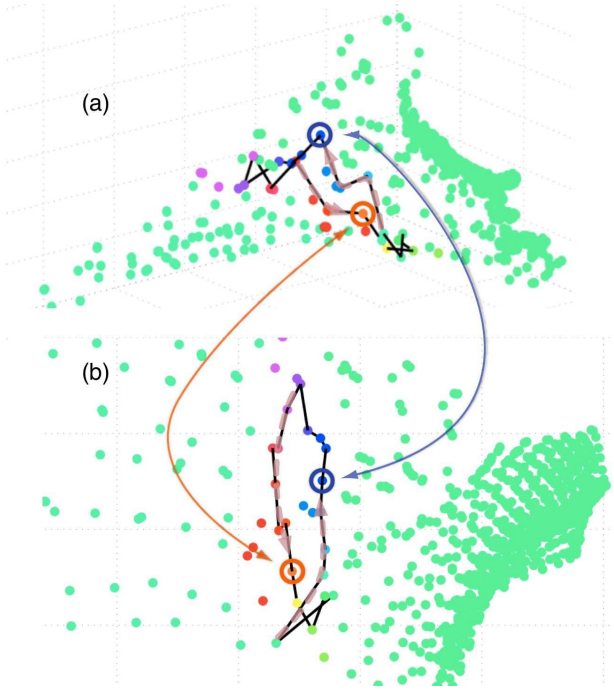

Fig. 15. Data from a combustion simulation. (a): 3D Isomap embedding of the circular structure with the highest persistence. We aid the visualization using an approximated cocycle generator. (b): 3D rotated view with approximated generator to guide visualization. This circular structure can be used to discover correlations between parameters. Two points are linked between each view to aid visualization.

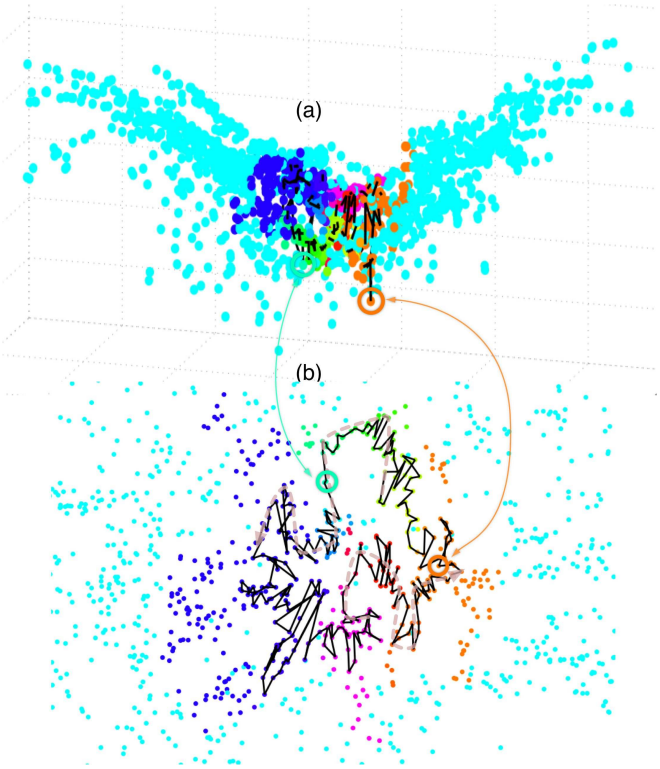

Fig. 16. Climate data: the second most significant circle-valued coordinate function (a) overlaid with approximated cocycle generator (b). Two points are linked between each view to aid visualization.

during the year 1995. In this period, 205 congressmen voted on 885 different issues. Using a Hamming distance metric, we constructed global circle-valued coordinates with respect to significant non-trivial 1-cocycles. We then embedded all 205 points into 3D using Multidimensional Scaling.

In Figure 14, we see that the most significant circular structure is formed using representatives Deal (D-GA), Laughlin (D-TX), and Reynolds (D-IL), who switched political affiliation or resigned during the year (marked with red arrows in the figure), as well as two tails from the core of the point cloud. This circular structure likely indicates that in addition to resignation or party switches, there are several modes in which a representative can differ from the mainstream of their party. Beyond not agreeing with their own party, representatives can seem absent through having heavy political duties or being absent from their work for other reasons. This cycle took 94.27 seconds to compute (92.15 for Rips calculation, 1.76 for persistence calculation).

6.6 Combustion and climate simulations

Advanced modeling and simulation tools have been used recently to reduce the need for large, expensive integrated physical and chemical experiments. As these simulations become more sophisticated and accurate, it becomes crucial to estimate the likelihood of a given prediction and quantify its uncertainty. Topological tools have been helpful in handling parameter choices and an analysis of the parameter spaces related to a simulation [32]. We believe that topological tools in general, and the analysis of global circular structures in particular, are helpful as feedback for domain scientists that provide extra information useful for uncertainty quantification and sensitivity analysis. We demonstrate here that within simulation data from both combustion simulations and climate simulations, there are significantly large circular structures present.

Combustion simulation. We use a subset of points from a combustion simulation and analyze its output parameter space. Points are located on a 16 by 16 grid with 4 simulation time steps. Each point is considered 16 dimensional, including parameters such as mixture fraction,

dissipation rate, heat release rate and temperature. We embedded the points into 3D using Isomap and visualized the significant circular features in the data. This is shown in Figure 15. This cycle took 17.04 seconds to compute (11.66 for Rips calculation, 2.96 for persistence calculation).

Climate simulation. We are also interested in finding features in the output parameter space of a climate simulation. This data set comprises 1612 simulation points, each with 8 output parameters, including total cloud percentage, precipitation rate, sea-level pressure, surface stress and temperature . We construct global circle-valued coordinate functions and visualize them in Landmark Isomap 3D projection. This is shown in Figure 16, where the significant circular structure is concentrated at the basin of the projection. This cycle took 0.09 seconds to compute (0.05 for Rips calculation, 0.02 for persistence calculation). For this particular cycle, its birth time is low compared to the range of function values. Therefore, the calculation is very efficient due to needing only a small Rips complex to fully capture the cocycle.

These global circle-valued coordinates indicate potential non-local correlations among simulation parameters since they indicate high dimensional path along which they may be changed in different ways to achieve the same conditions. This type of non-local correlations are challenging to find and provide insight to the scientists on the complexities of their simulations. We will work with domain scientists to validate the consequences that the emergence of such cycles implies.

6.7 Feasibility, performance and limitations

We shall discuss the runtime complexities of each step in our algorithm:

- Step 1: computing the 2-skeleton of the Vietoris-Rips complex. For $\varepsilon = \infty$, the complex has $O(v^3)$ simplices (by choosing all possible triangles), where v is the number of vertices. Generating the complex takes time worst case $O(v^3)$, this is however rare in practice [49]. Usually, ε is chosen with prior knowledge of the problem domain to be just large enough to detect the topology. This decreases the above bound to an expected linear or even constant behavior. When computing local cohomology, we only need to construct the complex in the $(r+\varepsilon)$-neighborhood of a center point, which radically reduces the v generating runtime complexity.
- Step 2: computing (local) persistence cohomology. The persistence algorithm runs in time $O(n^3)$, where n is the number of simplices [13]. It shows a roughly linear behavior in practice [6].
- Step 3: solving a LSQR optimization problem, which generally has low-complexity and is unlikely the performance bottleneck.

The ambient dimension plays a role in the computation of metrics, possibly as a pre-processing step. The rest of the algorithm is independent of the ambient dimension (or even the notion of a well-defined embedding space).

In terms of locating points of interest for branching detection, we assume that the prior knowledge suggests a set of candidate points. If not, the computation can be performed on a random sample, retaining any points that show an unexpected high rank for its dimension-1 local cohomology.

Our computation includes two user-specified parameters: the size of the neighborhood r and the maximal bound on the persistence parameter ε. While ε can be chosen as indicated in [15], keeping in mind that a too small r will miss the local structure, and a too large r may involve disjoint parts of the data set.

Our method captures continuous occurrences of circular structures (or branching structures) within a given data set. If the data set does not reflect such structures, that is, its 1-dimension (local) cohomology has rank 0, then no continuous circle-valued functions exist.

7 Discussions

We consider our work as a first step towards a more ambitious goal of combining dimensionality reduction with local or regionalized topological analysis of intrinsic structure. Our local circle-valued coordinate functions are similar to local dimensionality estimation in a way that they reflect detailed structural information which might have global effects. We ask the following questions: how do the local and global structures of data interact with one another? How can local analysis infer global structure? There are various open questions, and we address a few here.

Shortest local cocycle. Many algorithms exist to compute 1-cycles with geometric constraints, such as shortest by length or minimum by weight. Are these algorithm extendable to compute the shortest (local) 1-cocycles? The smoothing step described earlier obtains a 1-cocycle with minimum total variance. While persistent homology computes representative homology-generating cycles, these cycles can fluctuate drastically due to changes in the filtration or in the simplicial complex. Work in [9] tracks these cycles so that the changes are local with temporal coherence. We believe this line of work can be extended to (local) persistence cohomology computations.

Extending local parametrization. Using our algorithm, a point set $U \subseteq X$ in the r-neighborhood of $x \in X$ is parameterized by a circle-valued coordinate function $\theta : U \to \mathbb{S}^1$. We can extend such a parametrization by gradually increasing r until non-trivial topological changes take place. That is, we can extend $\theta : U \to \mathbb{S}^1$ to $\theta' : U' \to \mathbb{S}^1$ where $U \subseteq U'$. We can also obtain a partial ordering of all points in X by concatenating multiple local parametrizations. That is, given two circle-valued functions $\theta_1 : U_1 \to \mathbb{S}^1$ and $\theta_2 : U_2 \to \mathbb{S}^1$, where $U_1 \cap U_2 \neq \emptyset$, it might be possible to construct a gluing $\theta : U_1 \cup U_2 \to \mathbb{S}^1$ in a coherent manner. The notion of 1-cocycle is not only important in our context of circular coordinates, but also shows up in data ranking and discrete vector fields [8]. Does a total partial ordering obtained from "gluing" local 1-cocycles play a role in data ranking?

Computation efficiency. To guarantee theoretical correctness in computing local (co)homology, we need to use the Delaunay complex as detailed in [5, 7]. However it is impractical to compute Delaunay complexes in high dimensions. We believe that using Vietoris-Rips or witness complexes to compute local cohomology in high dimensions is the best available option. In particular, methods for fast constructions of Vietoris-Rips complexes have become available [49], and there has been theoretical advancement on the topology-preserving qualities of Vietoris-Rips complexes [3]. Efficient data structure for representing and simplifying simplicial complexes in high dimension has been proposed [2]. Other proximity graph constructions, such as the k-nearest neighbor graph or the Gabriel graph might be employed as well. The correctness guarantee associated with these constructions remains as an open question.

Visualizing branching and circular structures. As shown in Figure 15 and Figure 16, the default projection and viewing angles can not visualize the circular structures clearly. We obtain better visualization on these structures by approximating cocycles in high dimensional space and choose proper viewing angles, using the algorithm presented in Section 5.4. However, augmenting a low-dimensional projection with color maps still suffers visual drawbacks as some structural information can be hidden. We believe it is an interesting open question to develop visualization techniques that preserve and emphasize topological structures recovered in high-dimensions. In other words, we would like to develop topology-driven or feature-optimized projection in terms of found branches or cycles.

Acknowledgments

Thanks to the National Center for Atmospheric Research (NCAR) for the climate data and Jacqueline H. Chen of the Combustion Research Facility (CRF) and Sandia National Laboratories for the combustion data. Thanks to Peer-Timo Bremer, Oliver Ruebel, A.N.M. Imroz Choudhury, Paul Rosen, Vaidyanathan Krishnamoorthy, Attila Gyulassy, Joshua Levine, Wathsala Widanagamaachchi, for valuable discussions. BW, BS and VP were supported in part by National Science Foundation awards IIS-0904631, IIS-0906379, and CCF-0702817. This work was also performed under the auspices of the U.S. Department of Energy by the University of Utah under contract DE-SC0001922 and DE-FC02-06ER25781 and by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344.LLNL-JRNL-453051. This work was supported in part by the DOE Occie of Science, Biology and Environment (BER), and the Scientific Discovery through Advanced Computing (SciDAC) programs Visualization and Analytics Center for Enabling Technologies (VACET). MVJ was partially supported by the Office of Naval Research, through grant N00014-08-1-0931.

REFERENCES

[1] Free motion capture. http://gfx-motion-capture.blogspot.com/.

[2] D. Attali, A. Lieutier, and D. Salinas. Efficient data structure for representing and simplifying simplicial complexes in high dimensions. *Proceedings 27th Annual ACM Symposium on Computational Geometry*, pages 501–509, 2011.

[3] D. Attali, A. Lieutier, and D. Salinas. Vietoris-Rips complexes also provide topologically correct reconstructions of sampled shapes. *Proceedings 27th Annual ACM Symposium on Computational Geometry*, pages 491–500, 2011.

[4] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15:1373–1396, 2003.

[5] P. Bendich, D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and D. Morozov. Inferring local homology from sampled stratified spaces. *Proceedings 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 536–546, 2007.

[6] P. Bendich, H. Edelsbrunner, and M. Kerber. Computing robustness and persistence for images. *IEEE Transactions on Visualization and Computer Graphics*, 16:1251–1260, 2010.

[7] P. Bendich, B. Wang, and S. Mukherjee. Towards stratification learning through homology inference. *AAAI 2010 Fall Symposium on Manifold Learning and its Applications*, 2010.

[8] D. Burghelea and T. K. Dey. Defining and computing topological persistence for 1-cocycles. Manuscript, December 2010.

[9] O. Busaryev, T. K. Dey, and Y. Wang. Tracking a generator by persistence. *Discrete Mathematics, Algorithms and Applications*, 2(4):539–552, 2010.

[10] G. Carlsson, A. J. Zomorodian, A. Collins, and L. J. Guibas. Persistence barcodes for shapes. *Proceedings Eurographs/ACM SIGGRAPH Symposium on Geometry Processing*, pages 124–135, 2004.

[11] F. Chazal, D. Cohen-Steiner, M. Glisse, L. J. Guibas, and S. Y. Oudot. Proximity of persistence modules and their diagrams. *Proceedings 25th Annual Symposium on Computational Geometry*, pages 237–246, 2009.

[12] C. Chen and D. Freedman. Quantifying homology classes. *Proceedings 25th International Symposium on Theoritical Aspects of Computer Science*, 1:169–180, 2008.

[13] C. Chen and M. Kerber. Persistent homology computation with a twist. *Proceedings 27th European Workshop on Computational Geometry*, 2011.

[14] V. de Silva and G. Carlsson. Topological estimation using witness complexes. *Symposium on Point-Based Graphics*, pages 157–166, 2004.

[15] V. de Silva, D. Morozov, and M. Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Proceedings 25th Annual Symposium on Computational Geometry*, pages 227–236, 2009.

[16] V. de Silva, D. Morozov, and M. Vejdemo-Johansson. Dualities in persistent (co)homology. Manuscript, 2010.

[17] V. de Silva, D. Morozov, and M. Vejdemo-Johansson. Persistent cohomology and circular coordinates. *Discrete & Computational Geometry*, pages 1–23, 2011.

[18] T. K. Dey, A. N. Hirani, and B. Krishnamoorthy. Optimal homologous cycles, total unimodularity, and linear programming. *Proceedings 42nd ACM Symposium on Theory of Computing*, pages 221–230, 2010.

[19] T. K. Dey, K. Li, J. Sun, and D. Cohen-Steiner. Computing geometry-aware handle and tunnel loops in 3D models. *SIGGRAPH*, 45:1–9, 2008.

[20] T. K. Dey, J. Sun, and Y. Wang. Approximating loops in a shortest homology basis from point data. *Proceedings Annual Symposium on Computational Geometry*, pages 166–175, 2010.

[21] M. Dixon, N. Jacobs, and R. Pless. Finding minimal parametrizations of cylindrical image manifolds. *Proceedings Conference on Computer Vision and Pattern Recognition Workshop*, page 192, 2006.

[22] P. Dlotko. A fast algorithm to compute cohomology group generators of orientable 2-manifolds. *3rd International Workshop on Computational Topology in Image Context*, 2010.

[23] H. Edelsbrunner and J. Harer. Persistent homology - a survey. *Contemporary Mathematics*, 453:257–282, 2008.

[24] H. Edelsbrunner and J. Harer. *Computational Topology: An Introduction*. American Mathematical Society, Providence, RI, USA, 2010.

[25] H. Edelsbrunner, D. Letscher, and A. J. Zomorodian. Topological persistence and simplification. *Discrete and Computational Geometry*, 28:511–533, 2002.

[26] Éric Colin de Verdière and F. Lazarus. Optimal system of loops on an orientable surface. *Discrete Computational Geometry*, 33:627–636, 2005.

[27] J. Erickson, E. W. Chambers, and A. Nayyeri. Homology flows, cohomology cuts. *Proceedings 41st Annual ACM Symposium on Theory of Computing*, pages 273–282, 2009.

[28] J. Erickson, E. W. Chambers, and A. Nayyeri. Minimum cuts and shortest homologous cycles. *Proceedings 25th Annual ACM Symposium on Computational Geometry*, pages 377–385, 2009.

[29] J. Erickson and K. Whittlesey. Greedy optimal homotopy and homology generators. *Proceedings 16th Annual ACM-SIAM symposium on Discrete Algorithms*, pages 1038–1046, 2005.

[30] J. Erickson and P. Worah. Computing the shortest essential cycle. *Discrete and Computational Geometry*, 2010.

[31] R. K. Gabriel and R. R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18(3):259–278, 1969.

[32] S. Gerber, P.-T. Bremer, V. Pascucci, and R. Whitaker. Visual exploration of high dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1271–1280, 2010.

[33] P. Giblin. *Graphs, Surfaces and Homology*. Cambridge University Press, New York, NY, USA, 2010.

[34] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[35] O. M. C. Lab. Motion capture data sets.

[36] N. D. Lawrence. Probabilistic spectral dimensionality reduction. Manuscript, 2010.

[37] J. A. Lee and M. Verleysen. Nonlinear dimensionality reduction of data manifolds with essential loops. *Neurocomputing*, 67:29–53, 2005.

[38] D. Morozov. Dionysus library for computing persistent homology.

[39] J. R. Munkres. *Elements of algebraic topology*. Addison-Wesley, Redwood City, CA, USA, 1984.

[40] P. Oesterling, C. Heine, H. Jänicke, and G. Scheuermann. Visual analysis of high dimentional point clouds using topological landscapes. *IEEE Pacific Visualization Symposium*, pages 113–120, 2010.

[41] P. Oesterling, G. Scheuermann, S. Teresniak, G. Heyer, S. Koch, T. Ertl, and G. H. Weber. Two-stage framework for a topology-based projection and visualization of classified document collections. *Proceedings IEEE Visual Analytics Science and Technology*, pages 91–98, 2010.

[42] R. Pless and I. Simon. Embedding images in non-flat spaces. In *Conference on Imaging Science Systems and Technology*, pages 182–188, 2002.

[43] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[44] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998.

[45] S. Takahashi, I. Fujishiro, and M. Okada. Applying manifold learning to plotting approximate contour trees. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1185–1192, 2009.

[46] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[47] L. van der Maaten. Matlab toolbox for dimensionality reduction. http://homepage.tudelft.nl/19j49/.

[48] G. Weber, P.-T. Bremer, and V. Pascucci. Topological landscapes: A terrain metaphor for scientific data. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1416–1423, 2007.

[49] A. J. Zomorodian. Fast construction of the Vietoris-Rips complex. *Computers & Graphics*, 34(3):263–271, 2010.

[50] A. J. Zomorodian and G. Carlsson. Computing persistent homology. *Discrete and Computational Geometry*, 33(2):249–274, 2005.

[51] A. J. Zomorodian and G. Carlsson. Localized homology. *Computational Geometry: Theory and Applications*, 2008.