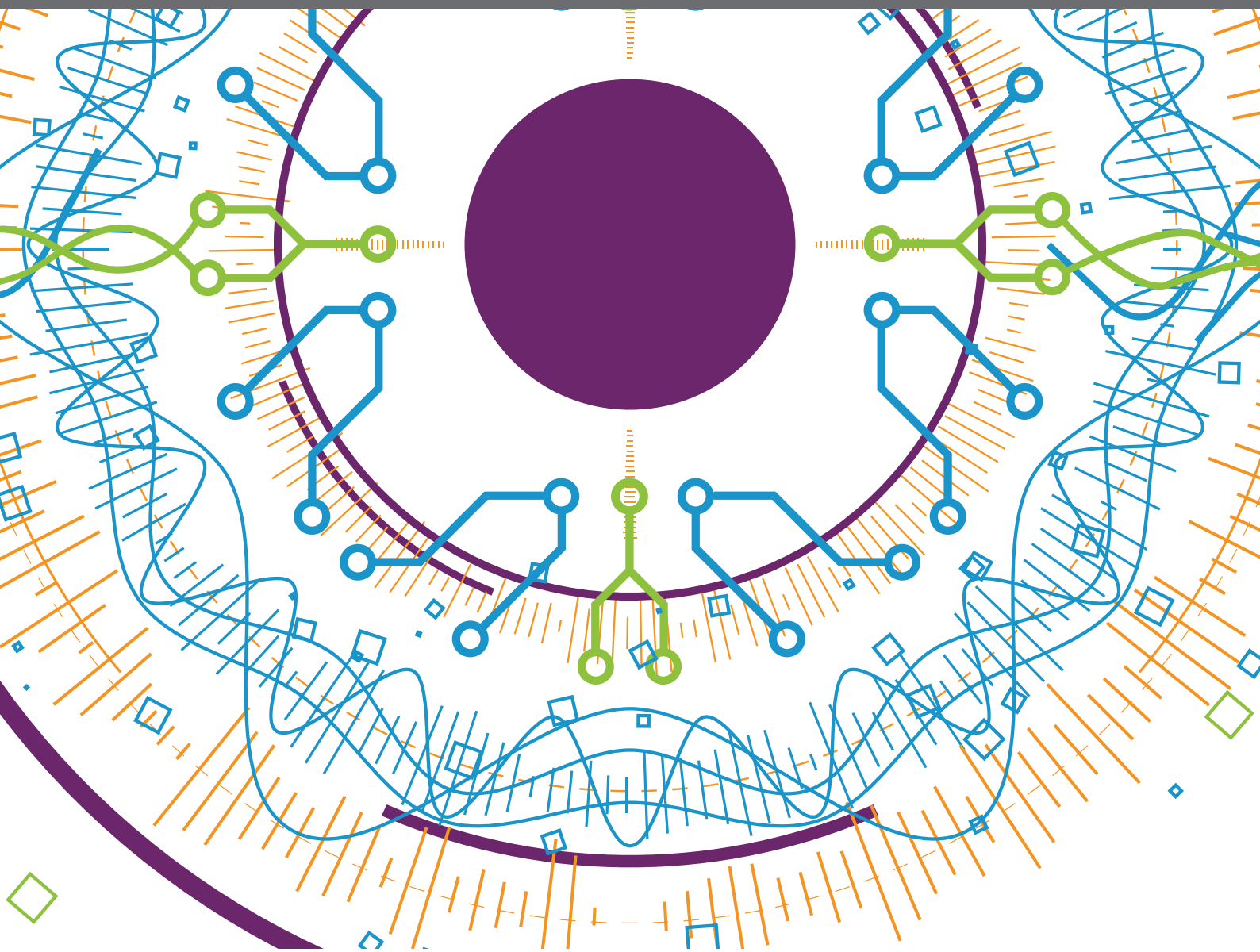
TOPOLOGY IN REAL-WORLD MACHINE LEARNING AND DATA ANALYSIS

EDITED BY: Kathryn Hess, Frédéric Chazal and Umberto Lupo
PUBLISHED IN: Frontiers in Artificial Intelligence, Frontiers in Big Data and Frontiers in Applied Mathematics and Statistics



frontiers Research Topics

frontiers

ISSN 1664-8714
ISBN 978-2-83250-412-3
DOI 10.3389/978-2-83250-412-3

About Frontiers

Frontiers is more than just an open-access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers Journal Series

The Frontiers Journal Series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the Frontiers Journal Series operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to Quality

Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews.
Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view. By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the Frontiers Journals Series: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area! Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers Editorial Office: frontiersin.org/about/contact

TOPOLOGY IN REAL-WORLD MACHINE LEARNING AND DATA ANALYSIS

Topic Editors:
Kathryn Hess, Swiss Federal Institute of Technology Lausanne, Switzerland
Frédéric Chazal, Inria Saclay - Île-de-France Research Centre, France
Umberto Lupo, Swiss Federal Institute of Technology Lausanne, Switzerland

Citation: Hess, K., Chazal, F., Lupo, U., eds. (2022). Topology in Real-World Machine Learning and Data Analysis. Lausanne: Frontiers Media SA.
doi: 10.3389/978-2-83250-412-3

Table of Contents

frontiers in Artificial Intelligence

REVIEW
published: 13 April 2021
doi: 10.3389/frai.2021.659037

Applications of Topological Data Analysis in Oncology

Anuraag Bukkuri[1]*, *Noemi Andor*[1] *and Isabel K. Darcy*[2]

[1] Department of Integrated Mathematical Oncology, Moffitt Cancer Center, Tampa, FL, United States, [2] Department of Mathematics, University of Iowa, Iowa City, IA, United States

The emergence of the information age in the last few decades brought with it an explosion of biomedical data. But with great power comes great responsibility: there is now a pressing need for new data analysis algorithms to be developed to make sense of the data and transform this information into knowledge which can be directly translated into the clinic. Topological data analysis (TDA) provides a promising path forward: using tools from the mathematical field of algebraic topology, TDA provides a framework to extract insights into the often high-dimensional, incomplete, and noisy nature of biomedical data. Nowhere is this more evident than in the field of oncology, where patient-specific data is routinely presented to clinicians in a variety of forms, from imaging to single cell genomic sequencing. In this review, we focus on applications involving persistent homology, one of the main tools of TDA. We describe some recent successes of TDA in oncology, specifically in predicting treatment responses and prognosis, tumor segmentation and computer-aided diagnosis, disease classification, and cellular architecture determination. We also provide suggestions on avenues for future research including utilizing TDA to analyze cancer time-series data such as gene expression changes during pathogenesis, investigation of the relation between angiogenic vessel structure and treatment efficacy from imaging data, and experimental confirmation that geometric and topological connectivity implies functional connectivity in the context of cancer.

Keywords: topological data analysis, persistent homology, oncology, single cell analysis, imaging, clonal evolution, tumor heterogeneity

OPEN ACCESS

Edited by:
Umberto Lupo,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Bala Krishnamoorthy,
Washington State University Vancouver, United States
Pablo G. Camara,
University of Pennsylvania, United States

***Correspondence:**
Anuraag Bukkuri
anuraag.bukkuri@moffitt.org

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *26 January 2021*
Accepted: *16 March 2021*
Published: *13 April 2021*

Citation:
Bukkuri A, Andor N and Darcy IK (2021) Applications of Topological Data Analysis in Oncology. Front. Artif. Intell. 4:659037. doi: 10.3389/frai.2021.659037

1. INTRODUCTION

With the advent of next-generation high-throughput sequencing (Roychowdhury et al., 2011; Reuter et al., 2015), improved medical imaging (Wang, 2016; Tahmassebi et al., 2018; Aiello et al., 2019), and an increased focus on personalized medicine (Dilsizian and Siegel, 2014; Gu and Taylor, 2014; Alyass et al., 2015; Suwinski et al., 2019), more data is being collected than ever before. Efficient data analysis techniques are critically needed to convert this data into meaningful, clinically translatable information. Topological data analysis (TDA) focuses on the shape of data, identifying both local and global structures at multiple scales. Consider a trivial example: suppose data points lie on a circle. The data points could represent customers' preferences or patient gene expression. In this case if a product or drug were targeted to the average person, the target would be the center of the circle and would thus miss the data set entirely. While this is a simple made-up example, it illustrates the importance of understanding the shape of data. TDA can be applied to high-dimensional and noisy data. While the output of TDA can be affected by incomplete data, it is still effective at distinguishing between data sets that have different shapes.

TDA has been successfully applied in a variety of medical contexts including to discover phenotype-biomarker associations in traumatic brain injury (Nielson et al., 2017), identify diagnostic factors for pulmonary embolism (Rucco et al., 2015), discriminate between healthy patients and those with diabetic retinopathy from retinal imaging (Garside et al., 2019), map human recombination at fine scales (Camara et al., 2016), identify novel pathological phenotypes of asthma (Siddiqui et al., 2018), and characterize the structure of chromatin conformation inside the nucleus (Emmett et al., 2016). In this review, we shall focus our attention on some recent applications of persistent homology, a main tool of TDA, to oncology. We specifically discuss treatment responses, clinical outcomes, disease classification, biomarker identification, and cellular architecture in cancer. We will also provide insights into possible future fruitful avenues of research, including analysis of time-series data to help with disease classification and identification of selection events, investigation of the relation between angiogenic vessel structure and treatment efficacy from imaging data, and experimental confirmation that geometric and topological connectivity implies functional connectivity in the context of cancer. Though we focus on persistent homology here, it is worth noting that there have been many notable successes of the application of other TDA methods, such as the *Mapper* algorithm (Singh et al., 2007). For example, *Mapper* was recently used to extract information from high-throughput microarray data and define a new subtype of breast cancer, c-MYB+, characterized by high c-MYB expression and low levels of innate inflammatory genes, with corresponding patients exhibiting 100% survival and no metastasis (Nicolau et al., 2007). In another study, *Mapper* was used to discover 38 new cancer-associated genes across tumor types, some of which were then confirmed to play a key role in tumorigenesis in mouse models (Rabadán et al., 2020). Before delving into the applications of persistent homology in cancer, we introduce some of the key mathematical underpinnings needed to understand these results.

2. WHAT IS PERSISTENT HOMOLOGY?

The mathematical definition of homology/homologous is very precise and often differs from the English common usage. Homology uses algebra to detect topological shapes. Topology is sometimes called rubber sheet geometry as two objects are topologically equivalent to each other if one can be deformed into the other without tearing or puncturing the objects. For example, the spherical and cubical surfaces are topologically equivalent per **Figure 1A**. The sphere is topologically different from the 3-dimensional ball that the sphere bounds. Homology detects this difference by noting that the 2-dimensional spherical surface bounds a void while the 3-dimensional ball is solid and thus does not bound any voids.

To describe homology, we will first focus on two quantities: β_0 = the number of connected components and β_1 = the number of 1-dimensional holes (a circle that has not been filled in). One does not need to understand the algebra of homology in order to understand the basics of persistent homology, thus we will only briefly introduce some concepts for the interested reader. Two points are homologous if they are in the same connected component. Thus, $\beta_0 = 1$ if the object is connected. To describe β_1, we will focus on **Figure 1B**. We can use addition to represent topological objects. For example, the rectangle in **Figure 1** is represented by the sum of edges: $e_1 + e_2 + e_3 + e_4$. Two 1-dimensional cycles are homologous to each other if they form the boundary of a surface. Thus, the rectangle is homologous to the cycle $e_5 + e_8 + e_{10} + e_{11}$ since these two cycles bound the green surface. The cycles $e_5+e_6+e_7+e_8$ and $e_9+e_{10}+e_{11}+e_{12}$ are also homologous since they bound the light green surface consisting of two crescent moons. In fact all these cycles are homologous to the rectangle $e_1+e_2+e_3+e_4$. One can see that this object contains many cycles, many of which are homologous to the rectangle (or a multiple of the rectangle, for example, $\sum_{i=5}^{12} e_i$ is homologous to $2\sum_{i=1}^{4} e_i$). A 1-dimensional cycle is homologous to 0 if it bounds a surface. Thus the cycles $e_5 + e_9 + e_{12} + e_8$ and $e_6 + e_7 + e_{11} + e_{10}$ are both homologous to 0 since they each form the boundary of a surface (the two crescent moons, waning or waxing, respectively). Since each of the cycles in this figure are homologous to 0 or to a multiple of the rectangle, its homology is generated by a single cycle (for example, the rectangle) and thus $\beta_1 = 1$.

The intuitive definition of homology is that β_n equals the number of $n-$dimensional holes[1]. Per the **Figure 1** caption, homology can be used to distinguish the following objects from each other: solid ball, sphere, higher dimensional balls and spheres, solid torus, and torus. Homology cannot distinguish all objects that are topologically different. For example,the 1-dimensional circle, the 2-dimensional surface in **Figure 1B**, and the 3-dimensional solid torus (**Figure 1C**) all have the same homology. For more on the mathematical definition of homology (please see Munkres, 1984; Hatcher, 2002; Ghrist, 2014).

We will illustrate with an elementary example how persistent homology can detect shape at multiple scales by noting the birth and death of topological features. Our dataset will consist of 5 points from a circle as shown in **Figure 2**. To detect the circle, we need to connect these points in some manner. For example, we could connect all points whose distance is less than some fixed ϵ. If one can visualize the data set, then the choice of ϵ may be clear. But more often, there is no obvious choice, so instead we analyze the data at multiple scales using persistent homology. The first box in **Figure 2** shows the five data points. At this stage, we have five components, one for each data point ($\beta_0 = 5$). These components are represented by the five red lines in the top part of this figure. These five red lines along with the blue segment is called the barcode for the data set. The barcode keeps track of the number of components (red bars) and number of 1-dimensional holes (blue bar) as the threshold for connecting

[1]While the intuitive definition will suffice for this paper, we have left out a number of details. For example if we use addition with $\mathbb{Z}_2$ coefficients, we can detect the Klein bottle surface ($\beta_2 = 1$), while if we use $\mathbb{Z}$ coefficients, $\beta_2 = 0$ since the Klein bottle does not bound a void. For computational speed, $\mathbb{Z}_2$ coefficients are frequently used when computing persistent homology.

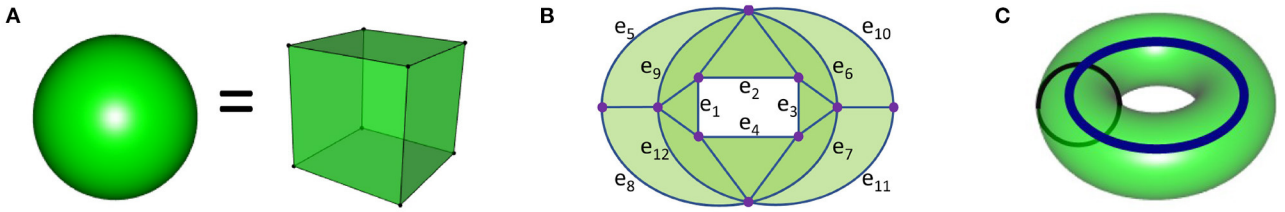


FIGURE 1 | (A) The solid ball and solid cube are topologically equivalent and thus have the same homology. Their surface boundaries also have the same homology since these surfaces are topologically equivalent. The solid ball has one connected component and thus $\beta_0 = 1$. The solid ball does not contain any voids, and thus $\beta_i = 0$ for all $i > 0$. The sphere, which is the boundary of the ball, has $\beta_0 = 1$ since it is connected, and $\beta_2 = 1$ since the 2-dimensional sphere bounds a void, while $\beta_i = 0$ for all other i since there are no lower or higher dimensional voids. For an $n + 1$-dimensional ball (for example, all points of distance less than or equal to 1 from the origin in R^{n+1}), $\beta_i = 0$ for all $i > 0$ since it does not contain any voids. The n-dimensional sphere, which is the boundary of the $n + 1$-dimensional ball, has $\beta_i = 1$ for $i = 0, n$ and $\beta_i = 0$ for all other i. Since the n-dimensional sphere contains a void, it is the n-dimensional object that generates β_n. **(B)** A surface with boundary that is topologically equivalent to an annulus. The annulus is a 2-dimensional surface that has the same homology as a 1-dimensional circle. Since this object has one connected component, $\beta_0 = 1$. We can use addition to represent a cycle. The cycle $e_5 + e_9 + e_{12} + e_8 = e_5 + e_8 + e_9 + e_{12}$ is homologous to 0 since it bounds a surface (the light green waning crescent moon). Since all 1-dimensional cycles are either homologous to 0 or to (a multiple of) the rectangle cycle $e_1 + e_2 + e_3 + e_4$, $\beta_1 = 1$. Since this object lives in the 2-dimensional plane, $\beta_i = 0$ for all $i > 1$. **(C)** The solid torus has $\beta_i = 1$ for $i = 0, 1$ and $\beta_i = 0$ for all other i while its boundary, the torus, has $\beta_i = 1$ for $i = 0, 2$, $\beta_1 = 2$, and $\beta_i = 0$ for all other i. The thick blue cycle is a 1-dimensional homology generator for both the solid torus and its boundary. The thiner black cycle is a homologous to 0 in the solid torus as it bounds a meridinal disk, while this black circle is a homology generator in the torus which is not homologous to the blue circle. The torus surface generates the 2-dimensional homology.

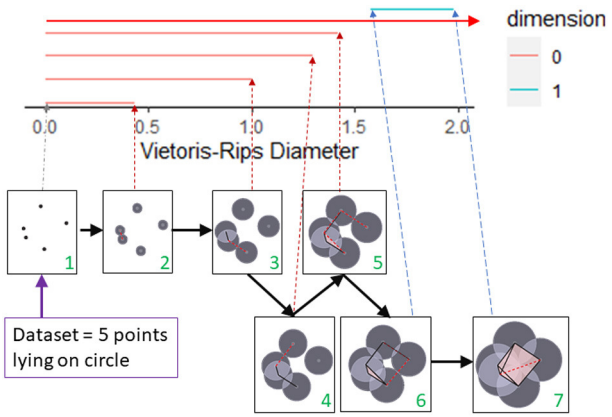


FIGURE 2 | A barcode captures topological features in a dataset at multiple scales. The topology of a dataset at a fixed scale is determined by joining pairs of data points with an edge if the distances between the pair of points is less than the fixed scale. If three edges form a triangle, then the triangle is filled in. This process is shown in the seven boxes as the scale for joining vertices increases from box 1 to box 7. The corresponding barcode is shown at the top of the figure. The persistence of a feature over multiple scales determines the length of the bar corresponding to that feature. The number of components (β_0) that exist at a particular scale is represented by the number of red bars that exists at the corresponding Rips diameter. The creation of the 1-dimensional cycle in box 6 is represented by the birth of the blue bar. The blue bar dies when this cycle is filled in with triangles (box 7). This figure was created by modifying the output of the R package TDAstat (Wadhwa et al., 2018) and latex code written by Catalina Betancourt.

data points increases. We can visualize the increasing threshold (or proximity parameter) by growing balls around each data point and connecting pairs of points as soon as their respective balls intersect. Thus, in the second box, an edge joins the two closest points, reducing the number of connected components by one. Thus, one bar ends (dies), and only 4 bars ($\beta_0 = 4$) continue past this threshold. Observe that every time an edge joins two components, a bar dies (and β_0 reduces by one). In the timepoint just before 1.5 (box labeled 5), two edges are added. One connects two components, but the third forms a triangle with two previously created edges. These three edges surround a small hole, but we fill in this hole (shaded in pink) as we only want to detect large holes. We are forming a Rips complex where whenever a triangle is formed, it is immediately filled in and thus triangles do not contribute to β_1. In the timepoint after 1.5 (box labeled 6), a cycle containing four edges is formed. This is indicated in the barcode by the start (birth) of the blue bar. As more edges are added, eventually this region is divided into two triangles and the blue bar dies at timepoint close to 2 (corresponding to box labeled 7). Note we have one infinitely long bar (top red bar with arrow) since after time 1.5 we have one connected component.

To summarize, this example of a TDA pipeline consists of taking a dataset, creating a sequence of Rips complexes, and outputting a barcode (Edelsbrunner et al., 2002; Carlsson et al., 2005; Zomorodian and Carlsson, 2005). A Rips complex is a generalization of a graph. While in our example we only looked at adding edges and triangles, we can also add higher dimensional simplices. A n-simplex in a Rips complex is a collection of $n + 1$ points where each pair of points is connected by an edge. Thus an edge is a 1-simplex, a triangle is a 2-simplex, and a tetrahedron is a 3-simplex. In our circle example, when all pairs of the 5 points are connected by edges, we add a 4-simplex even though the data set lives in 2-dimensions. The existence of an n-simplex means that (all pairs of) $n + 1$ points are close together according to a given threshold. The Rips complex is also called a clique complex, the latter term coming from graph theory where a clique is a graph where every pair of vertices is connected. Thus, our simplices correspond to clique subgraphs. Other names for Rips complex include Vietoris-Rips complex and flag complex.

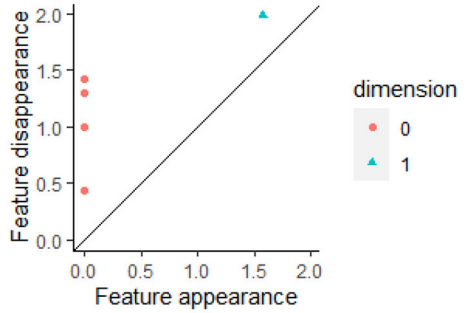


FIGURE 3 | A barcode can be converted into a persistent diagram. Each bar with finite length in a barcode is represented by a point in the persistent diagram. If a bar is born at time b and dies at time d, then the bar is represented by the point (b, d). In **Figure 2**, there are four finite red bars plus one infinite red bar. These bars are all born at time 0. In the persistent diagram, the four finite red bars are represented by the four red points all of which have $b = 0$. The one blue bar in **Figure 2** is represented by the blue triangle in this persistent diagram. This figure was created using the R package TDAstat (Wadhwa et al., 2018).

There are other ways to form a simplicial complex from data. For the Rips complex, an n-simplex is formed at threshold r when all pairs of $n + 1$ points are of distance less than r (so that each pair of points is connected by an edge). This is equivalent to requiring every pair of balls of radius r centered around the $n+1$ points to intersect. If we require the intersection of all these balls to be nonempty in order to form an n-simplex, we instead form the Čech complex. Thus, to form a 2-simplex (triangle), the Rips complex only requires non-empty pairwise intersection of three balls while the Čech complex requires the intersection of all three balls to be nonempty. Thus, the Čech complex is similar to the Rips complex, but an n-simplex is formed at a slightly larger threshold in the Čech complex. Under certain conditions, the Čech complex is guaranteed to have the same homology as the union of all balls of radius r centered around data points (Hatcher, 2002). But the Rips complex has much smaller computer memory requirements as only the edges need to be stored to determine the Rips complex, and thus the Rips complex is normally used when calculating persistent homology. A very different TDA technique called Mapper uses a completely different method to create a simplicial complex from data (Singh et al., 2007). For Mapper, each vertex represents a cluster of data points. If $n+1$ of these clusters have a common intersection, then an n-simplex is formed. Mapper can be used to reduce the size of a data set and to visualize it.

The example in **Figure 2** focused on β_0 and β_1. For data that lives in a higher dimensional space, we can similarly calculate β_n = the number of n-dimensional holes. For example, $\beta_2 = 1$ for both the sphere and torus as these are 2-dimensional surfaces that bound voids in space. For more details regarding persistent homology and barcodes (please see Ghrist, 2008; Carlsson, 2009; Edelsbrunner and Harer, 2010; Otter et al., 2017).

In order to use persistent homology in machine learning, we need a distance between barcodes. We first convert barcodes to persistence diagrams as described in the next section and use these diagrams to define a distance between barcodes. In this section, we show how persistent homology is stable with respect to noise: small perturbations in the data have only a small effect on the barcode (Cohen-Steiner et al., 2007). In section 2.2, we discuss the advantages/disadvantages of persistent homology with regard to how it handles noise, incomplete data, and computational complexity. In section 2.3, we discuss one method (persistent images) of converting a persistence diagram into a vector that can be used in machine learning. We also give references to many other methods for using persistent homology in machine learning.

While we have discussed the basic method for converting Euclidean data into barcodes, there are a number of other methods for obtaining barcodes from data. All one needs is a method to determine when to add an edge between pairs of data points. Thus, the data do not need to live in Euclidean space. We also assumed that small holes correspond to noise, but there are applications where the point of using persistent homology is to detect small holes (Bendich et al., 2016). We also had only one infinite bar corresponding to the one connected component we obtained when all our data points were connected by edges. If one is working with Euclidean data, eventually all holes will be filed in and thus eventually a Rips complex with only one component and no holes will be formed. But in other applications, holes may persist forever, resulting in infinite bars. One can also obtain additional information by looking at the group structure of the filtered homology groups, and prove stability properties using interleaving distance (Bauer and Lesnick, 2014; Bubenik and Scott, 2014; Oudot, 2015; Chazal et al., 2016).

2.1. Persistence Diagrams and Stability

While barcodes are useful for visualizing changes in homology, barcodes are generally converted into persistence diagrams for statistical and machine learning analysis (Edelsbrunner et al., 2002; Mileyko et al., 2011). The start of a bar represents the birth of a cycle while the end represents its death. The plot of the points (birth time, death time) in 2-dimensional space is called the persistent diagram (PD). The persistent diagram corresponding to the barcode in **Figure 2** is shown in **Figure 3**. A persistence diagram also includes the diagonal as shown in this figure as the diagonal is used when computing distances between PDs. A PD can be a multiset if multiple bars have the same birth time b and death time d, so that the point (b, d) occurs multiple times in the PD.

The formula for the bottleneck distance for a fixed β_i between two persistence diagrams, P_1 and P_2, is $d_B(P_1, P_2) := \inf_{\gamma : P_1 \to P_2} \sup_{x \in P_1} \|x - \gamma(x)\|_\infty$. To compute this distance we first create a matching γ between these diagrams for the fixed β_i as shown in **Figure 4**. In this figure the blue triangles represent features with the fixed β_i from one data set while the purple stars represent features from a different data set for the same β_i. A matching $\gamma : P_1 \to P_2$ is a bijective function from P_1 to P_2 where both persistence diagrams include the diagonal. Features that are close to the diagonal get matched to the diagonal unless they are closer to another feature that does not have a better

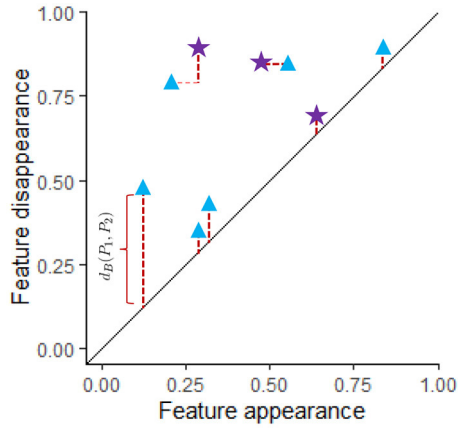


FIGURE 4 | Two persistence diagram, P_1 and P_2, are shown for a single dimension (for example, β_1). The blue triangles correspond to P_1 while purple stars are used for P_2. Both persistence diagrams include the diagonal. A matching between P_1 and P_2 is shown where the red dotted lines indicate features that have been matched where some of the features are matched to the diagonal. The length of the thicker dark red dotted lines indicate the distance between matched features. The distance between a feature and the diagonal is the persistence of the feature, $d - b$, where b = birth time and d = death time of that feature. If feature (b, d) is matched with feature (β, δ), then the distance between these features is $\max(|b - \beta|, |d - \delta|)$. Since the best matching is shown, $d_B(P_1, P_2)$ equals the length of the longest of the thick dark red dotted lines. Any other matching would have a matched pair of features with larger distance.

matching than to the diagonal. If $x = (b, d) \in P_1$ is matched to the point (β, δ), then the distance between these features is $\|x - \gamma(x)\|_\infty = \max(|b - \beta|, |d - \delta|)$. To find the distance for a particular matching γ, we calculate $\sup_{x \in P_1} \|x - \gamma(x)\|_\infty =$ the largest distance between a point x in P_1 and its match $\gamma(x)$ in P_2. The bottleneck distance is obtained by taking the infimum of this distance over all possible matchings. In **Figure 4**, red dotted lines indicate best matches between features from P_1 and P_2.

If P_1 is the PD for the data set X and P_2 is the PD for the data set Y, the stability theorem states that $d_B(P_1, P_2) \leq d_H(X, Y) = \inf\{\varepsilon \geq 0\,;\ X \subseteq Y_\varepsilon \text{ and } Y \subseteq X_\varepsilon\}$ where $X_\varepsilon := \bigcup_{x \in X} \{z \in M\,;\ d(z, x) \leq \varepsilon\}$ (Cohen-Steiner et al., 2007). In other words, if each data point is perturbed by at most a distance ϵ, then the persistence of a feature will change by at most 2ϵ since the birth and death times can change by at most ϵ. Features with persistence $<2\epsilon$ may disappear, while new features with persistence less than 2ϵ may be created.

2.2. Benefits and Limitations of Persistent Homology

That persistent homology is stable with respect to noise is, of course, a major advantage. But any method that uses Euclidean distance is affected by the curse of dimensionality due to the effect of noise on distance. For example, suppose a data point should be at the origin, but due to noise, each coordinate is perturbed by 0.01 units, then the point which should be at the origin is now $\Sigma_{i=1}^{n}(0.01^2)$ units away from the origin if the data lives in $\mathbb{R}^n$. Thus, for example if $n = 10,000$, then the data point is perturbed by a distance $\Sigma_{i=1}^{10,000}(0.01^2) = 1$. While the change in persistent homology is bounded by the distance between the original data set and the perturbed data set, the latter can be quite large, depending on the amount of noise and the dimension of the dataset. Thus, performing PCA or t-SNE or other dimension reduction technique first may lead to stronger results.

In order to recover the shape of an object, one must have sufficient coverage. Some holes detected by persistent homology may be due to incomplete data. If these are small, then they only result in short bars which may be considered noise. But in high dimensional spaces, one has many degrees of freedom, so even recovering the shape of simple objects in high dimensions can be impossible as obtaining a sufficient number of data points may not be feasible. However, differences between data sets may still be detected even if coverage is lacking. For example, one may have insufficient coverage to recover the topology of a torus if one uniformly under-samples data points from a torus. However, the resulting barcode will likely be very different than the barcode obtained from uniformly under-sampling points from a sphere. Also, coverage can be less of an issue if you have some information regarding the shape of the data such as periodicity (for example, Dequeant et al., 2008). Thus, in practice, topological data analysis has proven to be quite robust. For more on complexity and topological inference (see Weinberger, 2014).

Due to computational complexity, most analysis using TDA restricts to the use of β_i for $i \leq 4$. Often only β_0 and β_1 are used, but faster algorithms such as Ripser (Bauer, 2019) are becoming available. To calculate persistent homology of a point cloud, one first needs to create simplicial complexes. The number of simplices grows rapidly with the number of data points as well as the homology dimension (not the dimension of the data set, but the dimension of the holes one wishes to detect—in order to calculate β_i, one needs i-dimensional and $i + 1$ dimensional simplices). The TDA pipeline also requires the computation of distances between data points. The dimension in which the data lives can affect this step, but after distances are calculated, it is the shape of the data that can have the largest effect, sometimes even larger than the number of data points as there are several algorithms that can greatly simplify the simplicial complex (Zomorodian, 2010; Mischaikow and Nanda, 2013; Wilkerson et al., 2014; Boissonnat and Pritam, 2020). The effectiveness of these simplification algorithms depends on both the topology and geometry of the data set. For example, suppose one takes n data points equally spaced on a straight line. The topology of the line is the same as the topology of a point. Thus, to calculate the homology of the line, one can remove all simplices except for a single vertex. For more on computational complexity of persistent homology (see Otter et al., 2017).

If all the data points enter at time 0, the β_0 bars all start at time 0. Thus the barcode for β_0 can be created from a single linkage hierarchical clustering dendrogram as the merge heights of the dendrogram become the lengths of the β_0 bars. Hence the β_0 barcode contains less information than a single linkage hierarchical clustering dendrogram. However, there are applications where the data points enter at different times such as time series data. Thus, the β_0 barcode can be applied to a

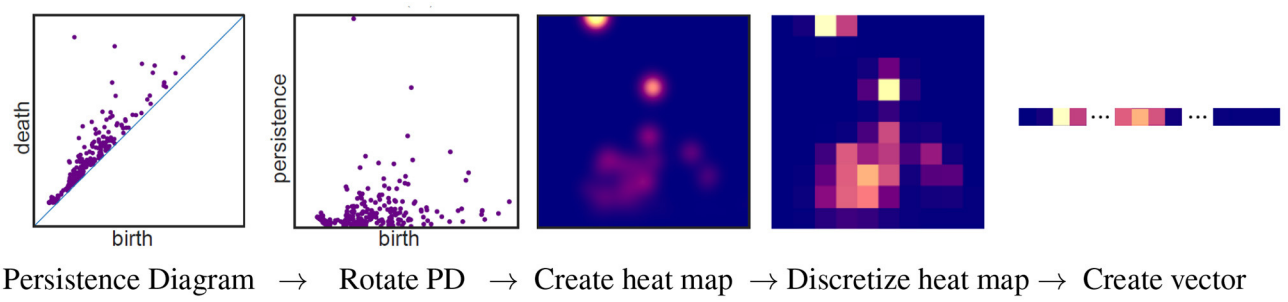


FIGURE 5 | Pipeline for vectorizing a persistent diagram using persistent images. This figure is a modification of **Figure 1** from Adams et al. (2017) which is licensed under CC BY 4.0.

wider variety of applications than standard clustering techniques. Clustering also cannot capture holes and voids; the higher dimensional barcodes capture structure that other methods such as clustering miss.

2.3. Persistent Homology and Machine Learning

The barcode can be used as a topological signature to identify structure in data. While homology is built to detect topology and not geometry, persistent homology can be implemented in a variety of ways to distinguish geometrical shapes (e.g., Turner et al., 2014; Li et al., 2018; Bubenik et al., 2020). Machine learning can be applied to a collection of persistent diagrams to distinguish between data sets with different structures. Many machine learning algorithms take a vector as input. There are many ways to create a vector from persistent homology. A pipeline to create a vector using persistence images (Adams et al., 2017) is illustrated in **Figure 5**. A persistent diagram is first rotated by 45° so that the diagonal becomes the horizontal axis (2nd panel of **Figure 5**). Thus the horizontal axis represents the birth time, while the vertical axis represents persistence = death - birth. A heat map is then created using a Gaussian distribution (or other weight function) about each point (3rd panel). The height of the Gaussian distribution is indicated with color in the heat map and is dependent on the persistence of the feature. Points closest to the diagonal are considered to be the result of noise and are thus given no intensity. Hence the bottom of the heat map will always have the color corresponding to zero intensity, in this case blue. In other words, points close to the diagonal have no effect on the heat map. Observe that the point furthest from the diagonal in the first panel corresponds to a feature with the largest persistence per second panel. Thus, in the heat map in the 3rd panel, the color at this point is given the highest intensity (yellow). As shown in the fourth panel, the heat map is discretized by partitioning the heat map into $n \times n$ squares where the color of each square corresponds to the average value of the corresponding square in the heat map. In the discretized heat map (4th panel), the yellowish region from the 3rd panel corresponding to the most persistent feature is partitioned between two squares with the yellow square in the top row of this heat map containing a larger portion than the pinkish square next to it in the same row. In the final panel, an n^2-dimensional vector is created by concatenating the rows of the discretized heat map.

Other methods for using persistent homology in machine learning include persistent landscapes (Bubenik, 2015, 2020), persistent curves (Chung and Lawson, 2019), and kernel functions (for example, Reininghaus et al., 2015; Kusano et al., 2016; Carrière et al., 2017; Chazal et al., 2017).

3. TREATMENT RESPONSES AND PROGNOSIS

What impedes the success of cancer therapies is often the coexistence of therapy resistant cells along with therapy sensitive tumor cell populations. When administered separately, all currently adopted therapeutic strategies—ranging from cytotoxic chemotherapies to molecular targeted therapies—impose a dramatic, yet homogeneous selective pressure on an often heterogeneous group of tumor cells. Despite varying resistance mechanisms contingent upon therapy-type and tumor composition, every therapeutic intervention inevitably selects for resistant cells, which expand and become the dominant cell type of recurrent tumors, that cease to respond to therapy (Maley and Reid, 2005; Aparicio and Caldas, 2013; Bukkuri, 2020). The increased resolution on the clonal architecture of intermixed tumor cell populations that has just now become available calls for prognostic and therapeutic benefits. High intra-tumor diversity in pre-malignant lesions has been shown to predict progression to malignant growths and poor outcome (Maley et al., 2006; Laurie et al., 2012). The therapeutic significance of intratumoral heterogeneity (ITH) is exemplified in a recent study that measured genetic and transcriptional diversity of breast cancer tumors before and after therapy based on four genetic markers and two transcriptional markers. The study provided proof-of-principle that therapy-induced phenotypic changes can be predicted based on the characterization of coexisting tumor subpopulations (Almendro et al., 2014). Another recent study used RNA interference to model heterogeneous tumors and tested the efficacy of predicted drug combinations in eliminating coexisting tumor subpopulations (Zhao et al., 2014). Their findings suggest that the most effective drug combination for a

given tumor cannot be achieved by targeting the predominant subpopulation alone, but requires detailed characterization of the genetic makeup of branched subpopulations and their contribution to the tumor bulk.

Techniques from computational homology have been used to develop a new algorithm to characterize comparative genomic hybridization (CGH) profiles and identify the frequency of cancer recurrence in early stage breast cancer patients through identification of recurrent copy number aberrations (CNAs) in cancer (DeWoskin et al., 2010), which serve as markers of genomic instability and thus cancer prognosis (Hanahan and Weinberg, 2000; Han et al., 2006). Specifically, the method uses a sliding window algorithm to associate a set of point clouds to each array CGH. Different window sizes allow one to analyze the data at various scales by considering different dimensional point clouds. Then, persistent homology is applied to these point clouds for classification. It was found, in accordance with prior results (Climent et al., 2007), that the Betti numbers of the zero dimensional homology groups (β_0) can distinguish between recurrent and non-recurrent groups in patients who did not receive anthracycline-based chemotherapy after surgery but not in patients who were treated with anthracycline. Note that, in this approach, no segmentation of the data was required.

In another study, a novel statistic called the smooth Euler characteristic transform (SECT), which allows shape information to be integrated into traditional statistical models, was developed and applied to predict disease free survival in glioblastoma multiforme (GBM) based on tumor shape from post-contrast T1 axial magnetic resonance imaging (MRI) (Crawford et al., 2020). SECT is a variation of the persistent homology transform (PHT) introduced in Turner et al. (2014) that was created to overcome the difficulties in integration with traditional statistical models. Specifically, the output of SECT is a collection of smooth vectors, while the output of PHT is a collection of persistence diagrams (Edelsbrunner et al., 2002), thus having a complicated representation and geometry which does not lend itself easily into integration with statistical models. In the GBM application, the statistical model used was a Bayesian linear mixed model (BLMM) (Ishwaran and Rao, 2005; Guan and Stephens, 2011; Zhou et al., 2013). When this topological approach was applied to the GBM MRI data, it was found to outperform gene expression, volumetric, and morphological summaries in predicting disease free survival.

Clinically, there is a great importance in the identification of biomarkers which can serve as predictors for metastasis and patient prognosis in cancer. To this end, researchers have recently used persistent homology techniques, in an exploratory data analysis fashion, to identify biologically meaningful geometric properties of single cell data (Lockwood and Krishnamoorthy, 2015). In this method, data was first transposed and analyzed in its dual space with each gene represented in a much lower dimensional sample space, thus circumventing the problem of high dimensionality that is typical of single cell data. A small set of genes (120–200) were then selected as landmarks (De Silva and Carlsson, 2004) and a family of nested simplicial complexes was constructed, indexed by a proximity parameter. Unlike many other methods which focus on the analysis of zero dimensional homology groups (DeWoskin et al., 2010; Nicolau et al., 2011), thus performing analyses which are topologically equivalent to clustering, this study focused their efforts on identifying loops of one dimensional homology groups which persist over a large range of values of the proximity parameter, hypothesizing that connections around holes imply nontrivial interactions among genes and biological functions which could have implications for tumorigenesis. Repeating this process for various landmarks, features which remain stable over large ranges of both the proximity parameter and number of landmarks could be detected. Applying these techniques to five different cancer data sets from brain, breast, ovarian, and acute myeloid leukemia cancers, many members of the significant loops in the one dimensional homology groups that were found have been previously shown to be accurate biomarkers for cancer biogenesis, while others serve as potential new markers which have yet to be experimentally validated.

4. TUMOR SEGMENTATION AND COMPUTER-AIDED DIAGNOSIS

Computerized methods can efficiently and effectively identify quantitative image features that are otherwise difficult to spot by manual inspection (Yu et al., 2016). Quantitative morphological features extracted from H&E stained slides, such as Zernike shape features, have been shown to predict survival in lung adeno- and squamous cell carcinoma (Yu et al., 2016). Recent advances in next-generation sequencing technologies gave rise to a plethora of approaches that quantify and characterize the genotypic diversity within a given tumor. Evidence supporting a quantitative relation between genotypic and morphological ITH followed. A quantitative image analysis approach that complements genomic profiling with geographical information was developed (Yuan et al., 2012; Andor et al., 2016). Furthermore, the authors characterized cellular heterogeneity by distinguishing between well-defined cell-populations (stromal cells, lymphocytes, cancer cells). However, so far qualitative details of how this diversity in morphology is structured (i.e., how many subpopulations are present and what their geographical boundaries are on the H&E slide) are unknown.

As a step toward a computer-aided cancer diagnosis system, persistent homology has been used to develop an automated tumor segmentation approach for Hematoxylin & Eosin (H&E) stained colorectal cancer histology whole slide images (WSI) (Qaiser et al., 2016). The authors exploit the fact that nuclei in tumor regions have atypical characteristics such as non-uniform chromatin texture, irregularity in shape and size, and clustering of nuclei, and use persistent homology profiles to characterize the degree of connectivity among nuclei and to classify cancerous regions based on this information. Specifically, once a WSI has been obtained, it is first divided into patches, each of which has a persistent homology profile. Given two patches, the symmetrized Kullback-Leibler divergence (KLD) can be computed between the respective persistent homology profiles, which serves as a metric for interpatch distance. Then an input patch is classified as cancerous or non-cancerous by

a kNN classifier, based on KLD distances between its persistent homology patch and those of each representative patches. These exemplar patches are chosen by training a CNN and selecting patches whose activation during training is large (separately for cancerous and non-cancerous classes). The benefit of this approach over previous approaches is that only the subset of highly activated patches from the convolutional layers are used as exemplars rather than the set of all patches in the training data. This method was compared against standard CNN and HyMaP (Khan et al., 2013) approaches on 74 H&E stained WSIs of colorectal cancers; in addition to being computationally less expensive than the other two methods, it was also shown to have better precision and segmentation accuracy.

Another example of tumor segmentation and algorithmic diagnosis is a recent study which aimed to segment a diseased area of skin and classify the type of skin lesion into one of seven classes in a given dermatoscopic image (Tschandl et al., 2018) using persistent homology (Chung et al., 2018). Like the colorectal image segmentation study (Qaiser et al., 2016), the segmentation algorithm used is a concept similar to persistent homology (Edelsbrunner et al., 2002). Linear support vector machines (SVMs) were used for classification on the persistence statistics (Chung et al., 2018) and persistence curves (Chung and Lawson, 2019) were derived from persistence diagrams. Specifically, given an image, a segmentation algorithm was first implemented to obtain an image mask: a binary image in which each pixel is colored either white (if it part of the healthy skin) or black (if it is part of a lesion). Once the mask was applied to the original image, the RGB color space is transformed into an RGB, HSV, or XYZ color space and each channel was extracted. Persistent homology software was then used to compute persistence diagrams for each channel; from each diagram, persistence statistics and curves were computed as features. Finally, a multi-class SVM was used to classify the input into one of the seven types of skin lesions. When this approach was applied to a validation set of 5,000 images, the highest resulting accuracy scores were 65.6, 66, and 67.2%.

Similar persistent homology techniques were used to classify H&E stained stage T3 and stage T4 colorectal adenocarcinomas images as benign or malignant (Chittajallu et al., 2018). To do this, given an image, it was first color normalized (Reinhard et al., 2001) and the nuclear stain and minimum cross entropy thresholding (Li and Tam, 1998) for nuclear foreground segmentation were extracted using an unsupervised color deconvolution method (Macenko et al., 2009). Then, a fast difference-of-Gaussian implementation of the scale-adaptive Laplacian-of-Gaussian filter of Al-Kofahi et al. (2010) was performed to detect nuclei centroids. Then, by considering the set of nuclei centroids as a point cloud, the persistence diagram of its Vietoris-Rips filtration for the one dimensional homology groups (loops) was computed using a fast multiscale approach (Doyle et al., 2008). Then, persistence landscape (Bubenik, 2015) and image (Adams et al., 2017) representations were computed and used as features to characterize loops formed by glandular epithelial cell nuclei. Then given training images with benign/malignant labels, a random forest classifier was trained using these topological features. PCA was used to reduce the dimensionality of each feature group so as to preserve 99% of the variance. Hyperparameter optimization was also performed via cross-validation using a tree-structured parzen estimator (Bergstra et al., 2011). When this method was applied to testing data consisting of 80 images, an accuracy of 85%, AUC of 0.85, precision of 78%, and recall of 95% was obtained, an improvement over the traditional cell graph property approach in all areas (Doyle et al., 2008).

5. DISEASE CLASSIFICATION

Cancers of unknown primary represent 3–5% of all cancer cases, whereby physicians find one or multiple metastases but fail to locate the primary tumor. Pathologic evaluation of a metastatic biopsy often does not provide a definitive answer. Molecular data ranging from gene expression to somatic mutations have been shown to significantly aid classification of metastatic biopsies to their corresponding primary tumor site (Ferracin et al., 2011; Marquard et al., 2015; Vikeså et al., 2015; Moran et al., 2016; Søndergaard et al., 2017).

One study used persistent homology on 150 non-contrast-enhanced fat-suppressed 3D T1-weighted magnetic resonance (MR) images to classify hepatic tumors into three classes: hepatocellular carcinomas (HCC), metastatic tumors (MT), and hepatic hemangiomas (HH) (Oyama et al., 2019). To do this, for each image, a 3D region of interest (ROI) in the shape of a rectangular solid enclosing the entire lesion was created by an experienced radiologist. Then, gray-scale values of the voxels in each ROI were normalized and persistence diagrams were created for dimensions 0, 1, and 2 using HomCloud (Kimura et al., 2018; Obayashi and Hiraoka, 2018). These diagrams were vectorized into persistence images (Adams et al., 2015). Feature vectors were then obtained from these images and inputted into logistic regression with an elastic net penalty and extreme gradient boosting machine learning models for classification. The results from classification showed that dimension 1 persistence images had the highest accuracy rates: 85% for classifying HCC and MT, 84% for HCC and HH, and 74% for HH and MT.

An alternative method to accurately classify tumor subtypes is through the use of high throughput genomics (Nutt et al., 2003; Freije et al., 2004). Aiming to produce more robust algorithms than traditional classification methods, given gene expression profile data, researchers used statistical invariants and persistent homology to identify core patient groups associated with the classical, mesenchymal, and proneural subtypes of GBM and a compact set of genes most useful for this partitioning (Seemann et al., 2012). To do this, a sufficient, but compact, panel of genes to be used for clustering was predetermined using non-dimensionalized standard deviation (to ensure bimodality of gene expression distribution across patient samples; Phillips et al., 2006; Verhaak et al., 2010) and persistent homology (to find groups of genes whose expression levels change coherently among patient samples; Carlsson, 2009; Horak et al., 2009). Then, a hierarchical partitioning of patient samples based on gene expression levels is performed using persistent homology; specifically, samples are repeatedly bisected until

further partitioning is not possible, thus obtaining the number of clusters that exists and some notion of genetic proximity of the clusters. Each bisection was implemented using 30 genes. A predictive model was then implemented to assign cancer subtypes to each cluster. Applying this approach to the 20 GBM test samples, fifteen predictions were in accordance with results from standard clustering calculations (Verhaak et al., 2010), five of which were unassigned by both algorithms. Of the remaining five samples, four were classified as "neural" by the clustering algorithm, but were unassigned by this approach since the neural group was not found in a single cluster.

Another example of the use of persistent and computational homology on gene expression data is in Arsuaga et al. (2012), whereby, upon application to a breast cancer gene expression dataset, the algorithm was able to distinguish among most breast cancer subtypes. This paper extended the work of DeWoskin et al. (2010) to gene expression data, under the assumption that gene expression is a measure of the underlying copy number changes (Neve et al., 2006; Horlings et al., 2010). Before applying the sliding window algorithm developed in DeWoskin et al. (2010) to gene expression data, theoretical work was done to show that under idealized conditions, the point cloud defined by the algorithm is a good representation of the original data. Hence, analysis of the point cloud is applicable to the original data set. This was done using Taken's embedding theorem, an extension of Whitney's embedding theorem to dynamical systems theory, and a circularization technique. To apply the sliding window algorithm to gene expression data, instead of pre-selecting differentially expressed genes like traditional clustering algorithms, all genes were ordered by their location in the genomes. Then, the sliding window algorithm was applied to generate point clouds, upon which topological and statistical analysis was performed. It was shown that when only β_0 was used, the algorithm could distinguish between less aggressive subtypes, like normal and luminal-A, and more aggressive ones, such as luminal B, basal-like, and Her2. It was also noted that the algorithm could not distinguish luminal B from Her2 and basal-like, implying the close similarities among these subtypes. Thus, it was noted that breast cancer subtypes can not only be classified by specific sets of genes, but also by certain global relationships among all genes.

6. CELLULAR ARCHITECTURE

Imaging is an essential part of cancer clinical protocols, providing physicians with morphological, structural, and metabolic information about patient tumors, thereby assisting in clinical decision making and treatment planning (Fass, 2008). The development of new image segmentation tools (Zhang et al., 2001; Hong and Brady, 2003; Xiaohua et al., 2004) and quantitative multiplex immunofluorescence (Stack et al., 2014; Dimitriou et al., 2019; Abousamra et al., 2020) have set the stage for topological data analysis and persistent homology techniques to be harnessed for interpretation of high-dimensional information in histopathological imaging data.

One example of this is using persistent homology techniques to investigate architectural characteristics of cellular organization and nuclear arrangements from microarray tissue samples to distinguish among genetically derived breast cancer subtypes (Basal, Luminal A, Luminal B, and HER2; Singh et al., 2014). This was done through distinct topological characterizations such as nuclear connectivity (generators of zero dimensional homology groups) and loops (generators of one dimensional homology groups) based on Vietoris-Rips filtration of nuclei centers (Mischaikow and Nanda, 2013). When its performance was compared to a standard distance weighted discrimination classifier (Marron and Todd, 2007), nearly a four times improvement in classification accuracy was noted. Furthermore, for certain combinations of feature weightings, it was shown that topological features provide complementary information to patch based image appearance features. By using such topological features, they solve/address two main challenges in obtaining accurate cellular architectural characterization: the heterogeneity of spatial arrangements, both among patients and within single tumor samples, and differences in stain intensity which require manually determined phenotypic thresholds (Engers, 2007; Truesdale et al., 2011; Goodman et al., 2012; Helpap et al., 2012; Truong et al., 2013; Epstein et al., 2016; Evans et al., 2016). This improves performance over existing standard classifiers, which are more sensitive to noise, cannot model stain concentration variations, and have issues with larger cell arrangements (Aukerman et al., 2020).

In another paper, researchers used TDA to cluster prostate cancer histology into architectural groups consistent with the continuum of Gleason patterns, the most widely accepted system for evaluating prostate cancer architecture (Humphrey, 2004; Lawson et al., 2019). Persistent homology was used to compute persistence intensity diagrams (of zero and one dimensional components) of purely graded prostate cancer histopathology images of Gleason patterns 3–5. This revealed key insights into characteristics such as nuclei density, glandular shape, and inter-glandular arrangement. Furthermore, persistent homology was able to cluster these images into architectural groups through a rank descending persistence vector–the six resulting clusters provided a stable architectural continuum from well differentiated to poorly differentiated adenocarcinoma at an even finer level than the standard Gleason scale.

Persistent homology has also been used to characterize the spatial arrangement of immune and epithelial (tumor) cells within the breast cancer immune microenvironment from quantitative multiplex immunofluorescence (qmIF) imaging (Aukerman et al., 2020). Stain intensities and spatial coordinates of individual cells were collected from qmIF through nuclear segmentation, cytoplasmic definition, and stain quantification. In order to incorporate these stain intensities, instead of directly using a Rips or Cech filtration on the point cloud data (Chazal et al., 2009), a discretization process was first implemented to convert the point cloud data with stain intensity values into an image. Then, persistence diagrams were created from these images by using the opposite of the pixel stain intensity as the filter function. These diagrams were assessed as potential biomarkers of cancer subtype and prognostic biomarkers of

overall survival using kernel mean embeddings (Gretton et al., 2012) with the sliced Wasserstein kernel (Carrière et al., 2017) and were shown to outperform the standard nearest neighbor analysis with a standard Gaussian kernel. Furthermore, a correlation analysis using constrained covariance (Herbrich et al., 2005) showed that the correlation between nearest neighbor and persistence diagrams were always <0.1, implying the features are nearly statistically independent and thus complementary.

7. DISCUSSION

As we have seen in this paper, TDA has proven to be a powerful tool, yielding critical insights in the treatment prognosis, tumor segmentation and diagnosis, disease classification, and cellular architecture of cancer. But despite the many recent successes of TDA in the field of oncology, it is still a nascent field with much fruitful work yet to be done. Experimentally, to biologically validate the TDA methodology and results, it would be worth performing thorough studies to assess whether geometric and topological connectivity implies functional connectivity. Computationally, one area which deserves further exploration is the use of TDA to analyze time-series data (Ravishanker and Chen, 2019) in cancer. This has been done extensively in several other fields including climate analysis (Berwald et al., 2014), tracking stability of dynamical systems (Khasawneh and Munch, 2016), clustering populations of Tribolium flour beetles (Pereira and de Mello, 2015), analyzing motion sensor data during sports activities (Stolz et al., 2017), and financial time series data (Gidea, 2017; Truong, 2017; Gidea and Katz, 2018; Gidea et al., 2020). Though time series oncological data have been analyzed with varying degrees of success (Aoto et al., 2018; Kourou et al., 2020), TDA techniques of any sort have yet to be applied. Applying persistent homology techniques to time series microarray, cell anatomy imaging, or gene/pathway expression data, for example, may further help in disease classification, identifying intra-tumoral selection events, and contribute to a greater understanding of tumorigenesis. Another possible avenue of research is to investigate the process of angiogenesis, an inherently geometric and spatially dependent process, using persistent homology techniques. Specifically, we anticipate that TDA will help us understand the changes that occur in tumor vasculature morphology during cancer progression and under treatments. More importantly, we hope that connections between cancer vessel network and treatment prognosis can be found, such as by testing vessel normalization theory (Jain, 2005). In addition to the ideas presented above, it is worth noting that research into the use of TDA in oncology is sparse and, as such, there is much important and clinically relevant work to be done in simply applying well-understood persistent homology algorithms to broader classes of cancer data sets (note that most TDA analyses have been concentrated in just melanoma, brain, breast, and colorectal cancers) and in performing longitudinal studies across several cancer types.

AUTHOR CONTRIBUTIONS

AB conceptualized the project and wrote the sections 3–7. AB and ID wrote the section 1. ID wrote the section 2. NA wrote the sections 1, 3, 4, and 5. All authors contributed to the article and approved the submitted version.

FUNDING

AB was supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1746051. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

ACKNOWLEDGMENTS

The authors would like to thank Ethan Rooke and Hind Benmerabet for their insightful comments on a draft of this manuscript.

REFERENCES

Abousamra, S., Fassler, D., Hou, L., Zhang, Y., Gupta, R., Kurc, T., et al. (2020). "Weakly-supervised deep stain decomposition for multiplex IHC images," in *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, 481–485. doi: 10.1109/ISBI45749.2020.9098652

Adams, H., Chepushtanova, S., Emerson, T., Hanson, E., Kirby, M., Motta, F., et al. (2015). Persistence images: a stable vector representation of persistent homology. *J. Mach. Learn. Res.* 18, 1–35. Available online at: http://jmlr.org/papers/v18/16-337.html

Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., et al. (2017). Persistence images: a stable vector representation of persistent homology. *J. Mach. Learn. Res.* 18, 1–35. Available online at: http://jmlr.org/papers/v18/16-337.html

Aiello, M., Cavaliere, C., D'Albore, A., and Salvatore, M. (2019). The challenges of diagnostic imaging in the era of big data. *J. Clin. Med.* 8:316. doi: 10.3390/jcm8030316

Al-Kofahi, Y., Lassoued, W., Lee, W., and Roysam, B. (2010). Improved automatic detection and segmentation of cell nuclei in histopathology images. *IEEE Trans. Bio-Med. Eng.* 57, 841–852. doi: 10.1109/TBME.2009.2035102

Almendro, V., Cheng, Y. K., Randles, A., Itzkovitz, S., Marusyk, A., Ametller, E., et al. (2014). Inference of tumor evolution during chemotherapy by computational modeling and in situ analysis of genetic and phenotypic cellular diversity. *Cell Rep.* 6, 514–527. doi: 10.1016/j.celrep.2013.12.041

Alyass, A., Turcotte, M., and Meyre, D. (2015). From big data analysis to personalized medicine for all: challenges and opportunities. *BMC Med. Genomics* 8:33. doi: 10.1186/s12920-015-0108-y

Andor, N., Graham, T. A., Jansen, M., Xia, L. C., Aktipis, C. A., Petritsch, C., et al. (2016). Pan-cancer analysis of the extent and consequences of intratumor heterogeneity. *Nat. Med.* 22, 105–113. doi: 10.1038/nm.3984

Aoto, Y., Okumura, K., Hachiya, T., Hase, S., Wakabayashi, Y., Ishikawa, F., et al. (2018). Time-series analysis of tumorigenesis in a murine skin carcinogenesis model. *Sci. Rep.* 8:12994. doi: 10.1038/s41598-018-31349-x

Aparicio, S., and Caldas, C. (2013). The implications of clonal genome evolution for cancer medicine. *N. Engl. J. Med.* 368, 842–851. doi: 10.1056/NEJMra1204892

Arsuaga, J., Baas, N. A., Daniel DeWoskin, Mizuno, H., Pankov, A., and Park, C. (2012). Topological analysis of gene expression arrays identifies high risk molecular subtypes in breast cancer. *Applicable Algebra in Engineering, Communications and Comput*. 23, 3–15. doi: 10.1007/s00200-012-0166-8

Aukerman, A., Carriére, M., Chen, C., Gardner, K., Rabadán, R., and Vanguri, R. (2020). "Persistent homology based characterization of the breast cancer immune microenvironment: a feasibility study," in *36th International Symposium on Computational Geometry, Vol. 11* (Dagstuhl), 1–11.

Bauer, U. (2019). Ripser: efficient computation of Vietoris-Rips persistence barcodes. *arXiv: 1908.02518v1*.

Bauer, U., and Lesnick, M. (2014). "Induced matchings of barcodes and the algebraic stability of persistence," in *Computational Geometry (SoCG'14)* (New York, NY: ACM), 355–364. doi: 10.1145/2582112.2582168

Bendich, P., Marron, J. S., Miller, E., Pieloch, A., and Skwerer, S. (2016). Persistent homology analysis of brain artery trees. *Ann. Appl. Stat*. 10, 198–218. doi: 10.1214/15-AOAS886

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for hyper-parameter optimization. *Adv. Neural Inform. Process. Syst*. 24, 1–9. Available online at: https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf

Berwald, J. J., Gidea, M., and Vejdemo-Johansson, M. (2014). Automatic recognition and tagging of topologically different regimes in dynamical systems. *Discont. Nonlin. Complex*. 3, 413–426. doi: 10.5890/DNC.2014.12.004

Boissonnat, J.-D., and Pritam, S. (2020). "Edge collapse and persistence of flag complexes," in *36th International Symposium on Computational Geometry (SoCG 2020)*, *Vol. 164 of Leibniz International Proceedings in Informatics (LIPIcs)*, eds S. Cabello and D. Z. Chen (Dagstuhl: Schloss Dagstuhl-Leibniz-Zentrum für Informatik), 19:1–19:15.

Bubenik, P. (2015). Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res*. 16, 77–102. Available online at: http://jmlr.org/papers/v16/bubenik15a.html

Bubenik, P. (2020). "The persistence landscape and some of its properties," in *Topological Data Analysis*, eds N. Baas, G. Carlsson, G. Quick, M. Szymik, M. Thaule (Geiranger: Springer), 97–117. doi: 10.1007/978-3-030-43408-3_4

Bubenik, P., Hull, M., Patel, D., and Whittle, B. (2020). Persistent homology detects curvature. *Inverse Probl*. 36:025008. doi: 10.1088/1361-6420/ab4ac0

Bubenik, P., and Scott, J. A. (2014). Categorification of persistent homology. *Discrete Comput. Geom*. 51, 600–627. doi: 10.1007/s00454-014-9573-x

Bukkuri, A. (2020). Optimal control analysis of combined chemotherapy-immunotherapy treatment regimens in a PKPD cancer evolution model. *Biomath* 9, 1–12. doi: 10.11145/j.biomath.2020.02.137

Camara, P. G., Rosenbloom, D. I., Emmett, K. J., Levine, A. J., and Rabadan, R. (2016). Topological data analysis generates high-resolution, genome-wide maps of human recombination. *Cell Syst*. 3, 83–94. doi: 10.1016/j.cels.2016.05.008

Carlsson, G. (2009). Topology and data. *Bull. Am. Math. Soc*. 46, 255–308. doi: 10.1090/S0273-0979-09-01249-X

Carlsson, G., Zomorodian, A., Collins, A., and Guibas, L. J. (2005). Persistence barcodes for shapes. *Int. J. Shape Model*. 11, 149–187. doi: 10.1142/S0218654305000761

Carriére, M., Cuturi, M., and Oudot, S. (2017). "Sliced Wasserstein kernel for persistence diagrams," in *Proceedings of Machine Learning Research* (Sydney, NSW).

Chazal, F., Cohen-Steiner, D., Glisse, M., Guibas, L., and Oudot, S. (2009). "Proximity of persistence modules and their diagrams," in *Proceedings of the Twenty-Fifth Annual Symposium on Computational Geometry* (Aarhus: ACM), 237–246. doi: 10.1145/1542362.1542407

Chazal, F., de Silva, V., Glisse, M., and Oudot, S. (2016). *The Structure and Stability of Persistence Modules*. SpringerBriefs in Mathematics. Cham: Springer. doi: 10.1007/978-3-319-42545-0_2

Chazal, F., Fasy, B., Lecci, F., Michel, B., Rinaldo, A., and Wasserman, L. (2017). Robust topological inference: distance to a measure and kernel distance. *J. Mach. Learn. Res*. 18:40. Available online at: http://jmlr.org/papers/v18/15-484.html

Chittajallu, D. R., Siekierski, N., Lee, S., Gerber, S., Beezley, J., Manthey, D., et al. (2018). "Vectorized persistent homology representations for characterizing glandular architecture in histology images," in *2018 IEEE 15th International Symposium on Biomedical Imaging* (Washington, DC). doi: 10.1109/ISBI.2018.8363562

Chung, Y.-M., Hu, C.-S., Lawson, A., and Smyth, C. (2018). "Topological approaches to skin disease image analysis," in *IEEE International Conference on Big Data (Big Data)* (Seattle, WA), 100–105. doi: 10.1109/BigData.2018.8622175

Chung, Y.-M., and Lawson, A. (2019). Persistence curves: a canonical framework for summarizing persistence diagrams. *arXiv: 1904.07768*.

Climent, J., Dimitrow, P., Fridlyand, J., Palacios, J., Siebert, R., Albertson, D. G., et al. (2007). Deletion of chromosome 11q predicts response to anthracycline-based chemotherapy in early breast cancer. *Cancer Res*. 67, 818–826. doi: 10.1158/0008-5472.CAN-06-3307

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2007). Stability of persistence diagrams. *Discrete Comput. Geom*. 37, 103–120. doi: 10.1007/s00454-006-1276-5

Crawford, L., Monod, A., Chen, A. X., Mukherjee, S., and Rabadán, R. (2020). Predicting clinical outcomes in glioblastoma: an application of topological and functional data analysis. *J. Am. Stat. Assoc*. 115, 1139–1150. doi: 10.1080/01621459.2019.1671198

De Silva, V., and Carlsson, G. (2004). "Topological estimation using witness complexes," in *Eurographics Symposium on Point-Based Graphics* (Zurich), 157–166. doi: 10.2312/SPBG/SPBG04/157-166

Dequeant, M.-L., Ahnert, S., Edelsbrunner, H., Fink, T. M., Glynn, E. F., Hattem, G., et al. (2008). Comparison of pattern detection methods in microarray time series of the segmentation clock. *PLoS ONE* 3:e2856. doi: 10.1371/journal.pone.0002856

DeWoskin, D., Climent, J., Cruz-White, I., Vazquez, M., Park, C., and Arsuaga, J. (2010). Applications of computational homology to the analysis of treatment response in breast cancer patients. *Topol. Appl*. 157, 157–164. doi: 10.1016/j.topol.2009.04.036

Dilsizian, S. E., and Siegel, E. L. (2014). Artificial intelligence in medicine and cardiac imaging: harnessing big data and advanced computing to provide personalized medical diagnosis and treatment. *Curr. Cardiol. Rep*. 16:441. doi: 10.1007/s11886-013-0441-8

Dimitriou, N., Arandjelović, O., and Caie, P. D. (2019). Deep learning for whole slide image analysis: an overview. *Front. Med*. 6:264. doi: 10.3389/fmed.2019.00264

Doyle, S., Agner, S., Madabhushi, A., Feldman, M., and Tomaszewski, J. (2008). "Automated grading of breast cancer histopathology using spectral clustering with textural and architectural image features," in *5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro* (Paris), 496–499. doi: 10.1109/ISBI.2008.4541041

Edelsbrunner, H., and Harer, J. (2010). *Computational Topology: An Introduction*. Providence, RI: American Mathematical Society. doi: 10.1090/mbk/069

Edelsbrunner, H., Letscher, D., and Zomorodian, A. (2002). Topological persistence and simplification. *Discr. Comput. Geom*. 28, 511–533. doi: 10.1007/s00454-002-2885-2

Emmett, K., Schweinhart, B., and Rabadan, R. (2016). "Multiscale topology of chromatin folding," in *Proceedings of the 9th EAI Conference on Bio-inspired Information and Communications Technologies* (New York, NY), 177–180. doi: 10.4108/eai.3-12-2015.2262453

Engers, R. (2007). Reproducibility and reliability of tumor grading in urological neoplasms. *World J. Urol*. 25, 595–605. doi: 10.1007/s00345-007-0209-0

Epstein, J. I., Zelefsky, M. J., Sjoberg, D. D., Nelson, J. B., Egevad, L., Magi-Galluzzi, C., et al. (2016). A contemporary prostate cancer grading system: a validated alternative to the Gleason score. *Eur. Urol*. 69, 428–435. doi: 10.1016/j.eururo.2015.06.046

Evans, S. M., Patabendi Bandarage, V., Kronborg, C., Earnest, A., Millar, J., and Clouston, D. (2016). Gleason group concordance between biopsy and radical prostatectomy specimens: a cohort study from Prostate Cancer Outcome Registry-Victoria. *Prost. Int*. 4, 145–151. doi: 10.1016/j.prnil.2016.07.004

Fass, L. (2008). Imaging and cancer: a review. *Mol. Oncol*. 2, 115–152. doi: 10.1016/j.molonc.2008.04.001

Ferracin, M., Pedriali, M., Veronese, A., Zagatti, B., Gafá, R., Magri, E., et al. (2011). MicroRNA profiling for the identification of cancers with unknown primary tissue-of-origin. *J. Pathol*. 225, 43–53. doi: 10.1002/path.2915

Freije, W. A., Castro-Vargas, F. E., Fang, Z., Horvath, S., Cloughesy, T., Liau, L. M., et al. (2004). Gene expression profiling of gliomas strongly predicts survival. *Cancer Res*. 64, 6503–6510. doi: 10.1158/0008-5472.CAN-04-0452

Garside, K., Henderson, R., Makarenko, I., and Masoller, C. (2019). Topological data analysis of high resolution diabetic retinopathy images. *PLoS ONE* 14:e217413. doi: 10.1371/journal.pone.0217413

Ghrist, R. (2008). Barcodes: the persistent topology of data. *Bull. Am. Math. Soc.* 45, 61–75. doi: 10.1090/S0273-0979-07-01191-3

Ghrist, R. W. (2014). *Elementary Applied Topology, Vol. 1*. Createspace Seattle.

Gidea, M. (2017). "Topology data analysis of critical transitions in financial networks," in *3rd International Winter School and Conference on Network Science* (Tel Aviv), 47–59. doi: 10.1007/978-3-319-55471-6_5

Gidea, M., Goldsmith, D., Katz, Y., Roldan, P., and Shmalo, Y. (2020). Topological recognition of critical transitions in time series of cryptocurrencies. *Phys. A* 548:123843. doi: 10.1016/j.physa.2019.123843

Gidea, M., and Katz, Y. (2018). Topological data analysis of financial time series: landscapes of crashes. *Phys. A* 491, 820–834. doi: 10.1016/j.physa.2017.09.028

Goodman, M., Ward, K. C., Osunkoya, A. O., Datta, M. W., Luthringer, D., Young, A. N., et al. (2012). Frequency and determinants of disagreement and error in gleason scores: a population-based study of prostate cancer. *Prostate* 72, 1389–1398. doi: 10.1002/pros.22484

Gretton, A., Borgwardt, K. M., Rasch, M. J., Smola, A., Schölkopf, B., and Smola Gretton, A. (2012). A kernel two-sample test. *J. Mach. Learn. Res.* 13, 723–773. Available online at: http://jmlr.org/papers/v13/gretton12a.html

Gu, J., and Taylor, C. R. (2014). Practicing pathology in the era of big data and personalized medicine. *Appl. Immunohistochem. Mol. Morphol.* 22, 1–9. doi: 10.1097/PAI.0000000000000022

Guan, Y., and Stephens, M. (2011). Bayesian variable selection regression for genome-wide association studies and other large-scale problems. *Ann. Appl. Stat.* 5, 1780–1815. doi: 10.1214/11-AOAS455

Han, W., Han, M. R., Kang, J. J., Bae, J. Y., Lee, J. H., Bae, Y. J., et al. (2006). Genomic alterations identified by array comparative genomic hybridization as prognostic markers in tamoxifen-treated estrogen receptor-positive breast cancer. *BMC Cancer* 6:92. doi: 10.1186/1471-2407-6-92

Hanahan, D., and Weinberg, R. A. (2000). *The Hallmarks of Cancer*. Technical report. 100, 57–70. doi: 10.1016/S0092-8674(00)81683-9

Hatcher, A. (2002). *Algebraic Topology*. Cambridge: Cambridge University Press.

Helpap, B., Kristiansen, G., Beer, M., Köllermann, J., Oehler, U., Pogrebniak, A., et al. (2012). Improving the reproducibility of the gleason scores in small foci of prostate cancer - Suggestion of diagnostic criteria for glandular fusion. *Pathol. Oncol. Res.* 18, 615–621. doi: 10.1007/s12253-011-9484-6

Herbrich, R., Smola, A., Bousquet, O., Schölkopf Bernhardschoelkopf, B., Gretton, A., and Schölkopf Gretton, B. (2005). Kernel methods for measuring independence. *J. Mach. Learn. Res.* 6, 2075–2129. Available online at: http://jmlr.org/papers/v6/gretton05a.html

Hong, B.-W., and Brady, M. (2003). "A topographic representation for mammogram segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Montreal, QC), 730–737. doi: 10.1007/978-3-540-39903-2_89

Horak, D., Maletić, S., and Rajković, M. (2009). Persistent homology of complex networks. *J. Stat. Mech.* 2009:P03034. doi: 10.1088/1742-5468/2009/03/P03034

Horlings, H. M., Lai, C., Nuyten, D. S., Halfwerk, H., Kristel, P., Van Beers, E., et al. (2010). Integration of DNA copy number alterations and prognostic gene expression signatures in breast cancer patients. *Clin. Cancer Res.* 16, 651–663. doi: 10.1158/1078-0432.CCR-09-0709

Humphrey, P. A. (2004). Gleason grading and prognostic factors in carcinoma of the prostate. *Modern Pathol.* 17, 292–306. doi: 10.1038/modpathol.3800054

Ishwaran, H., and Rao, J. S. (2005). Spike and slab variable selection: frequentist and bayesian strategies. *Ann. Stat.* 33, 730–773. doi: 10.1214/009053604000001147

Jain, R. K. (2005). Normalization of tumor vasculature: an emerging concept in antiangiogenic therapy. *Sci. Rev.* 307, 58–62. doi: 10.1126/science.1104819

Khan, A., El-Daly, H., Simmons, E., and Rajpoot, N. (2013). HyMaP: A hybrid magnitude-phase approach to unsupervised segmentation of tumor areas in breast cancer histology images. *J. Pathol. Inform.* 4(Suppl):S1. doi: 10.4103/2153-3539.109802

Khasawneh, F. A., and Munch, E. (2016). Chatter detection in turning using persistent homology. *Mech. Syst. Signal Process.* 70–71, 527–541. doi: 10.1016/j.ymssp.2015.09.046

Kimura, M., Obayashi, I., Takeichi, Y., Murao, R., and Hiraoka, Y. (2018). Non-empirical identification of trigger sites in heterogeneous processes using persistent homology. *Sci. Rep.* 8:3553. doi: 10.1038/s41598-018-21867-z

Kourou, K., Rigas, G., Papaloukas, C., Mitsis, M., and Fotiadis, D. I. (2020). Cancer classification from time series microarray data through regulatory Dynamic Bayesian Networks. *Comput. Biol. Med.* 116:103577. doi: 10.1016/j.compbiomed.2019.103577

Kusano, G., Hiraoka, Y., and Fukumizu, K. (2016). "Persistence weighted Gaussian kernel for topological data analysis," in *International Conference on Machine Learning* (New York, NY), 2004–2013.

Laurie, C. C., Laurie, C. A., Rice, K., Doheny, K. F., Zelnick, L. R., McHugh, C. P., et al. (2012). Detectable clonal mosaicism from birth to old age and its relationship to cancer. *Nat. Genet.* 44, 642–650. doi: 10.1038/ng.2271

Lawson, P., Sholl, A. B., Brown, J. Q., Fasy, B. T., and Wenk, C. (2019). Persistent homology for the quantitative evaluation of architectural features in prostate cancer histology. *Sci. Rep.* 9:1139. doi: 10.1038/s41598-018-36798-y

Li, C. H., and Tam, P. K. (1998). An iterative algorithm for minimum cross entropy thresholding. *Pattern Recogn. Lett.* 19, 771–776. doi: 10.1016/S0167-8655(98)00057-9

Li, M., An, H., Angelovici, R., Bagaza, C., Batushansky, A., Clark, L., Coneva, V., et al. (2018). Topological data analysis as a morphometric method: Using persistent homology to demarcate a leaf morphospace. *Front. Plant Sci.* 9:553. doi: 10.3389/fpls.2018.00553

Lockwood, S., and Krishnamoorthy, B. (2015). "Topological features in cancer gene expression data," in *Pacific Symposium on Biocomputing* (Kohala Coast).

Macenko, M., Niethammer, M., Marron, J., Borland, D., Woosley, J. T., and Guan, X. (2009). "A method for normalizing histology slides for quantitative analysis," in *IEEE International Symposium on Biomedical Imaging: From Nano to Macro* (Boston, MA), 1107–1110. doi: 10.1109/ISBI.2009.5193250

Maley, C. C., Galipeau, P. C., Finley, J. C., Wongsurawat, V. J., Li, X., Sanchez, C. A., et al. (2006). Genetic clonal diversity predicts progression to esophageal adenocarcinoma. *Nat. Genet.* 38, 468–473. doi: 10.1038/ng1768

Maley, C. C., and Reid, B. J. (2005). Natural selection in neoplastic progression of Barrett's esophagus. *Semin. Cancer Biol.* 15, 474–483. doi: 10.1016/j.semcancer.2005.06.004

Marquard, A. M., Birkbak, N. J., Thomas, C. E., Favero, F., Krzystanek, M., Lefebvre, C., et al. (2015). TumorTracer: a method to identify the tissue of origin from the somatic mutations of a tumor specimen. *BMC Med. Genomics* 8:58. doi: 10.1186/s12920-015-0130-0

Marron, J. S., and Todd, M. (2007). Distance-weighted discrimination. *J. Am. Stat. Assoc.* 102, 1267–1271. doi: 10.1198/016214507000001120

Mileyko, Y., Mukherjee, S., and Harer, J. (2011). Probability measures on the space of persistence diagrams. *Inverse Probl.* 27:124007. doi: 10.1088/0266-5611/27/12/124007

Mischaikow, K., and Nanda, V. (2013). Morse theory for filtrations and efficient computation of persistent homology. *Discr. Comput. Geom.* 50, 330–353. doi: 10.1007/s00454-013-9529-6

Moran, S., Martínez-Cardús, A., Sayols, S., Musulén, E., Bala ná, C., Estival-Gonzalez, A., et al. (2016). Epigenetic profiling to classify cancer of unknown primary: a multicentre, retrospective analysis. *Lancet Oncol.* 17, 1386–1395. doi: 10.1016/S1470-2045(16)30297-2

Munkres, J. R. (1984). *Elements of Algebraic Topology*. Menlo Park, CA: Addison-Wesley Publishing Company.

Neve, R. M., Chin, K., Fridlyand, J., Yeh, J., Baehner, F. L., Fevr, T., et al. (2006). A collection of breast cancer cell lines for the study of functionally distinct cancer subtypes. *Cancer Cell* 10, 515–527. doi: 10.1016/j.ccr.2006.10.008

Nicolau, M., Levine, A. J., and Carlsson, G. (2011). Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proc. Natl. Acad. Sci. U.S.A.* 108, 7265–7270. doi: 10.1073/pnas.1102826108

Nicolau, M., Tibshirani, R., Børresen-Dale, A. L., and Jeffrey, S. S. (2007). Disease-specific genomic analysis: identifying the signature of pathologic biology. *Bioinformatics* 23, 957–965. doi: 10.1093/bioinformatics/btm033

Nielson, J. L., Cooper, S. R., Yue, J. K., Sorani, M. D., Inoue, T., Yuh, E. L., et al. (2017). Uncovering precision phenotype-biomarker associations in traumatic brain injury using topological data analysis. *PLoS ONE* 12:e169490. doi: 10.1371/journal.pone.0169490

Nutt, C. L., Mani, D. R., Betensky, R. A., Tamayo, P., Cairncross, J. G., Ladd, C., et al. (2003). Gene expression-based classification of malignant gliomas correlates better with survival than histological classification. *Cancer Res.* 63, 1602–1607.

Obayashi, I., and Hiraoka, Y. (2018). Persistence diagrams with linear machine learning models. *J. Appl. Comput. Topol.* 1, 421–449. doi: 10.1007/s41468-018-0013-5

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., and Harrington, H. A. (2017). A roadmap for the computation of persistent homology. *EPJ Data Sci.* 6:17. doi: 10.1140/epjds/s13688-017-0109-5

Oudot, S. Y. (2015). *Persistence Theory: From Quiver Representations to Data Analysis, Vol. 209 of Mathematical Surveys and Monographs*. Providence, RI: American Mathematical Society. doi: 10.1090/surv/209

Oyama, A., Hiraoka, Y., Obayashi, I., Saikawa, Y., Furui, S., Shiraishi, K., et al. (2019). Hepatic tumor classification using texture and topology analysis of non-contrast-enhanced three-dimensional T1-weighted MR images with a radiomics approach. *Sci. Rep.* 9:8764. doi: 10.1038/s41598-019-45283-z

Pereira, C. M., and de Mello, R. F. (2015). Persistent homology for time series and spatial data clustering. *Expert Syst. Appl.* 42, 6026–6038. doi: 10.1016/j.eswa.2015.04.010

Phillips, H. S., Kharbanda, S., Chen, R., Forrest, W. F., Soriano, R. H., Wu, T. D., et al. (2006). Molecular subclasses of high-grade glioma predict prognosis, delineate a pattern of disease progression, and resemble stages in neurogenesis. *Cancer Cell* 9, 157–173. doi: 10.1016/j.ccr.2006.02.019

Qaiser, T., Sirinukunwattana, K., Nakane, K., Tsang, Y. W., Epstein, D., and Rajpoot, N. (2016). "Persistent homology for fast tumor segmentation in whole slide histology images," in *Procedia Computer Science, Vol. 90* (Loughborough: Elsevier B.V.), 119–124. doi: 10.1016/j.procs.2016.07.033

Rabadán, R., Mohamedi, Y., Rubin, U., Chu, T., Alghalith, A. N., Elliott, O., et al. (2020). Identification of relevant genetic alterations in cancer using topological data analysis. *Nat. Commun.* 11, 1–10. doi: 10.1101/2020.01.30.922310

Ravishanker, N., and Chen, R. (2019). Topological data analysis (TDA) for time series. *arXiv: 1909.10604*.

Reinhard, E., Ashikhmin, M., Gooch, B., and Shirley, P. (2001). Color transfer between images. *IEEE Comput. Graph. Appl.* 21, 34–41. doi: 10.1109/38.946629

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. (2015). "A stable multi-scale kernel for topological machine learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Boston, MA), 4741–4748. doi: 10.1109/CVPR.2015.7299106

Reuter, J. A., Spacek, D. V., and Snyder, M. P. (2015). High-throughput sequencing technologies. *Mol. Cell* 58, 586–597. doi: 10.1016/j.molcel.2015.05.004

Roychowdhury, S., Iyer, M. K., Robinson, D. R., Lonigro, R. J., Wu, Y. M., Cao, X., et al. (2011). Personalized oncology through integrative high-throughput sequencing: a pilot study. *Sci. Transl. Med.* 3, 1–12. doi: 10.1126/scitranslmed.3003161

Rucco, M., Merelli, E., Herman, D., Ramanan, D., Petrossian, T., Falsetti, L., et al. (2015). Using Topological Data Analysis for diagnosis pulmonary embolism. *arXiv:1409.5020v1*. 9, 41–55.

Seemann, L., Shulman, J., and Gunaratne, G. H. (2012). A robust topology-based algorithm for gene expression profiling. *ISRN Bioinform.* 2012:381023. doi: 10.5402/2012/381023

Siddiqui, S., Shikotra, A., Richardson, M., Doran, E., Choy, D., Bell, A., et al. (2018). Airway pathological heterogeneity in asthma: visualization of disease microclusters using topological data analysis. *J. Aller. Clin. Immunol.* 142, 1457–1468. doi: 10.1016/j.jaci.2017.12.982

Singh, G., Mémoli, F., and Carlsson, G. (2007). "Topological methods for the analysis of high dimensional data sets and 3D object recognition," in *Eurographics Symposium on Point-Based Graphics* (Prague).

Singh, N., Couture, H. D., Marron, J. S., Perou, C., and Niethammer, M. (2014). "Topological descriptors of histology images," in *Machine Learning in Medical Imaging* (Boston, MA). doi: 10.1007/978-3-319-10581-9_29

Søndergaard, D., Nielsen, S., Pedersen, C. N., and Besenbacher, S. (2017). Prediction of primary tumors in cancers of unknown primary. *J. Integr. Bioinform.* 14:20170013. doi: 10.1515/jib-2017-0013

Stack, E. C., Wang, C., Roman, K. A., and Hoyt, C. C. (2014). Multiplexed immunohistochemistry, imaging, and quantitation: a review, with an assessment of Tyramide signal amplification, multispectral imaging and multiplex analysis. *Methods* 70, 46–58. doi: 10.1016/j.ymeth.2014.08.016

Stolz, B. J., Harrington, H. A., and Porter, M. A. (2017). Persistent homology of time-dependent functional networks constructed from coupled time series. *Chaos* 27:047410. doi: 10.1063/1.4978997

Suwinski, P., Ong, C. K., Ling, M. H., Poh, Y. M., Khan, A. M., and Ong, H. S. (2019). Advancing personalized medicine through the application of whole exome sequencing and big data analytics. *Front. Genet.* 10:49. doi: 10.3389/fgene.2019.00049

Tahmassebi, A., Schulte, M. H., Gandomi, A. H., Goudriaan, A. E., McCann, I., and Meyer-Baese, A. (2018). "Deep learning in medical imaging: FMRI big data analysis via convolutional neural networks," in *ACM International Conference Proceeding Series* (Pittsburgh, PA: Association for Computing Machinery), 1–4. doi: 10.1145/3219104.3229250

Truesdale, M. D., Cheetham, P. J., Turk, A. T., Sartori, S., Hruby, G. W., Dinneen, E. P., et al. (2011). Gleason score concordance on biopsy-confirmed prostate cancer: is pathological re-evaluation necessary prior to radical prostatectomy? *BJU Int.* 107, 749–754. doi: 10.1111/j.1464-410X.2010.09570.x

Truong, M., Slezak, J. A., Lin, C. P., Iremashvili, V., Sado, M., Razmaria, A. A., et al. (2013). Development and multi-institutional validation of an upgrading risk tool for Gleason 6 prostate cancer. *Cancer* 119, 3992–4002. doi: 10.1002/cncr.28303

Truong, P. (2017). *An exploration of topological properties of high-frequency onedimensional financial time series data using TDA* (Ph.D. thesis). KTH Royal Institute of Technology, Stockholm, Sweden.

Tschandl, P., Rosendahl, C., and Kittler, H. (2018). Data descriptor: the HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci. Data* 5:180161. doi: 10.1038/sdata.2018.161

Turner, K., Mukherjee, S., and Boyer, D. M. (2014). Persistent homology transform for modeling shapes and surfaces. *Inf. Inference* 3, 310–344. doi: 10.1093/imaiai/iau011

Verhaak, R. G., Hoadley, K. A., Purdom, E., Wang, V., Qi, Y., Wilkerson, M. D., et al. (2010). Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in PDGFRA, IDH1, EGFR, and NF1. *Cancer Cell* 17, 98–110. doi: 10.1016/j.ccr.2009.12.020

Vikeså, J., Møller, A. K., Kaczkowski, B., Borup, R., Winther, O., Henao, R., et al. (2015). Cancers of unknown primary origin (CUP) are characterized by chromosomal instability (CIN) compared to metastasis of know origin. *BMC Cancer* 15:151. doi: 10.1186/s12885-015-1128-x

Wadhwa, R. R., Williamson, D. F. K., Dhawan, A., and Scott, J. G. (2018). TDAstats: R pipeline for computing persistent homology in topological data analysis. *J. Open Source Softw.* 3:860. doi: 10.21105/joss.00860

Wang, G. (2016). A perspective on deep imaging. *IEEE Access* 4, 8914–8924. doi: 10.1109/ACCESS.2016.2624938

Weinberger, S. (2014). The complexity of some topological inference problems. *Found. Comput. Math.* 14, 1277–1285. doi: 10.1007/s10208-013-9152-1

Wilkerson, A. C., Chintakunta, H., and Krim, H. (2014). "Computing persistent features in big data: a distributed dimension reduction approach," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (Florence), 11–15. doi: 10.1109/ICASSP.2014.6853548

Xiaohua, C., Brady, M., and Rueckert, D. (2004). "Simultaneous segmentation and registration for medical image," in *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Saint-Malo), 663–670. doi: 10.1007/978-3-540-30135-6_81

Yu, K. H., Zhang, C., Berry, G. J., Altman, R. B., Ré, C., Rubin, D. L., et al. (2016). Predicting non-small cell lung cancer prognosis by fully automated microscopic pathology image features. *Nat. Commun.* 7, 1–10. doi: 10.1038/ncomms12474

Yuan, Y., Failmezger, H., Rueda, O. M., Raza Ali, H., Gräf, S., Chin, S. F., et al. (2012). Quantitative image analysis of cellular heterogeneity in breast tumors complements genomic profiling. *Sci. Transl. Med.* 4:157ra143. doi: 10.1126/scitranslmed.3004330

Zhang, Y., Brady, M., and Smith, S. (2001). Segmentation of brain MR images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE Trans. Med. Imaging* 20:45. doi: 10.1109/42.906424

Zhao, B., Pritchard, J. R., Lauffenburger, D. A., and Hemann, M. T. (2014). Addressing genetic tumor heterogeneity through computationally

predictive combination therapy. *Cancer Discov*. 4, 166–174. doi: 10.1158/2159-8290.CD-13-0465

Zhou, X., Carbonetto, P., and Stephens, M. (2013). Polygenic modeling with bayesian sparse linear mixed models. *PLoS Genet*. 9:e1003264. doi: 10.1371/journal.pgen.1003264

Zomorodian, A. (2010). "The tidy set: a minimal simplicial set for computing homology of clique complexes [extended abstract]," in *Computational Geometry (SCG'10)* (New York, NY: ACM), 257–266. doi: 10.1145/1810959.1811004

Zomorodian, A., and Carlsson, G. (2005). Computing persistent homology. *Discrete Comput. Geom*. 33, 249–274. doi: 10.1007/s00454-004-1146-y

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Applied Mathematics and Statistics

ORIGINAL RESEARCH
published: 22 April 2021
doi: 10.3389/fams.2021.651467

Optimization of Spectral Wavelets for Persistence-Based Graph Classification

Ka Man Yim and Jacob Leygonie*

Mathematical Institute, University of Oxford, Oxford, United Kingdom

A graph's spectral wavelet signature determines a filtration, and consequently an associated set of extended persistence diagrams. We propose a framework that optimizes the choice of wavelet for a dataset of graphs, such that their associated persistence diagrams capture features of the graphs that are best suited to a given data science problem. Since the spectral wavelet signature of a graph is derived from its Laplacian, our framework encodes geometric properties of graphs in their associated persistence diagrams and can be applied to graphs without a priori node attributes. We apply our framework to graph classification problems and obtain performances competitive with other persistence-based architectures. To provide the underlying theoretical foundations, we extend the differentiability result for ordinary persistent homology to extended persistent homology.

Keywords: topological data analysis, graph classification, graph Laplacian, extended persistent homology, spectral wavelet signatures, radial basis neural network

OPEN ACCESS

Edited by:
Umberto Lupo,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Bastian Rieck,
ETH Zürich, Switzerland
Primoz Skraba,
Queen Mary University of London, United Kingdom

****Correspondence:***
Ka Man Yim
yim@maths.ox.ac.uk

Specialty section:
This article was submitted to Mathematics of Computation and Data Science, a section of the journal Frontiers in Applied Mathematics and Statistics

Received: *09 January 2021*
Accepted: *24 February 2021*
Published: *22 April 2021*

Citation:
Yim KM and Leygonie J (2021) Optimization of Spectral Wavelets for Persistence-Based Graph Classification. Front. Appl. Math. Stat. 7:651467. doi: 10.3389/fams.2021.651467

1. INTRODUCTION

1.1. Background

Graph classification is a challenging problem in machine learning. Unlike data represented in Euclidean space, there is no easily computable notion of distance or similarity between graphs. As such, graph classification requires techniques that lie beyond mainstream machine learning techniques focused on Euclidean data. Much research has been conducted on methods such as graph neural networks (GNNs) [1] and graph kernels [2, 3] that embed graphs in Euclidean space in a consistent manner.

Recently, *persistent homology* [4, 5] has been applied as a feature map that explicitly represents topological and geometric features of a graph as a set of *persistence diagrams* (a.k.a. *barcodes*). In the context of our discussion, the persistent homology of a graph $G = (V, E)$ depends on a vertex function $f : V \to \mathbb{R}$. In the case where a vertex function is not given with the data, several schemes have been proposed in the literature to assign vertex functions to graphs in a consistent way. For example, vertex functions can be constructed using local geometric descriptions of vertex neighborhoods, such as discrete curvature [6], heat kernel signatures [7] and Weisfeiler–Lehman graph kernels [8].

However, it is often difficult to know *a priori* whether a heuristic vertex assignment scheme will perform well in addressing different data science problems. For a single graph, we can optimize the vertex function over $|V|$ many degrees of freedom in $\mathbb{R}^V$. In recent years, there have been many other examples of persistence optimization in data science applications. The first two examples of persistence optimization are the computation of Fréchet mean of barcodes using gradients on Alexandrov spaces [9], and that of point cloud inference [10], where a point cloud is optimized so

that its barcode fits a target fixed barcode. The latter is an instance of topological inverse problems (see Oudot and Solomon [11] for a recent overview of such). Another inverse problem is that of surface reconstruction [12]. Besides, in the context of shape matching [13], persistence optimization is used in order to learn an adequate function between shapes. Finally, there are also many recent applications of persistence optimization in Machine Learning, such as the incorporation of topological information in Generative Modeling [14–16] or in Image Segmentation [17, 18], the design of topological losses for Regularization in supervised learning [19] or for dimension reduction [20].

Each of these applications can be thought of as minimizing a certain *loss* function over a manifold $\mathcal{M}$ of parameters:

$$\min_{\theta\in\mathcal{M}} \mathcal{L}(\theta),$$

where $\mathcal{L}:\mathcal{M} \to \mathbf{Bar}^N \to \mathbb{R}$ factors through the space $\mathbf{Bar}^N$ of N-tuples of barcodes. The aim is to find the parameter θ that best fits the application at hand. Gradient descent is a very popular approach in minimization, but it requires the ability to differentiate the loss function. In fact, Leygonie et al. [21] provide notions of differentiability for maps in and out **Bar** that are compatible with smooth calculus, and show that the loss functions $\mathcal{L}$ corresponding the applications cited in the above paragraph are generically differentiable. The use of (stochastic) gradient descent is further legitimated by Carriere et al. [22], where convergence guarantees on persistence optimization problems are devised, using a recent study of stratified non-smooth optimization problems [23]. In practice, the minimization of $\mathcal{L}$ can be unstable due to its non-convexity and partial non-differentiability. Some research has been conducted in order to smooth and regularize the optimization procedure [24, 25].

In a supervised learning setting, we want to optimize our vertex function assignment scheme over many individual graphs in a dataset. Since graphs may not share the same vertex set and come in different sizes, optimizing over the $|V|$ degrees of freedom of any one graph is not conducive to learning a vertex function assignment scheme that can generalize to another graph. The degrees of freedom in any practical vertex assignment scheme should be independent of the number of vertices of a graph. However, a framework for parameterizing and optimizing the vertex functions of many graphs over a common parameter space $\mathcal{M}$ is not immediately apparent.

The first instance of a graph persistence optimization framework (GFL) [26] uses a one layer graph isomorphism network (GIN) [1] to parameterize vertex functions. The GIN learns a vertex function by exploiting the local topology around each vertex. In this paper, we propose a different framework for assigning and parameterizing vertex functions, based on a graph's Laplacian operator. Using the Laplacian, we can explicitly take both local and global structures of the graph into consideration in an interpretable and transparent manner.

1.2. Outline and Contributions

We address the issue of vertex function parameterization and optimization using *wavelet signatures*. Wavelet signatures are vertex functions derived from the eigenvalues and eigenvectors of the graph Laplacian and encode multiscale geometric information about the graph [27]. The wavelet signature of a graph is dependent on a choice of wavelet $g:\mathbb{R} \to \mathbb{R}$, a function on the eigenvalues of the graph's Laplacian matrix. We can thus obtain a parameterization of vertex functions for any graph $\mathrm{F}:\mathcal{M} \to \mathbb{R}^V$ by parameterizing g. Consequently, the *extended persistence* of a graph—which has only four non-trivial persistence diagrams—can be varied over the parameter space $\mathcal{M}$. If we have a function $\mathsf{Out}:\mathbf{Bar}^4 \to \mathbb{R}$ on persistence diagrams that we wish to minimize, we can optimize over $\mathcal{M}$ to minimize the loss function

$$\mathcal{L}:\mathcal{M} \xrightarrow{\mathrm{F}} \mathbb{R}^V \xrightarrow{\mathsf{EPH}} \mathbf{Bar}^4 \xrightarrow{\mathsf{Out}} \mathbb{R}. \quad (1)$$

If $\mathcal{L}$ is generically differentiable, we can optimize the wavelet signature parameters $\theta \in \mathcal{M}$ using gradient descent methods. We illustrate an application of this framework to a graph classification problem in **Figure 1**, where the loss function $\mathcal{L}$ is the classification error of a graph classification prediction model based on the graph's extended persistence diagrams.

In section 2, we describe the assignment of vertex functions $\mathrm{F}:\mathcal{M} \to \mathbb{R}^V$ by reviewing the definition of wavelet signatures. While spectral wavelets have been used in graph neural network architectures that predict vertex features [1] and compress vertex functions [28], they have not been considered in a persistent homology framework for graph classification. We describe several ways to parameterize wavelets. We also show in Proposition 2.2 that wavelet signatures are independent of the choice of eigenbasis of the graph Laplacian from which it is derived, ensuring that it is well-defined. We prove this result in Appendix B in **Supplementary Material**.

In section 3, we describe the theoretical basis for optimizing the *extended* persistent homology of a vertex function $\mathsf{EPH}:\mathbb{R}^V \to \mathbf{Bar}^4$ and elucidate what it means for $\mathcal{L}$ to be differentiable. In Proposition 3.3, we generalize the differentiability formalism of ordinary persistence [21] to extended persistence. We prove this result in Appendix A in **Supplementary Material**.

Finally, in section 4, we apply our framework to graph classification problems on several benchmark datasets. We show that our model is competitive with state-of-the-art persistence-based models. In particular, optimizing the vertex function appreciably improves the prediction accuracy on some datasets.

2. FILTER FUNCTION PARAMETERIZATION

We describe our recipe for assigning vertex functions to any simplicial graph $G = (V, E)$ based on a parameterized spectral wavelet, the first part F of the loss function

$$\mathcal{L}:\mathcal{M} \xrightarrow{\mathrm{F}} \mathbb{R}^V \xrightarrow{\mathsf{EPH}} \mathbf{Bar}^4 \xrightarrow{\mathsf{Out}} \mathbb{R}.$$

(Equation 1 recalled)

Our recipe is based on a graph's wavelet signature, a vertex function derived from the graph's Laplacian. The wavelet signature also depends on a so-called 'wavelet function' in

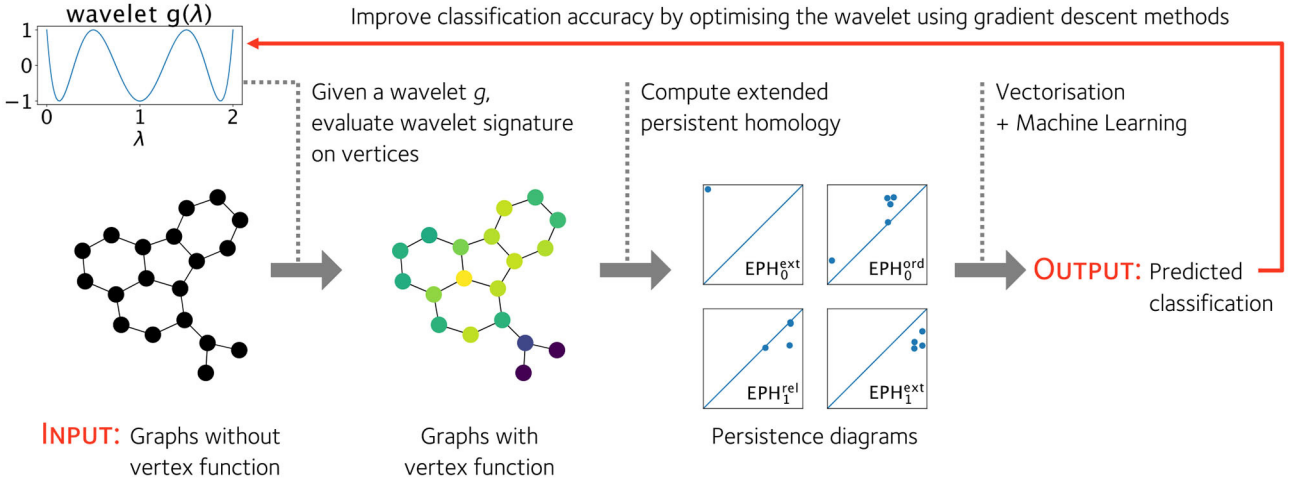


FIGURE 1 | Given a wavelet $g : \mathbb{R} \to \mathbb{R}$, we can equip any graph with a non-trivial vertex function. This allows us to compute the extended persistence diagrams of a graph and use the diagrams as features of the graph to predict a graph's classification in some real world setting. The wavelet g can be optimized to improve the classification accuracy of a graph classification pipeline based on the extended persistence diagrams of a graph's vertex function.

$g : \mathbb{R} \to \mathbb{R}$, which is independent of the graph. By modulating the wavelet, we can jointly vary the wavelet signature across many graphs. We parameterize the wavelet using a finite linear combination of basis functions, such that the wavelet signature can be manipulated in a computationally tractable way. In the following section, we define the wavelet signature and describe our linear approach to wavelet parameterization.

2.1. Wavelet Signatures

The wavelet signature is a vertex function initially derived from wavelet transforms of vertex functions on graphs [29], a generalization of wavelet transforms for square integrable functions on Euclidean space [30, 31] for signal analysis [32]. Wavelet signatures for graphs have been applied to encode geometric information about meshes of 3D shapes [27, 32]. Special cases of wavelets signatures, such as the heat kernel signature [33] and wave kernel signature [34], have also been applied to describe graphs and 3D shapes [35, 36].

The wavelet signature of a graph is constructed from the graph's Laplacian operator. A graph's normalized Laplacian $L \in \mathbb{R}^{V \times V}$ is a symmetric positive semi-definite matrix, whose entries are given by

$$L_{uv} = \begin{cases} 1 & u = v \\ -\frac{1}{\sqrt{k_u k_v}} & (u, v) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where k_u is the degree of vertex u. The Laplacian's eigenvalues λ and eigenvectors ϕ are known to encode various topological and geometric information about the graph [37, 38]; for example, the number of zero eigenvalues corresponds to the number of connected components of the graph. The spectrum of the normalized Laplacian have real eigenvalues in $[0, 2]$ [37]. As such, any function $g : \mathbb{R} \to \mathbb{R}$ evaluated on the eigenvalues need only be defined on $[0, 2]$. Moreover, functions on a compact domain are easily parameterized using convenient bases.

Definition 2.1. (Wavelet Signature [27]) Let $L \in \mathbb{R}^{V \times V}$ be the normalized Laplacian of a simplical graph $G = (V, E)$. Let $\phi_1, \ldots, \phi_{|V|}$ be an orthonormal eigenbasis for L and $\lambda_1, \ldots, \lambda_{|V|}$ be their corresponding eigenvalues. The wavelet signature $W : \mathbb{R}^{[0,2]} \to \mathbb{R}^V$ maps a function $g : [0, 2] \to \mathbb{R}$, which we refer to as a *wavelet*, to a vertex function $W(g) \in \mathbb{R}^V$ linearly, where the value of $W(g)$ on vertex v is given by

$$W(g)_v = \sum_{i=1}^{|V|} g(\lambda_i)(\phi_i)_v^2, \tag{3}$$

and $(\phi_i)_v$ denotes the component of eigenvector ϕ_i corresponding to vertex v.

If the eigenvalues of L have geometric multiplicity one (i.e., their eigenspaces are one dimensional), then the orthonormal eigenvectors are uniquely defined up to a choice of sign. It is then apparent from Equation (3) that the wavelet signature is independent of the choice of sign. However, if some eigenvalues have geometric multiplicity greater than one, then the orthonormal eigenvectors of L are uniquely defined up to orthonormal transformations in the individual eigenspaces. However, the wavelet signature is well-defined even when the multiplicities of eigenvalues are greater than one. This is the content of the next Proposition, whose proof is deferred to Appendix B in **Supplementary Material**.

PROPOSITION 2.2. *The wavelet signature of a graph is independent of the choice of orthonormal eigenbasis for the Laplacian.*

Remark 2.3. In addition to the traditional view of wavelets from a spectral signal processing perspective [29], we can also relate the

wavelet signature of a vertex v to the degrees of vertices in some neighborhood of v prescribed by g. Consider a wavelet $g:[0,2]\to\mathbb{R}$. On a finite graph G, the normalized Laplacian L has at most $|V|$ many distinct eigenvalues. As such, there exists a polynomial $\hat{g}(x)=\sum_{n=0}^{p}a_n x^n$ of finite order that interpolates g at the eigenvalues $g(\lambda_i)=\hat{g}(\lambda_i)$. Therefore, $W(g)=W(\hat{g})$. Moreover, the vertex values assigned by $W(\hat{g})$ are the diagonal entries of the matrix polynomial $\hat{g}(L)$:

$$\begin{aligned}\hat{g}(L)_{vv} &= \sum_{n=0}^{p} a_n\left(L^n\right)_{vv} = \sum_{i=1}^{|V|}\hat{g}(\lambda_i)(\phi_i)_v^2 = \sum_{i=1}^{|V|} g(\lambda_i)(\phi_i)_v^2 \\ &= W(g)_{vv}. \end{aligned} \tag{4}$$

Furthermore, we can also write the matrix polynomial $\hat{g}(L)$ as a matrix polynomial in $A=I-L$, the *normalized adjacency matrix*. From the definition of L, we can compute the diagonal entry of a monomial A^r corresponding to vertex v as an inverse degree weighted count of paths[1] $[v_0,v_1,\ldots,v_r]$ on the graph which begin and end on vertex $v=v_0=v_r$ [39]:

$$\left(A^r\right)_{vv}=\frac{1}{k_v}\sum_{[v,v_1,\ldots,v_{r-1},v]}\left(\prod_{l=1}^{r-1}\frac{1}{k_{v_l}}\right). \tag{5}$$

By expressing the wavelet signature as a matrix polynomial in A, we see that g controls how information at different length scales of the graph contribute to the wavelet signature. For instance, if g were an order p polynomial, then $W(g)_v$ only takes the degrees of vertices that are $\lfloor p/2\rfloor$ away from v into account. As a corollary, since $W(g)$ can be specified by replacing g with a polynomial $\hat{g}$ of order at most $|V|-1$, the wavelet signature at a vertex is only dependent on the subgraph of G that is within $\lfloor |V|-1\rfloor/2$ steps away from v.

2.2. Parameterizing the Wavelet

We see from Remark 2.3 that the choice of wavelet g determines how the topology and geometry of the graph is reflected in the vertex function. Though the space of wavelets is potentially infinite dimensional, here we only consider wavelets $g_\theta(x)$ that are parameterized by parameters θ in a finite dimensional manifold, so that we can easily optimize them using computational methods. In particular, we focus on wavelets written as a linear combination of m *basis functions* $h_1,\ldots,h_m:[0,2]\to\mathbb{R}$

$$g_\theta(x):=\sum_{j=1}^{m}\theta_j h_j(x) \tag{6}$$

This parameterization of wavelets in turn defines a parameterization of vertex functions $\mathrm{F}:\mathbb{R}^m\to\mathbb{R}^V$ for our optimization pipeline in Equation (1)

$$\mathrm{F}:\quad \theta\in\mathbb{R}^m \quad\longmapsto\quad \mathrm{F}(\theta):=W\left(g_\theta\right)\quad\in\mathbb{R}^V. \tag{7}$$

[1]Here a path refers to a sequences of vertices that are connected to the next vertex in the sequence by an edge.

Since $W(g)$ is a linear function of the wavelet g, F is a linear transformation:

$$\mathrm{F}(\theta)=W\left(\sum_{j=1}^{m}\theta_j h_j(x)\right)=\sum_{j=1}^{m}\theta_j W\left(h_j\right). \tag{8}$$

We can write F as a $|V|\times m$ matrix acting on a vector $[\theta_1,\ldots\theta_m]^\intercal\in\mathbb{R}^m$, whose columns are the vertex functions $W(h_j)$.

Example 2.4 (Chebyshev Polynomials)**.** Any Lipschitz continuous function on an interval can be well-approximated by truncating its Chebyshev series at some finite order [40]. The Chebyshev polynomials $T_n:[-1,1]\to\mathbb{R}$

$$T_n(x)=\cos(n\arccos(x))\quad n\in\mathbb{N}_{\geq 0}. \tag{9}$$

form an orthonormal set of functions. We can thus consider $h_j(\lambda)=T_j(\lambda-1)$, $j=0,2,\ldots,m$ as a naïve basis for wavelets. We exclude $T_1(x)=x$ in the linear combination as $W(T_1(1-x))=0$ for graphs without self loops.

Example 2.5 (Radial Basis Functions)**.** In the machine learning community, a *radial function* refers loosely to a continuous monotonically decreasing function $\rho:\mathbb{R}_{\geq 0}\to\mathbb{R}_{\geq 0}$. There are many possible choices for ρ, for example, the inverse multiquadric

$$\rho(r)=\left(\left(\frac{r}{\epsilon}\right)^2+1\right)^{-\frac{1}{2}} \tag{10}$$

where $\epsilon\neq 0$ is a width parameter. We can obtain a naïve wavelet basis $h_j(x)=\rho\left(\|x-x_j\|\right)$ using copies of ρ offset by a collection of centroids $x_j\in\mathbb{R}$ along $\mathbb{R}$. In general, the centroids are parameters that could be optimized, but we fix them in this study. This parameterization can be considered as a *radial basis function neural network*. RBNNs are well-studied in function approximation and subsequently machine learning; we refer readers to [41, 42] for further details.

2.3. The Choice of Wavelet Basis

The choice of basis functions determines the space of wavelet signatures and also the numerical stability of the basis function coefficients which serve as the wavelet signature parameters. The stability of the parameterization depends on the graphs as much as the choice of wavelet basis $h_1,\ldots,h_m$. We can analyse the stability of a parameterization F by its the singular value decomposition

$$\mathrm{F}=\sum_{k=1}^{r}\sigma_k u_k v_k^\intercal \tag{11}$$

where $\sigma_1,\ldots,\sigma_r$ are the non-zero singular values of the matrix, and $u_k\in\mathbb{R}^{|V|}$ and $v_k\in\mathbb{R}^m$ are orthonormal sets of vectors, respectively. If the distribution of singular values span many orders of magnitude, we say the parameterization is *ill-conditioned*. An ill-conditioned parameterization interferes with

the convergence of gradient descent algorithms on a loss function evaluated on wavelet signatures. We discuss the relationship between the conditioning of F and the stability of gradient descent in detail in Remark 2.7.

We empirically observe that the coefficients of a naïve choice of basis functions, such as Chebyshev polynomials or radial basis functions, are numerically ill-conditioned. In **Figure A2** (**Appendix** in **Supplementary Material**.), we can see that the singular values of radial basis function and Chebyshev polynomial parameterizations, respectively, are distributed across a large range on the logarithmic scale for some datasets of graphs in machine learning. We address this problem by picking out a new wavelet basis

$$h'_k(x) = \frac{1}{\sigma_k} \sum_{j=1}^{m} (v_k)_j h_j(x), \quad k = 1, \ldots, r, \tag{12}$$

where σ_k are the singular values of F and v_k are the associated vectors in $\mathbb{R}^m$ from the singular value decomposition of matrix F in Equation (11). Then the parameterization $\mathrm{F}' : \mathbb{R}^r \to \mathbb{R}^V$

$$\mathrm{F}'(\theta') = \sum_{k=1}^{r} \theta'_k W(h'_k). \tag{13}$$

have singular values equal to one, since this is a linear combination of orthonormal vectors $u_k \in \mathbb{R}^V$:

$$W(h'_k) = \sum_{j=1}^{m} \frac{1}{\sigma_k} (v_k)_j W(h_j) = \frac{1}{\sigma_k} \mathrm{F} v_k = u_k. \tag{14}$$

As an example, we plot the new wavelet basis h'_k derived from a 12 parameter radial basis function parameterization for the MUTAG dataset in **Figure A3** in **Appendix B** in **Supplementary Material**.

Remark 2.6 (Learning a Wavelet Basis for Wavelet Signatures on Multiple Graphs)**.** In the case where the wavelet coefficients parameterize the wavelet signatures over graphs $G_1, \ldots, G_N$, we can view the maps $F_1, \ldots, F_N$ that map wavelet basis coefficients to vertex functions of graphs $G_1, \ldots, G_N$, respectively, as a parameterization for the disjoint union $\bigsqcup_i G_i$:

$$f = \begin{bmatrix} f_1 \\ \vdots \\ f_N \end{bmatrix} = \begin{bmatrix} \mathrm{F}_1 \\ \vdots \\ \mathrm{F}_N \end{bmatrix} \theta =: \mathrm{F}\theta. \tag{15}$$

We can then perform a singular value decomposition of the parameterization F on $\bigsqcup_i G_i$ and derive a new, well-conditioned basis.

Remark 2.7 (Why the Conditioning of F Matters)**.** Let us optimize a loss function $\mathcal{L}$ on the parameter space of wavelet coefficients θ using a gradient descent algorithm. In a gradient descent step of step size s, the wavelet coefficients are updated to $\theta \mapsto \theta - s\nabla_\theta \mathcal{L}$. Using the singular value decomposition of F (Equation 11), we can write

$$\nabla_\theta \mathcal{L} = \nabla_\theta f^\intercal \nabla_f \mathcal{L} = \mathrm{F}^\intercal \nabla_f \mathcal{L} = \sum_{k=1}^{r} \sigma_k \langle \nabla_f \mathcal{L}, u_k \rangle v_k. \tag{16}$$

The change in the vertex function is simply the matrix F applied to the change in wavelet parameters. Hence, the vertex function is updated to $f \mapsto f - s\mathrm{F}\nabla_\theta \mathcal{L}$, where

$$\mathrm{F}\nabla_\theta \mathcal{L} = \sum_{k=1}^{r} \sigma_k \langle \nabla_f \mathcal{L}, u_k \rangle \mathrm{F} v_k = \sum_{k=1}^{r} \sigma_k^2 \langle \nabla_f \mathcal{L}, u_k \rangle u_k. \tag{17}$$

If the loss function $\mathcal{L}$ has large second derivatives– for example, due to non-linearities in the function on persistence diagrams $\mathsf{Out} : \mathbf{Bar}^4 \to \mathbb{R}$—the projections $\langle \nabla_f \mathcal{L}, u_k \rangle$ in Equations (16) and (17) may change dramatically from one gradient descent update to another. If the smallest singular value is much smaller than the largest, then updates to the wavelet signature can be especially unstable throughout the optimization process. This source of instability can be removed if we choose a parameterization with uniform singular values $\sigma_k = 1$. In this case, the update to f is simply the projection of $\nabla_f \mathcal{L}$ onto the space of wavelet signatures spanned by $u_1, \ldots, u_r$, without any distortion introduced by non-uniform singular values:

$$f \mapsto f - s \sum_{k=1}^{r} \langle u_k, \nabla_f \mathcal{L} \rangle u_k. \tag{18}$$

3. EXTENDED PERSISTENT HOMOLOGY

The homology of a given graph is a computable vector space whose dimension counts the number of connected components or cycles in the graph. Finer information can be retained by filtering the graph and analyzing the evolution of the homology throughout the filtration. This evolution is described by a set of *extended persistence diagrams* (a.k.a. *extended barcodes*), a multiset of points $\langle b, d \rangle$ that record the birth b and death of homological features in the filtration. In this section, we begin by summarizing these constructions. We refer the reader to Zomorodian and Carlsson [4], Edelsbrunner and Harer [5], and Cohen-Steiner et al. [43] for full treatments of the theory of Persistence.

Compared to *ordinary persistence*, extended persistence is a more informative and convenient feature map for graphs. Extended persistence encodes strictly more information than ordinary persistence. For instance, the cycles of a graph are represented as points with $d = \infty$ in ordinary persistence. Thus, only the birth coordinate b of such points contain useful information about the cycles. In contrast, the corresponding points in extended persistence are each endowed with a finite death time d, thus associating extra information to the cycles. The points at infinity in ordinary persistence also introduce obstacles to vectorization procedures, as often arbitrary finite cutoffs are needed to 'tame' the persistence diagrams before vectorization.

3.1. Extended Persistent Homology

Let $G = (V, E)$ be a finite graph without double edges and self-loops. For the purposes of this paper, the associated *extended persistent homology* is a map

$$\mathsf{EPH} : \mathbb{R}^V \to \mathbf{Bar}^4$$

from functions $f \in \mathbb{R}^V$ on its vertices to the space of four *persistence diagrams* or *barcodes*, which we define below. The map arises from a *filtration* of the graph, a sequential attachment of vertices and edges in ascending or descending order of f. We extend f on each edge $e = (v, v')$ by the maximal value of f over the vertices v and v', and we then let $G_t \subset G$ be the sub graph induced by vertices taking value less than t. Then we have the following sequence of inclusions:

$$\emptyset \longrightarrow \cdots \longrightarrow G_s \xrightarrow{s\leq t} G_t \longrightarrow \cdots \longrightarrow G. \quad (19)$$

Similarly, the sub graphs $G^t \subset G$ induced by vertices taking value greater than t assemble into a sequence of inclusions:

$$G \longleftarrow \cdots \longleftarrow G^s \xleftarrow{s\leq t} G^t \longleftarrow \cdots \longleftarrow \emptyset\ . \quad (20)$$

The changes in the topology of the graph along the filtration in ascending and descending order of f can be detected by its *extended persistence module*, indexed over the poset $\mathbb{R} \cup \{\infty\} \cup \mathbb{R}^{\mathsf{op}}$:

$$\mathsf{V}_p(f): \begin{array}{ccccccccc} 0 = H_p(\emptyset) & \longrightarrow \cdots \longrightarrow & H_p(G_s) & \xrightarrow{s\leq t} & H_p(G_t) & \longrightarrow \cdots \longrightarrow & H_p(G) \\ & & & & \downarrow\cong & & \\ 0 = H_p(G,G) & \longleftarrow \cdots \longleftarrow & H_p(G,G^s) & \xleftarrow{s\leq t} & H_p(G,G^t) & \longleftarrow \cdots \longleftarrow & H_p(G,\emptyset) \end{array}, \quad (21)$$

where H_p is the singular (relative) homology functor in degree $p \in 0, 1$ with coefficients in a fixed field, chosen to be $\mathbb{Z}/2\mathbb{Z}$ in practice. In general terms, the modules $\mathsf{V}_0(f)$ and $\mathsf{V}_1(f)$ together capture the evolution of the connected components and loops in the sub graphs of G induced by the function f.

Each module $\mathsf{V}_p(f)$ is completely characterized by a finite multi-set $\mathsf{EPH}_p(f)$ of pairs of real numbers $\langle b, d\rangle$ called *intervals* representing the birth and death of homological features. Following Cohen-Steiner et al. [44], the intervals in $\mathsf{EPH}_p(f)$ are further partitioned according to the type of homological feature they represent:

$$\mathsf{EPH}_p(f) = \underbrace{\{\langle b,d\rangle \mid b < d < \infty\}}_{=\mathsf{EPH}_p^{\mathsf{ord}}(f)} \sqcup \underbrace{\{\langle b,d\rangle \mid b < \infty < d\}}_{=\mathsf{EPH}_p^{\mathsf{ext}}(f)} \sqcup \underbrace{\{\langle b,d\rangle \mid \infty < b < d\}}_{=\mathsf{EPH}_p^{\mathsf{rel}}(f)}. \quad (22)$$

Each of the three finite multiset $\mathsf{EPH}_p^{\mathsf{k}}(f)$, for $k \in \{\text{ord}, \text{ext}, \text{rel}\}$, is an element in the space **Bar** of so-called *barcodes* or *persistence diagrams*. However, $\mathsf{EPH}_0^{\mathsf{rel}}(f)$ and $\mathsf{EPH}_1^{\mathsf{ord}}(f)$ being trivial for graphs, we refer to the collection of four remaining persistence diagrams

$$\mathsf{EPH}(f) = \left[\mathsf{EPH}_0^{\mathsf{ord}}(f), \mathsf{EPH}_0^{\mathsf{ext}}(f), \mathsf{EPH}_1^{\mathsf{ext}}(f), \mathsf{EPH}_1^{\mathsf{rel}}(f)\right] \in \mathbf{Bar}^4 \quad (23)$$

as the *extended barcode* or *extended persistence diagram* of f. We have thus defined the *extended persistence map*

$$\mathsf{EPH}: \mathbb{R}^V \to \mathbf{Bar}^4.$$

Remark 3.1. If we only apply homology to the filtration of Equation (19), we get an *ordinary persistence module* indexed over the real line, which is essentially the first row in Equation (21). This module is characterized by a unique barcode $\mathsf{PH}_p(f) \in \mathbf{Bar}$. We refer to the map

$$\mathsf{PH}: f \in \mathbb{R}^V \longmapsto \left[\mathsf{PH}_0(f), \mathsf{PH}_1(f)\right] \in \mathbf{Bar}^2 \quad (24)$$

as the *ordinary persistence map*.

3.2. Differentiability of Extended Persistence

The extended persistence map can be shown to be locally Lipschitz by the Stability theorem [44]. The Rademacher theorem states that any real-valued function that is locally Lipschitz is differentiable on a full measure set. Thus, so is our loss function

$$\mathcal{L}: \mathcal{M} \xrightarrow{\mathrm{F}} \mathbb{R}^V \xrightarrow{\mathsf{EPH}} \mathbf{Bar}^4 \xrightarrow{\mathsf{Out}} \mathbb{R}\ .$$

(Equation 1 recalled)

as long as Out and F are smooth or locally Lipschitz[2]. If a loss function $\mathcal{L}$ is locally Lipschitz, we can use stochastic gradient descent as a paradigm for optimization. Nonetheless, the theorem above does not rule out dense sets of non differentiability in general.

In this section, we show that the set where EPH is not differentiable is not pathological. Namely, we show that EPH is *generically* differentiable, i.e., differentiable on an open dense subset. This property guarantees that local gradients yield reliable descent directions in a neighborhood of the current iterate. We recall from Leygonie et al. [21] the definition of differentiability for maps to barcodes.

We call a map $\mathrm{F}: \mathcal{M} \to \mathbb{R}^V$ a *parameterization*, as it corresponds to a selection of filter functions over G parameterized by the manifold $\mathcal{M}$. Then $B := \mathsf{EPH} \circ \mathrm{F}$ is the barcode valued map whose differentiability properties are of interest in applications.

Definition 3.2. A map $B: \mathcal{M} \to \mathbf{Bar}$ on a smooth manifold $\mathcal{M}$ is said to be differentiable at $\theta \in \mathcal{M}$ if for some neighborhood U of θ, there exists a finite collection of differentiable maps[3] $b_i, d_i: U \to \mathbb{R} \cup \{\infty\}$, called a *local coordinate system* for B at θ, such that

$$\forall \theta' \in U,\ B(\theta') = \left\{\left\langle b_i(\theta'), d_i(\theta')\right\rangle \mid b_i(\theta') \neq d_i(\theta')\right\}.$$

For $N \in \mathbb{N}$, we say that a map $B: \mathcal{M} \to \mathbf{Bar}^N$ is differentiable at θ if all its components are so.

In Leygonie et al. [21], it is proven that the composition $\mathsf{PH} \circ \mathrm{F}$ is generically differentiable as long as F is so. It is possible to show that $\mathsf{EPH} \circ \mathrm{F}$ is generically differentiable along the

[2] In practice, a locally Lipschitz Out can be constructed out of Lipschitz stable vectorization methods, such as Persistence Landscapes [45] and Persistence Images [46].

[3] By convention, a differentiable map that takes the value ∞ is constant.

same lines, but we rather provide an alternative argument in the Appendix. Namely, we rely on the fact that the extended persistence of G can be decoded from the ordinary persistence of the cone complex $\mathsf{C}(G)$, a connection first noted in Cohen-Steiner et al. [44] for computational purposes.

PROPOSITION 3.3. *Let $F: \mathcal{M} \to \mathbb{R}^V$ be a generically differentiable parameterization. Then the composition $\mathsf{EPH} \circ F$ is generically differentiable.*

For completeness, the proof provided in the Appendix treats the general case of a finite simplicial complex K of arbitrary dimension.

4. BINARY GRAPH CLASSIFICATION

We investigate whether optimizing the extended persistence of wavelet signatures can be usefully applied to graph classification problems, where persistence diagrams are used as features to predict discrete, real life attributes of networks. In this setting, we aim to learn $\theta \in \mathcal{M}$ that minimize the classification error of graphs over a training dataset.

We apply our wavelet optimization framework to classification problems on the graph datasets `MUTAG` [47, 48], `COX2` [49], `DHFR` [49], `NCI1` [50, 51], `PROTEINS` [52, 53], and `IMDB-B` [54]. The former five datasets are biochemical molecules and `IMDB-B` is a collection of social ego networks. In our models, we use persistence images [46] as a fixed vectorization method and use a feed forward neural network to map the persistence images to classification labels. We also include the eigenvalues of the graph Laplacian as additional features; model particulars are described in the sections below.

To illustrate the effect of wavelet optimization on different classification problems, we also perform a set of *control experiments* where for the same model architecture, we fix the wavelet and only optimize the parameters of the neural network. The control experiment functions as a baseline against which we assess the efficacy of wavelet optimization.

We benchmark our results with two existing persistence based architectures, `PersLay` [7] and `GFL` [26]. `Perslay` optimizes the vectorization parameters and use two heat kernel signatures as fixed rather than optimizable vertex functions for computing extended persistence. `GFL` optimizes and parameterizes vertex functions using a graph isomorphism network [1], and computes *ordinary* sublevel and superlevel set persistence instead of extended persistence.

4.1. Model Architecture

We give a high level description of our model and relegate details and hyperparameter choices of the vectorization method and neural network architecture to Appendix C in **Supplementary Material**. In our setting, the extended persistence diagrams of the optimizable wavelet signatures for each graph are vectorized as persistence images. We also include the static persistence images of a *fixed* heat kernel signature, $W(e^{-0.1x})$, as an additional set of features, alongside some non-persistence features. Both the optimized and static persistence diagrams are transformed into the persistence images using identical hyperparameters. We feed the optimizable and static persistence images into two separate convolutional neural networks (CNNs) with the same architecture. Similarly, we feed the non-persistence features as a vector into a separate multilayer perceptron. The outputs of the CNNs are concatenated with the outputs of the multi-layer perceptron. Finally, an affine transformation sends the concatenated vector to a real number whose sign determines the binary classification.

TABLE 1 | Binary classification accuracy of our model where we vary whether non-Persistence features are included and whether the wavelet is optimized.

	Persistence only		Non-persistence features incl.	
	Control	**Wavelet Opt.**	**Control**	**Wavelet Opt.**
`MUTAG`	89.2 ± 0.6	89.8 ± 0.8	89.0 ± 0.6	90.4 ± 0.4
`COX2`	79.6 ± 1.0	79.4 ± 0.7	80.8 ± 1.0	80.8 ± 1.0
`DHFR`	79.9 ± 0.4	80.4 ± 0.4	80.3 ± 0.9	81.0 ± 0.9
`NCI1`	73.7 ± 0.2	74.3 ± 0.5	74.3 ± 0.3	74.4 ± 0.3
`PROTEINS`	72.9 ± 0.3	73.0 ± 0.4	74.5 ± 0.4	74.6 ± 0.6
`IMDB-B`	68.3 ± 0.5	68.6 ± 0.7	71.6 ± 0.9	72.0 ± 0.7

*The reported accuracies are the mean over 10 ten-folds, recorded at epochs reported in **Table C1**. We also provide standard deviations of the 10 mean accuracies of each ten-fold. See section 4.1.2 for the particulars about the non-persistence features.*

4.1.1. Wavelet Parametserization

We choose a space of wavelets spanned by 12 inverse multiquadric radial basis functions

$$h_j(x) = \left(\left(\frac{x - x_j}{\epsilon} \right)^2 + 1 \right)^{-\frac{1}{2}} \tag{25}$$

whose centroids x_j are located at $x_j = 2(j-1)/9$, $j = 0, \ldots, 11$. The width parameter is chosen to be the distance between the centroids, $\epsilon = 2/9$. On each dataset, we derive a numerically stable parameterization using the procedure described in section 2.2; the parameters we optimize are the coefficients of the new basis given by Equation (12). We initialize the parameters by fitting them via least squares to the heat kernel signature $W(e^{-10x})$ on the whole dataset of graphs.

4.1.2. Non-Persistence Features

We also incorporate the eigenvalues of the normalized Laplacian as additional, fixed features of the graph. Since the number of eigenvalues for a given graph is equal to the number of vertices, it differs between graphs in the same dataset. To encode the information represented in the eigenvalues as a fixed length vector, we first sort the eigenvalues into a time-series; we then compute the log path signature of the time series up to level four, which is a fixed length vector in $\mathbb{R}^8$. The log-signature captures the geometric features of the path; we refer readers to Chevyrev and Kormilitzin [55] for details about path signatures. For `IMDB-B` in particular, we also include the maxima and minima of the heat kernel signatures $W(e^{-10x})$ and $W(e^{-0.1x})$, respectively, of each graph.

4.2. Experimental Set Up

We employ a *10 ten-fold* test-train split scheme on each dataset to measure the accuracy of our model. Each ten-fold is a set of ten

TABLE 2 | Binary classification accuracy on datasets of graphs.

	Non-persistence state-of-the-art				Persistence based				
	P-SAN [57]	RetGK [58]	GIN [1]	FGSD [59]	PWL [8]	GFL [26]	Perslay [7]	Control This paper	Wavelet Opt. This paper
Node attr.	Yes		No		Yes		No		
MUTAG	92.6	90.3 ± 1.1	89.4	92.1	90.5 ± 1.3	–	–	89.8 ± 0.9	89.0±0.6	**90.4**±1.3
COX2	–	81.4 ± 0.6	–	–	–	–	–	**80.9** ± 1.0	80.8±0.4	80.8±1.0
DHFR	–	82.5 ± 0.8	–	–	–	–	–	80.3 ± 0.8	80.0±0.4	**81.0**±0.9
NCI1	78.6	84.5 ± 0.2	82.7	79.8	85.6 ± 0.3	77.2	71.2	73.5 ± 0.3	74.3±0.3	**74.4**±0.3
PROTEINS	75.9	78.0 ± 0.3	76.2	73.4	75.9 ± 0.8	73.4	74.1	**74.8** ± 0.3	74.5±0.4	74.6±0.6
IMDB-B	71.0	72.3 ± 0.6	75.1	73.6	73.0 ± 1.0	–	**74.5**	71.2 ± 0.7	71.6±0.9	72.0±0.7
# Ten-folds	10	10	1	1	10	1	1	10	10	10

The best accuracy of persistence-based models without *using node attributes is made bold for each dataset. The performance of our model is reported in the column Wavelet Opt. on the right hand side. The accuracies of the control model, where the wavelet parameters are fixed to the initial values, are shown in the column Control. Both these models use additional features (see section 4.1.2). The accuracies of our model are the means over 10 ten-folds, recorded at epochs reported in* ***Table C1****. We also provide the standard deviations of the 10 mean accuracies of each ten-fold. For other architectures, we indicate whether their accuracies were reported as averages over 1 ten-fold or 10 ten-fold in the bottom row of the table. To avoid confusion, we leave out the errors reported for* `P-SAN`, `GIN` *and* `GFL` *and refer the reader to the original sources, as they were calculated using a different formula. Errors were not reported in [59] for* `FGSD`.

experiments, corresponding to a random partition of the dataset into ten portions. In each experiment, a different portion is selected as the test set while the model is trained on the remaining nine portions. We perform 10 ten-folds to obtain a total of 10×10 experiments, and report the accuracy of the classifier as the average accuracy over 100 such experiments. The epochs at which the accuracies were measured are specified in **Table C1**.

Across all experiments, we use binary cross entropy as the loss function. We use the `Adam` optimizer [56] with learning rate `lr = 1e-3` to optimize the parameters of the neural network. The wavelet parameters are updated using stochastic gradient descent with learning rate `lr = 1e-2`, for all datasets except for `IMDB-B`, where the learning rate is set to `lr = 1e-1`. The batch sizes for each experiment are shown in **Table C2**. In all experiments, we stop the optimization of wavelet parameters at epoch 50 while the neural network parameters continue to be optimized.

We use the `GUDHI` library to compute persistence, and make use of the optimization and machine learning library `PyTorch` for the construction of the graph classifications models.

4.3. Results and Discussion

In **Table 1**, we present the classification accuracies of our model. For each dataset, we perform four experiments using our model, varying whether the wavelet parameter is optimized and whether additional features are included. In **Table 2**, we show the test accuracy of our model alongside two persistence-based graph classification architectures, `Perslay` and `GFL`, as well as other state-of-the-art graph classification architectures.

We first compare the performances of our model between cases where we optimize and fix the wavelets. In **Table 1**, we see that on `MUTAG` and `DHFR`, optimizing the wavelet improves the classification accuracy regardless of whether extra features are included. On `NCI1`, wavelet optimization improves the classification accuracy only persistence features are included. When we include non-persistence features in the model, the performances of the optimized and control models are statistically indistinguishable for `NCI1`, suggesting that the non-persistence features play a more significant role in the classification. As for `COX2`, `PROTEINS`, and `IMDB-B`, optimizing the wavelet coefficients do not bring about statistically significant improvements. This indicates that the initial wavelet signature—the heat kernel signature $W(e^{-10x})$—is a locally optimal choice of wavelet for our neural network classifier.

We now compare our architecture to other persistence based architectures, `Perslay` and `GFL`, where node attributes are excluded from their vertex function models. Except on `PROTEINS`, our wavelet optimized model matches or exceeds `Perslay`. While our model architecture and choice of wavelet initialization is similar to that of `Perslay`, we differ in two important respects. `Perslay` fixes the vertex functions but optimizes the weights assigned to points on the persistence diagrams, as well as the parameters of the persistence images. Our improvements on `Perslay` for `MUTAG`, `DHFR`, and `IMDB-B` indicate that vertex function optimization yields improvements that cannot be obtained through vectorization optimization alone on some datasets of graphs.

Compared to `GFL` (without node attributes), both `Perslay` and our architecture achieves similar or higher classification accuracies on `PROTEINS` and `NCI1`. This supports wavelet signatures being viable models for vertex functions on those datasets. On the other hand, both `Perslay` and our model lag behind `GFL` on `IMDB-B`. We attribute this to the fact that `IMDB-B`, unlike the other bioinformatics datasets, consists of densely connected graphs. The graphs in `IMDB-B` have diameter at most two and 14% of the graphs are cliques. This fact has two consequences. First, we expect the one-layer GIN used in `GFL`—a

local topology summary—to be highly effective in optimizing for the salient features of a graph with small diameter. Second, the extended persistence modules for cliques have zero persistence, since all vertices are assigned the same function value due to symmetry. In contrast, ordinary persistence used in GFL is able to capture the cycles in a complete graph as points with infinite persistence.

Compared to non-persistence state-of-the-art architectures in **Table 1**, our model achieves competitive accuracies on MUTAG, COX2, and DHFR. For NCI1 and PROTEINS, all persistence architectures listed that exclude additional node attributes perform poorly in comparison, though PWL was able to achieve leading results with node attributes.

All in all, we observe that wavelet signatures can be an effective parameterization of vertex functions when we use extended persistence as features for graph classification. In particular, on some bioinformatics datasets, we show that optimizing the wavelet signature can lead to improvements in classification accuracy. The wavelet signature approach is complementary to the GFL approach to vertex function parameterization as they show strengths on different datasets.

5. CONCLUSION

We have presented a framework for equipping any graph G with a set of extended persistence diagrams $\mathsf{EPH} \circ \mathrm{F} : \mathcal{M} \to \mathbf{Bar}^4$ parameterized over a manifold $\mathcal{M}$, a parameter space for the graph's wavelet signature. We described how wavelet signatures can be parameterized and interpreted. Given a function on extended persistence diagrams $\mathsf{Out} : \mathbf{Bar}^4 \to \mathbb{R}$ that is differentiable, we have shown how a loss function $\mathcal{L} = \mathsf{Out} \circ \mathsf{EPH} \circ \mathrm{F}$ can be generically differentiable with respect to $\theta \in \mathcal{M}$ as $\mathcal{L}$. Thus, we can apply gradient descent methods to optimize the extended persistence diagrams of a graph to minimize $\mathcal{L}$.

We applied this framework to a graph classification architecture where the wavelet signature is optimized for classification accuracy. We are able to demonstrate an increase in accuracy on several benchmark datasets where the wavelet is optimized, and perform competitively with state-of-the-art persistence based graph classification architectures.

DATA AVAILABILITY STATEMENT

The code for the computational experiments in section 4 can be found in the GitHub repository https://github.com/kmyim/Persistence_Opt_Spectral_Wavelets. The datasets we use are publicly available at the repository TUDatasets https://chrsmrrs.github.io/datasets/ [60].

AUTHOR CONTRIBUTIONS

The overall framework was jointly conceived by both authors. KY was responsible for developing wavelet signatures as a vertex function parameterization framework, along with the experimental design and analysis. The proof of the differentiability of extended persistence is due to JL. Both authors participated in the writing of the article.

FUNDING

KY was funded by the EPSRC Centre For Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) with industrial sponsorship from Elsevier. JL was funded by the EPSRC grant EP/R513295/1. Both authors are members of the Centre for Topological Data Analysis, which is supported by the EPSRC grant New Approaches to Data Science: Application Driven Topological Data Analysis EP/R018472/1.

ACKNOWLEDGMENTS

The authors would like to thank Ulrike Tillmann and Heather Harrington for their close guidance and thoughtful advice on this project. In addition, the authors would like to thank Vidit Nanda, Peter Grindrod CBE, Andrew Mellor, Steve Oudot, Mathieu Carrière, and Theo Lacombe for fruitful discussions on this subject. Finally, we are indebted to the reviewers for their thoughtful and constructive comments, which led to many improvements of the paper.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fams.2021.651467/full#supplementary-material

REFERENCES

1. Xu B, Shen H, Cao Q, Qiu Y, Cheng X. Graph wavelet neural network. *arXiv [Preprint]*. (2019) arXiv:1904.07785.
2. Vishwanathan SVN, Schraudolph NN, Kondor R, Borgwardt KM. Graph kernels. *J Mach Learn. Res.* (2010) 11:1201–42.
3. Shervashidze N, Vishwanathan SVN, Petri T, Mehlhorn K, Borgwardt K. Efficient graphlet kernels for large graph comparison. In: van Dyk D, Welling M, editors. *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, Vol. 5*. Clearwater Beach, FL: PMLR (2009). p. 488–95. Available online at: http://proceedings.mlr.press/v5/shervashidze09a.html
4. Zomorodian A, Carlsson G. Computing persistent homology. *Discr Comput Geom.* (2005) 33:249–74. doi: 10.1007/s00454-004-1146-y
5. Edelsbrunner H, Harer J. Persistent homology-a survey. *Contemp Mathe.* (2008) 453:257–82. doi: 10.1090/conm/453/08802
6. Zhao Q, Wang Y. Learning metrics for persistence-based summaries and applications for graph classification. In: Wallach H. Larochelle H, Beygelzimer A, dAlché-Buc F, Fox E, Garnett R, editors. *Advances in Neural Information Processing Systems,* Vol. 32. Red Hook, NY: Curran Associates, Inc. (2019). p. 9859–70. Available online at: https://proceedings.neurips.cc/paper/2019/file/12780ea688a71dabc284b064add459a4-Paper.pdf
7. Carrière M, Chazal F, Ike Y, Lacombe T, Royer M,and Umeda Y. Perslay: a neural network layer for persistence diagrams and new graph topological

signatures. In: Chiappa S, Calandra R, editors. *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics,* Vol. 108. PMLR (2020). p. 2786–96. Available online at: http://proceedings.mlr.press/v108/carriere20a.html.
8. Rieck B, Bock C, Borgwardt K. A persistent weisfeiler-lehman procedure for graph classification. In: *International Conference on Machine Learning.* PMLR (2019). p. 5448–58.
9. Turner K, Mileyko Y, Mukherjee S, Harer J. Fréchet means for distributions of persistence diagrams. *Discr Comput Geom.* (2014) 52:44–70. doi: 10.1007/s00454-014-9604-7
10. Gameiro M, Hiraoka Y, Obayashi I. Continuation of point clouds via persistence diagrams. *Phys D.* (2016) 334:118–32. doi: 10.1016/j.physd.2015.11.011
11. Oudot S, Solomon E. Inverse problems in topological persistence. In: Baas NA. Carlsson GE, Quick G, Szymik M, Thaule M, editors. *Topological Data Analysis.* Cham: Springer (2020). p. 405–33. doi: 10.1007/978-3-030-43408-3_16
12. Brüel-Gabrielsson R, Ganapathi-Subramanian V, Skraba P, Guibas LJ. Topology-aware surface reconstruction for point clouds. *Comp Graph Forum.* (2020) 39:197–207. doi: 10.1111/cgf.14079
13. Poulenard A, Skraba P, Ovsjanikov M. Topological function optimization for continuous shape matching. *Comp Graph Forum.* (2018) 37:13–25. doi: 10.1111/cgf.13487
14. Moor M, Horn M, Rieck B, Borgwardt K. Topological autoencoders. In: *International Conference on Machine Learning.* PMLR (2020). p. 7045–54.
15. Hofer C, Kwitt R, Niethammer M, Dixit M. Connectivity-optimized representation larning via persistent homology. In: Chaudhuri K, Salakhutdinov R, editors. *Proceedings of the 36th International Conference on Machine Learning.* Long Beach, CA: PMLR (2019). p. 2751–60. Available online at: http://proceedings.mlr.press/v97/hofer19a.html
16. Gabrielsson RB, Nelson BJ, Dwaraknath A, Skraba P. A topology layer for machine learning. In: *International Conference on Artificial Intelligence and Statistics.* PMLR (2020). p. 1553–63.
17. Hu X, Li F, Samaras D, Chen C. Topology-preserving deep image segmentation. In: *Advances in Neural Information Processing Systems.* Red Hook, NY: Curran Associates, Inc. (2019). p. 5658–69. Available online at: https://proceedings.neurips.cc/paper/2019/file/12780ea688a71dabc284b064add459a4-Paper.pdf
18. Clough JR, Oksuz I, Byrne N, Schnabel JA, King AP. Explicit topological priors for deep-learning based image segmentation using persistent homology. In: *International Conference on Information Processing in Medical Imaging.* Hong Kong: Springe (2019). p. 16–28. doi: 10.1007/978-3-030-20351-1_2
19. Chen C, Ni X, Bai Q, Wang Y. A topological regularizer for classifiers via persistent homology. In: *The 22nd International Conference on Artificial Intelligence and Statistics.* Naha (2019). p. 2573–82.
20. Kachan O. Persistent homology-based projection pursuit. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops.* (2020). p. 856–7. doi: 10.1109/CVPRW50498.2020.00436
21. Leygonie J, Oudot S, Tillmann U. A framework for differential calculus on persistence barcodes. *arXiv [Preprint].* (2019) arXiv:1910.00960.
22. Carriere M, Chazal F, Glisse M, Ike Y, Kannan H. A note on stochastic subgradient descent for persistence-based functionals: convergence and practical aspects. *arXiv preprint arXiv:201008356.* (2020).
23. Davis D, Drusvyatskiy D, Kakade S, Lee JD. Stochastic subgradient method converges on tame functions. *Found Comput Math.* (2020) 20:119–54. doi: 10.1007/s10208-018-09409-5
24. Solomon E, Wagner A, Bendich P. A fast and robust method for global topological functional optimization. *arXiv [Preprint].* (2020) arXiv:2009.08496.
25. Corcoran P, Deng B. Regularization of persistent homology gradient computation. *arXiv [Preprint].* (2020) arXiv:2011.05804.
26. Hofer C, Graf F, Rieck B, Niethammer M, Kwitt R. Graph filtration learning. In: Daumé III H, and Singh A, editors. *Proceedings of the 37th International Conference on Machine Learning.* PMLR (2020). p. 4314–23. Available online at: http://proceedings.mlr.press/v119/hofer20b.html
27. Li C, Hamza AB. A multiresolution descriptor for deformable 3D shape retrieval. *Visu Comp.* (2013) 29:513–524. doi: 10.1007/s00371-013-0815-3
28. Rustamov RM, Guibas LJ. Wavelets on graphs via deep learning. In: Stanković L, Sejdic E, editors. *Vertex-Frequency Analysis of Graph Signals.* Cham: Springer (2019). p. 207–22. doi: 10.1007/978-3-030-03574-7
29. Hammond DK, Vandergheynst P, Gribonval R. Wavelets on graphs via spectral graph theory. *Appl Computat Harm Anal.* (2011) 30:129–50. doi: 10.1016/j.acha.2010.04.005
30. Graps A. An introduction to wavelets. *IEEE Comput Sci Eng.* (1995) 2:50–61. doi: 10.1109/99.388960
31. Chui CK, Chan AK, Liu SJ. *An Introduction to Wavelets.* San Diego, CA: Academic Press (1992).
32. Akansu AN, Haddad RA. *Multiresolution Signal Decomposition : Transforms, Subbands, and Wavelets, 2nd Edn.* San Diego, CA: Academic Press (2001). doi: 10.1016/B978-012047141-6/50002-1
33. Sun J, Ovsjanikov M, Guibas L. A concise and provably informative multi-scale signature based on heat diffusion. *Comp Graph Forum.* (2009) 28:1383–92. doi: 10.1111/j.1467-8659.2009.01515.x
34. Aubry M, Schlickewei U, Cremers D. The wave kernel signature: a quantum mechanical approach to shape analysis. in: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops).* IEEE (2011). p. 1626–33. doi: 10.1109/ICCVW.2011.6130444
35. Bronstein MM, Kokkinos I. Scale-invariant heat kernel signatures for non-rigid shape recognition. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* San Francisco, CA: IEEE (2010). p. 1704–11. doi: 10.1109/CVPR.2010.5539838
36. Hu N, Rustamov RM, Guibas L. Stable and informative spectral signatures for graph matching. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* Columbus, OH: IEEE (2014). p. 2305–12. doi: 10.1109/CVPR.2014.296
37. Chung FRK, Graham FC. *Spectral Graph Theory.* Providence, RI: American Mathematical Society (1997).
38. Biyikoglu, Leydold T. J., Stadler PF. *Laplacian Eigenvectors of Graphs: Perron-Frobenius and Faber-Krahn Type Theorems.* Berlin: Springer (2007). doi: 10.1007/978-3-540-73510-6
39. Newman M. *Networks.* Oxford University Press (2018). doi: 10.1093/oso/9780198805090.001.0001
40. Trefethen LN, Bau D III. *Numerical Linear Algebra.* Vol. 50. SIAM (1997). doi: 10.1137/1.9780898719574
41. Chen S, Cowan CFN. Grant PM. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Trans Neural Netw.* (1991) 2:302–9. doi: 10.1109/72.80341
42. Park J, Sandberg IW. Universal approximation using radial-basis-function networks. *Neural Comput.* (1991) 3:246–57. doi: 10.1162/neco.1991.3.2.246
43. Cohen-Steiner D, Edelsbrunner H, Harer J. Stability of persistence diagrams. *Discr Comput Geom.* (2007) 37:103–20. doi: 10.1007/s00454-006-1276-5
44. Cohen-Steiner D, Edelsbrunner H, Harer J. Extending persistence using Poincaré and Lefschetz duality. *Found Comput Mathe.* (2009) 9:79–103. doi: 10.1007/s10208-008-9027-z
45. Bubenik P. Statistical topological data analysis using persistence landscapes. *J Mach Learn Res.* (2015) 16:77–102.
46. Adams H, Emerson T, Kirby M, Neville R, Peterson C, Shipman P, et al. Persistence images: a stable vector representation of persistent homology. *J Mach Learn Res.* (2017) 18:218–52.
47. Debnath AK, Lopez de Compadre RL, Debnath G, Shusterman AJ, Hansch C. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *J Med Chem.* (1991) 34:786–97. doi: 10.1021/jm00106a046
48. Kriege N, Mutzel P. Subgraph matching kernels for attributed graphs. In: *Proceedings of the 29th International Coference on International Conference on Machine Learning.* Madison, WI: Omnipress (2012). p. 291–8.
49. Sutherland JJ, O'brien LA, Weaver DF. Spline-fitting with a genetic algorithm: A method for developing classification structure- activity relationships. *J Chem Inf Comp Sci.* (2003) 43:1906–15. doi: 10.1021/ci034143r
50. Wale N, Watson IA, Karypis G. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowl Inf Syst.* (2008) 14:347–75. doi: 10.1007/s10115-007-0103-5
51. Shervashidze N, Schweitzer P, Van Leeuwen EJ, Mehlhorn K, Borgwardt KM. Weisfeiler-lehman graph kernels. *J Mach Learn Res.* (2011) 12:2539–61.

52. Borgwardt KM, Ong CS, Schönauer S, Vishwanathan SVN, Smola AJ, Kriegel HP. Protein function prediction via graph kernels. *Bioinformatics.* (2005) 21:i47–i56. doi: 10.1093/bioinformatics/bti1007
53. Dobson PD, Doig AJ. Distinguishing enzyme structures from non-enzymes without alignments. *J Mol Biol.* (2003) 330:771–83. doi: 10.1016/S0022-2836(03)00628-4
54. Yanardag P, Vishwanathan SVN. Deep graph kernels. in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* Sydney, NSW (2015). p. 1365–74. doi: 10.1145/2783258.2783417
55. Chevyrev I, Kormilitzin A. A primer on the signature method in machine learning. *arXiv [Preprint].* (2016) arXiv:1603.03788.
56. Kingma DP, Ba J. Adam: a method for stochastic optimization. *arXiv [Preprint].* (2014) arXiv:1412.6980.
57. Niepert M, Ahmed M, Kutzkov K. Learning convolutional neural networks for graphs. In Balcan MF, Weinberger QK, editors. *Proceedings of The 33rd International Conference on Machine Learning. Vol.48.* New York, NY: PMLR (2016). p. 2014–23. Available online at: http://proceedings.mlr.press/v48/niepert16.html
58. Zhang Z, Wang M, Xiang Y, Huang Y, Nehorai A. RetGK: graph kernels based on return probabilities of random walks. In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems.* Red Hook, NY: Curran Associates Inc. (2018). p. 3968–78.
59. Verma S, Zhang ZL. Hunt for the unique, stable, sparse and fast feature learning on graphs. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems.* Red Hook, NY: Curran Associates Inc. (2017). p. 87–97.
60. Morris C, Kriege NM, Bause F, Kersting K, Mutzel P, Neumann M. TUDataset: a collection of benchmark datasets for learning with graphs. In: *ICML 2020 Workshop on Graph Representation Learning and Beyond (GRL+ 2020).* (2020). Available online at: www.graphlearning.io

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Artificial Intelligence

REVIEW
published: 14 May 2021
doi: 10.3389/frai.2021.668302

Topology Applied to Machine Learning: From Global to Local

*Henry Adams and Michael Moy**

Department of Mathematics, Colorado State University, Fort Collins, CO, United States

Through the use of examples, we explain one way in which applied topology has evolved since the birth of persistent homology in the early 2000s. The first applications of topology to data emphasized the *global* shape of a dataset, such as the three-circle model for 3×3 pixel patches from natural images, or the configuration space of the cyclo-octane molecule, which is a sphere with a Klein bottle attached via two circles of singularity. In these studies of global shape, short persistent homology bars are disregarded as sampling noise. More recently, however, persistent homology has been used to address questions about the *local* geometry of data. For instance, how can local geometry be vectorized for use in machine learning problems? Persistent homology and its vectorization methods, including persistence landscapes and persistence images, provide popular techniques for incorporating both local geometry and global topology into machine learning. Our meta-hypothesis is that the short bars are as important as the long bars for many machine learning tasks. In defense of this claim, we survey applications of persistent homology to shape recognition, agent-based modeling, materials science, archaeology, and biology. Additionally, we survey work connecting persistent homology to geometric features of spaces, including curvature and fractal dimension, and various methods that have been used to incorporate persistent homology into machine learning.

Keywords: persistent homology, topological data analysis, machine learning, local geometry, applied topology

OPEN ACCESS

Edited by:
Umberto Lupo,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Vasileios Maroulas,
The University of Tennessee, Knoxville, United States
Ashleigh Thomas,
Georgia Institute of Technology, United States

***Correspondence:**
Michael Moy
michael.moy@colostate.edu

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *16 February 2021*
Accepted: *15 April 2021*
Published: *14 May 2021*

Citation:
Adams H and Moy M (2021) Topology Applied to Machine Learning: From Global to Local. Front. Artif. Intell. 4:668302. doi: 10.3389/frai.2021.668302

1. INTRODUCTION

Applied topology is designed to measure the *shape* of data—but what is shape? Early examples in applied topology found low-dimensional structures in high-dimensional datasets, such as the three circle and Klein bottle models for grayscale natural image patches. These models are global: they parameterize the entire dataset, in the sense that most of the data points look like some point in the model, plus noise. In more recent applications, however, the shape that is being measured is not global, but instead local. Local features include texture, small-scale geometry, and the structure of noise.

Indeed, for the first decade after the invention of persistent homology, the primary story was that significant features in a dataset corresponded to long bars in the persistence barcode, whereas shorter bars generally corresponded to sampling noise. This story has evolved as applied topology has become incorporated into the machine learning pipeline. In machine learning applications, many researchers have independently found (as we survey in sections 4–6) that the short bars are often the most discriminating—the shape of the noise, or of the local geometry, is what often enables high classification accuracy. We want to emphasize that short bars do matter. Indeed,

the short bars in persistent homology are currently one of the best out-of-the-box methods for summarizing local geometry for use in machine learning. Though humans may not be able to interpret short persistent homology bars on our own (there may be too many short bars for the human eye to count), machine learning algorithms can be trained to do so. In this way, persistent homology has greatly expanded in scope during the second decade after its invention: persistent homology has important applications as a descriptor not only of global shape, but also of local geometry.

In this perspective article, we begin by outlining some of the most famous early applications of persistent homology in the global analysis of data, in which short bars were disregarded as noise. Our meta-hypothesis, however, is that short bars do matter, and furthermore, they matter crucially when combining topology with machine learning. As a partial defense for this claim, we provide a selected survey on the use of persistent homology in measuring texture, noise, local geometry, fractal dimension, and local curvature. We predict that the applications of persistent homology to machine learning will continue to advance in number, impact, and scope, as persistent homology is a mathematically motivated out-of-the-box tool that one can use to summarize not only the global topology but also the local geometry of a wide variety of datasets.

2. POINT CLOUD AND SUBLEVEL SET PERSISTENT HOMOLOGY

What is a persistent homology bar? The homology of a space, roughly speaking, records how many holes that a space has in each dimension. A 0-dimensional hole is a connected component, a 1-dimensional hole is a loop, a 2-dimensional hole is a void enclosed by a surface like a sphere or a torus, etc. Homology becomes persistent when one is instead given a *filtration*, i.e., an increasing sequence of spaces. Each hole is now represented by a bar, where the start (resp. end) point of the bar corresponds to the first (resp. last) stage in the filtration where the topological feature is present (Edelsbrunner et al., 2000). Short bars correspond to features with short lifetimes, which are quickly filled-in after being created. By contrast, long bars correspond to more *persistent* features.

Perhaps the two most frequent contexts in which persistent homology is applied are point cloud persistent homology and sublevel set persistent homology. In *point cloud persistent homology*, the input is a finite set of points (a point cloud) residing in Euclidean space or some other metric space (Carlsson, 2009). For any real number $r > 0$, we consider the union of all balls of radius r centered at some point in our point cloud (see **Figure 1**). This union of balls provides our filtration as the radius r increases[1]. A typical interpretation of the resulting persistent homology, from the global perspective, is that the long persistent homology bars recover the homology of the "true" underlying space from which the point cloud was sampled (Chazal and Oudot, 2008). A more modern but increasingly utilized perspective is that the short persistent homology bars recover the local geometry—i.e., the texture, curvature, or fractal dimension of the point cloud data.

In *sublevel set persistent homology*, the input is instead a real-valued function $f\colon Y \to \mathbb{R}$ defined on a space Y (Cohen-Steiner et al., 2007). For example, Y may be a Euclidean space of some dimension. The filtration arises by considering the sublevel sets $\{y \in Y \mid f(y) \leq r\}$. As the threshold r increases, the sublevel sets grow. One can think of f as encoding an *energy*, in which case sublevel set persistent homology encodes the shape of low-energy configurations (Mirth et al., 2021). The length of a bar then measures how large of an energy barrier must be exceeded in order for a topological feature to be filled-in: a short bar corresponds to a feature that is quickly filled-in by exceeding a low energy barrier, whereas a long bar corresponds to a topological feature that persists over a longer range of energies (see **Figure 2**). Sublevel set persistent homology is frequently applied to grayscale image data or matrix data, where a real-valued entry of the image or matrix is interpreted as the value of the function f on a pixel.

We remark that the "union of balls" filtration for point cloud persistent homology can be viewed as a version of sublevel set persistent homology: a union of balls of radius r is the sublevel set at threshold r of the distance function to the set of points in the point cloud.

Persistent homology can be represented in two equivalent ways: either as a persistence barcode or as a persistence diagram (see **Figure 3**). Each interval in the persistence barcode is represented in the persistence diagram by a point in the plane, with its birth coordinate on the horizontal axis and with its death coordinate on the vertical axis[2]. As the death of each feature is after its birth, persistence diagram points all lie above the diagonal line $y = x$. Short bars in the barcode correspond to persistence diagram points close to the diagonal, and long bars in the barcode correspond to persistence diagram points far from the diagonal.

3. EXAMPLES MEASURING GLOBAL SHAPE

The earliest applications of topology to data measured the global shape of a dataset. In these examples, the long persistent homology bars represented the true homology underlying the data, whereas the small bars were ignored as artifacts of sampling noise.

What do we mean by "global shape"? Consider, for example, conformations of the cyclo-octane molecule C_8H_{16}, which consists of a ring of eight carbons atoms, each bonded to a pair of hydrogen atoms (see **Figure 4**, left). The locations of the carbon atoms in a conformation approximately determine the locations of the hydrogen atoms via energy minimization, and hence each molecule conformation can be mapped to a point in $\mathbb{R}^{24} = \mathbb{R}^{8\cdot 3}$, as the location of each carbon atom can be specified by three

[1] In practice, the union of balls is stored or approximated by a *simplicial complex*, for example a *Čech* or *Vietoris–Rips* complex (Chazal et al., 2014).

[2] Barcodes allow for open or closed endpoints of intervals. This information can be also be encoded in a *decorated* persistence diagram (Chazal et al., 2016).

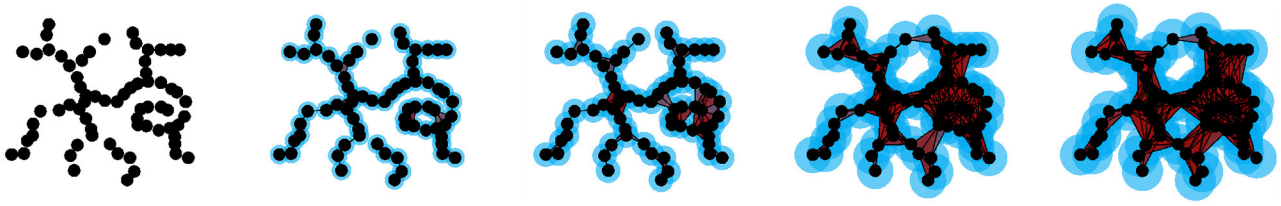


FIGURE 1 | A point cloud, the surrounding union of balls, and its Čech complexes at different choices of scale.

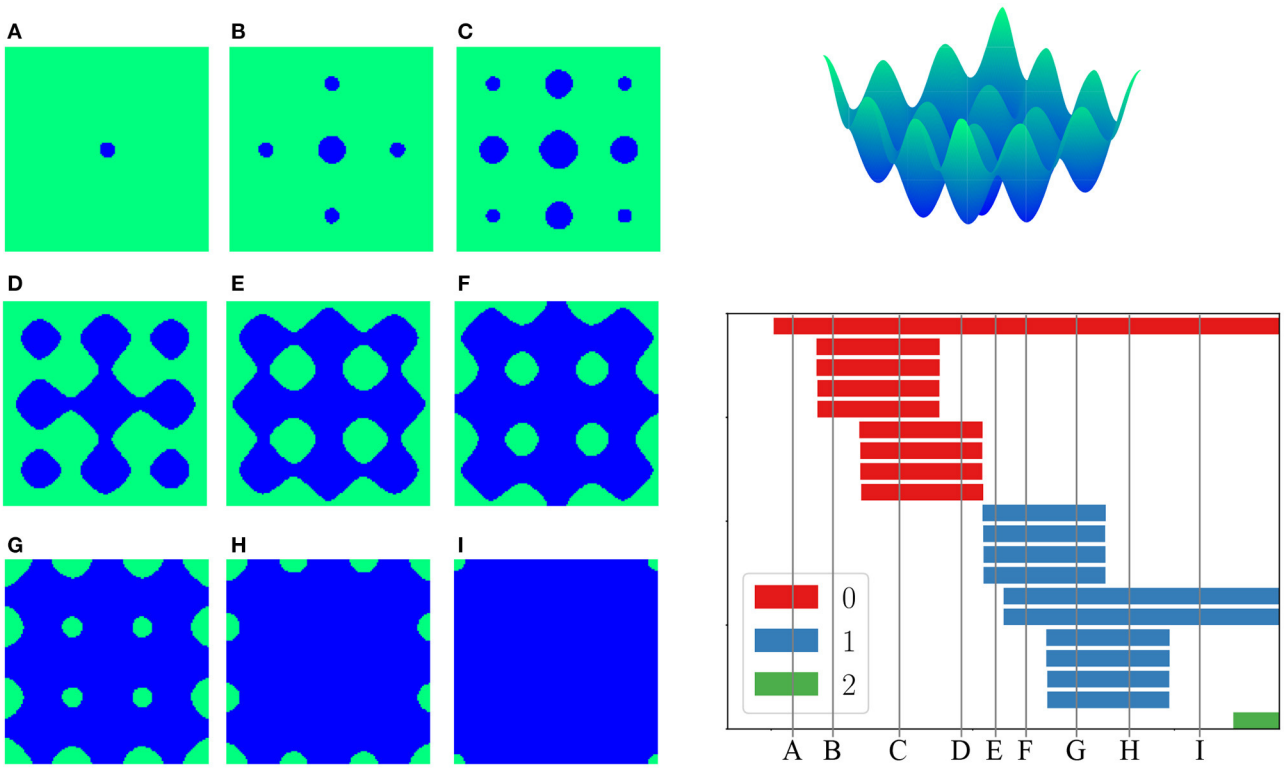


FIGURE 2 | (Top right) An energy function for the molecule pentane. The domain is a torus, i.e., a square with periodic boundary conditions, as there are two circular degrees of freedom (dihedral angles) in the molecule. **(Left)** Nine different sublevel sets of energy. **(Bottom right)** The sublevel set persistent homology of this energy function on the torus, with 0-dimensional homology in red, 1-dimensional homology in blue, 2-dimensional homology in green. Image from Mirth et al. (2021).

coordinates. This map realizes the conformation space of cyclo-octane as a subset of $\mathbb{R}^{24}$, and then we mod out by rigid rotations and translations. Topologically, the conformation space of cyclo-octane turns out to be the union of a sphere with a Klein bottle, glued together along two circles of singularities (see **Figure 4**, right). This model was obtained by Martin et al. (2010), Martin and Watson (2011), and Brown et al. (2008), who furthermore obtain a triangulation of this dataset (a representation of the dataset as a union of vertices, edges, and triangles).

A Klein bottle, like a sphere, is a 2-dimensional manifold. Whereas, a sphere can be embedded in 3-dimensional space, a Klein bottle requires at least four dimensions in order to be embedded without self-intersections. When a sphere and Klein bottle are glued together along two circles, the union is no longer a manifold. Indeed, near the gluing circles, the space does not look like a sheet of paper, but instead like the tail of a dart with four fins, i.e., the letter "X" crossed with the interval $[0, 1]$. However, the result is still a 2-dimensional stratified space. In **Figure 5**, we compute the persistent homology of a point cloud dataset of 1,000,000 cyclo-octane molecule configurations. The short bars are interpreted as noise, whereas the long bars are interpreted as attributes of the underlying shape. We obtain a single connected component, a single 1-dimensional hole, and two 2-dimensional homology features. These homology signatures agree with the homology of the union of a sphere with a Klein bottle, glued together along two circles of singularities.

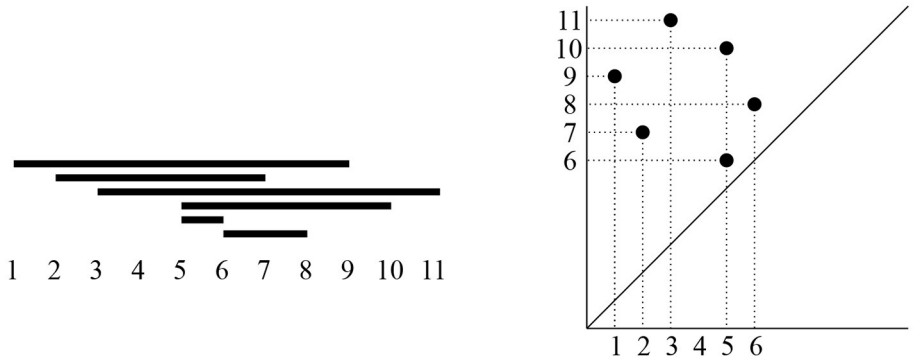


FIGURE 3 | (Left) A persistent homology barcode, with the birth and death scale of each bar indicated on the horizontal axis. **(Right)** Its corresponding persistence diagram, i.e., a collection of points in the first quadrant above the diagonal, with birth coordinates on the horizontal axis and death coordinates on the vertical axis.

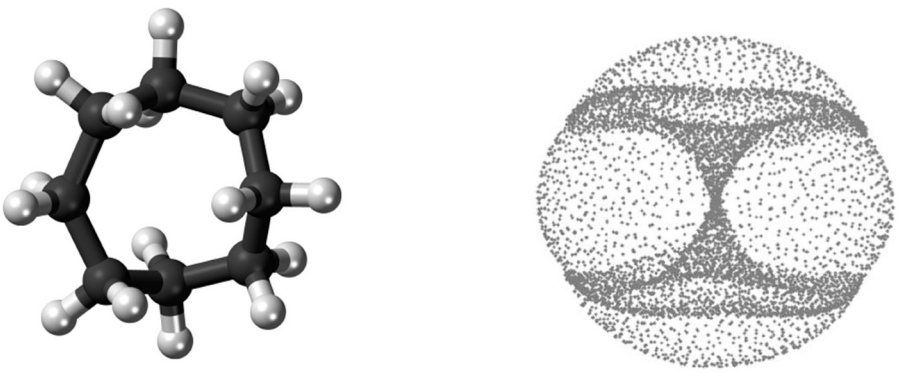


FIGURE 4 | (Left) The cyclo-octane molecule consists of a ring of 8 carbon atoms (black), each bonded to a pair of hydrogen atoms (white). **(Right)** A PCA projection of a dataset of different conformations of the cyclo-octane molecule; this shape is a sphere glued to a Klein bottle (the "hourglass") along two circles of singularity. The right image is from Martin et al. (2010).

One of the first applications of persistent homology was to measure the global shape of a dataset of image patches (Carlsson et al., 2008). This dataset of natural 3×3 pixel patches from black-and-white photographs from indoor and outdoor scenes in fact has three different global shapes! The most common patches lie along a circle of possible directions of linear gradient patches (varying from black to gray to white). The next most common patches lie along a three circle model, additionally including a circle's worth of horizontal quadratic gradients, and a circle's worth of vertical quadratic gradients. At the next level of resolution, the most common patches in some sense lie along a Klein bottle. All three of these models—the circle, the three circles, and the Klein bottle—are global models, summarizing the global shape of the dataset at different resolutions.

4. EXAMPLES MEASURING LOCAL GEOMETRY

Though a single long bar in persistent homology may carry a lot of information, a single small bar typically does not. However, together a collection of small bars may unexpectedly carry a large amount of geometric content. A long bar is a trumpet solo—piercing through to be heard over the orchestra with ease. The small bars are the string section—each small bar on its own is relatively quiet, but in concert the small bars together deliver a powerful message. We survey several modern examples where small persistent homology bars are now the signal, instead of the noise.

Birds, fish, and insects move as flocks, schools, and hordes in a way which is determined by *collective motion*: each animal's next motion is a random function of the location of its nearby neighbors. In a flock of thousands of birds, there is an impressively large amount of time-varying geometry, including for example all $\binom{n}{2}$ pairwise distances, where n is the number of birds (see **Figure 6**). How can one summarize this much geometric content for use in machine learning tasks, say to predict how the motion of the flock will vary next, or to predict some of the parameters in a mathematical model approximately governing the motion of the birds? Persistent homology has been used in Topaz et al. (2015), Ulmer et al. (2019), Bhaskar et al. (2019), Adams et al. (2020b), and Xian et al. (2020) to reduce a large collection of geometric content down to a concise summary. These datasets of animal swarms do not lie along

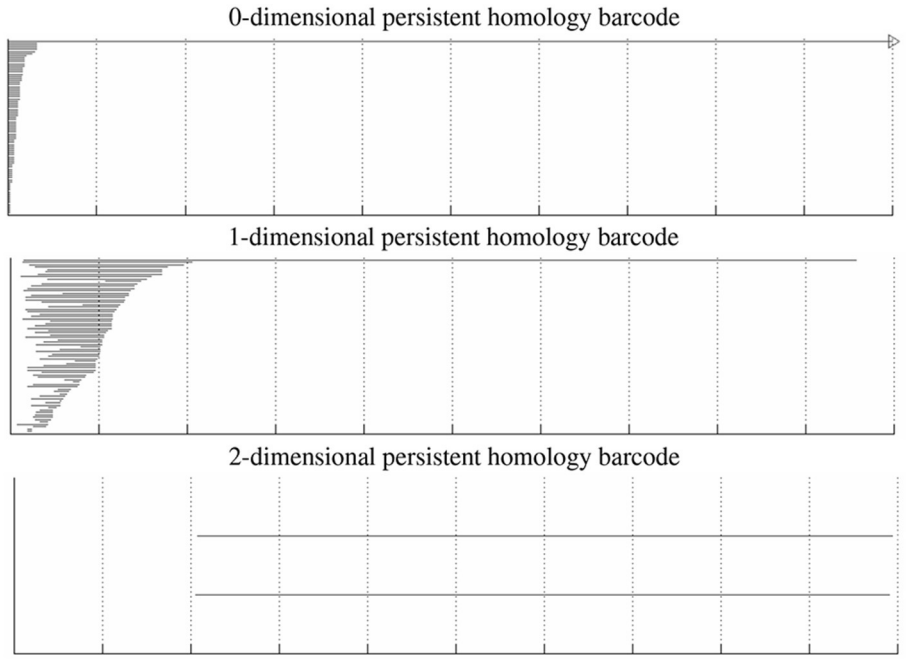


FIGURE 5 | 0-, 1-, and 2-dimensional persistent homology barcodes for the cyclo-octane dataset. The horizontal axis corresponds to the birth and death scale of the bars, and the vertical axis is an arbitrary ordering of the bars (here by death scale).

FIGURE 6 | A large amount of local and global geometric information is contained in a flock of birds.

beautiful manifolds (global shapes), but nevertheless there is a wealth of information in the local geometry as measured by the short persistent homology bars. For example, Ulmer et al. (2019) show via time-varying persistent homology[3] that a control model for aphid motion, in which aphids move independently at random, does not fit experimental data as well as a model incorporating social interaction (distances to nearby neighbors) between the aphids.

Other recent work has used persistent homology to characterize the complexity of geometric objects. Bendich et al. (2016b) apply sublevel set persistent homology to the study of brain artery trees, examining the effects of age and sex on the barcodes generated from artery trees. While younger brains have artery trees containing more local twisting and branching, older brains are sparser with fewer small branches and leaves. The authors use the 100 longest bars in dimensions 0 and 1 in their analysis, and they further examine which lengths of bars give the highest correlation with age and sex. For instance, when examining age, they find it is not the longest bars, but instead the bars of medium length (roughly the 21st through 40th longest bars) that are the most discriminatory.

In other datasets where points are nearly evenly spaced, barcodes will consist of bars with mostly similar birth and death times. Consider for instance the point cloud persistent homology for a square grid of points in the plane: all non-infinite 0-dimensional bars are identical and adding a small amount of noise to the points will result in a small change to the bars. The same is true for 1-dimensional bars. With this in mind, Motta et al. (2018) use persistent homology to measure the order, or regularity, of lattice-like datasets, focusing on hexagonal grids formed by ion bombardment of solid surfaces (see **Figure 7**). The authors' techniques use the variance of 0-dimensional homology bar lengths, and the sum of the lengths of 1-dimensional homology bars, as well as a particular linear combination of the two especially suited to hexagonal lattices. Their results suggest that techniques based on persistent homology can provide useful measures of order that are sensitive to both large scale and small scale defects in lattices. Point cloud persistence has also been used to summarize the local order and randomness in other materials science and chemistry

[3]In particular, the *crocker plot* (Topaz et al., 2015).

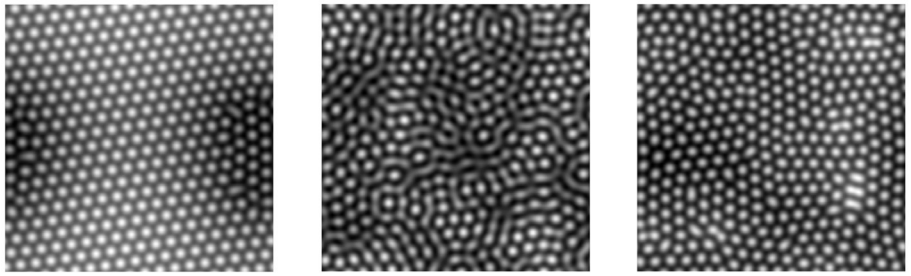


FIGURE 7 | Hexagonal lattices, of varying degree of regularity, created by ion bombardment. Figures from Motta et al. (2018).

contexts, including amorphous solids and glass (Nakamura et al., 2015; Hiraoka et al., 2016; Hirata et al., 2020), nanoporous materials used in gas adsorption (Krishnapriyan et al., 2020), crystal structure (Maroulas et al., 2020), and protein folding (Xia and Wei, 2014; Cang and Wei, 2018).

Though the above examples focus on point cloud persistence, sublevel set persistent homology has also been used to detect the local geometry of functions. Kramár et al. (2016) use sublevel set persistence to summarize the complicated spatio-temporal patterns that arise from dynamical systems modeling fluid flow, including turbulence (Kolmogorov flow) and heat convection (Rayleigh-Bénard convection). With sublevel set persistence, Zeppelzauer et al. (2016) improve 3D surface classification, including on an archaeology task of segmenting engraved regions of rock from the surrounding natural rock surface. In a task of tracking automobiles, Bendich et al. (2016a) use the sublevel set persistent homology of driver speeds in order to characterize driver behaviors and prune out improbable paths from their multiple hypothesis tracking framework.

5. THEORY OF HOW PERSISTENT HOMOLOGY MEASURES LOCAL GEOMETRY

Recent work has begun to formalize the idea that persistent homology measures local geometry. Bubenik et al. (2020) explore the effect of the curvature of a space on the persistent homology of a sample of points, focusing on disks in spaces with constant curvature. Their work includes theoretical results about the persistence of triangles in these spaces, and they are also able to demonstrate experimentally that persistent homology in dimensions 0 and 1 can be used to accurately estimate the curvature given a random sample of points. Since the disks in spaces with different curvature are homeomorphic, the differences in persistent homology cannot be due to topology, but rather result from the geometric features of the spaces.

Fractal dimension is another measure of local geometry, and indeed some of the earliest applications of persistent homology in Vanessa Robins' Ph.D. thesis were motivated as a way to capture the fractal dimension of an infinite set in Euclidean space (Robins, 2000; MacPherson and Schweinhart, 2012). Can this also be applied to datasets, i.e., to random collections of finite sets of points? Given a random sample of points from a measure, Adams et al. (2020a) use persistent homology to detect the fractal dimension of the support of the measure. This notion of *persistent homology fractal dimension* agrees with the Hausdorff/box-counting dimension for 0-dimensional persistent homology and a restricted class of measures; see Schweinhart (2019, 2020) for further theoretical developments.

A related line of work studies what can be proven about the topology of random point clouds, typically as the number of points in the point cloud goes to infinity (Kahle, 2011; Adler et al., 2014; Bobrowski and Kahle, 2014; Bobrowski et al., 2017). The magnitude (Leinster, 2013) and magnitude homology (Hepworth and Willerton, 2017; Leinster and Shulman, 2017) of a metric space measure both local and global properties; recent and ongoing work is being done to connect magnitude with persistent homology (Otter, 2018; Govc and Hepworth, 2021). See also Weinberger (2019) for connections between sublevel set persistent homology and the geometry of spaces of functions, including Lipschitz constants of functions. We predict that much more work demonstrating how local geometric features can be recovered from persistent homology barcodes will take place over the next decade.

6. MACHINE LEARNING

Because persistent homology gives a concise description of the shape of data, it is not surprising that recent work has incorporated persistent homology into machine learning. When might one consider using persistent homology in concert with machine learning, as opposed to other more classical machine learning techniques measuring shape such as clustering (Xu and Wunsch, 2005) or nonlinear dimensionality reduction (Roweis and Saul, 2000; Tenenbaum et al., 2000; Kohonen, 2012; McInnes et al., 2018)? We recommend persistent homology when one desires either (i) a quantitative reductive summary of local geometry, (ii) an estimate of the number or size of more global topological features in a dataset, or (iii) a way to explore if either local geometry or global topology may be discriminatory for the machine learning task at hand. Researchers have taken at least three distinct approaches: persistence barcodes have been adapted to be input to machine learning algorithms, topological methods have been used to create new algorithms, and persistent homology has been used to analyze machine learning algorithms.

Perhaps the most natural of these approaches is inputting persistence data into a machine learning algorithm. Though the persistent homology bars provide a summary of both local geometry and global topology, for a quantitative summary to be fully applicable it needs to be amenable for use in machine learning tasks. The space of persistence barcodes is not immediately appropriate for machine learning. Indeed, averages of barcodes need not be unique (Mileyko et al., 2011), and the space of persistence barcodes does not coarsely embed into any Hilbert space (Bubenik and Wagner, 2020). These limitations have initiated a large amount of research on transforming persistence barcodes into more natural formats for machine learning. From barcodes, Bubenik (2015) creates *persistence landscapes*, which live in a Banach space of functions[4]. Persistence landscapes are created by rotating a persistence diagram on its side—so that the diagonal line $y = x$ becomes as flat as the horizon—and then using the persistence diagram points to trace out the peaks in a mountain landscape profile. A landscape can then be discretized by taking a finite sample of the function values, allowing it to be used in machine learning tasks (see for instance Kovacev-Nikolic et al. (2016)). From barcodes, Adams et al. (2017) create *persistence images*, a Euclidean vectorization enabling a diverse class of machine learning tools to be applied (see also Chen et al., 2015; Reininghaus et al., 2015). A persistence image is created by taking a sum of Gaussians, one centered on each point in a persistence diagram, and then pixelating that surface to form an image. By analogy, recall that in point cloud persistent homology, one "blurs their vision" when looking at a dataset by replacing each data point with a ball—this is similar to the process of "blurring one's vision" when looking at a persistence diagram in order to create a persistence image.

Persistence landscapes were defined as part of an effort to give a firm statistical foundation to persistent homology. In fact, Bubenik (2015) proves a strong law of large numbers and a central limit theorem for persistence landscapes. This allows one to discuss hypothesis testing with persistent homology. Another approach to hypothesis testing is given by Robinson and Turner (2017). Other statistical approaches include Fasy et al. (2014), which describes confidence intervals and a statistical approach to distinguishing important features from noise, Divol and Polonik (2019) and Maroulas et al. (2019), which consider probability density functions for persistence diagrams, and Maroulas et al. (2020), which describes a Bayesian framework. See Wasserman (2018) for a review of statistical techniques in the context of topological data analysis.

Persistence landscapes and images are only two of the many different methods that have recently been invented in order to transform persistence barcodes into machine learning input. Algorithms that require only a distance matrix, such as many clustering or dimensionality reduction algorithms, can be applied on the bottleneck or Wasserstein distances between persistence barcodes (Cohen-Steiner et al., 2007; Mileyko et al., 2011; Kerber et al., 2017). Other techniques for vectorizing persistence barcodes involve heat kernels (Carrière et al., 2015), entropy (Merelli et al., 2015; Atienza et al., 2020), rings of algebraic functions (Adcock et al., 2016), tropical coordinates (Kališnik, 2019), complex polynomials (Di Fabio and Ferri, 2015), and optimal transport (Carrière et al., 2017), among others. Some of these techniques, including those by Zhao and Wang (2019) and Divol and Polonik (2019), allow one to learn the vectorization parameters that are best suited for a machine learning task on a given dataset. Others allow one to plug persistent homology information directly into a neural network (Hofer et al., 2017). Recent research on incorporating persistence as input for machine learning is vast and varied, and the above collection of references is far from complete.

As for the creation of new algorithms, persistent homology has recently been applied to regularization, a technique used in machine learning that penalizes overly complicated models to avoid overfitting. Chen et al. (2019) propose a "topological penalty function" for classification algorithms, which encourages a topologically simple decision boundary. Their method is based on measuring the relative importance of various connected components of the decision boundary via 0-dimensional persistent homology. They show how the gradient of such a penalty function can be computed, which is important for use in machine learning algorithms, and demonstrate their method on several examples. Similar work using topological methods to examine a decision boundary can also be found in Varshney and Ramamurthy (2015) and Ramamurthy et al. (2019).

Finally, other recent work has used persistent homology to analyze neural networks. Naitzat et al. (2020) provide experimental evidence that neural networks operate by simplifying the topology of a dataset. They examine the topology of a dataset and its images at the various layers of a neural network performing classification, finding that the corresponding barcodes become simpler as the data progresses though the network. Additionally, they observe the effects of different shapes of neural networks and different activation functions. They find that deeper neural networks have a tendency to simplify the topology of a dataset more gradually than shallow networks, and that networks with ReLU activation tend to simplify topology more in the earlier layers of a network than other activation functions.

7. CONCLUSION

Topological tools are often described as being able to stitch local data together in order to describe global features: from local to global. The history of applied topology, however, has in some sense gone in the reverse direction—from global to local—as surveyed above! Applied topology was developed in part to summarize global features in a point cloud dataset, as in the examples of the conformations of the cyclo-octane molecule or the collection of 3×3 pixel patches from images. If global shapes are the focus, long persistent homology bars are interpreted as the relevant features, while small bars are often disregarded as sampling artifacts or noise. However, in more recent applications, and in particular when using applied topology in concert with

[4]In practice, a different metric is sometimes chosen to map landscapes into a Hilbert space, though the restrictions of Bubenik and Wagner (2020) apply.

machine learning, it is often many short persistent homology bars that *together* form the signal. One of the biggest benefits of applied topology is that one need not choose a scale beforehand: persistent homology provides a useful summary of both the local and global features in a dataset, and this summary has been made accessible for use in machine learning tasks.

We have seen how the short bars can be a measure of local geometry, texture, curvature, and fractal dimension; their sensitivity to various features of datasets leads to the wide variety of applications surveyed here. Because persistent homology provides a concise, reductive view of the geometry of a dataset, for instance in the examples studying brain artery trees or hexagonal grids, it is not hard to imagine the potential applications to machine learning problems. This has led to recent techniques that turn barcodes into machine learning input, exemplified by persistence landscapes and persistence images. We hope that this wealth of recent work, which has shifted more attention to short persistent homology bars and the geometric information they summarize, will inspire further research at the intersection of applied topology, local geometry, and machine learning.

AUTHOR CONTRIBUTIONS

HA originally presented this material in conference talks. HA and MM reviewed literature and contributed to the writing of the article. All authors contributed to the article and approved the submitted version.

FUNDING

This material was based upon work supported by the National Science Foundation under Grant Number 1934725.

REFERENCES

Adams, H., Aminian, M., Farnell, E., Kirby, M., Mirth, J., Neville, R., et al. (2020a). "A fractal dimension for measures via persistent homology," in *Topological Data Analysis*, eds N. A. Baas, G. E. Carlsson, G. Quick, M. Szymik, and M. Thaule (Cham: Springer International Publishing), 1–31. doi: 10.1007/978-3-030-43408-3_1

Adams, H., Chepushtanova, S., Emerson, T., Hanson, E., Kirby, M., Motta, F., et al. (2017). Persistence images: A vector representation of persistent homology. *J. Mach. Learn. Res.* 18, 1–35.

Adams, H., Ciocanel, M.-V., Topaz, C. M., and Ziegelmeier, L. (2020b). Topological data analysis of collective motion. *SIAM News* 53, 1–4. Available online at: https://sinews.siam.org/Details-Page/topological-data-analysis-of-collective-motion

Adcock, A., Carlsson, E., and Carlsson, G. (2016). The ring of algebraic functions on persistence bar codes. *Homot. Homol. Appl.* 18, 341–402. doi: 10.4310/HHA.2016.v18.n1.a21

Adler, R. J., Bobrowski, O., and Weinberger, S. (2014). Crackle: The homology of noise. *Discr. Comput. Geometry* 52, 680–704. doi: 10.1007/s00454-014-9621-6

Atienza, N., González-Díaz, R., and Soriano-Trigueros, M. (2020). On the stability of persistent entropy and new summary functions for TDA. *Pattern Recognit.* 107:107509. doi: 10.1016/j.patcog.2020.107509

Bendich, P., Chin, S. P., Clark, J., Desena, J., Harer, J., Munch, E., et al. (2016a). Topological and statistical behavior classifiers for tracking applications. *IEEE Trans. Aerosp. Electron. Syst.* 52, 2644–2661. doi: 10.1109/TAES.2016.160405

Bendich, P., Marron, J. S., Miller, E., Pieloch, A., and Skwerer, S. (2016b). Persistent homology analysis of brain artery trees. *Ann. Appl. Stat.* 10:198. doi: 10.1214/15-AOAS886

Bhaskar, D., Manhart, A., Milzman, J., Nardini, J. T., Storey, K. M., Topaz, C. M., et al. (2019). Analyzing collective motion with machine learning and topology. *Chaos* 29:123125. doi: 10.1063/1.5125493

Bobrowski, O., and Kahle, M. (2014). Topology of random geometric complexes: a survey. *J. Appl. Comput. Topol.* 1, 331–364. doi: 10.1007/s41468-017-0010-0

Bobrowski, O., Kahle, M., and Skraba, P. (2017). Maximally persistent cycles in random geometric complexes. *Ann. Appl. Probab.* 27, 2032–2060. doi: 10.1214/16-AAP1232

Brown, M. W., Martin, S., Pollock, S. N., Coutsias, E. A., and Watson, J. P. (2008). Algorithmic dimensionality reduction for molecular structure analysis. *J. Chem. Phys.* 129:064118. doi: 10.1063/1.2968610

Bubenik, P. (2015). Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res.* 16, 77–102.

Bubenik, P., Hull, M., Patel, D., and Whittle, B. (2020). Persistent homology detects curvature. *Inverse Probl.* 36:025008. doi: 10.1088/1361-6420/ab4ac0

Bubenik, P., and Wagner, A. (2020). Embeddings of persistence diagrams into hilbert spaces. *J. Appl. Comput. Topol.* 4, 339–351. doi: 10.1007/s41468-020-00056-w

Cang, Z., and Wei, G.-W. (2018). Integration of element specific persistent homology and machine learning for protein-ligand binding affinity prediction. *Int. J. Numer. Methods Biomed. Eng.* 34, e2914. doi: 10.1002/cnm.2914

Carlsson, G. (2009). Topology and data. *Bull. Am. Math. Soc.* 46, 255–308. doi: 10.1090/S0273-0979-09-01249-X

Carlsson, G., Ishkhanov, T., de Silva, V., and Zomorodian, A. (2008). On the local behavior of spaces of natural images. *Int. J. Comput. Vis.* 76, 1–12. doi: 10.1007/s11263-007-0056-x

Carriére, M., Cuturi, M., and Oudot, S. (2017). "Sliced wasserstein kernel for persistence diagrams," in *International Conference on Machine Learning, PMLR* (Sydney, VIC), 664–673.

Carriére, M., Oudot, S. Y., and Ovsjanikov, M. (2015). "Stable topological signatures for points on 3d shapes," in *Computer Graphics Forum, Vol. 34* (Wiley Online Library), 1–12. doi: 10.1111/cgf.12692

Chazal, F., de Silva, V., Glisse, M., and Oudot, S. (2016). *The Structure and Stability of Persistence Modules*. Springer. doi: 10.1007/978-3-319-42545-0

Chazal, F., de Silva, V., and Oudot, S. (2014). Persistence stability for geometric complexes. *Geometr. Dedic.* 173, 193–214. doi: 10.1007/s10711-013-9937-z

Chazal, F., and Oudot, S. (2008). "Towards persistence-based reconstruction in Euclidean spaces," in *Proceedings of the 24th Annual Symposium on Computational Geometry* (College Park, MD: ACM), 232–241. doi: 10.1145/1377676.1377719

Chen, C., Ni, X., Bai, Q., and Wang, Y. (2019). "A topological regularizer for classifiers via persistent homology," in *Proceedings of Machine Learning Research, Vol. 89*, eds K. Chaudhuri and M. Sugiyama (Naha), 2573–2582.

Chen, Y.-C., Wang, D., Rinaldo, A., and Wasserman, L. (2015). Statistical analysis of persistence intensity functions. *arXiv preprint arXiv:1510.02502*.

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2007). Stability of persistence diagrams. *Discr. Comput. Geomet.* 37, 103–120. doi: 10.1007/s00454-006-1276-5

Di Fabio, B., and Ferri, M. (2015). "Comparing persistence diagrams through complex vectors," in *International Conference on Image Analysis and Processing* (Genoa: Springer), 294–305. doi: 10.1007/978-3-319-23231-7_27

Divol, V., and Polonik, W. (2019). On the choice of weight functions for linear representations of persistence diagrams. *J. Appl. Comput. Topol.* 3, 249–283. doi: 10.1007/s41468-019-00032-z

Edelsbrunner, H., Letscher, D., and Zomorodian, A. (2000). "Topological persistence and simplification," in *41st Annual Symposium on Foundations of Computer Science, 2000* (Redondo Beach, CA: IEEE), 454–463. doi: 10.1109/SFCS.2000.892133

Fasy, B. T., Lecci, F., Rinaldo, A., Wasserman, L., Sivaraman, B., and Singh, A. (2014). Confidence sets for persistence diagrams. *Ann. Stat.* 42, 2301–2339. doi: 10.1214/14-AOS1252

Govc, D., and Hepworth, R. (2021). Persistent magnitude. *J. Pure Appl. Algeb.* 225:106517. doi: 10.1016/j.jpaa.2020.106517

Hepworth, R., and Willerton, S. (2017). Categorifying the magnitude of a graph. *Homol. Homotopy Appl.* 19, 31–60. doi: 10.4310/HHA.2017.v19.n2.a3

Hiraoka, Y., Nakamura, T., Hirata, A., Escolar, E. G., Matsue, K., and Nishiura, Y. (2016). Hierarchical structures of amorphous solids characterized by persistent homology. *Proc. Natl. Acad. Sci. U.S.A.* 113, 7035–7040. doi: 10.1073/pnas.1520877113

Hirata, A., Wada, T., Obayashi, I., and Hiraoka, Y. (2020). Structural changes during glass formation extracted by computational homology with machine learning. *Commun. Mater.* 1, 1–8. doi: 10.1038/s43246-020-00100-3

Hofer, C., Kwitt, R., Niethammer, M., and Uhl, A. (2017). "Deep learning with topological signatures, in *Advances in Neural Information Processing Systems*, eds I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Long Beach, CA: Curran Associates, Inc), 30, 1634–1644.

Kahle, M. (2011). Random geometric complexes. *Discr. Comput. Geometry* 45, 553–573. doi: 10.1007/s00454-010-9319-3

Kališnik, S. (2019). Tropical coordinates on the space of persistence barcodes. *Found. Comput. Math.* 19, 101–129. doi: 10.1007/s10208-018-9379-y

Kerber, M., Morozov, D., and Nigmetov, A. (2017). Geometry helps to compare persistence diagrams. *ACM. J. Exp. Algorithmics.* 22, 1–20. doi: 10.1145/3064175

Kohonen, T. (2012). *Self-Organizing Maps, Vol. 30*. Berlin: Springer Science & Business Media.

Kovacev-Nikolic, V., Bubenik, P., Nikolić, D., and Heo, G. (2016). Using persistent homology and dynamical distances to analyze protein binding. *Stat. Appl. Genet. Mol. Biol.* 15, 19–38. doi: 10.1515/sagmb-2015-0057

Kramár, M., Levanger, R., Tithof, J., Suri, B., Xu, M., Paul, M., et al. (2016). Analysis of Kolmogorov flow and Rayleigh-Bénard convection using persistent homology. *Phys. D* 334, 82–98. doi: 10.1016/j.physd.2016.02.003

Krishnapriyan, A. S., Montoya, J., Hummelshøj, J., and Morozov, D. (2020). Persistent homology advances interpretable machine learning for nanoporous materials. *arXiv preprint arXiv:2010.00532*.

Leinster, T. (2013). The magnitude of metric spaces. *Doc. Math.* 18, 857–905.

Leinster, T., and Shulman, M. (2017). Magnitude homology of enriched categories and metric spaces. *arXiv preprint arXiv:1711.00802*.

MacPherson, R., and Schweinhart, B. (2012). Measuring shape with topology. *J. Math. Phys.* 53:073516. doi: 10.1063/1.4737391

Maroulas, V., Mike, J. L., and Oballe, C. (2019). Nonparametric estimation of probability density functions of random persistence diagrams. *J. Mach. Learn. Res.* 20, 1–49.

Maroulas, V., Nasrin, F., and Oballe, C. (2020). A Bayesian framework for persistent homology. *SIAM J. Math. Data Sci.* 2, 48–74. doi: 10.1137/19M1268719

Martin, S., Thompson, A., Coutsias, E. A., and Watson, J.-P. (2010). Topology of cyclo-octane energy landscape. *J. Chem. Phys.* 132:234115. doi: 10.1063/1.3445267

Martin, S., and Watson, J. P. (2011). Non-manifold surface reconstruction from high-dimensional point cloud data. *Comput. Geometry* 44, 427–441. doi: 10.1016/j.comgeo.2011.05.002

McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*. doi: 10.21105/joss.00861

Merelli, E., Rucco, M., Sloot, P., and Tesei, L. (2015). Topological characterization of complex systems: Using persistent entropy. *Entropy* 17, 6872–6892. doi: 10.3390/e17106872

Mileyko, Y., Mukherjee, S., and Harer, J. (2011). Probability measures on the space of persistence diagrams. *Inverse Probl.* 27:124007. doi: 10.1088/0266-5611/27/12/124007

Mirth, J., Zhai, Y., Bush, J., Alvarado, E. G., Jordan, H., Heim, M., et al. (2021). Representations of energy landscapes by sublevelset persistent homology: an example with n-alkanes. *J. Chem. Phys.* 154:114114.

Motta, F. C., Neville, R., Shipman, P. D., Pearson, D. A., and Bradley, R. M. (2018). Measures of order for nearly hexagonal lattices. *Phys. D* 380, 17–30. doi: 10.1016/j.physd.2018.05.005

Naitzat, G., Zhitnikov, A., and Lim, L.-H. (2020). Topology of deep neural networks. *J. Mach. Learn. Res.* 21, 1–40.

Nakamura, T., Hiraoka, Y., Hirata, A., Escolar, E. G., and Nishiura, Y. (2015). Persistent homology and many-body atomic structure for medium-range order in the glass. *Nanotechnology* 26:304001. doi: 10.1088/0957-4484/26/30/304001

Otter, N. (2018). Magnitude meets persistence. Homology theories for filtered simplicial sets. *arXiv preprint arXiv:1807.01540*.

Ramamurthy, K. N., Varshney, K., and Mody, K. (2019). "Topological data analysis of decision boundaries with application to model selection," in *Proceedings of the 36th International Conference on Machine Learning, Vol. 97 of Proceedings of Machine Learning Research*, eds K. Chaudhuri and R. Salakhutdinov (Long Beach, CA), 5351–5360.

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. (2015). "A stable multi-scale kernel for topological machine learning," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Boston, MA), 4741–4748. doi: 10.1109/CVPR.2015.7299106

Robins, V. (2000). *Computational topology at multiple resolutions: Foundations and applications to fractals and dynamics* (Ph.D. thesis). Boulder, CO: University of Colorado.

Robinson, A., and Turner, K. (2017). Hypothesis testing for topological data analysis. *J. Appl. Comput. Topol.* 1, 241–261. doi: 10.1007/s41468-017-0008-7

Roweis, S. T., and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science* 290, 2323–2326. doi: 10.1126/science.290.5500.2323

Schweinhart, B. (2019). Persistent homology and the upper box dimension. *Discr. Comput. Geometry* 65, 331–364. doi: 10.1007/s00454-019-00145-3

Schweinhart, B. (2020). Fractal dimension and the persistent homology of random geometric complexes. *Adv. Math.* 372:107291. doi: 10.1016/j.aim.2020.107291

Tenenbaum, J. B., De Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 2319–2323. doi: 10.1126/science.290.5500.2319

Topaz, C. M., Ziegelmeier, L., and Halverson, T. (2015). Topological data analysis of biological aggregation models. *PLoS ONE* 10:e0126383. doi: 10.1371/journal.pone.0126383

Ulmer, M., Ziegelmeier, L., and Topaz, C. M. (2019). A topological approach to selecting models of biological experiments. *PLoS ONE* 14:e0213679. doi: 10.1371/journal.pone.0213679

Varshney, K. R., and Ramamurthy, K. N. (2015). "Persistent topology of decision boundaries," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Processing* (Brisbane, QLD), 3931–3935. doi: 10.1109/ICASSP.2015.7178708

Wasserman, L. (2018). Topological data analysis. *Annu. Rev. Stat. Appl.* 5, 501–532. doi: 10.1146/annurev-statistics-031017-100045

Weinberger, S. (2019). Interpolation, the rudimentary geometry of spaces of Lipschitz functions, and geometric complexity. *Found. Comput. Math.* 19, 991–1011. doi: 10.1007/s10208-019-09416-0

Xia, K., and Wei, G.-W. (2014). Persistent homology analysis of protein structure, flexibility, and folding. International *J. Numer. Methods Biomed. Eng.* 30, 814–844. doi: 10.1002/cnm.2655

Xian, L., Adams, H., Topaz, C. M., and Ziegelmeier, L. (2020). Capturing dynamics of time-varying data via topology. *arXiv preprint arXiv:2010.05780*.

Xu, R., and Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Trans. Neural Netw.* 16, 645–678. doi: 10.1109/TNN.2005.845141

Zeppelzauer, M., Zieliński, B., Juda, M., and Seidl, M. (2016). "Topological descriptors for 3d surface analysis," in *International Workshop on Computational Topology in Image Context* (Springer), 77–87. doi: 10.1007/978-3-319-39441-1_8

Zhao, Q., and Wang, Y. (2019). "Learning metrics for persistence-based summaries and applications for graph classification," in *Advances in Neural Information Processing Systems*, 9859–9870.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Artificial Intelligence

REVIEW
published: 26 May 2021
doi: 10.3389/frai.2021.681108

A Survey of Topological Machine Learning Methods

Felix Hensel[1,2], *Michael Moor*[1,2] *and Bastian Rieck*[1,2]*

[1] Machine Learning and Computational Biology Laboratory, ETH Zurich, Zurich, Switzerland, [2] Swiss Institute of Bioinformatics, Lausanne, Switzerland

The last decade saw an enormous boost in the field of computational topology: methods and concepts from algebraic and differential topology, formerly confined to the realm of pure mathematics, have demonstrated their utility in numerous areas such as computational biology personalised medicine, and time-dependent data analysis, to name a few. The newly-emerging domain comprising topology-based techniques is often referred to as topological data analysis (TDA). Next to their applications in the aforementioned areas, TDA methods have also proven to be effective in supporting, enhancing, and augmenting both classical machine learning and deep learning models. In this paper, we review the state of the art of a nascent field we refer to as "topological machine learning," i.e., the successful symbiosis of topology-based methods and machine learning algorithms, such as deep neural networks. We identify common threads, current applications, and future challenges.

Keywords: computational topology, persistent homology, machine learning, topology, survey, topological machine learning

OPEN ACCESS

Edited by:
Kathryn Hess,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Raphael Reinauer,
École Polytechnique Fédérale de Lausanne, Switzerland
Matteo Caorsi,
L2F SA, Switzerland

***Correspondence:**
Bastian Rieck
bastian.rieck@bsse.ethz.ch

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *15 March 2021*
Accepted: *13 April 2021*
Published: *26 May 2021*

Citation:
Hensel F, Moor M and Rieck B (2021) A Survey of Topological Machine Learning Methods. Front. Artif. Intell. 4:681108. doi: 10.3389/frai.2021.681108

1. INTRODUCTION

Topological machine learning recently started to emerge as a field at the interface of topological data analysis (TDA) and machine learning. It is driven by improvements of computational methods, which make the calculation of topological features (via persistent homology, for instance) increasingly flexible and scalable to more complex and larger data sets.

Topology is colloquially often referred to as encoding the overall shape of data. Hence, as a complement to localised and generally more rigid geometric features, topological features are suitable to capture multi-scale, global, and intrinsic properties of data sets. This utility has been recognised with the rise of TDA, and topological information is now generally accepted to be relevant in the context of data analysis. Numerous works aim to leverage such information to gain a fundamentally different perspective on their data sets. We want to focus on a recent "outgrowth" of TDA, i.e., the integration of topological methods to *enhance* or *augment* both classical machine learning methods and deep learning models.

Our survey therefore discusses this ongoing synthesis of topology and machine learning, giving an overview of recent developments in the field. As an emerging research topic, topological machine learning is highly active and rapidly developing. Our survey is therefore explicitly not intended as a formal and complete review of the field. We rather want to identify, present, and discuss some of the main directions of developments, applications, and challenges in topological machine learning as we perceive it based on our own research background. Our aim is to provide newcomers to the field with a high-level overview of some of the central developments and techniques that have been developed, highlighting some "nuggets," and outlining common threads and future challenges. We

focus on publications in major machine learning conferences (such as AISTATS, ICLR, ICML, and NeurIPS) and journals (such as JMLR) but want to note that the selection of topics and papers presented here reflects our own preferences and knowledge. In particular, we decided against the inclusion of unpublished work in this area.

The survey is broadly structured as follows: we first provide a brief mathematical background on persistent homology, one of the core concepts of topological data analysis, in section 2. Following the introduction, the main part of the survey is in section 3. Section 3.2 focuses on what we term *extrinsic topological features* in machine learning. These methods are mainly concerned with the transformation of topological descriptors of data into feature vectors of fixed dimensionality, permitting their use as features in machine learning frameworks. This is in contrast to *intrinsic topological features*, portrayed in section 3.3, which employ topological features to analyse or influence the machine learning model itself, for instance by architectural choices or regularisation. Finally, section 4 discusses future directions and challenges in topological machine learning.

2. BACKGROUND ON ALGEBRAIC TOPOLOGY AND PERSISTENT HOMOLOGY

This section provides some background on basic concepts from algebraic topology and persistent homology. For in-depth treatments of the subject matter, we refer to standard literature (Bredon, 1993; Hatcher, 2000; Edelsbrunner and Harer, 2010). Readers familiar with algebraic topology and the concept of persistent homology may safely skip this section.

A basic hypothesis in data analysis which drives current research is that data has *shape*, or put differently, that data is sampled from an underlying manifold—the so-called "manifold hypothesis" (Fefferman et al., 2013). Instead of restricting the analysis to statistical descriptors, *topological data analysis* (TDA) aims to analyse data from a fundamentally different perspective by investigating this underlying manifold structure in an algebraic fashion. Namely, one computes descriptors of data sets which are *stable* under perturbation and encode intrinsic *multi-scale* information on the their shape. TDA is a rapidly developing field of mathematics aiming to leverage concepts of the well-established field of (algebraic) topology toward applications for real-world data sets and machine learning.

Topology studies invariant properties of (topological) spaces under homeomorphisms (i.e., continuous transformations); in the following, we restrict ourselves to topological manifolds, so as to simplify the exposition. A fundamental problem in topology is about classification: *How can two manifolds be distinguished from each other?* Algebraic topology (Bredon, 1993; Hatcher, 2000) provides sophisticated and powerful tools to study this question. The basic idea being to associate computable *algebraic structures* (e.g., groups or vector spaces) to a manifold that remain *invariant* under homeomorphisms. A very important class of algebraic invariants are the *homology groups*, which encode a great deal of information while still being efficiently computable in many cases. Homology groups arise from combinatorial representations of the manifold, the *chain complexes*.

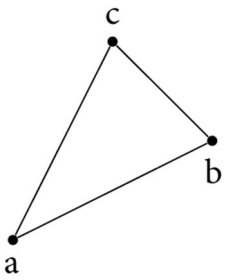


FIGURE 1 | A simplicial complex modelling a triangle.

2.1. Chain Complexes and Homology

The *standard k-simplex* Δ^k is defined as the convex hull of the standard basis vectors in $\mathbb{R}^{k+1}$, i.e.,

$$\Delta^k := \left\{ (x_0, \ldots, x_k) \in \mathbb{R}^{k+1} \;\middle|\; \sum_{i=0}^{k} x_i = 1, \quad x_i \geq 0 \ \forall i \right\}.$$

Similarly, a general *k-simplex* $[v_0, \ldots, v_k]$ is the convex hull of $k+1$ affinely independent points $v_0, \ldots, v_k$ in a Euclidean space. Note that deleting one of the *vertices* v_i from a k-simplex $[v_0, \ldots, v_k]$ yields a $(k-1)$-simplex $[v_0, \ldots, \hat{v}_i, \ldots, v_k]$ which is determined by the remaining vertices and called the *i-th face* of $[v_0, \ldots, v_k]$. Simplices are the basic building blocks of chain complexes that are used in algebraic topology for the computation of homological invariants. Any *topological manifold* X can be topologically modelled using simplices (see **Figure 1**). A *singular k-simplex* in X is a continuous map $\sigma : \Delta^k \to X$. It is not required that σ is an embedding, for instance any constant map, mapping so a single point in X is a valid singular simplex. The inclusion of the i-th face of Δ^k is an important singular simplex in Δ^k, which we will denote by $F_i^k : \Delta^{k-1} \to \Delta^k$. To keep the exposition simple we will restrict ourselves to working over the two element field $\mathbb{F}_2 := \mathbb{Z}/2\mathbb{Z}$ in what follows. Given any space X, its *singular k-chains* are elements of the $\mathbb{F}_2$-vector space $C_k(X)$ generated by the set of all singular k-simplices in X. Elements in $C_k(X)$ are thus "formal sums" of simplices. The *singular chain complex* $(C(X), \partial)$ of X is the sequence of spaces

$$\ldots \xrightarrow{\partial_{d+1}} C_d(X) \xrightarrow{\partial_d} C_{d-1}(X) \xrightarrow{\partial_{d-1}}$$

$$\ldots \xrightarrow{\partial_2} C_1(X) \xrightarrow{\partial_1} C_0(X) \xrightarrow{\partial_0} 0,$$

together with the *boundary maps* $\partial_k : C_k(X) \to C_{k-1}(X)$ given by

$$\partial_k(\sigma) := \sum_i \sigma \circ F_i^k$$

on the basis elements and extended linearly. A crucial property of the boundary maps is that they compose to 0, that is

$\partial_k \circ \partial_{k-1} = 0$. Elements of $Z_k(X) := \ker(\partial_k)$ are called *k-cycles* and those of $B_k(X) := \mathrm{im}(\partial_{k+1})$ are called *k-boundaries* and their well-defined quotient

$$H_k(X) := Z_k(X)/B_k(X)$$

is the *k-th singular homology group* of X (despite the name, this is still technically a quotient vector space; however, the group-theoretical viewpoint is more convenient and prevalent in algebraic topology). The homology groups are *topological invariants*, i.e., they remain invariant under homeomorphisms and therefore encode intrinsic information on the topology of X. Thus, homology groups and simpler invariants derived from them, such as the *Betti-numbers* $\beta_k := \dim H_k(X)$, are useful in studying the classification question raised above. For example, the 0-th Betti number β_0 is a count of the connected components of a space, while β_1 is a count of the number of cycles.

2.1.1. Brief Example

Using the simplicial complex in **Figure 1**, we briefly illustrate some of the aforementioned concepts. Let $X = \{\{a\},\{b\},\{c\},\{a,b\},\{b,c\},\{a,c\},\{a,b,c\}\}$ be the representation of the simplicial complex. The *boundary* of the triangle is non-trivial, i.e., $\partial_2\{a,b,c\} = \{b,c\} + \{a,c\} + \{a,b\}$ The boundary of this chain of edges is trivial, though, because duplicate simplices cancel each other out. We get $\partial_1\big(\{b,c\} + \{a,c\} + \{a,b\}\big) = \{c\} + \{b\} + \{c\} + \{a\} + \{b\} + \{a\} = 0$, which is consistent with the property of compatible boundary maps to compose to 0. To compute $H_1(X) := Z_1(X)/B_1(X)$, we only have to calculate $Z_1(X)$; the boundary group $B_1(X)$ does not contain any non-trivial simplices because X does not contain any 2-simplices. By definition, $Z_1(X) = \ker(\partial_1) = \mathrm{span}\big(\{a,b\} + \{b,c\} + \{a,c\}\big)$. This is the *only* cycle in X (which we can easily verify either by inspection or based on combinatorics). Hence $H_1(X) = Z_1(X) = \mathbb{F}_2$ and $\beta_1 = 1$; the triangle therefore exhibits a single cycle, which aligns with our intuition.

2.2. Persistent Homology

Persistent homology (Edelsbrunner et al., 2000; Zomorodian and Carlsson, 2005) is the flagship tool of TDA. In the analysis of real-world data, it is typically not a priori clear at what *scale* interesting topological features occur. By using a filtration (connected to the scale parameter) persistent homology is able to capture topological changes across the whole range of scales and store this information in so-called persistence diagrams.

Persistent homology is an extension of homology to the setting of filtered chain complexes. A *filtered chain complex* is a (not-necessarily strictly) ascending sequence of chain complexes $C^{\varepsilon_0} \subset C^{\varepsilon_1} \subset C^{\varepsilon_2} \subset \ldots$ with inclusion maps $\iota^i : C^{\varepsilon_i} \hookrightarrow C^{\varepsilon_{i+1}}$ and $\iota^{i,j} := \iota^j \circ \iota^{j-1} \circ \cdots \circ \iota^i : C^{\varepsilon_i} \hookrightarrow C^{\varepsilon_j}$ for $i < j$. Filtered chain complexes naturally arise in situations where we have a sequence of inclusions of spaces $X^{\varepsilon_0} \subset X^{\varepsilon_1} \subset X^{\varepsilon_2} \subset \ldots$. Such cases, for instance, occur if we consider the sublevel sets $X^\varepsilon := f^{-1}(\mathbb{R}_{<\varepsilon})$ of a so-called *filtration* function $f : X \to \mathbb{R}$, or if we consider a point cloud Y in a metric space (M, d) and set

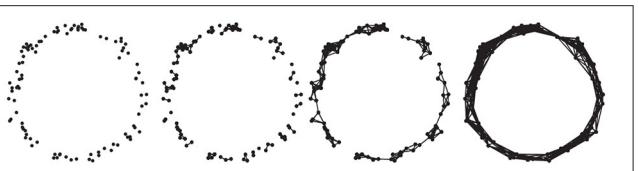


FIGURE 2 | Different stages of a Vietoris–Rips filtration for a simple "circle" point cloud. From left to right, connectivity of the underlying simplicial complex increases as ϵ increases.

$$Y^\varepsilon := \bigcup_{y \in Y} B_\varepsilon(y) = g^{-1}(\mathbb{R}_{<\varepsilon})$$

with filtration function $g : M \to \mathbb{R}$ given by $g(m) := \inf_{y \in Y} \mathrm{d}(m,y)$. Here $B_\varepsilon(y)$ denotes the open ball of radius ε centred at y and we implicitly identify $\varepsilon \simeq \varepsilon'$ if X^ε (resp. Y^ε) is canonically homeomorphic to X^δ (resp. Y^δ) for all $\delta \in [\varepsilon, \varepsilon']$. An important property of (singular) homology is that it is *functorial* (see e.g., Bredon, 1993), which implies that the inclusion maps $\iota^{i,j}$ induce maps on the respective homology groups $H_k(\iota^{i,j}) : H_k(C^{\varepsilon_i}) \to H_k(C^{\varepsilon_j})$. **Figure 2** depicts the Vietoris–Rips complex construction based on a distance filtration, a standard construction in TDA. The *k-th persistent homology groups* are the images of these inclusions, that is

$$H_k^{i,j} := \mathrm{im}\, H_k(\iota^{i,j}) = Z_k(C^{\varepsilon_i})/(B_k(C^{\varepsilon_j}) \cap Z_k(C^{\varepsilon_i})),$$

and thus precisely consist of the k-th homology classes of C^{ε_i} that still exist after taking the inclusion $H_k(\iota^{i,j})$. A homology class $\alpha \in H_k(C^{\varepsilon_i})$ is said to be *born* at C^{ε_i} if $\alpha \notin H_k^{i-1,i}$, i.e., if it is not in the image of $H_k(\iota^{i-1,i})$. If α is born at C^{ε_i}, it is said to *die* at C^{ε_j} if $H_k(\iota^{i,j-1})(\alpha) \notin H_k^{i-1,j-1}$ and $H_k(\iota^{i,j})(\alpha) \in H_k^{i-1,j}$. The *persistence* of α is given by $\varepsilon_j - \varepsilon_i$ and set to infinity if it never dies. The *persistent Betti-numbers*, defined by $\beta_k^{i,j} := \dim H_k^{i,j}$, carry information on how the homology (and thus the topology) changes across the filtration.

This information can be captured in a so-called *persistence diagram*, a multiset in $\overline{\mathbb{R}}^2 := \mathbb{R}^2 \cup \mathbb{R} \times \{\infty\}$. Specifically, the persistence diagram of (homological) *dimension k* is given by the points $(\varepsilon_i, \varepsilon_j) \in \overline{\mathbb{R}}^2$ with multiplicity

$$\mu_k^{i,j} := (\beta_k^{i,j-1} - \beta_k^{i,j}) - (\beta_k^{i-1,j-1} - \beta_k^{i-1,j})$$

for all $i < j$. The multiplicity $\mu_k^{i,j}$ counts the number of k-th homology classes that are born at C^{ε_i} and die at C^{ε_j}. **Figure 3** depicts a simple persistence diagram, calculated from the Vietoris–Rips complex in **Figure 2**. The axes of this diagram correspond to the ϵ values at which topological features are created and destroyed, respectively. The single point of high persistence corresponds to the primary topological feature of the point cloud, namely its circular shape. Other topological features occur at smaller scales—lower values of ϵ—and hence form a small dense cluster in the lower-left corner of the persistence diagram. The persistent Betti-numbers can be recovered from the persistence diagram itself; see Edelsbrunner and Harer, 2010.

A crucial fact that makes persistent homology valuable for application in data analysis is its *stability with respect to perturbations* of the filtration function. This means that persistent homology is robust to noise and constitutes an encoding of intrinsic topological properties of the data. More precisely, the space of persistence diagrams can be endowed with a metric induced by the *bottleneck distance* (or the *Wasserstein distances*) Edelsbrunner and Harer, 2010. A celebrated stability theorem (Cohen-Steiner et al., 2007) states that the L_∞-distance of two real-valued functions f and g is an upper bound for the bottleneck distance W_∞ of their respective persistence diagrams $\mathcal{D}_f$ and $\mathcal{D}_g$, i.e., $W_\infty(\mathcal{D}_f, \mathcal{D}_g) \leq \|f - g\|_\infty$. The stability theorem and its variants (Skraba and Turner, 2020) are highly relevant for applications because they imply that the behaviour of persistent homology under noise is known; descriptors such as persistence diagrams change continuously as the input function is varied, and the "amplitude" of their change is bounded from above via the stability theorem.

3. SURVEY

This section comprises the main part of the paper, where we gather and discuss pertinent methods and tools in topological machine learning. We broadly group the methods into the following categories. First, in section 3.2, we discuss methods that deal with *extrinsic topological features*. By the qualification *extrinsic*, we mean that no analysis of the topology of the machine learning model or the neural network itself is incorporated. These methods are instead mainly concerned with enabling the use of topological features, extracted from a given data set, in downstream machine learning models. This can be achieved through *vectorisation* of topological features or by designing specialised layers of neural networks that are capable of handling such features. Next, section 3.3 discusses *intrinsic topological features*. Those are methods that incorporate the topological analysis of aspects of the machine learning model itself. Whenever applicable, we further classify methods into *observational* and *interventional* methods. This sub-classification specifies *how* the methods are applied in a machine learning framework. Observational methods "observe" the topology of the data or model but they do not *directly* influence the model training or architecture. Interventional methods, by contrast, apply topological properties of the data, as well as *post-hoc* analysis of topological features of machine learning models, in order to inform the architectural design and/or model training. See **Figure 4** for an overview of the methods and their categories, as well as **Table 1** for the classification of all papers mentioned in this survey.

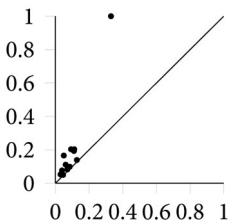


FIGURE 3 | A persistence diagram containing 1-dimensional topological features (cycles).

3.1. Limitations

Our paper selection is a cross-section over major machine learning conferences and machine learning journals. We refrain from comparing methods on certain tasks—such as classification—because there is considerable heterogeneity in the experimental setup, precluding a *fair* assessment of such methods.

3.2. Extrinsic Topological Features in Machine Learning

This section gives an overview of methods that aim at suitably representing topological features in order to use them as input features for machine learning models. We will refer to this class of methods as *extrinsic topological features in machine learning*, as they take topological information of the data sets into account, as opposed to intrinsic topological information of the machine learning framework itself (see section 3.3). A large class of such methods is comprised of *vectorisation* methods, that aim to transform persistent homology information into a feature vector form in order to make use of it in machine learning models.

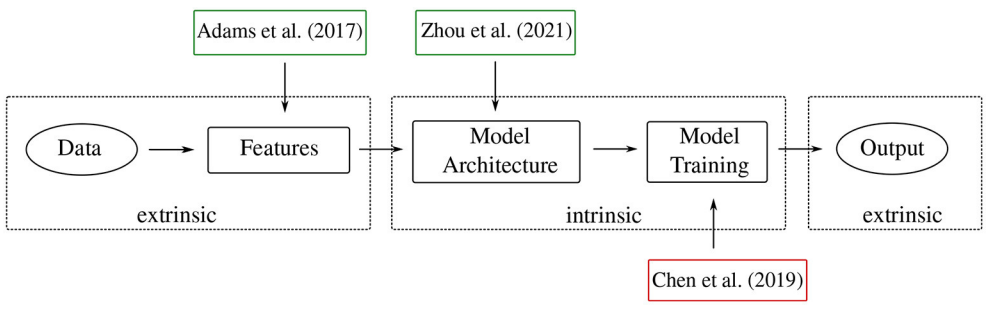


FIGURE 4 | This overview figure shows examples of methods discussed in the survey and their range of influence. Green (red) boxes signify *observational* (*interventional*) methods. **Table 1** provides a more in-depth classification of all methods.

TABLE 1 | The categorisation of the approaches discussed in the present survey.

Extrinsic		Intrinsic	
Observational	**Interventional**	**Observational**	**Interventional**
Adams et al., 2017	Carrière et al., 2020	Gabrielsson and Carlsson, 2019	Chen et al., 2019
Bubenik, 2015	Kim et al., 2020	Khrulkov and Oseledets, 2018	Hofer et al., 2017
Carrière et al., 2015	Zhao and Wang, 2019	Zhou et al., 2021	Hofer C. et al., 2019
Carrière et al., 2017			Hofer et al., 2020a
Kusano et al., 2018			Hofer et al., 2020b
Reininghaus et al., 2015			Moor et al., 2020
Rieck et al., 2020a			Ramamurthy et al., 2019
Rieck et al., 2020b			Rieck et al., 2019b
Umeda, 2017			Zhao et al., 2020

It is interesting to note that intrinsic features tend to be used more in interventional settings, whereas extrinsic features remain observational for the most part.

However, alternative representations of topological descriptors, such as kernels or function-based representations, are also discussed in this section.

3.2.1. Vector-Based and Function-Based Representations

Persistence diagrams (see section 2) constitute useful descriptors of homological information of data. However, being multisets, they cannot be used *directly* as input data for machine learning models in the usual sense (recent paradigm shifts in machine learning, namely the introduction of *deep sets* (Zaheer et al., 2017), challenge this assumption somewhat, as we will later see in section 3.2.3). One first needs to suitably represent—or *vectorise*—persistence diagrams (PDs) in order to use them for downstream machine learning tasks. There are two predominant strategies for facilitating the integration of topological features into machine learning algorithms, namely (i) different representations that ideally give rise to feature vectors, and (ii) kernel-based methods that permit the integration into certain classifiers. Notice that these two strategies are not necessarily exclusionary; some representations, for example, also give rise to a kernel-based method.

Representations and kernel-based methods should ideally be efficiently computable, satisfy similar stability properties as the persistence diagrams themselves—hence exhibiting robustness properties with respect to noise—as well as provide some interpretable features. The stability of such representations is based on the fundamental stability theorem by Cohen-Steiner et al. (2007). In recent years, a multitude of suitable representation methods have been introduced; we present a selection thereof, focusing on representations that have already been used in machine learning contexts. As a somewhat broad categorisation, we observe that persistence diagrams are often mapped into an auxiliary *vector space*, e.g., by discretisation (Anirudh et al., 2016; Adams et al., 2017), or by mapping into a (Banach- or Hilbert-) *function space* (Chazal et al., 2014; Bubenik, 2015; Di Fabio and Ferri, 2015). Alternatively, there are several *kernel methods* (Reininghaus et al., 2015; Carrière et al., 2017; Kusano et al., 2018) that enable the efficient calculation of a similarity measure between persistence diagrams. Representations and kernel-based methods fall into the category of what we denote "observational" methods. The only exception is given by PersLay (Carrière et al., 2020), which informs the layers of the model and thus is an "interventional" method.

Arguably the most simple form of employing topological descriptors in machine learning tasks uses *summary statistics*, such as the total persistence of a persistence diagram (Cohen-Steiner et al., 2010), its p-norm (Chen and Edelsbrunner, 2011), or its persistent entropy (Atienza et al., 2019), i.e., the Shannon entropy of the individual persistence values in a diagram. While all of these approaches result in scalar-valued summary statistics, they are often not directly applicable to complex machine learning tasks, which require more expressive representations. We note, however, that such statistics give rise to hypothesis testing (Blumberg et al., 2014) based on topological information and we envision that this field will become more prominent as topological features find their use case for data analysis. A simple and stable representation of persistence diagrams, suitable for machine learning tasks, is provided by what are commonly called *Betti curves*. Given a persistence diagram $\mathcal{D}$, and a weight function $w : \mathbb{R}^2 \to \mathbb{R}$, its Betti curve is the function $\beta : \mathbb{R} \to \mathbb{R}$ defined by

$$\beta(t) \ := \sum_{(b,d)\in\mathcal{D}} w(b,d) \cdot \mathbb{1}_{[b,d]}(t), \tag{1}$$

where

$$\mathbb{1}_{[b,d]}(t) \ := \begin{cases} 1, \text{ if } t \in [b,d] \\ 0, \text{ else} \end{cases} \tag{2}$$

is the indicator function. The Betti curve was often informally used to analyse data (Umeda, 2017); recently, Rieck et al. (2020a) provided a summarising description of their features. **Figure 5** depicts a simple illustration of the calculation of Betti curves. Betti curves are advantageous because they permit the calculation of a *mean* curve, next to providing an easy-to-evaluate distance and kernel method. Chevyrev et al. (2018) used this representation—and related "paths" derived from a persistence diagram and its representations—to solve classification tasks, using random forests and support vector machine classifiers. One

drawback of the Betti curves is their limited expressive power. Being a summary statistic of a persistence diagram, the mapping from a diagram to a curve is not injective; moreover, the curve only contains *counts* of topological features and does not permit tracking single features, for example.

A more fundamental technique, developed by Carrière et al. (2015), *directly* generates a high-dimensional feature vector from a persistence diagram. The main idea is to obtain a vector representation of some persistence diagram $\mathcal{D}$ based on the distribution of pairwise distances of its elements, including points on the diagonal $\Delta := \{(x,x) \mid x \in \mathbb{R}\} \subset \mathbb{R}^2$. More precisely, for each pair (p,q) of points in $\mathcal{D}$, they compute $m(p,q) := \min\{d_\infty(p,q), d_\infty(p,\Delta), d_\infty(q,\Delta)\}$ and associate to $\mathcal{D}$ the vector of these values, sorted in descending order. As persistence diagrams may be of different sizes, they enlarge each of these vectors by zeros so that its length matches the length of the longest vector in the set. Hence, the set of persistence diagrams one considers needs to be fixed a priori. This vectorisation does not necessarily scale well to large data sets, but it can provide a good baseline to furnish *any* machine learning classifier—including a neural network—with simple topology-based feature vectors. The use of this technique appears to be restricted at present; we hope that our article will help increase its adoption.

As a somewhat more complicated, but also more expressive, representation, Bubenik (2015) introduced topological descriptors called *persistence landscapes* that map persistence diagrams into a (Banach or Hilbert) function space in an invertible manner that satisfies stability properties with respect to the bottleneck distance of PDs. The *persistence landscape* $\lambda : \mathbb{N} \times \mathbb{R} \to \mathbb{R}$ of a PD $\mathcal{D} = \{(b_i, d_i)\}_{i \in I}$ can be defined in the following way. For $b < d$, we consider the auxiliary function $f_{(b,d)}(t) := \max\{0, \min\{t-b, d-t\}\}$ and define the persistence landscape as

$$\lambda(k,t) := \operatorname{kmax}\{f_{(b_i,d_i)}(t)\}_{i \in I},$$

where kmax denotes the k-th largest element of the set. In addition to injectivity and stability, persistence landscapes do not require any choice of auxiliary parameters in their construction (see **Figure 6** for a depiction of the persistence landscape computation process). They also afford various summary statistics, such as a norm calculation as well the calculation of both a kernel and a distance measure, making them a versatile representation of topological features. While, persistence landscapes have seen applications in time series analysis (Stolz et al., 2017), their most successful integration into machine learning algorithms is provided in the form of a new *layer*: persistence landscapes form the basis of a robust (with respect to noise) topological layer for deep neural networks, which is differentiable with respect to its inputs, the so-called PLLay (persistence landscape based topological layer) established in Kim et al. (2020). This layer exhibits good performance in image classification tasks as well as orbit classification, where it is shown to provide new state-of-the-art performance. We note that persistence landscapes are often considered in a vectorised form, which is obtained through binning their domain. While this is possible and useful for certain applications, we want to stress that the persistence landscape, as a lossless representation, should ideally be treated as such. The calculation of persistence landscapes imposes additional computational complexity, but the empirical performance reported by Kim et al. (2020) suggests that the landscapes are well-suited as a feature descriptor.

The *persistence images* (PIs), introduced by Adams et al. (2017), constitute an elegant hierarchical vectorisation step, representing a PD as a vector through the following steps. First the PD $\mathcal{D}$ is transformed from "birth–death"-coordinates into "birth–persistence"-coordinates via the transformation

$$T : \mathbb{R}^2 \longrightarrow \mathbb{R}^2 : (x,y) \longmapsto (x, y-x).$$

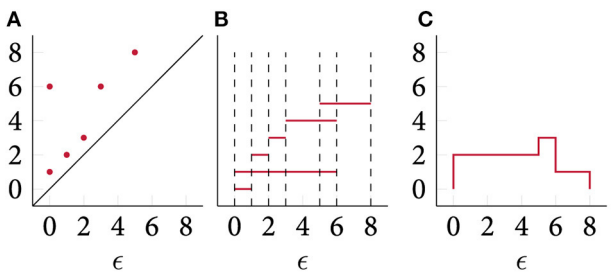


FIGURE 5 | A persistence diagram **(A)**, its persistence barcode **(B)**, and its corresponding Betti curve **(C)**. Notice that the *interpretation* of the axes of different plots is different, hence we exclude labels for the barcode representation.

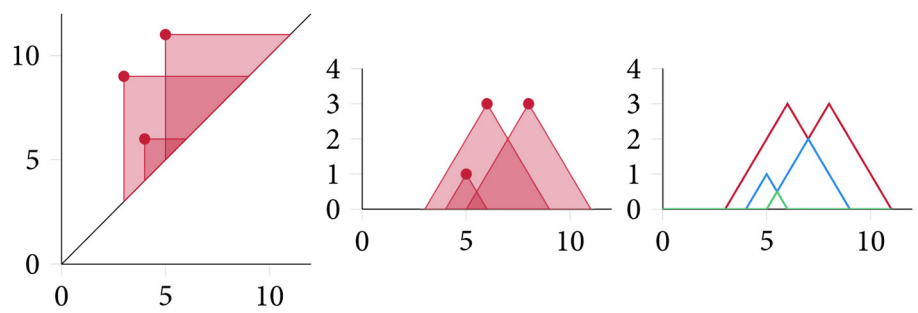


FIGURE 6 | Computing a *persistence landscape* involves calculating the "area of influence" of each topological feature in a persistence diagram. Each connected shaded region with at least k intersections forms the basis of the k-th persistence landscape, which can be obtained by "peeling off" layers in an iterative fashion.

FIGURE 7 | A persistence image arises as a discretisation of the density function (with appropriate weights) supported on a persistence diagram. It permits the calculation of an increasingly better-resolved sequence of images, which may be directly used as feature vectors.

Next, for each $u \in \mathbb{R}^2$ a differentiable probability density ϕ_u on $\mathbb{R}^2$ is chosen (the standard choice being a normalised symmetric Gaussian with $\mathbb{E}[\phi_u] = u$), as well as a weighting function $f : \mathbb{R}^2 \longrightarrow \mathbb{R}^2_{\geq 0}$ satisfying $f|_{\{0\}\times\mathbb{R}} \equiv 0$. Additionally one chooses a discretisation of a relevant subdomain of $\mathbb{R}^2$ by a standard grid. Each region R of this grid then corresponds to a pixel in the persistence image with value given by

$$\int_R \sum_{u \in T(\mathcal{D})} f(u)\phi_u(z)\mathrm{d}z.$$

In the process of generating persistence images, there are three non-canonical choices to be made. First, the choice of the weighting function, which is often chosen to emphasise features in the PD with large persistence value, next the distributions ϕ_u, and lastly the resolution of the discretisation grid. Adams et al. (2017) prove that PIs are stable with respect to the 1-Wasserstein distance between persistence diagrams. **Figure 7** illustrates their calculation. Persistence images are highly flexible and are often employed to make a classifier "topology-aware" to some extent (Zhao and Wang, 2019; Carrière and Blumberg, 2020; Rieck et al., 2020b). A paper by Zhao and Wang (2019), for instance, showcases their utility for graph classification. Interestingly, this paper constitutes also one of the few interventional approaches that employ extrinsic topological features; specifically, the authors use pre-defined filtrations to obtain graph-based persistence diagrams, and learn task-based weights for individual "pixels" (or "cells") in the diagram. This approach is seen to surpass several graph classification algorithms on standard benchmark data sets—a remarkable feat, considering that the method does not employ any label information. The main drawbacks of persistence images are their quadratic storage and computation complexity, as well as the choice of appropriate parameters. While recent work found them to be remarkably stable in practice with respect to the Gaussian kernel parameters (Rieck et al., 2020b), there are no guidelines for picking such hyperparameters, necessitating a (cross-validated) grid search, for instance.

3.2.2. Kernel-Based Representations

As an alternative to the previously-discussed representations, we now want to briefly focus on persistence diagrams again. The space of persistence diagrams can be endowed with metrics, such as the bottleneck distance. However, there is no natural Hilbert space structure on it, and such metrics tend to be computationally prohibitive or require the use of complex approximation algorithms (Kerber et al., 2017). Kernel methods provide a way of implicitly introducing such a Hilbert space structure to which persistence diagrams can be mapped via the feature map of the kernel. This then allows for a downstream use in machine learning models. To be more specific, given a set X, a function $k : X \times X \to \mathbb{R}$ is called a (positive definite) *kernel* if there exists a Hilbert space $\mathcal{H}_k$ together with a *feature map* $\phi : X \to \mathcal{H}_k$ such that $k(x_1, x_2) = \langle \phi(x_1), \phi(x_2) \rangle_{\mathcal{H}_k}$ for all $x_1, x_2 \in X$. Thus, by defining a kernel on the set of persistence diagrams, one obtains a vector representation via the feature map. However, in order for such a kernel to be useful in practice, it should additionally preserve the metric stability properties of persistence diagrams. Some pertinent examples of the kernel method are the following. Reininghaus et al. (2015) define a kernel on the set of persistence diagrams that is stable with respect to the 1-Wasserstein distance (Villani, 2009). The kernel is based on the idea of heat diffusion on a persistence diagram and offers a feature map that can be discretised (in fact, there are interesting similarities to persistence images). It was subsequently shown to satisfy *universality* (Kwitt et al., 2015), a desirable property for a kernel to have because it implies suitability for hypothesis testing. The *sliced Wasserstein kernel*, which is metric-preserving, was introduced by Carrière et al. (2017). It is based on the idea of the sliced Wasserstein distance (Kolouri et al., 2016), which ensures positive definiteness of the kernel through low-dimensional projections. Kusano et al. (2018) propose *persistence weighted Gaussian kernels* that incorporate a weighting and satisfy stability results with respect to the bottleneck distance and the 1-Wasserstein distance. The expressive power of kernels is in contrast to their computational complexity. Naïve implementations scale quadratically in the number of points, thus impeding the use of kernels for persistence diagrams with a large number of points. Some mitigation strategies exist (Greengard and Strain, 1991; Rahimi and Recht, 2008), but have not been adopted by implementations so far (moreover, their use is not always applicable, necessitating additional research). Nevertheless, such kernels are attractive because they are *not* limited with respect to the input data. Most of the papers exhibit good performance for shape classification or segmentation tasks, as well as in orbit classification.

While most of the aforementioned kernels are used to directly compare persistence diagrams, there are also examples of kernels *based on* topological information. An interesting example is provided by Rieck et al. (2019a), who introduce the Persistent Weisfeiler–Lehman (P-WL) kernel for graphs. It computes topological features during a Weisfeiler–Lehman (WL) procedure. The WL procedure refers to an iterative scheme in

which vertex label information is aggregated over the neighbours of each vertex, resulting in a label multiset. A perfect hashing scheme is now applied to every multiset and the graph is relabelled with the ensuing hashes. This process can be repeated until a pre-defined limit has been reached or until the labels do not change any more. While originally intended as a test for graph isomorphism, it turns out that there are non-isomorphic graphs that cannot be distinguished by the WL procedure. However, it turns out to be an exceptionally useful way of assessing the dissimilarity between two graphs in polynomial time, leading to the WL kernel framework (Shervashidze and Borgwardt, 2009; Shervashidze et al., 2011), which enjoys great popularity for graph learning tasks (Borgwardt et al., 2020; Kriege et al., 2020). The P-WL extension of WL is characterised by its capability to extract topological information of the graph with respect to the current node labelling for each WL iteration. This kernel is particularly notable since it constitutes the first (to our knowledge) method that imbues data-based labels into the calculation of persistent homology.

3.2.3. Integrating Topological Descriptors Into Neural Networks

One of the seminal methods that built a bridge between modern machine learning techniques and TDA is a work by Hofer et al. (2017). Using a differentiable projection function for persistence diagrams (with learnable parameters), the authors demonstrate that persistence diagrams of a data set can be easily integrated into *any* deep learning architecture. While the primary focus of the paper lies on developing such a projection function, the authors demonstrate the general feasibility of topological descriptors in both shape and graph classification tasks. A follow-up publication (Hofer C. D. et al., 2019) discusses more theoretical requirements for learning representations of topological descriptors.

This approach, as well as the development of the "DeepSets" architecture (Zaheer et al., 2017), which makes deep learning methods capable of learning *sets*, i.e., unordered sequences of varying cardinalities, spurred the development of *layers* that can be easily integrated into a deep learning workflow. An excellent example of such a layer is Carrière et al. (2020), which employs extended persistence (Cohen-Steiner et al., 2009) and heat kernel signatures to *learn* a vectorisation of persistence diagrams suited to the learning task at hand. PersLay is a neural network layer, defined by

$$\mathrm{PersLay}(\mathcal{D}) := \mathrm{op}(\{w(p)\cdot\phi(p)\}_{p\in\mathcal{D}}),$$

where $\mathcal{D}$ is a persistence diagram, op is and permutation invariant mapping, $w:\mathbb{R}^2 \longrightarrow \mathbb{R}$ is a weight function and $\phi:\mathbb{R}^2 \longrightarrow \mathbb{R}^d$ is a vector representation function. Its generic definition allows PersLay to subsume and recover many existing representations by appropriate choices of op and ϕ (Carrière et al., 2020).

3.3. Intrinsic Topological Features in Machine Learning

This section reviews methods that either incorporate topological information directly into the design of a machine learning model itself, or leverage topology to study aspects of such a model. We refer to such features as *intrinsic topological features*. The primary examples are regularisation techniques as well as techniques for analysing neural network architectures.

3.3.1. Regularisation Techniques

As a recent example, Moor et al. (2020) propose a topological autoencoder, which aims to preserve topological features of the input data in low-dimensional representations. This is achieved via a regularisation term that incentivises the persistence diagrams of both the latent and input space to be topologically similar. This method acts on the level of mini-batches, treating each of them as a point cloud. Persistence diagrams are obtained from the Vietoris–Rips complex of each space. By tracking the simplices that are relevant for the creation and destruction of topological features, and by consistently mapping simplices to a given edge in the Vietoris–Rips complex, each filtration can be interpreted as a selection of distances from the full distance matrix of the point cloud. The proposed regularisation term then compares the "selected" distances in the data space with the corresponding distances in the latent space (and vice versa). Finally, this regularisation is differentiable under the assumption that the persistence diagram is discrete (i.e., for each of its points, there is an infinitesimal neighbourhood containing no other points). The scheme can thus be directly integrated into the end-to-end training of an autoencoder, making it aware of the topology in the data space. This work can also be considered as an extension of previous work by Hofer C. et al. (2019), who introduced a differentiable loss term for one-class learning that controls the topology of the latent space; in effect, their loss term enforces a preferred "scale" for topological features in the latent space. It does not have to harmonise topological features *across* different spaces. It turns out that an autoencoder trained with this loss term on unlabelled data can be used on other data sets for one-class learning. This hints at the fact that enforcing a certain topological structure can be beneficial for learning tasks; we will later see that such empirical observations can also be furnished with a theoretical underpinning.

An approach by Chen et al. (2019) takes a different perspective. The authors develop a measure of the *topological complexity* (in terms of connected components) of the classification boundary of a given classifier. Said topological information is then used for regularisation in order to force the topological complexity of the decision boundary to be simpler, containing fewer features of low persistence. Thus, topological information serves as a penalty during classification such that training the classifier itself can be improved. In contrast to the aforementioned approach, differentiability is obtained through a "surrogate" piecewise linear approximation of the classifier. The method is seen to yield competitive results and the authors observe that the method performs well even in the presence of label noise. Analysing the decision boundary of a classifier also turns out to be advantageous for *model selection*, as we will later see in section 3.3.2.

Hofer et al. (2020a) analyse more fundamental principles of regularisation by means of topological features. Specifically, they study regularisation in a regime of small sample sizes

with over-parametrised neural networks. By developing a new topological constraint for per-class probability measures, mass concentration effects in the vicinity of the learned representations of training instances are observed, leading to overall improvements of generalisation performance. The authors observe that controlling topological properties of learned representations presents numerous avenues for future research. These theoretical findings validate the empirical improvements observed in previous works of this domain.

As a more involved example of methods that make use of intrinsic features, Zhao et al. (2020) include topological features of graph neighbourhoods into a standard graph neural network (GNN) architecture. Their method combines a shortest-path filtration with persistence images, which are subsequently compressed to a single scalar value using a multilayer perceptron. The resulting scalar is then used to re-weight the message passing scheme used in training the GNN, thus obtaining topologically-based representations of graph neighbourhoods. In contrast to the previously-described loss terms, this method is not end-to-end differentiable, though, because the conversion from persistence diagrams to persistence images involves non-continuous parameters, i.e., the image dimensions. Zhao et al. (2020) primarily propose this method for node classification tasks, but we hypothesise that other graph tasks would profit from the integration of topological features.

Last, to provide a somewhat complementary perspective to preceding work, a paper by Hofer et al. (2020b) discusses how to employ graph neural networks (GNNs) to *learn* an appropriate filtration in an end-to-end fashion. The authors demonstrate that a GNN can be used to successfully initialise a scalar-valued filtration function, which can then subsequently be trained under mild assumptions (specifically, injectivity at the vertices of the graph needs to hold). The learned filtration turns out to surpass fixed filtrations combined with a persistent homology baseline, thus demonstrating the benefits of making topological representations differentiable—and thus *trainable*.

3.3.2. Model Analysis

Shifting our view from regularisation techniques, topological analysis has been applied to evaluate generative adversarial networks (GANs). A GAN (Goodfellow et al., 2014) is comprised of two sub-networks, a generator and a discriminator. Given a data distribution P_{data}, the generators objective is to learn a distribution P_{model} with the same statistics, whereas the discriminator learns to distinguish generated samples from actual data samples. The topological evaluation of GANs is motivated by the manifold hypothesis (Fefferman et al., 2013), which poses that a data distribution P_{data} is sampled from an underlying manifold $\mathcal{M}_{\text{data}}$. The idea is to assess the topological similarity of $\mathcal{M}_{\text{data}}$ and the underlying manifold $\mathcal{M}_{\text{model}}$ of the model generated distribution P_{model}. Based on the persistent homology of witness complexes, Khrulkov and Oseledets (2018) introduce the *Geometry Score*, which is a similarity measure of the topologies of $\mathcal{M}_{\text{data}}$ and $\mathcal{M}_{\text{model}}$ and can be used to evaluate generative models. Later work by Zhou et al. (2021) generalises this approach and additionally extends it to the disentanglement evaluation of generative models in unsupervised settings.

In a different direction, the topological analysis of the intrinsic structure of a classifier, such as a neural network, makes it possible to improve a variety of tasks. This includes the analysis of training behaviour as well as model selection—or *architecture selection* in the case of neural networks.

While the literature dedicated to the better understanding of deep neural networks has typically focused on its functional properties, Rieck et al. (2019b) took a different perspective to focus on the graph structure of a neural network. Specifically, they treat a (feed-forward) neural network as a stack of bipartite graphs. From this view, they propose "neural persistence," a complexity measure which summarizes topological features that arise when calculating a filtration of the neural network graph where the filtration weights are given by the network parameters. They showed that neural persistence can distinguish between well-trained and badly-trained (i.e., diverged) networks. This measure is oblivious to the functional behaviour of the underlying network, but only focuses on its (weighted) *structure*. Nevertheless, Rieck et al. (2019b) showed that it can be used for guiding early stopping solely based on topological properties of the neural network, potentially saving validation data used for the early stopping decision.

Ramamurthy et al. (2019) employ labelled variants of simplicial complexes, such as a labelled Vietoris–Rips complex, to analyse the decision boundary (i.e., classification boundary) of a given classifier. The authors are able to provide theoretical guarantees that the correct homology of a decision boundary can be recovered from samples, thus paving the way for an efficient approximation scheme that incorporates local scale estimates of the data set. Such a construction is required because the density of available samples is not guaranteed to be uniform, leading to simplicial complexes with spurious simplices in high-density regions, while running the risk of "undersampling" low-density regions. Next to "matching" models based on the *Decision Boundary Topological Complexity* (DBTC) score, Ramamurthy et al. (2019) also enable matching data sets to pre-trained models. The underlying assumption is that a model that closely mimics the topological complexity of a data set is presumably a better candidate for this particular data set.

Gabrielsson and Carlsson (2019) utilise topological data analysis to analyse topological information encoded in the weights of convolutional neural networks (CNNs). They show that the weights of convolutional layers encode simple global structures which dynamically change during training of the network and correlate with the network's ability to generalise to unseen data. Moreover, they find that topological information on the trained weights of a network can lead to improvements in training efficiency and reflect the generality of the data set on which the training was performed.

4. OUTLOOK AND CHALLENGES

This survey provided a glimpse of the nascent field of *topological machine learning*. We categorised existing work depending on its intention (interventional vs. observational) and according to what type of topological features are being

calculated (extrinsic vs. intrinsic), finding that most extrinsic approaches are observational, i.e., they do not inform the choice of model afterwards, while most intrinsic approaches are interventional, i.e., they result in changes to the choice of model or its architecture.

Numerous avenues for future research exist. Of the utmost importance is the improvement of the "software ecosystem." Software libraries such as `GUDHI` (Maria et al., 2014) and `giotto-tda` (Tauzin et al., 2021) are vital ingredients for increasing the adoption of TDA methods, but we envision that there is a specific niche for libraries that integrate *directly* with machine learning frameworks such as `pytorch`. This will make it easier to disseminate knowledge and inspire more research. A challenge that the community yet has to overcome involves the overall scalability of methods, though. While certain improvements on the level of filtrations are being made (Sheehy, 2013; Cavanna et al., 2015), those improvements have yet to be integrated into existing algorithms. A more fundamental question is to what extent TDA has to rely on "isotropic" complexes such as the Vietoris–Rips complex, and whether scale-dependent complexes that incorporate sparsity can be developed.

On the side of applications, we note that several papers already target problems such as graph classification, but they are primarily based on fixed filtrations (with the notable exception of Hofer et al. (2020b), who learn a filtration end-to-end). We envision that future work could target more involved scenarios, such as the creation of "hybrid" GNNs, and the use of end-to-end differentiable features for other graph tasks, such as node classification, link prediction, or community detection.

As another upcoming topic, we think that the analysis of time-varying data sets using topology-based methods is long overdue. With initial work by Cohen-Steiner et al. (2006) on time-varying topological descriptors providing a theoretical foundation, there are nevertheless few topology-based approaches that address time series classification or time series analysis. Several—theoretical and practical—aspects for such an endeavour are addressed by Perea et al. (2015), who develop a persistence-based method for quantifying periodicity in time series. The method is based on the fundamental embedding theorem by Takens (1981) and is combined with a sliding window approach. Future work could build on such approaches, or find other ways to characterise time-series, for instance based on complex networks (Lacasa et al., 2008). This could pave the road toward novel applications of TDA such as anomaly detection.

AUTHOR CONTRIBUTIONS

FH, MM, and BR performed the literature search and revised the draft. FH and BR drafted the original manuscript. All authors contributed to the article and approved the submitted version.

FUNDING

This work was partially funded and supported by the Swiss National Science Foundation [Spark grant 190466, FH and BR]. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

REFERENCES

Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., et al. (2017). Persistence images: a stable vector representation of persistent homology. *J. Mach. Learn. Res.* 18, 1–35.

Anirudh, R., Venkataraman, V., Ramamurthy, K. N., and Turaga, P. (2016). "A Riemannian framework for statistical analysis of topological persistence diagrams," in *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1023-1031. doi: 10.1109/CVPRW.2016.132

Atienza, N., Gonzalez-Diaz, R., and Rucco, M. (2019). Persistent entropy for separating topological features from noise in vietoris-rips complexes. *J. Intell. Inform. Syst.* 52, 637–655. doi: 10.1007/s10844-017-0473-4

Blumberg, A. J., Gal, I., Mandell, M. A., and Pancia, M. (2014). Robust statistics, hypothesis testing, and confidence intervals for persistent homology on metric measure spaces. *Found. Comput. Math.* 14, 745–789. doi: 10.1007/s10208-014-9201-4

Borgwardt, K., Ghisu, E., Llinares-Lopez, F., O'Bray, L., and Rieck, B. (2020). Graph kernels: state-of-the-art and future challenges. *Found. Trends Mach. Learn.* 13, 531–712. doi: 10.1561/2200000076

Bredon, G. E. (1993). *Topology and Geometry, Volume 139 of Graduate Texts in Mathematics. New York, NY: Springer-Verlag.* doi: 10.1007/978-1-4757-6848-0

Bubenik, P. (2015). Statistical topological data analysis using persistence landscapes. *J. Mach. Learn. Res.* 16, 77–102.

Carrière, M., and Blumberg, A. (2020). "Multiparameter persistence image for topological machine learning," in *Advances in Neural Information Processing Systems*, Vol. 33, eds H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Curran Associates, Inc.), 22432–22444.

Carrière, M., Chazal, F., Ike, Y., Lacombe, T., Royer, M., and Umeda, Y. (2020). "PersLay: a neural network layer for persistence diagrams and new graph topological signatures," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, eds S. Chiappa and R. Calandra (PMLR), 2786–2796.

Carrière, M., Cuturi, M., and Oudot, S. (2017). "Sliced Wasserstein kernel for persistence diagrams," in *Proceedings of the 34th International Conference on Machine Learning* eds D. Precup and Y. W. The (Sydney, NSW: International Convention Centre; PMLR), 664–673.

Carrière, M., Oudot, S., and Ovsjanikov, M. (2015). Stable topological signatures for points on 3D shapes. *Comput. Graph. Forum* 34, 1–12. doi: 10.1111/cgf.12692

Cavanna, N. J., Jahanseir, M., and Sheehy, D. R. (2015). "*A geometric perspective on sparse filtrations*," in *Proceedings of the Canadian Conference on Computational Geometry* (Kingston, ON).

Chazal, F., Fasy, B. T., Lecci, F., Rinaldo, A., and Wasserman, L. (2014). "Stochastic convergence of persistence landscapes and silhouettes," in *Proceedings of the Thirtieth Annual Symposium on Computational Geometry, SOCG'14* (New York, NY: Association for Computing Machinery), 474–483. doi: 10.1145/2582112.2582128

Chen, C., and Edelsbrunner, H. (2011). "Diffusion runs low on persistence fast," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (Red Hook, NY: Curran Associates, Inc.), 423–430. doi: 10.1109/ICCV.2011.6126271

Chen, C., Ni, X., Bai, Q., and Wang, Y. (2019). "A topological regularizer for classifiers via persistent homology," in *Proceedings of Machine Learning Research*, eds K. Chaudhuri and M. Sugiyama (PMLR), 2573–2582.

Chevyrev, I., Nanda, V., and Oberhauser, H. (2018). Persistence paths and signature features in topological data analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 42, 192–202. doi: 10.1109/TPAMI.2018.2885516

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2007). Stability of persistence diagrams. *Discrete Comput. Geom.* 37, 103–120. doi: 10.1007/s00454-006-1276-5

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2009). Extending persistence using Poincaré and Lefschetz duality. *Found. Comput. Math.* 9, 79–103. doi: 10.1007/s10208-008-9027-z

Cohen-Steiner, D., Edelsbrunner, H., Harer, J., and Mileyko, Y. (2010). Lipschitz functions have L_p-stable persistence. *Found. Comput. Math.* 10, 127–139. doi: 10.1007/s10208-010-9060-6

Cohen-Steiner, D., Edelsbrunner, H., and Morozov, D. (2006). "Vines and vineyards by updating persistence in linear time," in *Proceedings of the Twenty-Second Annual Symposium on Computational Geometry, SCG '06* (New York, NY: Association for Computing Machinery), 119–126. doi: 10.1145/1137856.1137877

Di Fabio, B., and Ferri, M. (2015). "Comparing persistence diagrams through complex vectors," in *Image Analysis and Processing – ICIAP 2015*, eds V. Murino and E. Puppo (Cham: Springer International Publishing), 294–305. doi: 10.1007/978-3-319-23231-7_27

Edelsbrunner, H., and Harer, J. (2010). *Computational Topology: An Introduction. Providence, RI: American Mathematical Society*. doi: 10.1090/mbk/069

Edelsbrunner, H., Letscher, D., and Zomorodian, A. (2000). "Topological persistence and simplification," in *Proceedings 41st Annual Symposium on Foundations of Computer Science* (Redondo Beach, CA), 454–463. doi: 10.1109/SFCS.2000.892133

Fefferman, C., Mitter, S., and Narayanan, H. (2013). Testing the manifold hypothesis. *J. Am. Math. Soc.* 29, 983–1049. doi: 10.1090/jams/852

Gabrielsson, R. B., and Carlsson, G. (2019). "Exposition and interpretation of the topology of neural networks," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 1069–1076. doi: 10.1109/ICMLA.2019.00180

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, Vol. 27, eds Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (Curran Associates, Inc.).

Greengard, L., and Strain, J. (1991). The fast Gauss transform. *SIAM J. Sci. Stat. Comput.* 12, 79–94. doi: 10.1137/0912004

Hatcher, A. (2000). *Algebraic Topology*. Cambridge: Cambridge University Press.

Hofer, C., Graf, F., Niethammer, M., and Kwitt, R. (2020a). "Topologically densified distributions," in *Proceedings of the 37th International Conference on Machine Learning*, eds H. Daumé III and A. Singh (PMLR), 4304–4313.

Hofer, C., Graf, F., Rieck, B., Niethammer, M., and Kwitt, R. (2020b). "Graph filtration learning," in *Proceedings of the 37th International Conference on Machine Learning*, eds H. Daumé III and A. Singh (PMLR), 4314–4323.

Hofer, C., Kwitt, R., Niethammer, M., and Dixit, M. (2019). "Connectivity-optimized representation learning via persistent homology," in *Proceedings of the 36th International Conference on Machine Learning*, eds K. Chaudhuri and R. Salakhutdinov (PMLR), 2751–2760.

Hofer, C., Kwitt, R., Niethammer, M., and Uhl, A. (2017). "Deep learning with topological signatures," in *Advances in Neural Information Processing Systems*, Vol. 30, eds I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc.).

Hofer, C. D., Kwitt, R., and Niethammer, M. (2019). Learning representations of persistence barcodes. *J. Mach. Learn. Res.* 20, 1–45.

Kerber, M., Morozov, D., and Nigmetov, A. (2017). Geometry helps to compare persistence diagrams. *ACM J. Exp. Algorith.* 22. doi: 10.1145/3064175

Khrulkov, V., and Oseledets, I. (2018). "Geometry score: a method for comparing generative adversarial networks," in *Proceedings of the 35th International Conference on Machine Learning*, eds J. Dy and A. Krause (Stockholm: PMLR), 2621–2629.

Kim, K., Kim, J., Zaheer, M., Kim, J., Chazal, F., and Wasserman, L. (2020). "PLLay: efficient topological layer based on persistent landscapes," in *Advances in Neural Information Processing Systems*, Vol. 33, eds H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Curran Associates, Inc.), 15965–15977.

Kolouri, S., Zou, Y., and Rohde, G. K. (2016). "Sliced Wasserstein kernels for probability distributions," *in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5258–5267. doi: 10.1109/CVPR.2016.568

Kriege, N. M., Johansson, F. D., and Morris, C. (2020). A survey on graph kernels. *Appl. Netw. Sci.* 5:6. doi: 10.1007/s41109-019-0195-3

Kusano, G., Fukumizu, K., and Hiraoka, Y. (2018). Kernel method for persistence diagrams via kernel embedding and weight factor. *J. Mach. Learn. Res.* 18, 1–41.

Kwitt, R., Huber, S., Niethammer, M., Lin, W., and Bauer, U. (2015). "Statistical topological data analysis—a kernel perspective," in *Advances in Neural Information Processing Systems*, Vol. 28, eds C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Curran Associates, Inc.).

Lacasa, L., Luque, B., Ballesteros, F., Luque, J., and Nuno, J. C. (2008). From time series to complex networks: the visibility graph. *Proc. Natl. Acad. Sci. U.S.A.* 105, 4972–4975. doi: 10.1073/pnas.0709247105

Maria, C., Boissonnat, J.-D., Glisse, M., and Yvinec, M. (2014). "The GUDHI library: simplicial complexes and persistent homology," in *Mathematical Software-ICMS 2014*, eds H. Hong and C. Yap (Berlin; Heidelberg: Springer), 167–174. doi: 10.1007/978-3-662-44199-2_28

Moor, M., Horn, M., Rieck, B., and Borgwardt, K. (2020). "Topological autoencoders," in *Proceedings of the 37th International Conference on Machine Learning*, eds H. Daumé III and A. Singh (PMLR), 7045–7054.

Perea, J., Deckard, A., Haase, S., and Harer, J. (2015). SW1PerS: sliding windows and 1-persistence scoring; discovering periodicity in gene expression time series data. *BMC Bioinformatics* 16:257. doi: 10.1186/s12859-015-0645-6

Rahimi, A., and Recht, B. (2008). "Random features for large-scale kernel machines," in *Advances in Neural Information Processing Systems*, Vol. 20, eds J. Platt, D. Koller, Y. Singer, and S. Roweis (Curran Associates, Inc.).

Ramamurthy, K. N., Varshney, K., and Mody, K. (2019). "Topological data analysis of decision boundaries with application to model selection," in *Proceedings of the 36th International Conference on Machine Learning*, eds K. Chaudhuri and R. Salakhutdinov (PMLR), 5351–5360.

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. (2015). "A stable multi-scale kernel for topological machine learning," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4741–4748. doi: 10.1109/CVPR.2015.7299106

Rieck, B., Bock, C., and Borgwardt, K. (2019a). "A persistent Weisfeiler-Lehman procedure for graph classification," in *International Conference on Machine Learning*, eds K. Chaudhuri and R. Salakhutdinov (PMLR), 5448–5458.

Rieck, B., Sadlo, F., and Leitte, H. (2020a). "Topological machine learning with persistence indicator functions," in *Topological Methods in Data Analysis and Visualization V*, eds H. Carr, I. Fujishiro, F. Sadlo, and S. Takahashi (Cham: Springer), 87–101. doi: 10.1007/978-3-030-43036-8_6

Rieck, B., Togninalli, M., Bock, C., Moor, M., Horn, M., Gumbsch, T., et al. (2019b). "*Neural* persistence: a complexity measure for deep neural networks using algebraic topology," in *International Conference on Learning Representations*.

Rieck, B., Yates, T., Bock, C., Borgwardt, K., Wolf, G., Turk-Browne, N., et al. (2020b). "Uncovering the topology of time-varying fMRI data using cubical persistence," in *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 33, eds H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Curran Associates, Inc.), 6900–6912.

Sheehy, D. R. (2013). Linear-size approximations to the Vietoris-Rips filtration. *Discrete Comput. Geom.* 49, 778–796. doi: 10.1007/s00454-013-9513-1

Shervashidze, N., and Borgwardt, K. (2009). "Fast subtree kernels on graphs," *in Advances in Neural Information Processing Systems*, Vol. 22, eds Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (Curran Associates, Inc.), 1660–1668.

Shervashidze, N., Schweitzer, P., van Leeuwen, E. J., Mehlhorn, K., and Borgwardt, K. M. (2011). Weisfeiler-Lehman graph kernels. *J. Mach. Learn. Res.* 12, 2539–2561.

Skraba, P., and Turner, K. (2020). Wasserstein stability for persistence diagrams. *arXiv preprint arXiv:2006.16824*.

Stolz, B. J., Harrington, H. A., and Porter, M. A. (2017). Persistent homology of time-dependent functional networks constructed from coupled time series. *Chaos* 27:047410. doi: 10.1063/1.4978997

Takens, F. (1981). "Detecting strange attractors in turbulence," in *Dynamical systems and turbulence, Warwick 1980 (Coventry, 1979/1980)*, eds D. Rand and L. S. Young (Berlin; New York, NY: Springer), 366–381. doi: 10.1007/BFb0091924

Tauzin, G., Lupo, U., Tunstall, L., Perez, J. B., Caorsi, M., Medina-Mardones, A. M., et al. (2021). giotto-tda: a topological data analysis toolkit for machine learning and data exploration. *J. Mach. Learn. Res.* 22, 1–6.

Umeda, Y. (2017). Time series classification via topological data analysis. *Trans. Jpn. Soc. Artif. Intell.* 32, 1–12. doi: 10.1527/tjsai.D-G72

Villani, C. (2009). *Optimal Transport, Volume 338 of Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Berlin: Springer-Verlag. doi: 10.1007/978-3-540-71050-9

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). "Deep sets," in *Advances in Neural Information Processing Systems*, Vol. 30, eds I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Curran Associates, Inc.).

Zhao, Q., and Wang, Y. (2019). "Learning metrics for persistence-based summaries and applications for graph classification," in *Advances in Neural Information Processing Systems*, Vol. 32, eds H. Wallach, H. Larochelle, A, Beygelzimer, Florence d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc.)

Zhao, Q., Ye, Z., Chen, C., and Wang, Y. (2020). "Persistence enhanced graph neural network," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, eds S. Chiappa and R. Calandra (PMLR), 2896–2906.

Zhou, S., Zelikman, E., Lu, F., Ng, A. Y., Carlsson, G. E., and Ermon, S. (2021). "Evaluating the disentanglement of deep generative models through manifold topology," in *International Conference on Learning Representations*.

Zomorodian, A., and Carlsson, G. (2005). Computing persistent homology. *Discrete Comput. Geom.* 33, 249–274. doi: 10.1007/s00454-004-1146-y

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Applied Mathematics and Statistics

ORIGINAL RESEARCH
published: 28 May 2021
doi: 10.3389/fams.2021.668082

Topology-Inspired Method Recovers Obfuscated Term Information From Induced Software Call-Stacks

Kelly Maggs[1] *and Vanessa Robins*[2]*

[1]Mathematical Sciences Institute, Australian National University, Canberra, ACT, Australia, [2]Research School of Physics, Australian National University, Canberra, ACT, Australia

Fuzzing is a systematic large-scale search for software vulnerabilities achieved by feeding a sequence of randomly mutated input files to the program of interest with the goal being to induce a crash. The information about inputs, software execution traces, and induced call stacks (crashes) can be used to pinpoint and fix errors in the code or exploited as a means to damage an adversary's computer software. In black box fuzzing, the primary unit of information is the call stack: a list of nested function calls and line numbers that report what the code was executing at the time it crashed. The source code is not always available in practice, and in some situations even the function names are deliberately obfuscated (i.e., removed or given generic names). We define a topological object called the call-stack topology to capture the relationships between module names, function names and line numbers in a set of call stacks obtained via black-box fuzzing. In a proof-of-concept study, we show that structural properties of this object in combination with two elementary heuristics allow us to build a logistic regression model to predict the locations of distinct function names over a set of call stacks. We show that this model can extract function name locations with around 80% precision in data obtained from fuzzing studies of various linux programs. This has the potential to benefit software vulnerability experts by increasing their ability to read and compare call stacks more efficiently.

Keywords: fuzzing, crash-triage, software vulnerability research, call-stack analysis, topology, TDA, specialization pre-order

OPEN ACCESS

Edited by:
Frédéric Chazal,
Inria Saclay-Île-de-France Research Center, France

Reviewed by:
André Lieutier,
Dassault Systèmes, France
Vincent Rouvreau,
Inria Saclay-Île-de-France Research Center, France

****Correspondence:***
Vanessa Robins
vanessa.robins@anu.edu.au

Specialty section:
This article was submitted to Mathematics of Computation and Data Science, a section of the journal Frontiers in Applied Mathematics and Statistics

Received: *15 February 2021*
Accepted: *03 May 2021*
Published: *28 May 2021*

Citation:
Maggs K and Robins V (2021) Topology-Inspired Method Recovers Obfuscated Term Information From Induced Software Call-Stacks. Front. Appl. Math. Stat. 7:668082. doi: 10.3389/fams.2021.668082

1 INTRODUCTION

A black-box fuzzing campaign is one conducted without explicit knowledge of the source code or its intermediate representations. Generally, methods in this area require a brute-force generation of inputs. This can lead to masses of crashes where many are duplicates of one another. For practitioners, untangling the output of a black-box fuzzing campaign is a time-consuming task. The goal of this article is to investigate methods that alleviate the difficulty of comprehending such results.

1.1 Call Stacks

When a program crashes, the slew of error text it returns to the user is referred to as the call-stack (Example in **Figure 1**). The call-stack is a record of the nested functions traced out by the program in its final moments and is one of the few pieces of information available to us when analyzing black-box fuzzing. The lines in the call-stack are called frames, and while contingent on the operating system's debugging syntax, decompose roughly into three columns: 1) the module (or filename), 2)

Frame	Module	Function	Line Num.
0	parser.c	xmlParseCDSect	9750
1	parser.c	xmlParseContent	9806
2	parser.c	xmlParseElement	9995
3	parser.c	xmlParseContent	9822
4	parser.c	xmlParseElement	9995
5	parser.c	xmlParseDocument	10665
6	parser.c	xmlDoRead	15062
7	parser.c	xmlReadFile	15122
8	xmllimit.c	parseAnPrintFile	2382

Frame	Module	Function	Line Num.
0	parser.c	????	9750
1	parser.c	????	9806
2	parser.c	????	9995
3	parser.c	????	9822
4	parser.c	????	9995
5	parser.c	????	10665
6	parser.c	????	15062
7	parser.c	????	15122
8	xmllimit.c	????	2382

FIGURE 1 | A simplified call-stack in our data-set with and without function terms.

the function and 3) the line number. We will refer to the set of all constituent modules, functions and line numbers in a set of call-stacks $\mathcal{C}$ as the *terms* in $\mathcal{C}$.

Further complicating matters is that—depending on whether the source code is available—call-stacks may have partial information excluded. In particular, when fuzzing programs without possession of source code or full knowledge of terms, the partially obscured call-stack may be the only source of information available.

1.2 Goals

The ANU researchers [1] provided to us a data set of call stacks generated by fuzzing several Linux programs with the afl fuzzing algorithm (See [2]). They were interested in answering two key research questions:

1. Clustering and Deduplication: determining the extent to which there are discernible clusters in the set of call-stacks.
2. Term Removal: quantifying how much information about function terms can be recovered given they are obscured (for example, as in **Figure 1**).

While the first question has been studied in a number of contexts [3, 4], to the best of our knowledge no attempt has been made at the second. In this paper, we show that once the data has been suitably whitelisted, the set of crashes contain a high number of exact duplicate call-stacks. This observation highlights a fundamental lack of diversity in the data generated by fuzzing, and alone is enough to answer the first question to a large extent.

To address the question of function term removal, we introduce a model of call-stack information using finite topological spaces, posets and the theory of [5]. This not only helps to quantify the significance of removing function terms, but is a useful object to capture the dependencies between terms in the set of call-stacks.

2 DATA-SET OVERVIEW

Six common Linux programs were fuzzed using the program afl. Key aspects of the program: the binary name, file extension, and version are presented in **Table 1**. Call-stacks were generated within the framework of the GDB debugger using the AddressSanitizer (ASAN) [6] tool.

Upon recommendation by the ANU cyber-security researchers, we performed several pre-processing whitelisting steps to each call-stack text file. Firstly, frames appearing up to and including the ASAN error frame were considered superfluous and hence deleted. In crashes that did not call the ASAN module, frames up to assert_fail were deleted. For every file, the two final generic end-of-file frames were deleted. Finally, we extracted three salient features from each frame: the module, the function and intra-module line number, and discarded the other information in the call-stack file.

Unlike the afl protocol–where crashes are de-duplicated based on a hashing scheme–we labeled two call-stacks to be deplicates whenever their text files were identical after the pre-processing described above. A striking result of frequency analysis is that there are dramatically fewer distinct crashes relative to total crashes (see **Figure 1**). Further, the frequency of distinct crashes is unevenly distributed. Across programs, the call-stack data displayed largely the same pattern: most of the weight was distributed among a few crashes, with the rest rarely occurring (See **Figure 2**).

3 TOPOLOGICAL MODEL

In this section, we propose a model to frame the complex dependency relationships between the terms $\mathcal{T}$ appearing across a set of call-stacks $\mathcal{C}$. Our model is inspired by the work of [5] on finite topological spaces, where pre-orders, equivalence classes and posets capture certain topological interactions between points. Our applications will use primarily the poset representation of the data, but we have included the topological perspective which motivated the original idea with the hope that future work may be able to further incorpate the topological characteristics of the model.

Recall that in any topological space X, the open neighborhood $\mathcal{N}(x)$ of a point $x \in X$ is the set of open sets containing x. A rough intuition of point-set topologies over finite sets is that elements

TABLE 1 | The number of crashes generated by fuzzing each program, and the number of unique crashes after whitelisting and removed exact duplicates. Within the set of call-stacks, some files were either blank or unable to be opened. We discarded such files, as appears in the second column from the right.

Binary	Extension	Version	Call stacks	Discarded (per 1,000)	Distinct
SoX	mp3	14.4.2	40,017	1 (0.02)	12
Librsvg	Svg	2.40.20	6,276	94 (15)	68
Libtiff	Tiff	4.0.9	5,486	2 (0.36)	9
Freetype	Ttf	2.5.3	17,034	1 (0.06)	51
SoX	Wav	14.4.2	30,856	1 (0.03)	11
Libxml2	Xml	2.9.0	240,821	7,467 (31)	3,175

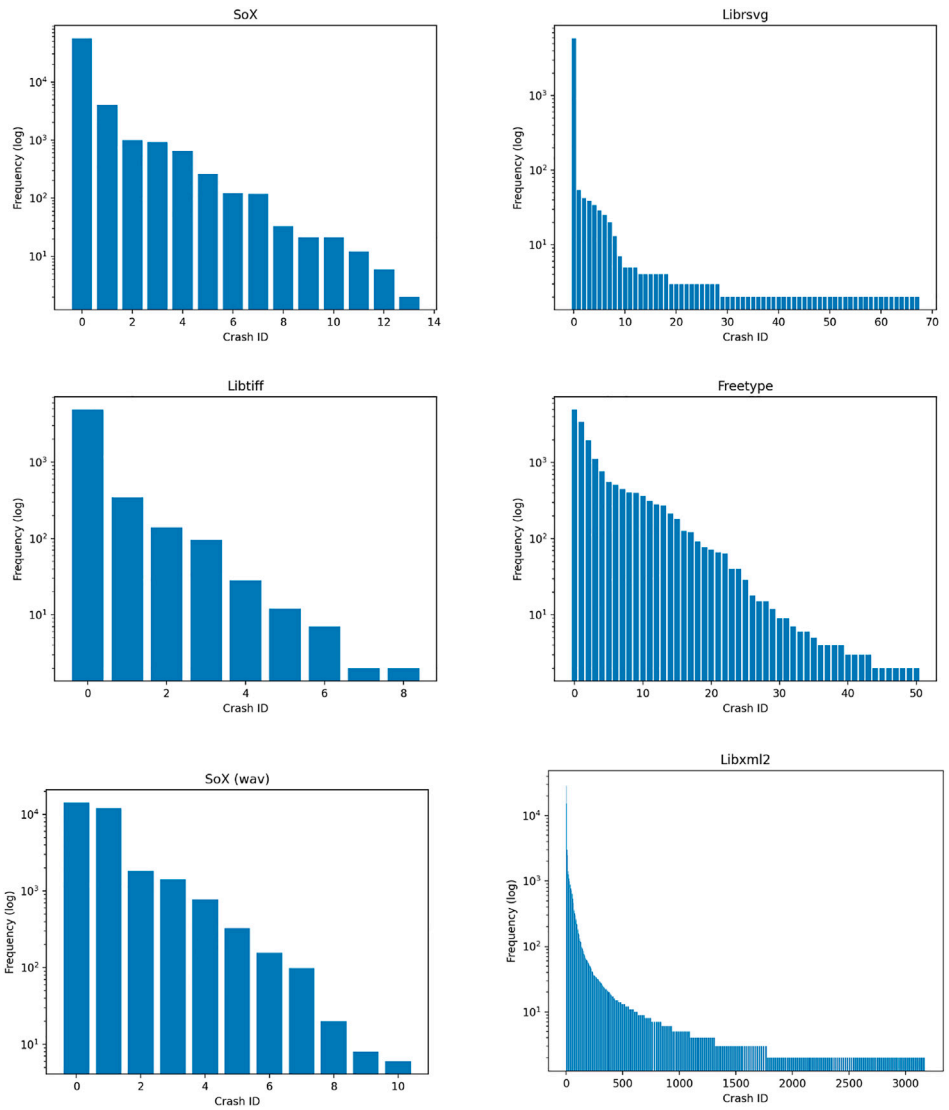


FIGURE 2 | Frequency (logarithmic scale) distribution of crashes in each binary, where the crash ID is with respect to distinct call-stacks.

are considered close when they have similar open neighbourhoods. Our goal is to create a topology over the set of terms $\mathcal{T}$ where those that occur in a similar set of call-stacks are close.

3.1 Call-Stack Topology

Given a set of call-stacks $\mathcal{C}$ comprised of terms $\mathcal{T}$, we define the call-stack topology $\mathcal{T}(\mathcal{C})$ on $\mathcal{T}$ to be that generated by treating each call-stack $c \in \mathcal{C}$ as an open set of terms. The complete

collection of open sets in $\mathcal{T}(\mathcal{C})$ is then formed by taking all possible intersections and unions of call-stacks from $\mathcal{C}$.

Unlike many topologies we are familiar with, e.g., topologies generated by open balls in a metric space, the call-stack topology is seldom Hausdorff. In our context, the looser criterion of a T_0 space is a more useful notion of point separation. Recall that a T_0 space $(X,\mathcal{N})$ is one where points may be distinguished by their open neighbourhoods; explicitly, for each pair of points $x, y \in X$ there exists either an open set in $\mathcal{N}$ containing x without y or y without x.

Since distinct terms can appear in the same subset of call-stacks, $\mathcal{T}(\mathcal{C})$ is not even T_0. However, we can transform $\mathcal{T}(\mathcal{C})$ into a T_0 space by taking the Kolmogorov quotient (see [7]). The Kolmogorov quotient $\tilde{X}$ is obtained from $(X,\mathcal{N})$ by the equivalence relation $x \sim y$ whenever they have the same open neighborhood $\mathcal{N}(x) = \mathcal{N}(y)$. It is known that $\tilde{X}$ and $(X,\mathcal{N})$ have the same homotopy type. By taking the Kolmogorov quotient of the call-stack topology, one reduces the object of study from a potentially large set of terms $\mathcal{T}$ into a more manageable set of equivalence classes of terms $\tilde{\mathcal{T}}$.

The following simple lemma shows that one may characterize equivalence classes in the Kolmogorov quotient of the call-stack topology by examining the set of call-stacks directly rather than the topology. For $t \in \mathcal{T}$, we refer to the set of call-stacks in $\mathcal{C}$ which contain t as the call-stack neighborhood, using the notation $\mathcal{C}(t)$.

Lemma 1. For a set of call-stacks $\mathcal{C}$ comprised of terms $\mathcal{T}$, two terms $t_1 \sim t_2$ are equivalent if and only if $\mathcal{C}(t_1) = \mathcal{C}(t_2)$.

Proof. The definition of the equivalence relation is $t_1 \sim t_2$ whenever $\mathcal{N}(x) = \mathcal{N}(y)$ in the call-stack topology. Hence, we need to show that open neighbourhoods $\mathcal{N}(x) = \mathcal{N}(y)$ are equal if and only if call-stack neighbourhoods $\mathcal{C}(t_1) = \mathcal{C}(t_2)$ are equal.

Suppose that $\mathcal{C}(t_1) \neq \mathcal{C}(t_2)$. Without loss of generality, suppose there exists $c \in \mathcal{C}$ such that $t_1 \in c$ and $t_2 \notin c$. By the definition of call-stack topology, c is open and hence a member of $\mathcal{N}(t_1)$. Since $t_2 \notin c$, it follows that $\mathcal{N}(t_1) \neq \mathcal{N}(t_2)$, proving one side of the statement.

Conversely, suppose that $\mathcal{C}(t_1) = \mathcal{C}(t_2)$, and further suppose that $U \in \mathcal{N}(t_1)$. All open sets in the topology generated by a set $\mathcal{C}$ may be expressed in the form

$$U = \bigcup_j \bigcap_i \mathcal{C}_{i,j} \tag{1}$$

where each $\mathcal{C}_{i,j} \in \mathcal{C}$ is a generating set. The assumption $U \in \mathcal{N}(t_1)$ implies that $t_1 \in U$ and further that there exists j such that $t_1 \in \mathcal{C}_{i,j}$ for all i. Since we have assumed that $\mathcal{C}(t_1) = \mathcal{C}(t_2)$, $t_1 \in \cap_i \mathcal{C}_{i,j}$ implies that $t_2 \in \cap_i \mathcal{C}_{i,j} \subseteq U$ as well. This implies that $\mathcal{N}(t_1) \subseteq \mathcal{N}(t_2)$. By the same argument $\mathcal{N}(t_2) \subseteq \mathcal{N}(t_1)$, thus $\mathcal{N}(t_1) = \mathcal{N}(t_2)$, and finishing the proof.

According to the above lemma, equivalence classes in the Kolmogorov quotient of the call-stack topology consist of terms that occur in the same set of call-stacks. The intuition is that by taking the Kolmogorov quotient, we only consider terms up to the information of which call-stacks they appear in. The composition of equivalence classes in such a quotient will be a key feature for analysis in our application.

In theory, calculating such equivalence classes requires knowledge of open neighbourhoods and, ergo, the entire gamut of open sets in the call-stack topology. Aside from providing useful intuition, the above lemma also ensures that we can avoid this computationally expensive task, attaining equivalence classes indirectly by comparing the call-stack neighbourhoods of pairs of terms.

Example 1. In **Figure 3**, we depict a set of three call-stacks. In the center of the Figure, the three circles each represent a generating set for the call-stack topology $\mathcal{T}(\mathcal{C})$ over the constituent terms $\mathcal{T}$ of $\mathcal{C}$. The coloring of the terms represents their partition into equivalence classes under the Kolmogorov quotient operation. Following Lemma 1, equivalence classes consist of terms sharing identical call-stack neighbourhoods. This example also highlights that both the ordering of terms in the call-stack and the frequency of each term within it are both disregarded by the model.

3.2 Call-Stack Partial Order

In this section we equip the set of call-stack terms with the additional structures of a pre-order and partial order. Our approach in later sections is to use this structure to examine relations between terms in different equivalence classes. For any topological space, one may use the structure of the open sets to define a pre-order over its points called the specialization pre-order. This may be defined in the following equivalent statements.

Definition 1. *For a topological space X, the* specialization pre-order $(X,<=)$ over X is given by either

$$x \leq y \text{ whenever } \mathcal{N}(y) \subseteq \mathcal{N}(x)$$

or equivalently

$$x \leq y \text{ whenever } y \in \bigcap_{U \in \mathcal{N}(x)} U$$

The specialization pre-order forms a partial order over X precisely when X is a T_0 space, with the T_0 condition ensuring that the order relation satisfies the anti-symmetry condition: $x \leq y$ and $y \leq x$ implies $x = y$.

Definition 2. The call-stack pre-order on a set of call-stacks $\mathcal{T} \leq (\mathcal{C})$ is the specialization pre-order over the call-stack topology $\mathcal{T}(\mathcal{C})$.

definition 3. The call-stack poset on a set of call-stacks $\tilde{\mathcal{T}} \leq (\mathcal{C})$ is the specialization pre-order over the Kolmogorov quotient $\tilde{T}(\mathcal{C})$ of the call-stack topology.

Unlike the call-stack topology $\mathcal{T}(\mathcal{C})$ in general, the Kolmogorov quotient $\tilde{T}(\mathcal{C})$ of the call-stack topology is guaranteed to be T_0 space (see [7] for a full survey of Kolmogorov quotients). Note here that the call-stack poset is defined over equivalence classes of terms within the call-stacks, rather than the individual terms themselves. In moving to this construction, we reduce the space of information we are working with; order theoretic notions are considered between blocks of terms rather than individual ones.

As is the case for equivalence classes, the call-stack partial order can be computed without explicitly calculating the open sets in the call-stack topology.

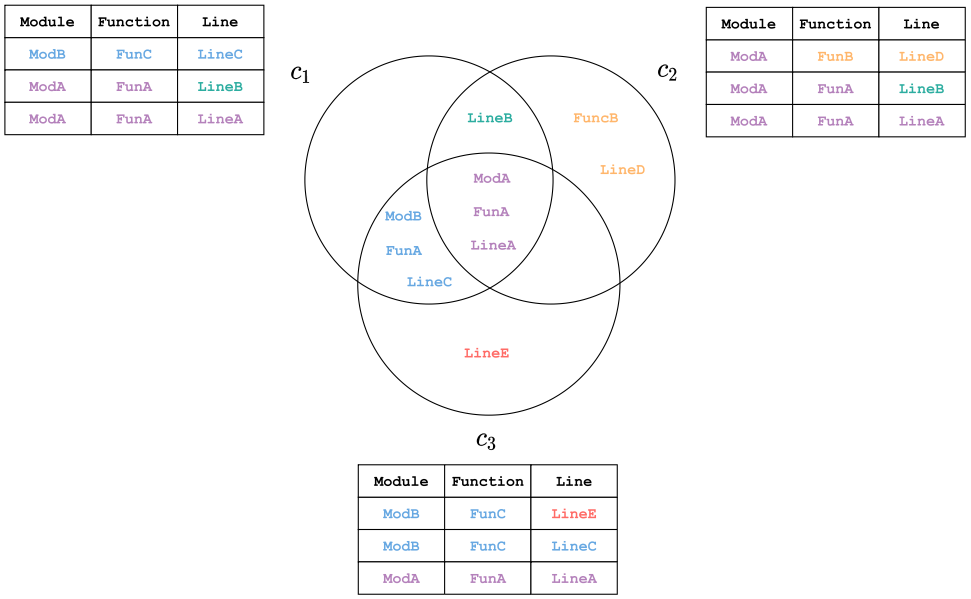


FIGURE 3 | Three call-stacks $\mathcal{C}$ and their corresponding generating sets over their constituent terms $\mathcal{T}$.

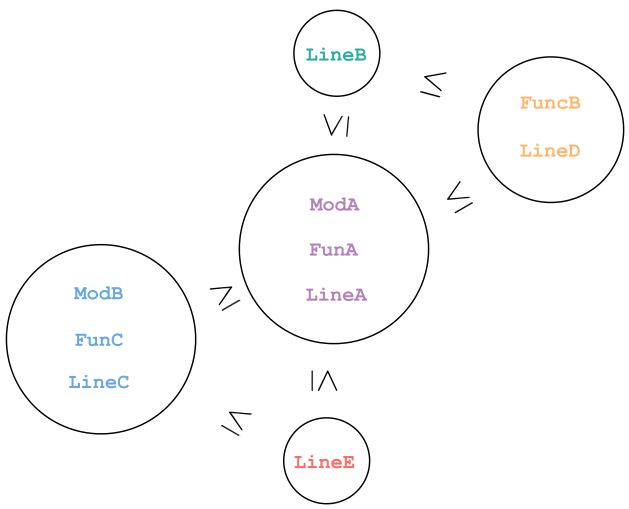


FIGURE 4 | The call-stack poset corresponding to the call-stack topology defined in Example 1.

LEMMA 2. *Two classes of terms* $[t_1], [t_2] \in \tilde{T}(\mathcal{C})$ *satisfy an order relation* $[t_1] \leq [t_2]$ *in the call-stack partial order if and only if* $\mathcal{C}(t_2) \subseteq \mathcal{C}(t_1)$.

PROOF. Suppose first that $\mathcal{C}(t_2) \subseteq \mathcal{C}(t_1)$. Let $U \in \mathcal{N}(t_2)$ be an open neighborhood of t_2 in the call-stack topology. As in Equation 1, U may be expressed in the form

$$U = \bigcup_j \bigcap_i \mathcal{C}_{i,j}$$

where there exists j such that $t_2 \in \cap_i \mathcal{C}_{i,j}$. Since $\mathcal{C}(t_2) \subseteq \mathcal{C}(t_1)$, it follows that $t_1 \in \cap_i \mathcal{C}_{i,j}$ also, implying that $U \in \mathcal{N}(t_1)$ and $\mathcal{N}(t_2) \subseteq \mathcal{N}(t_1)$. Thus, $[t_1] \leq [t_2]$ as required.

In the other direction, suppose that C_2 is not a subset of C_1. Then there exists $c \in \mathcal{C}(t_2)$ such that $c \notin \mathcal{C}(t_1)$. Since c is open in the call-stack topology, $c \in \mathcal{N}(t_2)$ and $c \notin \mathcal{N}(t_2)$, implying that $\mathcal{N}(t_2) \not\subseteq \mathrm{N}(t_1)$ and thus $[t_1] \leq [t_2]$.

The above lemma suggests how one should interpret the call-stack partial order: two sets of terms $[t_1], [t_2] \in [t_1], [t_2] \in \tilde{T}(\mathcal{C})$ satisfy an order relation $[t_1] \leq [t_2]$ when every call-stack containing the $[t_2]$ terms also contains the $[t_1]$ terms. In this sense, witnessing the terms in $[t_2]$ depends on witnessing the terms in $[t_2]$ across the call-stacks in $\mathcal{C}$.

Example 2. In **Figure 4**, we depict the call-stack poset corresponding to the example call-stack topology provided in Example 1. Each circle contains an equivalence class of terms that have identical call-stack neighbourhoods. Lemma 2 tells us that an order relation $[t_1] \leq [t_2]$ between classes occurs when $\mathcal{C}(t_2) \subseteq \mathcal{C}(t_1)$; namely, when all call-stacks containing t_2 also contain t_1.

3.3 Function Term Obfuscation

One of the key research questions is how much information can be extracted from the call-stack when terms are removed. Let $\mathcal{C}$ be a collection of call-stacks, tokenized into a set of terms $\mathcal{T}$. To consider the effect on the model of removing a single term $t \in \mathcal{T}$, let $\mathcal{T}' \triangleq \mathcal{T} \backslash \{t\}$ and

$$\mathcal{C}' \triangleq \{c \backslash (c \cap \{t\}) | c \in \mathcal{C}\}$$

be the set of call stacks with the term t removed. Note here that each $c \in \mathcal{C}$ is a set of terms $c \subseteq \mathcal{T}$, so the set notation $c \backslash (c \cap \{t\})$ makes sense. The following lemma describes the effect on the call-stack poset when t is removed.

Lemma 3. Suppose $t_1, t_2 \in \mathcal{T}$. Then

1. $[t_1] \neq [t_2]$ *in* $\tilde{\mathfrak{T}}(\mathfrak{C})$ if and only if $[t_1] \neq [t_2]$ *in* $\tilde{\mathcal{T}}'(\mathcal{C}')$.
2. $[t_1] \leq [t_2]$ *in* $\tilde{\mathcal{T}} \leq (\mathcal{C})$ if and only if $[t_1] \leq [t_2]$ *in* $\tilde{\mathcal{T}}'(\mathcal{C}')$.

Proof. For (1), $[t_1] \neq [t_2]$ in $\tilde{\mathfrak{T}}(\mathcal{C})$ if and only if there exists $c \in \mathcal{C}$ such that either $t_1 \in c$ and $t_2 \notin c$ or $t_1 \notin c$ and $t_2 \in c$. This occurs if and only if there exists $\mathcal{C}' \in \mathcal{C}$ such that either $t_1 \in \mathcal{C}'$ and $t_2 \notin \mathcal{C}'$ or $t_1 \notin \mathcal{C}'$ and $t_2 \in \mathcal{C}'$, which is equivalent to $[t_1] \neq [t_2]$ in $\tilde{\mathcal{T}}' \leq (\mathcal{C})$. For (2), $[t_1] \leq [t_2]$ in $\tilde{\mathcal{T}} \leq (\mathcal{C}) \Leftrightarrow \mathcal{C}(t_2) \subseteq \mathcal{C}(t_1) \Leftrightarrow \mathcal{C}'(t_2) \subseteq \mathcal{C}'(t_1) \Leftrightarrow [t_1] \leq [t_2]$ in $\tilde{\mathcal{T}}' \leq (\mathcal{C}')$.

One can summarize the above result as the fact that $\tilde{\mathcal{T}}'(\mathcal{C}')$ is isomorphic to $\tilde{\mathcal{T}}(\mathcal{C})$ whenever the singleton $\{t\}$ is not an equivalence class. When it is an equivalence class, it is the only difference between the two call-stack posets $\tilde{\mathcal{T}} \leq (\mathcal{C})$ and $\tilde{\mathcal{T}}' \leq (\mathcal{C}')$. By inductively removing all of the function terms, $\mathcal{F} \subset \mathcal{T}$, and applying the lemma at each step, we attain the following corollary.

Corollary 1. Let $\widetilde{\mathfrak{T}''}(\mathcal{C}'')$ be the call-stack poset over $\tilde{\mathcal{T}}'' \triangleq \mathcal{T} \backslash \mathcal{F}$ generated by

$$\mathcal{C}'' \triangleq \{c \backslash (c \cap \mathcal{F}) | c \in \mathcal{C}\}$$

Then $\tilde{\mathcal{T}}'' \leq (\mathcal{C}'')$ is the sub-poset of $\tilde{\mathfrak{T}}(\mathcal{C})$ spanned by equivalence classes

$$\left\{ [x] \in \tilde{\mathfrak{T}}(\mathcal{C}) \middle| [x] \not\subseteq \mathcal{F} \right\}$$

In other words, whenever we remove function terms from the model, the structure of the call-stack poset is unchanged away from classes comprised solely of function terms. When a function term t shares an equivalence class with non-function terms, these may be used to recover its structural dependency information even when t is removed. The point of the above theorems is to motivate the idea that many attributes of the call-stack poset are retained in the case where some terms are missing.

4 FUNCTION TERM RECONSTRUCTION

The goal of this paper is to reconstruct information about function terms from call-stacks in which they are obscured. In this section, we present a small-scale experiment on our linux data set using features extracted from the call-stack topology model.

Accordingly, we must first define what we mean by 'function information.' When the function names are missing, it is not possible to recover them explicitly from the call-stack data. The next best data, and what we choose to focus on in this paper, is to recover the set of positions within the call-stacks that share a common function name. This notion is captured in the following definition.

Definition 4. For a term $t \in \mathcal{T}$ within a set of call stacks $\mathcal{C}$, define the frame trace $\mathsf{FT}_{\mathcal{C}}(t)$ of t to be the set of pairs

$$\mathsf{FT}_{\mathcal{C}}(t) \triangleq \{(c, n) | t \in c[n]\}$$

where $c[n]$ is the nth frame of call stack c.

If t appears in multiple frames $c[n]$ and $c[n']$ within the same crash $c \in \mathcal{C}$, then both (c, n) and (c, n') are elements of $\mathsf{FT}_{\mathcal{C}}(t)$. For any pair of terms of the same type in a set of call stacks, their frame traces must be disjoint. It is impossible to guess the explicit names of obscured function terms. However, if one can recover the frame traces of every function, then one can generate call stacks that are equivalent up to re-naming function terms.

By performing logistic regression over features extracted from the call-stack model, we will show that a surprising number of function frame traces can be recovered without any explicit knowledge of function names. This is particularly striking given that the user also knows nothing about the internal structure of the program. Additionally, we provide an algorithm for generating fake function names based on the guessed frame traces, making sets of call stacks more human-readable in the setting where function names are missing.

4.1 Preliminary Analysis of Call-Stack Equivalence Classes and Poset Structure

To motivate the use of our novel tools in the task of recovering function frame traces, we first present a basic analysis of the data through the lens of the call-stack topology and poset. In particular, we study the characteristics of equivalence classes—their size and the types of terms of which they comprise—as well as the order relations and dependencies they exhibit on one another.

4.1.1 Basic Statistics

Recall that the equivalence classes in the call-stack topology consist of terms that occur in the same set of call stacks. **Table 2** shows the extent of reduction from the number of terms to their equivalence classes under the quotient operation.

Our primary interest is to understand the effect of obscuring function terms. Corollary 1 states that removing the function terms only alters the model's structure at equivalence classes consisting of function terms alone. Accordingly, we say a function term f is *retained* under the quotient operation when it is equivalent to a non-function term t. Notably, in the case $f \in \mathcal{F}$ is retained, there exists a term $t \in \mathcal{T} \backslash \mathcal{F}$ with call-stack set equivalent to f.

Table 2 shows that, on average, 86% of function terms are retained. Extensive term equivalences in the call-stack topology mean that a dramatic reduction in the available terms has little

TABLE 2 | Call-stack model and term statistics for each linux program.

	SoX (m)	Librsvg	Libtiff	Freetype	SoX (w)	Libxml	Mean
Modules	15	15	8	23	14	17	
Functions	36	92	17	48	34	151	
Line num	43	99	21	81	40	361	
Total terms	94	206	46	152	88	529	
Classes	33	113	14	66	27	343	
Reduction %	65%	45%	70%	56%	69%	35%	57%
Order relations	108	1,024	22	352	89	2,220	
F-loss	4	32	0	6	2	29	
F-retention %	89%	65%	100%	88%	94%	81%	86%

Frame	Module	Function	Line Num.
0	effects_i_dsp.c	lsx_make_lpf	367
1	effects_i_dsp.c	lsx_design_lpf	405
2	e.c	dft_stage_init	225
3	e.c	rate_init	443
4	rate.c	start	632
5	effects.c	sox_add_effect	157
6	sox.c	add_effect	708
7	sox.c	add_effects	1073
8	sox.c	process	1759

Frame	Module	Function	Line Num.
0	parser.c	xmlParseCDSect	9750
1	parser.c	xmlParseContent	9806
2	parser.c	xmlParseElement	9995
3	parser.c	xmlParseContent	9822
4	parser.c	xmlParseElement	9995
5	parser.c	xmlParseDocument	10665
6	parser.c	xmlDoRead	15062
7	parser.c	xmlReadFile	15122
8	xmllimit.c	parseAnPrintFile	2382

FIGURE 5 | Example call stacks from the Linux data exhibiting the behavior described by the two patterns. The left and right examples are taken from two separate libraries, where each of the seven colors represents an equivalence class in the call-stack topology; that is, a set of terms which have identical call-stack neighbourhoods.

effect on the call-stack poset structure. The main takeaway from this analysis is that function terms rarely occur in an equivalence class on their own.

4.1.2 Patterns Relating Line Numbers and Function Terms

When two terms are in the same equivalence class, they occur in the same set of call-stacks. However, our topological model encodes none of the information about which frame they occur in. Our toy example (Example 1) suggested that line numbers and functions in the same equivalence class tend to occupy similar frames in the call-stack. In a thorough examination of the data, we observed two patterns, demonstrating each with the example call stacks in **Figure 5**.

- Pattern 1: When multiple line numbers belong to an equivalence class, they are usually paired with functions in the same frame except for the line number in the bottom frame. The lowest line number appears to act as a switch point between blocks of terms, instigating a run of function calls that are either seen in only one call-stack or always together. In **Figure 5**, this occurs in the brown and green equivalence classes of the example on the left.
- Pattern 2: When a single line number is in an equivalence class with a function, it is likely paired with a function one frame above. In **Figure 5**, this occurs in the purple and orange boxes of the left call-stack, and the purple, orange and green boxes of the right.

It is important to state that neither pattern reflects an underlying mathematical truth. Rather, they seem to be a symptom of programming convention. Namely, as source code tends to decompose into many different simple functions nested within one another, runs of frames in the call-stack tend to cycle through distinct function names. Further, these patterns only apply in the case that a line number occurs in the same set of crashes as a function, in which case we assume that they describe how the frames of a function and line number are related.

4.2 Method

Our method for frame trace recovery is centered around leveraging structure of the call-stack topology and poset. To do so, we generate the call-stack equivalence classes $\widetilde{T''}(\mathcal{C})$ and poset $\widetilde{\mathfrak{T}''} \leq (\mathfrak{C})$ from the incomplete data $\mathcal{T}''$, i.e., the set of terms with function names omitted. The intuition of Corollary 1 — as well as the empirical observations of **Table 2** — suggest that such objects should be relatively similar to their counterparts $\tilde{\mathcal{T}}(\mathcal{C}), \tilde{\mathfrak{T}} \leq (\mathfrak{C})$ generated from the full data that we aim to partially reconstruct.

Once such objects are constructed, our approach consists of the following two steps.

1. Classifying Equivalence Classes: Within the incomplete data model, the terms of an equivalence class $[t] \in \widetilde{T''}(\mathcal{C})$ consist only of line numbers and module names. However, in the complete data model, there may exist function terms that are also in the corresponding class. The first step of

our method is to estimate the likelihood that an equivalence class $[t] \in \widetilde{T''}(\mathcal{C})$ in the incomplete data corresponds to an equivalence class containing a function term in the full data $\tilde{T}(\mathcal{C})$.

2. Pairing Frame Traces: The second step of our method is to apply our two observations above to obtain a heuristic for predicting frame trace locations of function terms. This is done by selecting line numbers within a given equivalence class whose frame traces are likely to be paired with a function term frame trace in the complete data. The frame traces of these line numbers serve as our set of predictions for function frame traces and enable us to partially reconstruct the data set.

4.2.1 Classifying Equivalence Classes Within Libxml

The first step in our method of frame trace recovery is learning to detect when an equivalence class contains a function term. Before tackling the task of classifying equivalence classes in the incomplete data, we restrict our focus to a small study of libxml, which offers the largest base of terms and equivalence classes from which to garner information. We outline our method here to examine the relationship between the structure of an equivalence class within the libxml call-stack poset and the types of terms that it contains.

Consider the following three binary classification problems over the equivalence classes $\tilde{\mathcal{T}}(\mathcal{C})$ in the call-stack topology.

1. Modules: each equivalence class $[t] \in \tilde{T}(\mathcal{C})$ is labeled 1 if there exists a module $m \in \mathcal{M}_{\cap}[t]$ and 0 otherwise.
2. Functions: each equivalence class $[t] \in \tilde{\mathcal{T}}(\mathcal{C})$ is labeled 1 if there exists a function $f \in \mathcal{F}_{\cap}[t]$ and 0 otherwise.
3. Line Numbers: each equivalence class $[t] \in \tilde{T}(\mathcal{C})$ is labeled 1 if there exists a line number $l \in \mathcal{L}_{\cap}[t]$ and 0 otherwise.

We address each of the above by performing a simple logistic regression based on four features in the call-stack model. For an equivalence class $[t] \in \tilde{T}(\mathcal{C})$, these are as follows.

1. The size $|\{t' \in [t]\}|$ of an equivalence class.
2. The frequency (number of call-stacks) of the class $|\{\mathcal{C} \in \mathcal{C} | [t] \subseteq \mathcal{C}\}|$.
3. The weighted in-degree

$$\sum_{[t']} \phi([t'] \leq [t])$$

of the class within the order graph of the call-graph poset $\tilde{\mathfrak{C}} \leq (\mathfrak{T})$.

4. The *weighted out-degree*

$$\sum_{[t']} \phi([t] \leq [t'])$$

of the class within the order graph of the call-graph poset $\tilde{\mathfrak{C}} \leq ([\mathfrak{T}])$.

The names given to features 3 and 4 reference the fact that a poset p can be represented as a graph whose nodes are elements of p and edges are order relations $p \leq q$. The in- and out-degree of $[t]$ are the number of equivalence classes that $[t]$ depends on and that depend on $[t]$ respectively. To incorporate the magnitude of such dependencies, the weight of an order relation is determined by the function

$$\phi([t] \leq [t']) = \frac{|\{\mathcal{C} \in \mathcal{C} | [t'] \subseteq \mathcal{C}\}|}{|\{\mathcal{C} \in \mathcal{C} | [t] \subseteq \mathcal{C}\}|}$$

Lastly, for normalization each of the four variables is scaled by minimum and maximum to lie within $[0, 1]$, making the logistic regression weights comparable across variables.

Since the classification labels are unbalanced, the classes were re-weighted according to the to sci-kit learn class re-weighting scheme. To prevent over-fitting, the data was randomly split into an 80% training set and 20% testing set. To measure results, we use the F1 score and Area Under (precision-recall) Curve, which is suggested to be the most sensible measurements when predicting heavily weighted classes in binary classification (See [8]).

As a baseline to compare the statistical significance of our method, we propose the following binary classification null-model. Firstly, we empirically derive three probabilities from the ratio of the number of equivalence classes containing each term over the total number of equivalence classes. For each type of term, the null-model randomly guesses whether each class contains that particular term type with the empirically derived probability

4.2.2 Classifying Incomplete-Data Equivalence Classes

Once we have attained logistic regression weights for the libxml data, we then apply them to other programs. An important point of this stage is that, unlike the libxml program experiment, we withhold the full-data with function names as a validation set. This means that the call-stack topology and poset are generated for each program from the call-stacks $\mathcal{C}''$ with function terms obscured $\mathcal{T}''$.

From each of these objects, we extract the same four features as above, normalizing in the same way to ensure that the learned libxml weights scale appropriately. The goal of this stage is to predict whether an equivalence class in the incomplete data is likely to contain a function in the full-data, thus predicting a set of call-stacks which share a common missing function term.

4.2.3 Pairing Line Numbers With Function Frame Traces

The outline of our approach to predicting function frame traces is to 1) guess when a line number in the incomplete-data model was likely to have been in an equivalence class with a function name and 2) generating predicted frame traces for functions from line number frame traces using our two heuristics. Algorithm LABEL: predict_FTs ties these two steps together, taking in the set of obscured call-stacks $\mathcal{T}''(\mathcal{C}'')$ and their frame traces $\mathsf{FT}_{\mathcal{C}''}$ then returning a set of predicted frame traces. The value p is a cut-off

TABLE 3 | Proportion of equivalence classes containing each term type, and logistic regression F1 score and AUC improvements on the null-model for the libxml call-stacks.

	Module	Function	Line number
% Equivalence Classes	4.76%	65.01%	88.06%
F1	0.40	0.89	0.92
Null model	0.00	0.40	0.90
AUC	0.20	0.94	0.97
Null model	0.04	0.43	0.93

likelihood for using logistic regression weights to decide when a pattern should be applied to predict a frame trace.
Algorithm 1PredictFTs$(\tilde{\mathcal{T}}''(\mathcal{C}''), \mathsf{FT}_{\mathcal{C}''}, p)$.

$\mathcal{CT}$
  predicted_fts = []
  $[t] \in \tilde{\mathcal{T}}''(\mathcal{C}'')$
    $\mathbb{P}(\exists f \in [t]) \geq p$
      $\triangleright$ $|[t] \cap \mathcal{L}| > 1$
        $\triangleright$ lines = $([t] \cap \mathcal{L})$.drop_bottom_linenum()
        $l \in$ lines
       predicted_fts = predicted_fts + $[\mathsf{FT}_{\mathcal{C}''}(l)$ for $l \in$ lines$]$
      $|[t] \cap \mathcal{L}| = 1$
        $\triangleright$ $l = [t] \cap \mathcal{L}$
     predicted_fts.append$((c, min(n-1, 0)) | (c, n) \in \mathsf{FT}_{\mathcal{C}''}(l))$
predicted_fts

In our algorithm, the logistic regression weights in Line 3 learned over libxml serve as the basis for detecting whether there exists a function in each line number's equivalence class. Using only the libxml weights ensures that when we predict function frame traces, we require no information about function names in other programs beforehand.

The logistic regression is learned over the full-term model of libxml then performed over the call-stack topology models generated without terms in other programs. There are two significant obstacles that the model must overcome to be successful. Firstly, the model must exhibit *transference* if the regression weights from libxml are to work for other programs. Secondly, the model must be *robust* to term removal given that it classifies over a model without function terms. On the second point, Corollary 1 states that the call-stack model retains much of its structure when function terms are removed, which suggests that the logistic regression weights have a chance of still being applicable.

The role of the logistic regression model is primarily to act as a gate-keeper, probabilistically determining when a given line number is *not* in the same class as any function. This prevents the model from over-predicting instances where a function's frame trace should be paired to that of a line number. The method drop bottom lineum in Line 5 removes the line number with the lowest average frame trace from the set, which is necessary to apply pattern 1.

4.3 Results

4.3.1 Learning Libxml Logistic Regression Weights

In **Table 3** we present the results of our binary classification experiment within the libxml data described in Subsection 4.2.1. The results show that the inclusion of call-stack topology features significantly improves the quality of prediction across terms when compared with the null model. To quantify the effect of each feature in classification, we plot the logistic regression weights in **Figure 6**. In all cases, the frequency of a term has little effect on classification. For individual term types, there are several observations about the model variables that detect its presence in an equivalence class.

- Modules are likely to be in smaller equivalence classes with lower weighted in-degree and higher weighted out-degree. This means more terms depend on them than they depend on.
- Functions are likely to be in large equivalence classes, with high out-degree. This means many terms are likely to depend on them.
- Line Numbers are likely to be in larger equivalence classes, with a low weighted out-degree. This means terms are unlikely to depend on line numbers.

Each observation agrees with the structure of library dependencies, where line numbers depend on functions, and functions depend on modules. The logistic regression model is notably adept at detecting the presence of function terms within a given equivalence class.

4.3.2 Frame Trace Recovery

The PREDICTFTS algorithm is run over each Linux program. Since the function term information in libxml was used to generate the logistic regression weights, we exclude it from the analysis. To measure the results, we compare the set of predicted function frame traces generated by the algorithm against the set of actual function frame traces in each set of call stacks.

Table 4 contains the results of each experiment with three different cut-off probabilities 0.4, 0.5 and 0.6. Despite the heavy reliance on fairly naïve heuristics, our model has a reasonable mean precision of above 0.75 is each case. Notably, both precision and recall of frame traces are relatively stable across each program. This suggests that the libxml logistic regression weights and the heuristics both exhibit some degree of transference across programs.

In **Figure 7**, we analyze the effect of the cut-off probability parameter in detail. When this parameter is high, the algorithm requires a large degree of confidence that an equivalence class contains a function term before predicting a frame trace. This is reflected in an increasing precision and decreasing recall as the cut-off probability increases.

The cut-off probability parameter indirectly allows the user to dictate the importance of precision at the expense of recall. Given that the purpose of our experiment is to reliably reconstruct what function names we can, the importance of precision outweighs that of recall. Indeed, there exist function names in the data that could not possibly be recalled from the module and line number information alone. For example, large swathes of function names are hidden behind the repeated line number 0 in the librsvg data (**Figure 8**), rendering their recall impossible by our method.

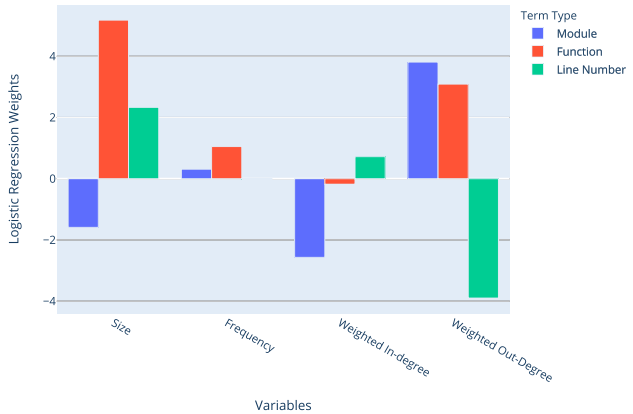


FIGURE 6 | Logistic regression weights for libxml fitted to each term type.

TABLE 4 | Precision and recall of PREDICTFTS algorithm at cut-off probabilities 0.4, 0.5 and 0.6.

	SoX (m)	Librsvg	Libtiff	Freetype	SoX (w)	Mean	Cut-off probability
Precision	0.71	0.57	1.0	0.78	0.67	0.75	
Recall	0.47	0.29	0.71	0.6	0.47	0.50	0.4
Precision	0.71	0.71	1.0	0.76	0.66	0.77	
Recall	0.47	0.22	0.71	0.52	0.47	0.48	0.5
Precision	0.73	0.89	1.0	0.86	0.74	0.84	
Recall	0.44	0.18	0.53	0.52	0.41	0.42	0.6

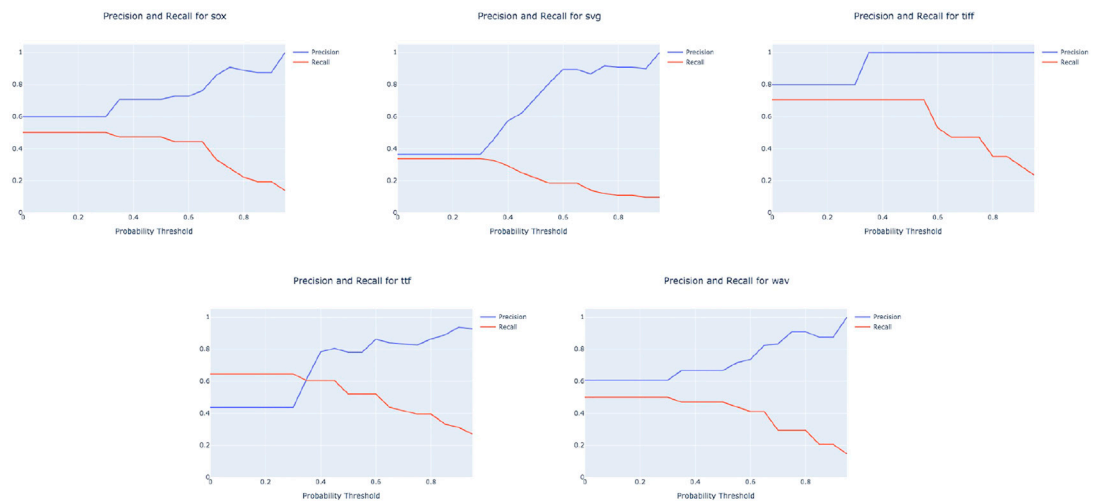


FIGURE 7 | Precision and recall for the PredictFTs algorithm on each program indexed by the cut-off probability parameter.

Example 3. To make call stacks without function terms more readable we insert random words into each predicted frame trace. Keeping with the afl theme, we sample random words from the surnames of champion players from the Richmond Tigers Australian Football League (AFL) team. These words are consistent across the set of call stacks, making it easier for the user to visually compare different call stacks.

Figure 9 demonstrates two reconstructed call stacks from the SoX program. In the original call-stacks, the coloring represents equivalence classes in the model. We color the reconstructed call-stacks the same, noting that when we attempt to reconstruct we do not know what equivalence classes will contain functions a priori.

As is evident, words representing functions are consistent across the set of call-stacks when frame traces are correctly predicted. The ???? terms in the reconstructed call-stacks represent function frame traces that the algorithm did not attempt to predict.

Frame	Module	Function	Line Num.
0	libc.so.6	raise	0
1	libc.so.6	abort	0
2	libglib-2.0.so.0	??	0
3	libglib-2.0.so.0	??	0
4	libglib-2.0.so.0	g_private_get	0
5	libglib-2.0.so.0	g_logv	0
6	libglib-2.0.so.0	g_log	0
7	libglib-2.0.so.0	g_malloc	0
8	libglib-2.0.so.0	g_strdup	0
...	...	...	...

FIGURE 8 | The top eight frames from an example librsvg call-stack.

Original Crashes

Frame	Module	Function	Line Num.
0	fft4g.c	bitrv2	721
1	fft4g.c	makewt	681
2	fft4g.c	____	350
3	effects_i_dsp.c	lsx_safe_rdft	219
4	e.c	dft_stage_init	239
5	e.c	rate_init	443
6	rate.c	start	632
7	effects.c	sox_add_effect	157
8	sox.c	add_effect	708
9	sox.c	add_effects	1073
10	sox.c	process	1759

Frame	Module	Function	Line Num.
0	fft4g.c	bitrv2	721
1	fft4g.c	lsx_rdft	681
2	effects_i_dsp.c	lsx_safe_rdft	219
3	e.c	dft_stage_init	239
4	e.c	rate_init	443
5	rate.c	start	632
6	effects.c	sox_add_effect	157
7	sox.c	add_effect	708
8	sox.c	add_effects	1073
9	sox.c	process	1759

Reconstructed Crashes

Frame	Module	Function Guesses	Line Num.
0	fft4g.c	COTCHIN	721
1	fft4g.c	MARTIN	681
2	fft4g.c	DELEDIO	350
3	effects_i_dsp.c	????	219
4	e.c	RIEWOLDT	239
5	e.c	RANCE	443
6	rate.c	????	632
7	effects.c	RICHARDSON	157
8	sox.c	HOULI	708
9	sox.c	EDWARDS	1073
8	sox.c	????	1759

Frame	Module	Function Guesses	Line Num.
0	fft4g.c	COTCHIN	721
1	fft4g.c	DELEDIO	681
2	effects_i_dsp.c	????	219
3	e.c	RIEWOLDT	239
4	e.c	RANCE	443
5	rate.c	????	632
6	effects.c	RICHARDSON	157
7	sox.c	HOULI	708
8	sox.c	EDWARDS	1073
9	sox.c	????	1759

FIGURE 9 | Two call stacks from the SoX data set, along with their reconstructions using the PredictFTs algorithm.

In **Figure 10**, we explain which fake function names were paired with which line numbers. We describe how each fake function name pairs with an original in the case that it was a correct prediction, as well as which of the two heuristics were used to pair it with the frame trace of a given line number. The only incorrect prediction was the fake function name DELEDIO, which erroneously predicted that the line number 219 was paired with a function in the original call-stack set via heuristic 2.

	Original Function	Function Guesses	Paired Line. Num	Heuristic Applied
Correct	bitrv2	COTCHIN	721	2
Correct	makewt	MARTIN	681	1
Incorrect	N/A	DELEDIO	219	2
Correct	dft_stage_init	RIEWOLDT	443	2
Correct	rate_init	RANCE	632	2
Correct	sox_add_effect	RICHARDSON	157	1
Correct	add_effect	HOULI	708	1
Correct	add_effects	EDWARDS	1073	1

FIGURE 10 | The list of functions whose frame trace was recovered, and the fake function names and line numbers that were paired.

5 RELATED WORK

There is a large collection of research centered on crash triage, in particular in crash de-duplication. The most common tasks are either 1) to automatically de-duplicate full bug-reports submitted to open-source software or 2) to bucket crashes by dissimilarity. In contrast to the setting considered in this paper, most research concerns full bug reports where call-stacks are only a subset of the entire information. In particular, the language in user-reported comments is incorporated, and in some cases the central object [9].

Some of the highest rates of an expert-validated crash duplicate recall are attributed when program execution traces are also recorded [10]. Including call stacks in bug report data has been shown to increase de-duplication recall of full bug reports significantly [11] validating that they are an important object of study in crash triage. We refrained from using the common methods outlined in the above research, showing that a reasonable whitelisting and de-duplication was enough to significantly reduce the number of call-stacks.

Other models of call-stacks exist, albeit with slightly different machinery. For example, the crash-graph defined in [12] serves as a way to graphically compare the similarity between call-stacks. The use of such a model for function frame trace recovery could be an avenue for future research.

6 LIMITATIONS

One of the main limitations of our work is that it is performed on a relatively small data-set. Indeed there are less than 100 distinct call-stacks in each Linux program we have tested, barring the libxml data used to generate the logisitic regression weights. When the set of call-stacks is not very diverse, there may be a tendency to only see function terms with a single line number, making them easier to recover using our method.

A second limitation of our model is that it relies on two fairly naïve patterns. It is not clear if 1) such patterns yield similar results on larger data-sets or 2) whether such patterns could be improved upon or replaced with a more scientific approach. At present, the heuristic method means that our model can only predict function terms that are consistently associated with a single line number across the call-stack set. Our hope is that more sophisticated pattern recognition techniques applied to our topological model could accommodate cases such as frames that pair a particular function with various line numbers. In particular, since the call-stack poset can be thought of as a graph, we expect that more sophisticated techniques from the graph-learning literature could be leveraged 1) in lieu of our logistic regression model and 2) to derive better heuristics and push recall beyond 40 – 50% while preserving precision.

7 CONCLUSION

In summary, our main contribution has been to present a novel topological model to address the problem of function term reconstruction in call-stack data. We performed a small-scale experiment, providing an algorithm to predict the frame-traces of function terms which have been obscured in the call-stack data. Despite the limitations, the performance of the model is relatively encouraging, showing that more information about obscured function terms can be recovered than one may initially suspect. In the future, we envision further research could be done within this framework to improve the recall of the PREDICTFTS algorithm.

We also showed that there is a fundamental lack of diversity in our call-stack data, and we hypothesize that the brute-force nature of fuzzing means that this will probably occur in most data-sets generated by a fuzzer. It is an open question whether our method will work on larger, more diverse call-stack data-sets. Given that some level of dependence between terms is required to form equivalence classes, there is no guarantee that similar results will be achieved.

Lastly, the topological model used here is an example of a larger framework defined in [13]. The extended model is used to tackle applications in gray-box fuzzing, with the goal being to help guide fuzzing campaigns to generate more diverse call-stack data. The use of these models of dependency relations may be applicable in broader contexts outside of fuzzing, such as analyzing dependencies between genes in medical data.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/**Supplementary Material**, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

VR conceived the project and helped source the data. KM conducted the research and wrote the manuscript with assistance and guidance from VR. Both authors discussed the results and analysis at length.

FUNDING

KM received funding from the Australian Commonwealth Department of Defense under the project title "Mathematical methods for analysis and classification of call-stack data sets". VR was supported by ARC Future Fellowship FT140100604 in the early stages of the project.

ACKNOWLEDGMENTS

The authors are indebted to W.P. Malcolm for suggesting that topological data analysis might be usefully applied to study call-stacks, for assistance in sourcing the data, and many fascinating conversations about probability theory. The paper would not exist without the data provided by Adrian Herrera. The authors gratefully acknowledge many helpful discussions with AH about the art of fuzzing, software compilers, and call-stack jargon.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/fams.2021.668082/full#supplementary-material

REFERENCES

1. Gunadi H, and Herrera A. *Experiment Data for MoonLight: Effective Fuzzing with Near-Optimal Corpus Distillation* (2019). doi:10.1109/icicos48119.2019.8982513
2. Zalewski M. *Technical Whitepaper for AFL-Fuzz* (2014). doi:10.3726/978-3-653-03549-0 https://lcamtuf.coredump.cx/afl/technical_details.txt.
3. Bartz K, Stokes J, and Platt J. *Finding Similar Failures Using Callstack Similarity*. Redmond WA: Microsoft Corporation (2009).
4. Modani N, Gupta R, Lohman G, Syeda-Mahmood T, and Mignet L. Automatically Identifying Known Software Problems. In: Proceedings - International Conference on Data Engineering (2007). Istanbul, Turkey. doi:10.1109/ICDEW.2007.4401026
5. McCord MC. Singular Homology Groups and Homotopy Groups of Finite Topological Spaces. *Duke Math J* (1966). 33:465–74. doi:10.1215/S0012-7094-66-03352-7
6. Serebryany K, Bruening D, Potapenko A, and Vyukov D. AddressSanitizer: A Fast Address Sanity Checker. In: Proceedings of the 2012 USENIX Conference on Annual Technical Conference. Boston, MA: USENIX Association, USENIX ATC'12 (2012). p. 28.
7. Pirttimäki T. *A Survey of Kolmogorov Quotients* (2019). https://arxiv.org/abs/1905.01157.
8. He H. *Imbalanced Learning*. John Wiley & Sons (2011). p. 44–107. doi:10.1002/9781118025604.ch3
9. Runeson P, Alexandersson M, and Nyholm O. Detection of Duplicate Defect Reports Using Natural Language Processing. In: Proceedings - International Conference on Software Engineering (2007). Barcelona, Spain. doi:10.1109/ICSE.2007.32
10. Xiaoyin W, Lu Z, Tao X, Anvik J, and Sun J. An Approach to Detecting Duplicate Bug Reports Using Natural Language and Execution Information. In: Proceedings - International Conference on Software Engineering (2008). Germany. doi:10.1145/1368088.1368151
11. Lerch J, and Mezini M. Finding Duplicates of Your yet Unwritten Bug Report. In: Proceedings of the European Conference on Software Maintenance and Reengineering. Geneva, Italy: CSMR (2013). doi:10.1109/CSMR.2013.17
12. Kim S, Zimmermann T, and Nagappan N. Crash Graphs: An Aggregated View of Multiple Crashes to Improve Crash Triage. In: 2011 IEEE/IFIP 41st International Conference on Dependable Systems Networks. Hong Kong: DSN (2011). p. 486–93. doi:10.1109/DSN.2011.5958261
13. Maggs K. *A Topological Model For Applications In Fuzzing*. Canberra: Mathematical Science Institute, ANU College of Science, The Australian National University (2021). Master's thesis.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Big Data

ORIGINAL RESEARCH
published: 16 June 2021
doi: 10.3389/fdata.2021.680535

Deep Graph Mapper: Seeing Graphs Through the Neural Lens

Cristian Bodnar, Cătălina Cangea* and Pietro Liò*

Department of Computer Science and Technology, University of Cambridge, Cambridge, United Kingdom

Graph summarization has received much attention lately, with various works tackling the challenge of defining pooling operators on data regions with arbitrary structures. These contrast the grid-like ones encountered in image inputs, where techniques such as max-pooling have been enough to show empirical success. In this work, we merge the Mapper algorithm with the expressive power of graph neural networks to produce topologically grounded graph summaries. We demonstrate the suitability of Mapper as a topological framework for graph pooling by proving that Mapper is a generalization of pooling methods based on soft cluster assignments. Building upon this, we show how easy it is to design novel pooling algorithms that obtain competitive results with other state-of-the-art methods. Additionally, we use our method to produce GNN-aided visualisations of attributed complex networks.

Keywords: mapper, graph neural networks, pooling, graph summarization, graph classification

OPEN ACCESS

Edited by:
Umberto Lupo,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Mustafa Hajij,
Santa Clara University, United States
Stanislav Schmidt,
Ecole polytechnique fédérale de Lausanne (EPFL), Switzerland

****Correspondence:***
Cristian Bodnar
cb2015@cam.ac.uk
Cătălina Cangea
ccc53@cam.ac.uk

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Big Data

Received: *14 March 2021*
Accepted: *24 May 2021*
Published: *16 June 2021*

Citation:
Bodnar C, Cangea C and Liò P (2021) Deep Graph Mapper: Seeing Graphs Through the Neural Lens. Front. Big Data 4:680535. doi: 10.3389/fdata.2021.680535

1 INTRODUCTION

The abundance of relational information in the real world and the success of deep learning techniques have brought renowned interest in learning from graph-structured data. Efforts in this direction have been primarily focused on replicating the hierarchy of convolutional filters and pooling operators, which have achieved previous success in computer vision Sperduti. (1994); Goller and Kuchler. (1996); Gori et al. (2005); Scarselli et al. (2009); Bruna et al. (2014); Li et al. (2015), within relational data domains. In contrast to image processing applications, where the signal is defined on a grid-like structure, designing graph coarsening (pooling) operators is a much more difficult problem, due to the arbitrary structure typically present in graphs.

In this work, we introduce Structural Deep Graph Mapper (SDGM)[1]—an adaptation of Mapper (Singh et al., 2007), an algorithm from the field of Topological Data Analysis (TDA) (Chazal and Michel, 2017), to graph domains. First, we prove that SDGM is a generalization of pooling methods based on soft cluster assignments, which include state-of-the-art algorithms like minCUT (Bianchi et al., 2019) and DiffPool (Ying et al., 2018). Building upon this topological perspective of graph pooling, we propose two pooling algorithms leveraging fully differentiable and fixed PageRank-based "lens" functions, respectively. We demonstrate that these operators achieve results competitive with other state-of-the-art pooling methods on graph classification benchmarks. Furthermore, we show how our method offers a means to flexibly visualize graphs and the complex data living on them through a GNN "lens" function.

[1]Code to reproduce models and experimental results is available at https://github.com/crisbodnar/dgm.

2 RELATED WORK

In this section, we investigate the existing work in the two broad areas that our method is part of—graph pooling (also deemed hierarchical representation learning) and network visualisations.

2.1 Graph Pooling

Algorithms have already been considerably explored within GNN frameworks for graph classification. Luzhnica et al. (2019) propose a topological approach to pooling, which coarsens the graph by aggregating its maximal cliques into new clusters. However, cliques are local topological features, whereas our methods leverage a global perspective of the graph during pooling. Two paradigms distinguish themselves among learnable pooling layers: Top-k pooling based on a learnable ranking (Gao and Ji, 2019), and learning the cluster assignment (Ying et al., 2018) with additional entropy and link prediction losses for more stable training (DiffPool). Following these two trends, several variants and incremental improvements have been proposed. The Top-k approach is explored in conjunction with jumping-knowledge networks (Cangea et al., 2018), attention (Huang et al., 2019; Lee et al., 2019) and self-attention for cluster assignment (Ranjan et al., 2019). Similarly to DiffPool, the method suggested by Bianchi et al. (2019) uses several loss terms to enforce clusters with strongly connected nodes, similar sizes and orthogonal assignments. A different approach is also proposed by Ma et al. (2019), who leverage spectral clustering.

2.2 Graph Visualization

Graph visualization is a vast topic in network science. We therefore refer the reader to existing surveys, for a complete view of the field (Nobre et al., 2019; von Landesberger et al., 2011; Beck et al., 2017), and focus here only on methods that, similarly to ours, produce node-link-based visual summaries through the aggregation of static graphs. Previous methods rely on grouping nodes into a set of predefined motifs (Dunne and Shneiderman, 2013), modules (Dwyer et al., 2013) or clusters with basic topological properties (Batagelj et al., 2010). Recent approaches have considered attribute-driven aggregation schemes for multivariate networks. For instance, PivotGraph (Wattenberg, 2006) groups the nodes based on categorical attributes, while van den Elzen and van Wijk. (2014) propose a more sophisticated method using a combination of manually specified groupings and attribute queries. However, these mechanisms are severely constrained by the simple types of node groupings allowed and the limited integration between graph topology and attributes. Closest to our work, Mapper-based summaries for graphs have recently been considered by Hajij et al. (2018). Despite the advantages provided by Mapper, their approach relies on hand-crafted graph-theoretic "lenses," such as the average geodesic distance, graph density functions or eigenvectors of the graph Laplacian. Not only are these functions unable to fully adapt to the graph of interest, but they are also computationally inefficient and do not take into account the attributes of the graph.

3 BACKGROUND AND FORMAL PROBLEM STATEMENT

3.1 Formal Problem Statement

Consider a dataset whose samples are formed by a graph $G_i = (V_i, E_i)$, A d-dimensional signal defined over the nodes of the graph $h_i : V \to \mathbb{R}^d$ and a label y_i associated with the graph, where $i \in I$, a finite indexing set for the dataset samples. We are interested in the setting where graph neural networks are used to classify such graphs using a sequence of (graph) convolutions and pooling operators. While convolutional operators act like filters of the graph signal, pooling operators coarsen the graph and reduce its spatial resolution. Unlike image processing tasks, where the inputs exhibit a regular grid structure, graph domains pose challenges for pooling. In this work, we design topologically inspired pooling operators based on Mapper. As an additional contribution, we also investigate graph pooling as a tool for the visualization of attributed graphs.

We briefly review the Mapper (Singh et al., 2007) algorithm, with a focus on graph domains (Hajij et al., 2018). We first introduce the required mathematical background.

Definition 3.1: Let X, Z be two topological spaces, $f : X \to Z$, a continuous function, and $\mathcal{U} = (U_i)_{i \in I}$ a cover of Z. Then, the pull back cover $f^{-1}(\mathcal{U})$ of X induced by $(f, \mathcal{U})$ is the collection of open sets $f^{-1}(U_i), i \in I$, for some indexing set I. For each $f^{-1}(U_i)$, let $\{C_{i,j}\}_{j \in J_i}$ be a partition of $f^{-1}(U_i)$ indexed by J_i. We refer to the elements of these partitions as clusters. The resulting collection of clusters forms another cover of X called the refined pull back cover $\mathcal{R}(f^{-1}(\mathcal{U})) = \{C_{i,j}\}_{i \in I, j \in J_i}$.

Definition 3.2: Let X be a topological space with an open cover $\mathcal{U} = (U_i)_{i \in I}$. The 1-skeleton of the nerve $\mathcal{N}(\mathcal{U})$ of $\mathcal{U}$, which we denote by $\mathrm{sk}_1(\mathcal{N}(\mathcal{U}))$, is the graph with vertices given by $(v_i)_{i \in I}$, where two vertices v_i, v_j are connected if and only if $U_i \cap U_j \neq \emptyset$.

3.2 Mapper

Given a topological space X, a carefully chosen lens function $f : X \to Z$ and a cover $\mathcal{U}$ of Z, Mapper produces a graph representation of the topological space by computing the 1-skeleton of the nerve of the refined pull back cover $\mathrm{sk}_1(\mathcal{N}(\mathcal{R}(f^{-1}(\mathcal{U}))))$, which we denote by $\mathcal{M}(f, \mathcal{U})$. We note that, more generally, the skeleton operator might be omitted, in which case the output of the algorithm becomes a simplicial complex. However, for the purpose of this work, we are only interested in graph outputs. Typically, the input to the Mapper algorithm is a point cloud and the connected components are inferred using a statistical clustering algorithm, with the help of a metric defined in the space where the points live.

Mapper for Graphs. More recently, Hajij et al. (2018) considered the case when the input topological space $X = G(V, E)$ is a also a graph with vertices V and edge set E. In a typical point cloud setting, the relationships between points are statistically inferred; in a graph setting, the underlying relationships are given by the edges of the graph. The adaptation of Mapper for graphs proposed by Hajij et al. (2018) uses a lens function $f : V \to \mathbb{R}$ based on graph-theoretic functions and a cover $\mathcal{U}$ formed of open intervals of the real line. Additionally, the connected

components $\{C_{i,j}\}_{j \in J_i}$ are given by the vertices of the connected components of the subgraph induced by $f^{-1}(U_i)$.

However, the graph version of Mapper described above has two main limitations. Firstly, the graph-theoretic functions considered for f are rather limited, not taking into account the signals which are typically defined on the graph in signal processing tasks, such as graph classification. Secondly, by using a pull back cover only over the graph vertices, as opposed to a cover of the entire graph, the method relies exclusively on the lens function to capture the structure of the graph and the edge-connections between the clusters. This may end up discarding valuable structural information, as we later show in **Section 7.7**.

4 STRUCTURAL DEEP GRAPH MAPPER

Structural Graph Mapper. One of the disadvantages of the graph version of Mapper (described in the background section) is that its output does not explicitly capture the connections between the resulting collections of clusters. This is primarily because the lens function f is defined only over the set of vertices V and, consequently, the resulting pull-back cover only covers V. In contrast, one should aim to obtain a cover for the graph G, which automatically includes the edges. While this could be resolved by considering a lens function over the geometric realization of the graph, handling only a finite set of vertices is computationally convenient.

To balance these trade-offs, we add an extra step to the Mapper algorithm. Concretely, we extend the refined pull back cover into a cover over both nodes and edges. Given the set of refined clusters $\{C_{i,j}\}_{i \in I, j \in J_i}$, we compute a new set of clusters $\{C\prime_{i,j}\}_{i \in I, j \in J_i}$ where each cluster $C\prime_{i,j}$ contains the elements of $C_{i,j}$ as well as all the edges incident to the vertices in $C_{i,j}$. We use $\mathcal{R}_E$ (the edge-refined pull back cover) to refer to this open cover of the graph G computed from $f^{-1}(\mathcal{U})$. Then, our algorithm can be written as $\mathrm{sk}_1(\mathcal{N}(\mathcal{R}_E(f^{-1}(\mathcal{U}))))$ and we denote it by $\mathrm{GM}(f, \mathcal{U})$.

Remark 1: We note that Structural Mapper, unlike the original Mapper method, encodes two types of relationships via the edges of the output graph. The semantic connections highlight a similarity between clusters, according to the lens function (that is, two clusters have common nodes—as before), while structural connections show how two clusters are connected (namely, two clusters have at least one edge in common). This latter type of connection is the result of considering the extended cover over the edges. The two types of connections are not mutually exclusive because two clusters might have both nodes and edges in common.

We now broadly outline our proposed method, using the three main degrees of freedom of the Mapper algorithm to guide our discussion: the lens function, the cover, and the clustering algorithm.

4.1 Lens

The lens is a function $f : V \to \mathbb{R}^d$ over the vertices, which acts as a filter that emphasizes certain features of the graph. Typically, d is a small integer—in our case, $d \in \{1, 2\}$. The choice of f depends on the graph properties that should be highlighted by the visualization. In this work, we leverage the recent progress in the field of graph representation learning and propose a parameterized lens function based on graph neural networks (GNNs). We thus consider a function $f_\theta(v) = g_\theta(V, E, X)_v$, where g is a GNN with parameters θ taking as input a graph $G = (V, E)$ with n nodes and node features $X \in \mathbb{R}^{n \times k}$. For visualization purposes, we often consider a function composition $f_\theta(v) = (r \circ g_\theta)_v$, where $r : \mathbb{R}^{n \times d'} \to \mathbb{R}^{n \times d}$ is a dimensionality reduction algorithm like t-SNE (van der Maaten and Hinton, 2008).

Unlike the traditional graph theoretic lens functions proposed by Hajij et al. (2018), GNNs can naturally learn to integrate the features associated with the graph and its topology, while also scaling computationally to large, complex graphs. Additionally, visualisations can be flexibly tuned for the task of interest, by adjusting the lens g_θ through the loss function of the model.

4.2 Cover

The cover U determines the resolution of the output graph. For most purposes, we leverage the usual cover choice for Mapper, $\mathbb{R}^d$. When $d = 1$, we use a set of equally sized overlapping intervals over the real line. When $d = 2$, this is generalized to a grid of overlapping cells in the real plane. Using more cells will produce more detailed visualisations, while higher overlaps between the cells will increase the connectivity of the output graph. When chosen suitably, these hyperparameters are a powerful mechanism for obtaining multi-scale visualisations.

Another choice that we employ for designing differentiable pooling algorithms is a set of RBF kernels, where the second arguments of kernel functions are distributed over the real line. We introduce this in detail in **Section 5.2**.

4.3 Clustering

Clustering statistically approximates the (topological) connected components of the cover sets U_i. Mapper does not require a particular type of clustering algorithm; however, when the input topological space X is a graph, a natural choice, also adopted by Hajij et al. (2018), is to take the connected components of the subgraphs induced by the vertices $f^{-1}(U_i), i \in I$. Therefore, in principle, there is no need to resort to statistical clustering techniques.

However, relying on the topological connected components introduces certain challenges when the aim is to obtained a coarsened graph. Many real-world graphs comprise thousands of connected components, which is a lower bound to the number of connected components of the graph produced by GM. In the most extreme case, a graph containing only isolated nodes (namely, a point cloud) would never be coarsened by this procedure. Therefore, it is preferable to employ statistical techniques where the number of clusters can be specified. In our pooling experiments, we draw motivation from the relationship with other pooling algorithms and opt to assign all the nodes to the same cluster (which corresponds to no clustering).

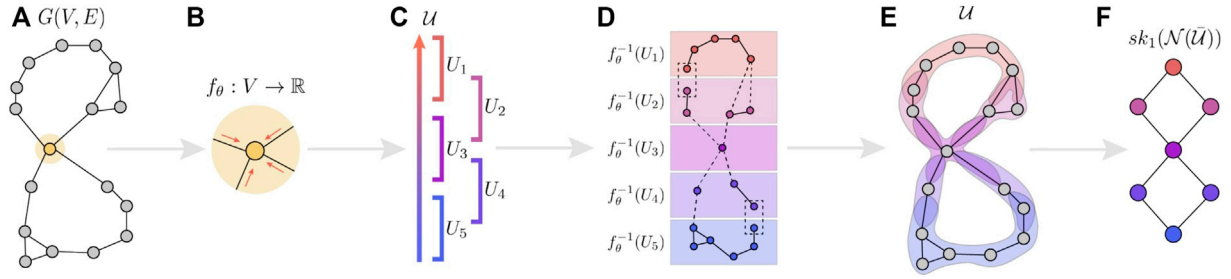


FIGURE 1 | A cartoon illustration of Structural Deep Graph Mapper (SDGM) where, for simplicity, a graph neural network (GNN) approximates a "height" function over the nodes in the plane of the diagram. The input graph **(A)** is passed through the GNN, which maps the vertices of the graph to a real number (the height) **(B–C)**. Given a cover $\mathcal{U}$ of the image of the GNN **(C)**, the edge-refined pull back cover $\overline{\mathcal{U}}$ is computed **(D–E)**. The dotted edges in **(D)** illustrate connections between the node clusters (strucutal connections), while the dotted boxes show nodes that appear in multiple clusters (semantic connections). The 1-skeleton of the nerve of the edge-refined pull back cover provides the pooled graph **(F)**.

We broadly refer to this instance of Structural Graph Mapper, with the choices described above, as Structural Deep Graph Mapper (SDGM). We summarize it step-by-step in the cartoon example in **Figure 1** and encourage the reader to refer to it.

5 STRUCTURAL GRAPH MAPPER FOR POOLING

We begin this section by introducing several theoretical results, which provide a connection between our version of Mapper and other graph pooling algorithms. We then use these results to show how novel pooling algorithms can be designed.

5.1 Relationship to Graph Pooling Methods

An early suggestion that Mapper could be suitable for graph pooling is given by the fact that it constitutes a generalization of binary spectral clustering, as observed by Hajij et al. (2018). This link is a strong indicator that Mapper can compute "useful" clusters for pooling. We formally restate this observation below and provide a short proof.

Proposition 5.1: Let L be the Laplacian of a graph $G(V,E)$ and l_2 the eigenvector corresponding to the second lowest eigenvalue of L, also known as the Fiedler vector (Fiedler, 1973). Then, for a function $f : V \to \mathbb{R}, f(v) = l_2(v)$, outputting the entry in the eigenvector l_2 corresponding to node v and a cover $\mathcal{U} = \{(-\infty, \varepsilon), (-\varepsilon, +\infty)\}$, Mapper produces a spectral bi-partition of the graph for a sufficiently small positive ϵ.

Proof: It is well known that the Fiedler vector can be used to obtain a "good" bi-partition of the graph based on the signature of the entries of the vector (i.e., $l_2(v) > 0$ and $l_2(v) < 0$) (please refer to Demmel. (1995) for a proof). Therefore, by setting ϵ to a sufficiently small positive number $\varepsilon < \min_v |l_2(v)|$, the obtained pull back cover is a spectral bi-partition of the graph.

The result above indicates that Mapper is a generalization of spectral clustering. As the latter is strongly related to min-cuts (Leskovec, 2016), the proposition also links them to Mapper. We now provide a much stronger result in that direction, showing that Structural Mapper is a generalization of all pooling methods based on soft-cluster assignments. Soft cluster assignment pooling methods use a soft cluster assignment matrix $S \in \mathbb{R}^{N \times K}$, where S_{ij} encodes the probability that node i belongs to cluster j, N is the number of nodes in the graph and K is the number of clusters. The adjacency matrix of the pooled graph is computed via $A' = S^T(A + I)S$. Below, we prove a helpful result concerning this class of methods.

Lemma 5.1: The adjacency matrix $A' = S^T(A + I)S$ defines a pooled graph, where the nodes corresponding to clusters encoded by S are connected if and only if there is a common edge (including self-loops) between them.

Proof: Let $L = \mathrm{AS}$. Then, $A'_{ij} = \sum_k^N S_{ik}^T L_{kj} = 0$ if and only if $S_{ik}^T = 0$ (node k does not belong to cluster i) or $L_{kj} = 0$ (node k is not connected to any node belonging to cluster j), for all k. Therefore, $A'_{ij} \neq 0$ if and only if there exists a node k such that k belongs to cluster i and k is connected to a node from cluster j. Due to the added self-loops, $A'_{ij} \neq 0$ also holds if there is a node k belonging to both clusters.

Proposition 5.2: $\mathrm{GM}(f, \mathcal{U})$ generalizes approaches based on soft-cluster assignments.

Proof: Let $s : V \to \triangle_{K-1}$ be a soft cluster assignment function that maps the vertices to the $(K-1)$-dimensional unit simplex. We denote by $s_k(v)$ the probability that vertex v belongs to cluster $k \leq K$ and $\sum_k^K s_k(v) = 1$. This function can be completely specified by a cluster assignment matrix $S \in \mathbb{R}^{N \times K}$ with $S_{ik} = s_k(i)$. This is the soft cluster assignment matrix computed by algorithms like minCut and DiffPool. Let $\mathcal{U} = \{U_i\}_{i \leq K}$ with $U_i = \{x \in \triangle_{K-1} x = \sum_j \lambda_j u_j, \sum_j \lambda_j = 1 \text{ and } \lambda_i > 0\}$ be an open cover of $\triangle_{K-1}$. Then consider an instance of GM where everything is assigned to a single cluster (i.e. same as no clustering). Clearly, there is a one-to-one correspondence between the vertices of $\mathrm{GM}(s, \mathcal{U})$ and the soft clusters. By Remark 1, the nodes corresponding to the clusters are connected only if the clusters share at least one node or at least one edge. Then, by **Lemma 5.1** the adjacency between the nodes of $\mathrm{GM}(s, \mathcal{U})$ are the same as those described by $A' = S^T(A + I)S$. Thus, the two pooled graphs are isomorphic.

We hope that this result will enable theoreticians to study pooling operators through the topological and statistical properties of Mapper (Dey et al., 2017; Carriere et al., 2018; Carrière and Oudot, 2018). At

the same time, we encourage practitioners to take advantage of it and design new pooling methods in terms of a well-chosen lens function f and cover $\mathcal{U}$ for its image. To illustrate this idea and showcase the benefits of this new perspective over graph pooling methods, we introduce two Mapper-based operators.

5.2 Differentiable Mapper Pooling

The main challenge for making pooling via Mapper differentiable is to differentiate through the pull back computation. To address this, we replace the cover of n overlapping intervals over the real line, described in the previous section, with a cover formed of overlapping RBF kernels $\phi(x, x_i) = \exp(-||x - x_i||^2/\delta)$, evaluated at n fixed locations x_i. The overlap between these kernels can be adjusted through the scale δ of the kernels. The soft cluster assignment matrix S is given by the normalized kernel values:

$$S_{ij} = \frac{\phi\left(\sigma\left(f_\theta\left(X_l\right)\right)_i, x_j\right)}{\sum_{j=1}^{n} \phi\left(\sigma\left(f_\theta\left(X_l\right)\right)_i, x_j\right)}, \tag{1}$$

where the lens function f_θ is a GNN layer, σ is a sigmoid function ensuring the outputs are in $[0, 1]$, and X_l are the node features at layer l. Intuitively, the more closely a node is mapped to a location x_i, the more it belongs to cluster i. By **Proposition 5.2**, we can compute the adjacency matrix of the pooled graph as $S^T(A + I)S$; the features are given by $S^T X$. This method can also be thought as a version of DiffPool (Ying et al., 2018), where the low-entropy constraint on the cluster assignment distribution is topologically satisfied, since a point cannot be equally close to many other points on a line. Therefore, each node will belong only to a few clusters if the scale δ is appropriately set.

In **Figure 2** we show two examples of RBF kernel covers for the output space. The scale of the kernel, δ, determines the amount of overlap between the cover elements. At bigger scales, there is a higher overlap between the clusters, as shown in the two plots. Because the line is one-dimensional, a point on the unit interval can only be part of a small number of clusters (that is, the kernels for which the value is greater than zero), assuming the scale δ is not too large. Therefore, DMP can be seen as a DiffPool variant where the low-entropy constraint on the cluster assignment is satisfied topologically, rather than by a loss function enforcing it.

5.3 Mapper-Based PageRank Pooling

To evaluate the effectiveness of the differentiable pooling operator, we also consider a fixed and scalable non-differentiable lens function $f : V \to \mathbb{R}$ that is given by the normalized PageRank (PR) (Page et al., 1999) of the nodes. The PageRank function assigns an importance value to each of the nodes based on their connectivity, according to the well-known recurrence relation:

$$f(X)_i \triangleq PR_i = \sum_{j \in N(i)} \frac{PR_j}{|\mathcal{N}(i)|}, \tag{2}$$

where $\mathcal{N}(i)$ represents the set of neighbors of the ith node in the graph and the damping factor was set to the typical value of $d = 0.85$. The resulting scores are values in $[0, 1]$ which reflect the probability of a random walk through the graph to end in a given node. Using the previously described overlapping intervals cover $\mathcal{U}$, the elements of the pull back cover form a soft cluster assignment matrix S:

$$S_{ij} = \frac{\mathbb{I}_{i \in f^{-1}(U_j)}}{\left|\{U_k | i \in f^{-1}(U_k)\}\right|} \tag{3}$$

where U_n is the nth cover set in the cover $\mathcal{U}$ of $[0, 1]$. It can be observed that the resulting clusters contain nodes with similar PageRank scores. Intuitively, this pooling method merges the (usually few) highly connected nodes in the graph, at the same time clustering the (typically many) dangling nodes that have a normalized PageRank score closer to zero. Therefore, this method favors the information attached to the most "important" nodes of the graph. The adjacency matrix of the pooled graph and the features are computed in the same manner as for DMP.

5.4 Model

For the graph classification task, each example G is represented by a tuple (X, A), where X is the node feature matrix and A is the adjacency matrix. Both our graph embedding and classification networks consist of a sequence of graph convolutional layers (Kipf and Welling, 2016); the lth layer operates on its input feature matrix as follows:

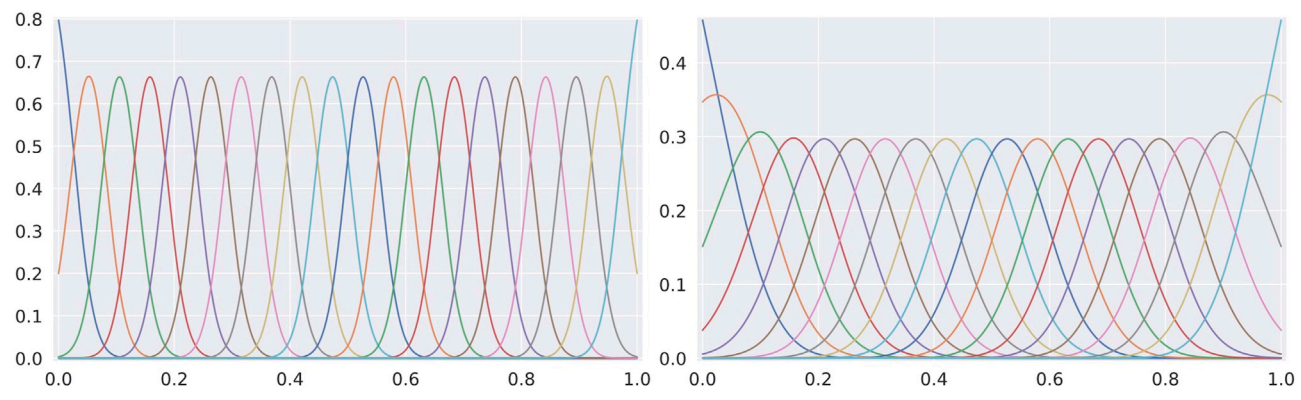


FIGURE 2 | Two covers of RBF kernels with different scales: $\delta = 0.002$ and $\delta = 0.01$. The x-axis corresponds to the unit interval where the nodes of the graph are mapped. The y-axis represents the value of the normalized RBF kernels.

TABLE 1 | Results obtained on classification benchmarks. Accuracy measures with 95% confidence intervals are reported. The highest result is bolded and the second highest is underlined. The first columns four are molecular graphs, while the others are social graphs. Our models perform competitively with other state of the art models.

Model	D&D	Mutag	NCI1	Proteins	Collab	IMDB-B	IMDB-M	Reddit-B	Reddit-5k
DMP (ours)	77.3 ± 3.6	<u>84.0 ± 8.6</u>	<u>70.4 ± 4.2</u>	75.3 ± 3.3	<u>81.4 ± 1.2</u>	73.8 ± 4.5	50.9 ± 2.5	86.2 ± 6.8	51.9 ± 2.1
MPR (ours)	78.2 ± 3.4	80.3 ± 6.0	69.8 ± 1.8	<u>75.2 ± 2.2</u>	81.5 ± 1.0	73.4 ± 2.7	50.6 ± 2.0	<u>86.3 ± 4.8</u>	<u>52.3 ± 1.6</u>
Top-*k*	75.1 ± 2.2	82.5 ± 6.8	67.9 ± 2.3	74.8 ± 3.0	75.0 ± 1.1	69.6 ± 3.8	45.0 ± 2.8	79.4 ± 7.4	48.5 ± 1.1
minCUT	77.6 ± 3.1	82.9 ± 6.0	68.8 ± 2.1	73.5 ± 2.9	79.9 ± 0.8	70.7 ± 3.5	50.6 ± 2.1	87.2 ± 5.0	52.9 ± 1.3
DiffPool	<u>77.9 ± 2.4</u>	94.7 ± 7.1	68.1 ± 2.1	74.2 ± 0.3	81.3 ± 0.1	72.4 ± 3.1	50.3 ± 1.8	79.0 ± 1.1	50.4 ± 1.7
WL	77.4 ± 2.6	74.5 ± 6.5	76.4 ± 2.7	74.7 ± 3.2	78.5 ± 1.1	72.1 ± 3.1	<u>50.7 ± 2.9</u>	66.7 ± 10.4	49.2 ± 1.4
Flat	69.9 ± 2.2	71.8 ± 4.3	65.5 ± 1.7	70.2 ± 2.6	80.9 ± 1.4	<u>73.6 ± 4.2</u>	48.5 ± 2.4	70.0 ± 10.8	49.5 ± 1.7
Avg-MLP	63.7 ± 1.4	69.1 ± 5.8	55.7 ± 2.8	61.8 ± 1.7	74.8 ± 1.3	71.5 ± 2.9	49.5 ± 2.2	53.6 ± 6.2	45.9 ± 1.6

$$X_{l+1} = \sigma\left(\widehat{D}^{-\frac{1}{2}}\widehat{A}\widehat{D}^{-\frac{1}{2}}X_l W_l\right), \tag{4}$$

where $\widehat{A} = A + I$ is the adjacency matrix with self-loops, $\widehat{D}$ is the normalized node degree matrix, W_l is the weight matrix of the l-th layer and σ is the activation function. After E layers, the embedding network simply outputs node features X_{L_E}, which are subsequently processed by a pooling layer to coarsen the graph. The classification network first takes as input node features of the Mapper-pooled graph,[2] X_{MG}, and passes them through L_C graph convolutional layers. Following this, the network computes a graph summary given by the feature-wise node average and applies a final linear layer which predicts the class:

$$y = \mathrm{softmax}\left(\frac{1}{|\mathrm{MG}|}\sum_{i=1}^{|\mathrm{MG}|} X_{L_C} W_f + b_f\right). \tag{5}$$

We note that either of these pooling operators could readily be adapted to the recently proposed message passing simplicial neural networks (MPSNs) (Bodnar et al., 2021) as a tool for coarsening simplicial complexes by dropping the 1-skeleton operator after computing the nerve. We leave this endeavor for future work.

5.5 Complexity

The topology of the output graph can be computed in $O(V + E)$ time when using a cover over the unit interval, as described above. The output graph can be computed via (sparse) matrix multiplication given by $S^T(A + I)S$, to take advantage of GPU parallelism and compute the coefficients associated with the edges.

6 POOLING EXPERIMENTS

6.1 Tasks

We illustrate the applicability of the Mapper-GNN synthesis within a pooling framework, by evaluating DMP and MPR in a variety of settings: social (IMDB-Binary, IMDB-Multi, Reddit-Binary, Reddit-Multi-5k), citation networks (Collab) and chemical data (D&D, Mutag, NCI1, Proteins) (Kersting et al., 2016).

[2]Note that one or more {embedding → pooling} operations may be sequentially performed in the pipeline.

6.2 Experimental Setup

We adopt a 10-fold cross-validation approach to evaluating the graph classification performance of DMP, MPR and other competitive state-of-the-art methods. The random seed was set to zero for all experiments (with respect to dataset splitting, shuffling and parameter initialisation), in order to ensure a fair comparison across architectures. All models were trained on a single Titan Xp GPU, using the Adam optimiser (Kingma and Ba, 2014) with early stopping on the validation set, for a maximum of 30 epochs. We report the classification accuracy using 95% confidence intervals calculated for a population size of 10 (the number of folds).

6.3 Baselines

We compare the performance of DMP and MPR to two other pooling methods that we identify mathematical connections with: minCUT (Bianchi et al., 2019) and DiffPool (Ying et al., 2018). Additionally, we include Graph U-Net (Gao and Ji, 2019) in our evaluation, as it has been shown to yield competitive results while performing pooling from the perspective of a learnable node ranking; we denote this approach by Top-k in the remainder of this section. The non-pooling baselines evaluated are the WL kernel (Shervashidze et al., 2011), a "flat" model (2 MP steps and global average pooling) and an average-readout linear classifier.

We optimize both DMP and MPR with respect to the cover cardinality n, the cover overlap (δ for DMP, overlap percentage g for MPR), learning rate and hidden size. The Top-k architecture is evaluated using the code provided in the official repository, where separate configurations are defined for each of the benchmarks. The minCUT architecture is represented by the sequence of operations described by Bianchi et al. (2019): MP(32)-pooling-MP(32)-pooling-MP(32)-GlobalAvgPool, followed by a linear softmax classifier. The MP(32) block represents a message-passing operation performed by a graph convolutional layer with 32 hidden units:

$$X^{(t+1)} = \mathrm{ReLU}\left(\tilde{A}X^{(t)}W_m + X^{(t)}W_s\right), \tag{6}$$

where $\tilde{A} = D^{-1/2}AD^{-1/2}$ is the symmetrically normalized adjacency matrix and W_m, W_s are learnable weight matrices representing

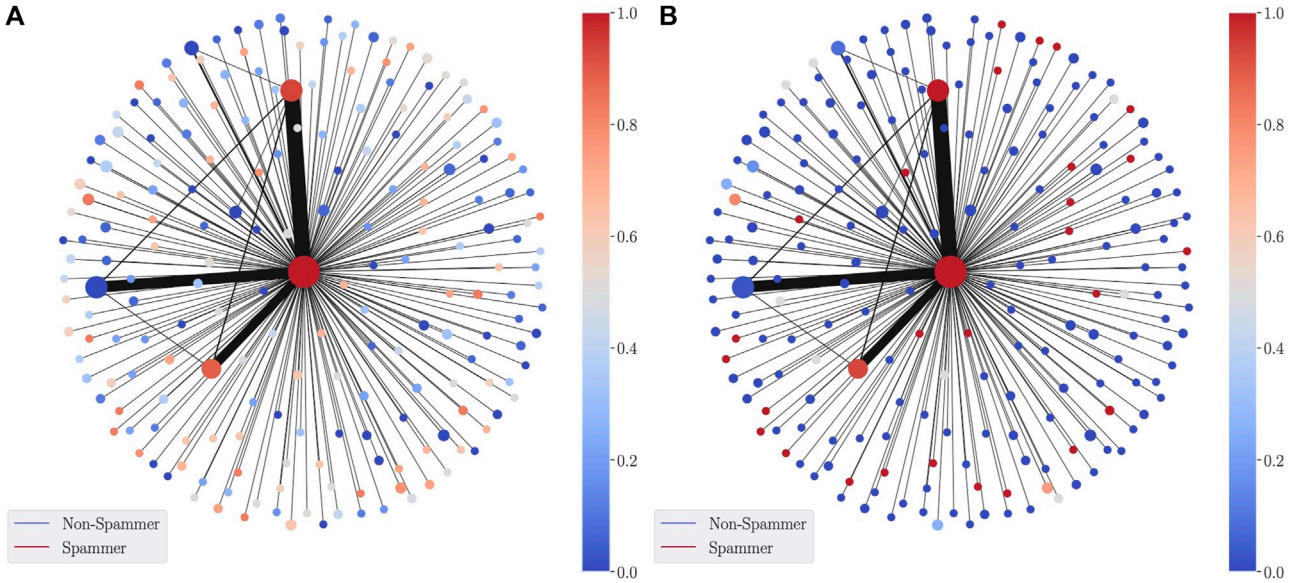


FIGURE 3 | SDGM visualization using as a lens function the GNN-predicted probability of a node in the network to be Spam. The **(A)** is colored with the average predicted spam probability in each cluster, whereas the **(B)** is colored by the proportion of true spammers in each node.

the message passing and skip-connection operations within the layer. The DiffPool model follows the same sequence of steps.

6.4 Evaluation Procedure

The best procedure for evaluating GNN pooling layers remains a matter of debate in the graph machine learning community. One may consider a fixed GNN architecture with a different pooling layer for each baseline; alternatively, the whole architecture can be optimized for each type of pooling layer. The first option, more akin to the typical procedure for evaluating pooling layers in CNNs on image domains, is used in papers like minCUT (Bianchi et al., 2019). The second option is more particular to GNNs and it is employed, for instance, by DiffPool (Ying et al., 2018). In this work, we choose the latter option for our evaluation.

We argue that for non-Euclidean domains, such as graph ones, the relationships between the nodes of the pooled graph and the ones of the input graph are semantically different from one pooling method to another. This is because pooling layers have different behaviors and may interact in various ways with the interleaved convolutional layers. Therefore, evaluating the same architecture with only the pooling layer(s) swapped is restrictive and might hide the benefits of certain operators. For example, Top-k pooling (one of our baselines) simply drops nodes from the input graph, instead of computing a smaller number of clusters from all nodes. Assume we fix the pooled graph to have only one node. Then Top-k would only select one node from the original graph. In contrast, DiffPool would combine the information from the entire graph in a single node. DiffPool would thus have access to additional information with respect to Top-k, so it would be unfair to conclude that one model is better than the other in such a setting. These differences implicitly affect the features of the output graph at that layer, which in turn affect the next pooling layer, as its computation depends on the features. This can have a cascading effect on the overall performance of the model. One might also argue that this procedure makes the evaluated models more homogeneous and, therefore, easier to compare. While this is true, the conclusions one can draw from such a comparison are much more limited because they are restricted to the particular architecture that was chosen.

For this reason, we have either run models with hyperparameters as previously reported by the authors, or optimized them ourselves end-to-end, where applicable. The best-performing configurations were (Appendix A details the hyperparameter search):

- MPR—learning rate $5e^{-4}$, hidden sizes {128, 128} (except for {64, 64} on IMDB-Binary and {32, 32} on IMDB-Multi), interval overlap 25% on Proteins, Reddit-Binary, Mutag, IMDB-Multi and 10% otherwise, batch size 32 (except for 128 on Proteins) and:
- D&D, Collab, Reddit-Binary, Reddit-Multi-5K: cover sizes {20, 5};
- Proteins, NCI1: cover sizes {8, 2};
- Mutag, IMDB-Binary, IMDB-Multi: cover sizes {4, 1};
- DMP—learning rate $5e^{-4}$, hidden sizes {128, 128}, $\delta = 1/(\text{cluster\_size})^2$ and:
- Proteins: cover sizes {8, 2}, batch size 128;
- Others: cover sizes {20, 5}, batch size 32;
- Top-k—specific dataset configurations, as provided in the official GitHub repository[3];

[3]https://github.com/HongyangGao/Graph-U-Nets/blob/48aa171b16964a2466fceaf4cb06fc940d649294/run_GUNet.sh

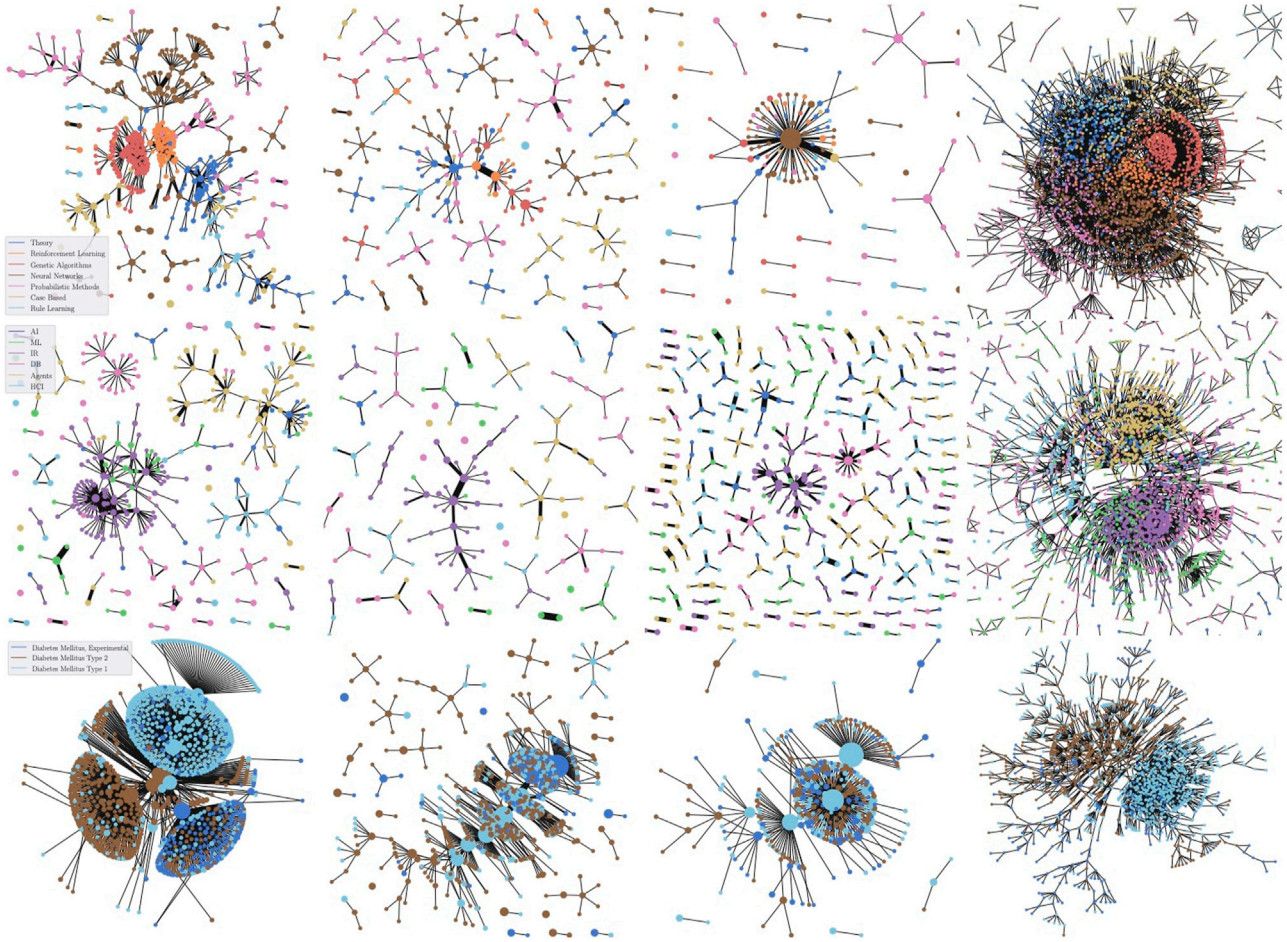


FIGURE 4 | Qualitative comparison between SDGM (first column), Mapper with an RBF graph density function (Hajij et al., 2018) (second), and Mapper with a PageRank function (Hajij et al., 2018) (third). The Graphviz visualization of the graph cores (fourth column) are added for reference. The rows show plots for Cora, CiteSeer, and PubMed, respectively. The graphs are colored based on the most frequent class in each cluster to aid the comparison. SDGM with unsupervised lens implicitly makes all dataset classes appear in the visualization more clearly separated. This does not happen in the baseline visualisations, which mainly focus on the class with the highest number of nodes from each graph.

- minCUT—learning rate $1e^{-3}$, same architecture as reported by the authors in the original work (Bianchi et al., 2019);
- DiffPool—learning rate $1e^{-3}$, hidden size 32, two pooling steps, pooling ratio $r = 0.1$ for D&D, Proteins, Collab and Reddit-Binary and $r = 0.25$ for Mutag, NCI1, IMDB-Binary, IMDB-Multi and Reddit-Multi-5K, global average mean readout layer, with the exception of Collab and Reddit-Binary, where the hidden size was 128;
- Flat: hidden size 32.

6.5 Pooling Results

The graph classification performance obtained by these models is reported in **Table 1**. We reveal that MPR ranks either first or second on all social datasets, or achieves accuracy scores within 0.5% of the best-performing model. This result confirms that PageRank-based pooling exploits the power-law distributions in this domain. The performance of DMP is similar on social data and generally higher on molecular graphs. We attribute this to the fact that all nodes in molecular graphs tend to have a similar PageRank score—MPR is therefore likely to assign all nodes to one cluster, effectively performing a readout. In this domain, DMP performs particularly well on Mutag, where it is second-best and improves by 3.7% over MPR, showing the benefits of having a differentiable lens in challenging data settings. Overall, MPR achieves the best accuracy on two datasets (D&D, Collab) and the next best result on three more (Proteins, Reddit-Binary and Reddit-Multi-5k). DMP improves on MPR by less than 1% on NCI1, Proteins, IDMB-Binary and IMDB-Multi, showing the perhaps surprising strength of the simple, fixed-lens pooling MPR operator.

7 MAPPER FOR VISUALISATIONS

Graph pooling methods and summarized graph visualisations methods can be seen as two sides of the same coin, since both aim

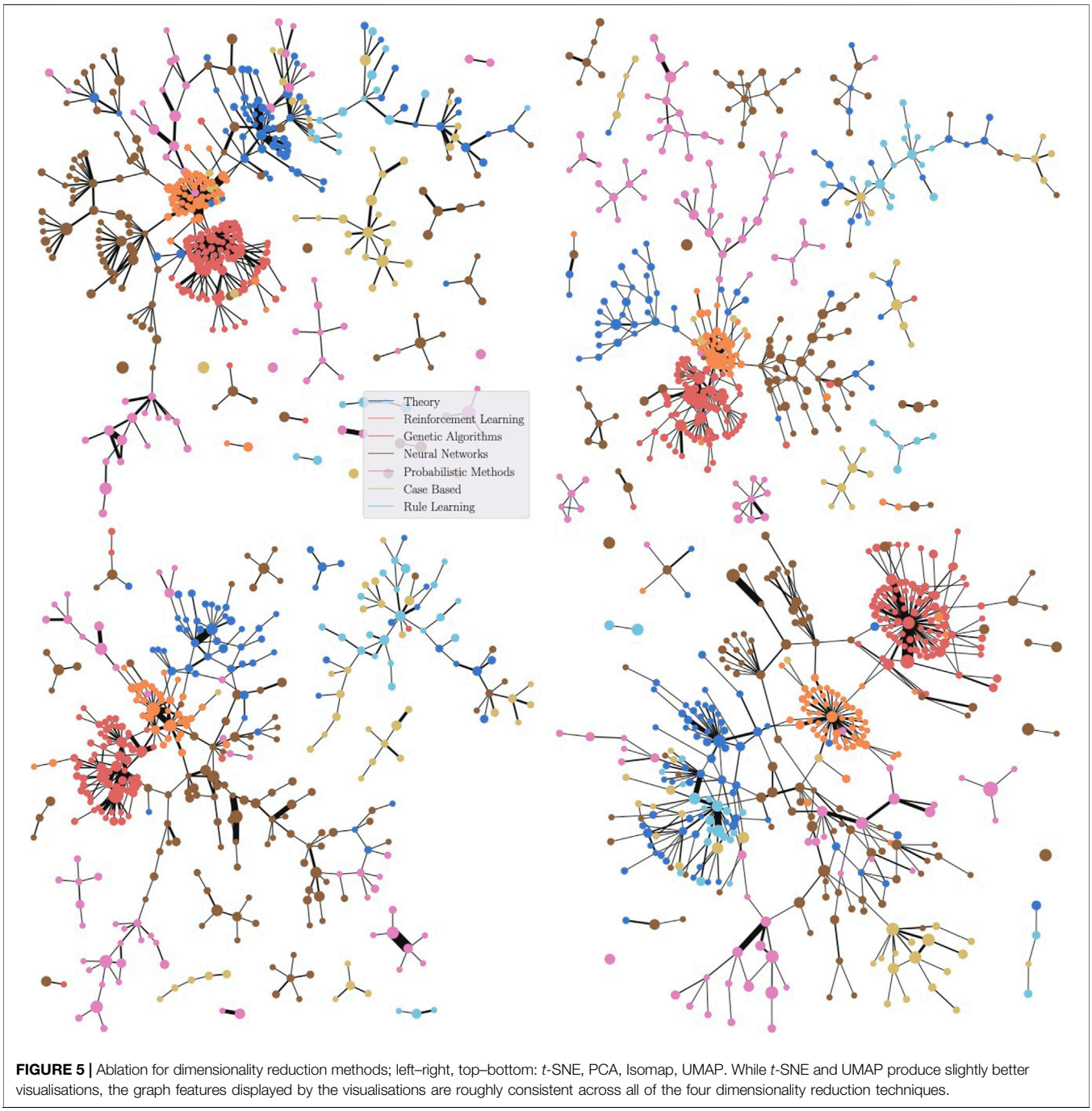


FIGURE 5 | Ablation for dimensionality reduction methods; left–right, top–bottom: *t*-SNE, PCA, Isomap, UMAP. While *t*-SNE and UMAP produce slightly better visualisations, the graph features displayed by the visualisations are roughly consistent across all of the four dimensionality reduction techniques.

to condense the information in the graph. We now turn our attention to the latter.

7.1 Visualisations in Supervised Learning

The first application of DGM is in a supervised learning context, where f_θ is trained via a cross entropy loss function to classify the nodes of the graph. When the classification is binary, $f_\theta : V \to [0, 1]$ outputs the probability that a node belongs to the positive class. This probability acts directly as the parameterization of the graph nodes. An example is shown in **Figure 3** (left) for a synthetic dataset a network formed of spammers and non-spammers. Spammers are highly connected to many other nodes in the network, whereas non-spammers generally have fewer neighbors. For the lens function, we use a Graph Convolutional Network (GCN) (Kipf and Welling, 2016) with four layers (with $32, 64, 128, 128$ hidden units) and ReLU activations trained to classify the nodes of the graph. For the spammer graph, the lens is given by the predicted spam probability of each node and the cover consists of 10 intervals over $[0, 1]$, with 10% overlap.

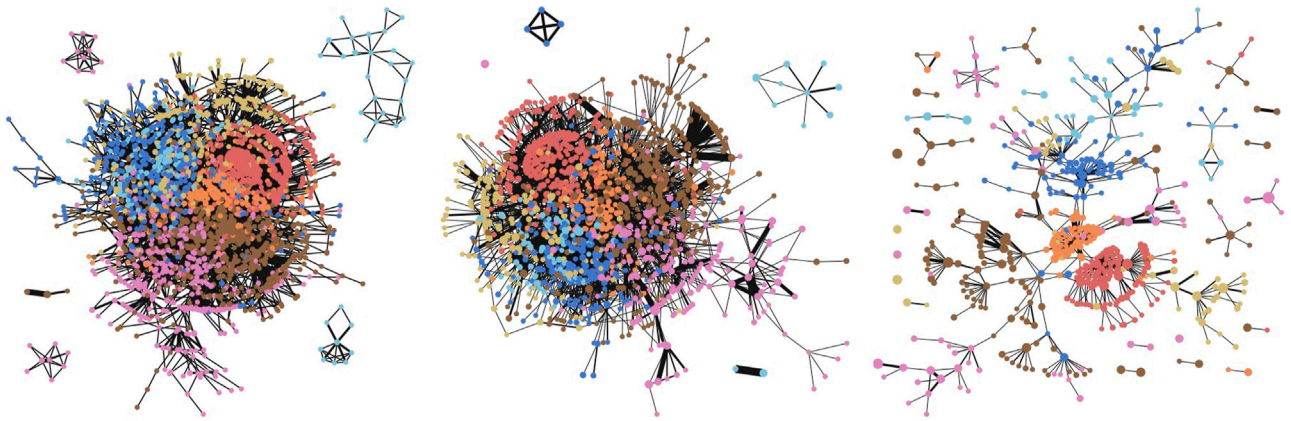


FIGURE 6 | Ablation for different types of unsupervised lenses (identity, untrained DGI, trained DGI). The trained DGI model significantly improves the quality of the visualisations.

Through the central cluster node, the SDGM visualization correctly shows how spammers occupy an essential place in the network, while non-spammers tend to form many smaller disconnected communities. When labels are available, we also produce visualisations augmented with ground-truth information. These visualisations can provide a label-driven understanding of the graph. For instance, in **Figure 3** (right) we color each node of the SDGM visualization according to the most frequent class in the corresponding cluster. This second visualization, augmented with the ground-truth information, can also be used to compare with the model predictions.

7.2 Visualization in Unsupervised Learning

The second application corresponds to an unsupervised learning scenario, where the challenge is obtaining a parameterization of the graph in the absence of labels. This is the typical use case for unsupervised graph representation learning models (Chami et al., 2020). The approach we follow is to train a model to learn node embeddings in $\mathbb{R}^{d'}$ (in our experiments, $d' = 512$), which can be reduced, as before, to a low-dimensional space via a dimensionality reduction method r. Unsupervised visualisations can be found in the qualitative evaluation in **Section 7.3**.

7.3 Qualitative Evaluation

In this section, we qualitatively compare SDGM against the two best-performing graph theoretic lens functions proposed by Hajij et al. (2018), on the Cora and CiteSeer (Sen et al., 2008) and PubMed (Yang et al., 2016) citation networks. Namely, we compare against a PageRank (Page et al., 1999) lens function and a graph density function $f(v) = \sum_{u \in V} \exp((-D(u,v)/\delta))$, where D is the distance matrix of the graph. For SDGM, we use a composition of an unsupervised Deep Graph Infomax (DGI) (Veličković et al., 2018) model $g_\theta : V \to \mathbb{R}^{512}$ and a dimensionality reduction function $r : \mathbb{R}^{512} \to \mathbb{R}^2$ based on t-SNE. To aid the comparison, we mark the nodes with the color of the most frequent class in the corresponding cluster. Additionally, we include a Graphviz (Gansner and North, 2000) plot of the full graph. We carefully fine-tuned the covers for each combination of model and graph.

As depicted by **Figure 4**, SDGM successfully summarizes many of the properties of the graphs that are also reflected by full graph visualisations. For instance, on Cora, Genetic Algorithms (in dark orange) are shown to be primarily connected to Reinforcement Learning (orange). At the same time, related classes that largely overlap in the full visualisation—Probabilistic Methods and Neural Networks (NNs) on Cora or Information Retrieval (IR) and ML on CiteSeer—are connected in the SDGM plot. In contrast, the baselines do not have the same level of granularity and fail to capture many such properties. Both PageRank and the graph density function tend to focus on the classes with the highest number of nodes, such as the IR class on CiteSeer or the NNs class on Cora, while largely de-emphasizing other classes.

7.3.1 Limitations

The proposed visualisations also present certain limitations. In an unsupervised learning setting, in the absence of any labels or attributes for coloring the graph, the nodes have to be colored based on a colourmap associated with the abstract embedding space, thus affecting the interpretability of the visualisations. In contrast, even though the graph theoretic lens functions produce lower quality visualisations, their semantics are clearly understood mathematically. This is, however, a drawback shared even by some of the most widely used data visualization methods, such as t-SNE or UMAP (McInnes et al., 2018). In what follows, we present additional visualisations and ablation studies.

7.4 Ablation Study for Dimensionality Reduction

We study how the choice of the dimensionality reduction method for the unsupervised visualisations affects the output. To test this, we consider the following dimensionality reduction methods: t-SNE (van der Maaten and Hinton, 2008), UMAP (McInnes

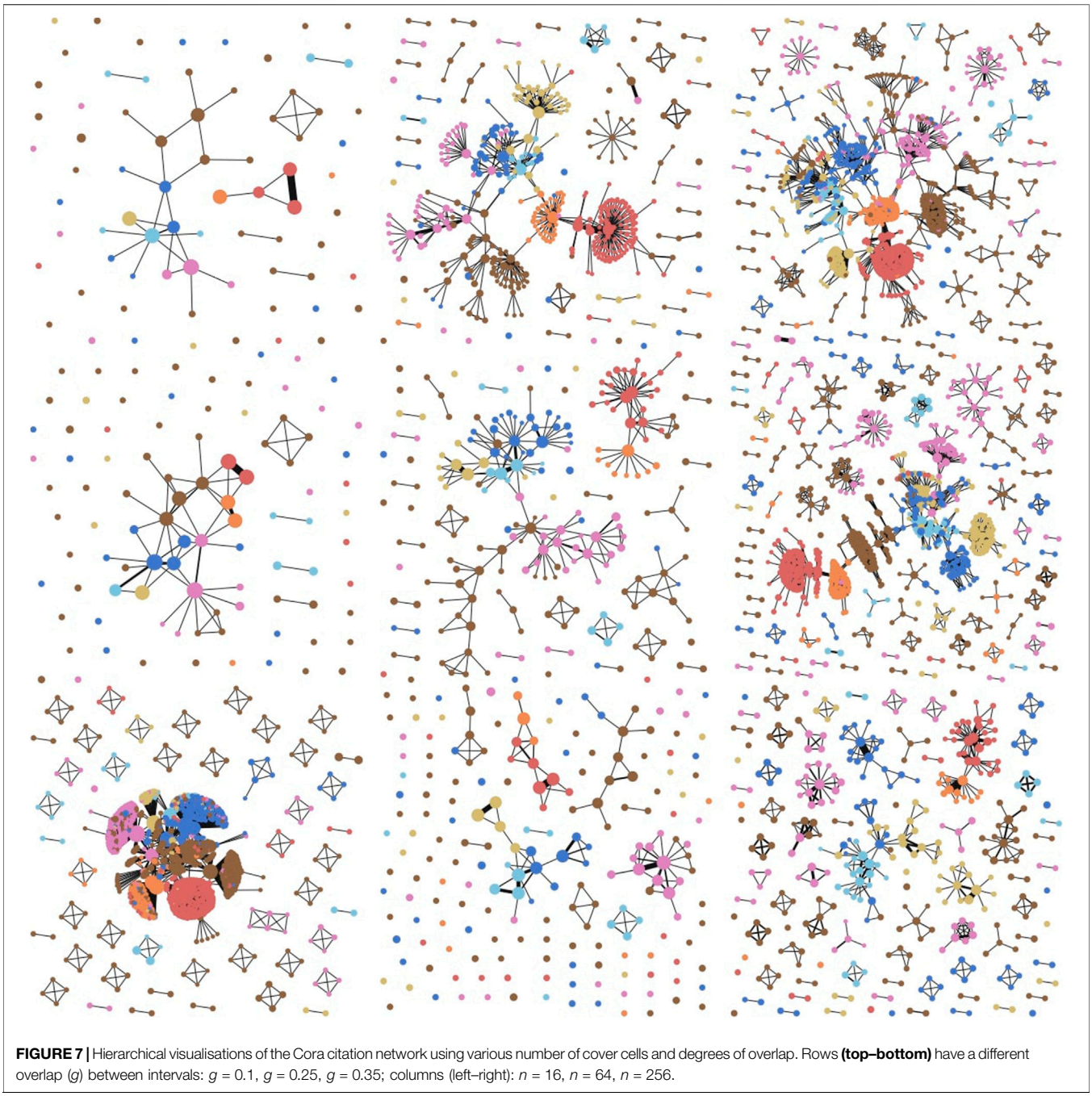


FIGURE 7 | Hierarchical visualisations of the Cora citation network using various number of cover cells and degrees of overlap. Rows **(top–bottom)** have a different overlap (g) between intervals: $g = 0.1$, $g = 0.25$, $g = 0.35$; columns (left–right): $n = 16$, $n = 64$, $n = 256$.

et al., 2018), IsoMap (Tenenbaum et al., 2000) and PCA. We use the same model as in **Section 7.2** and **Section 8**. 2D cells for the cover of all models. The overlap was set after fine-tuning to 0.2 for t-SNE and UMAP, and to 0.1 for the other two models. **Figure 5** displays the four visualisations. As expected, t-SNE and UMAP produce more visually pleasing outputs, due to their superior ability to capture variation in the GNN embedding space. However, the features highlighted by all visualisations are largely similar, generally indicating the same binary relations between clusters. This demonstrates that the GNN embedding space is robust to the choice of the dimensionality reduction method.

7.5 Ablation for the Unsupervised Lens

To better understand the impact of GNNs on improving the quality of the Mapper visualisations, we perform an ablation study on the type of unsupervised lens functions used within Mapper. The first model we consider is simply the identity function taking as input only graph features. The second model is a randomly initialized DGI model. Despite the

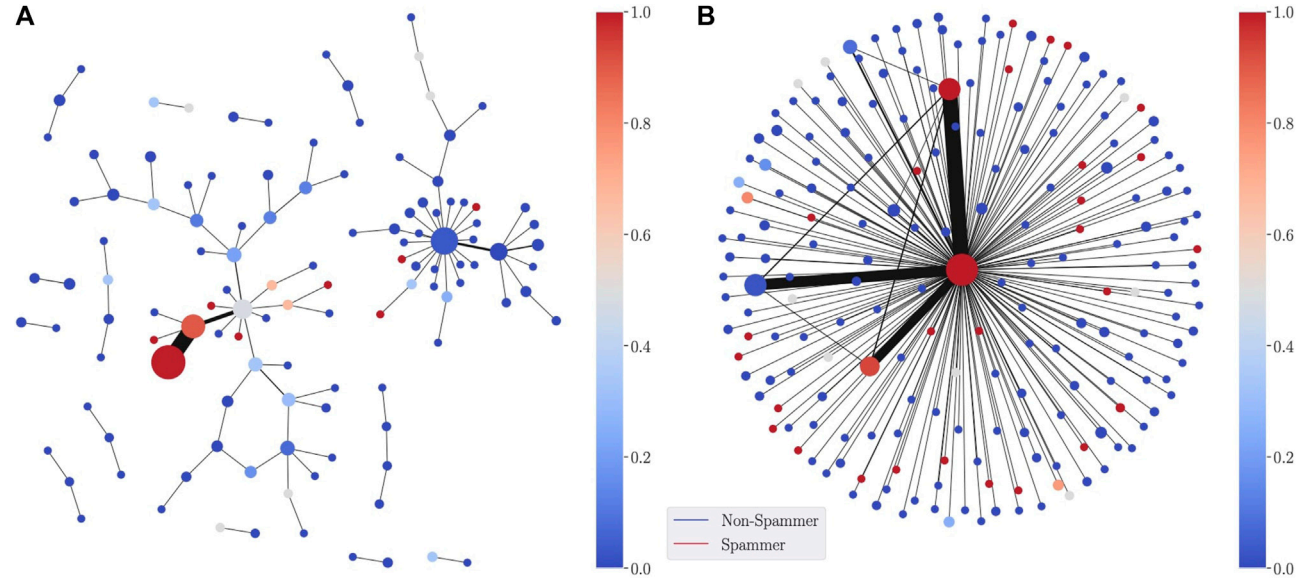


FIGURE 8 | DGM **(A)** vs SDGM **(B)** visualization of the sythetic spammer datasets. DGM does not capture important relational information between spammers and non-spammers.

apparent simplicity of a randomly initialized model, it was shown that such a method produces reasonably good embeddings, often outperforming other more sophisticated baselines (Veličković et al., 2018). Finally, we use our trained DGI model from **Section 7.2**. For all models, we perform a *t*-SNE reduction of their embedding space to obtain a 2D output space and use 81 overlapping cells that cover this space. An overlap of 0.2 is used across all models.

The three resulting visualisations are depicted in **Figure 6**. The identity model and the untrained DGI model do not manage to exploit the dataset structure and neither does particularly well. In contrast, the trained DGI model emphasizes all the classes in the visualization, together with their main interactions.

7.6 Hierarchical Visualisations

One of the most powerful features of Mapper is the ability to produce multi-resolution visualisations through the flexibility offered by the cover hyperparameters. As described in **Section 4**, having a higher number of cells covering the output space results in more granular visualisations containing more nodes, while a higher overlap between these cells results in increased connectivity. We highlight these trade-offs in **Figure 7**, where we visualize the Cora citation network using nine combinations of cells and overlaps. These kinds of hierarchical visualisations can help one identify the persistent features of the graph. For instance, when inspecting the plots that use $n = 64$ cells, the connections between the light blue class and the yellow class persist for all 3 degrees of overlap, which indicates that this is a persistent feature of the graph. In contrast, the connection between the red and orange classes is relatively reduced ($g = 0.25$) or none ($g = 0.1$) for low values of overlap, but it clearly appears at $g = 0.35$ in the top-right corner, suggesting that the semantic similarity between the two classes is very scale-sensitive (that is, less persistent).

7.7 The Importance of Capturing Structural Information

In this section, we revisit the synthetic spammer dataset to illustrate the importance of capturing structural information via the edge-refined pull back cover operator. To that end, we compare SDGM with a version using the usual refined pull back cover as in Hajij et al. (2018), while using the same lens function for both (a GCN classifier). We refer to the latter as DGM. The visualisations produced by the two models are included in **Figure 8**. We note that while both models capture the large cluster of spammers at the center of the network and the smaller communities of non-spammers, DGM does not capture the structural relationships between spammers and non spammers since it encodes only semantic relations.

8 CONCLUSION

We have introduced Deep Graph Mapper, a topologically grounded method for producing informative graph visualisations with the help of GNNs. We have shown these visualisations are not only useful for understanding various graph properties, but can also aid in visually identifying classification mistakes. Additionally, we have proved that Mapper is a generalization of soft cluster assignment methods, effectively providing a bridge between graph pooling and the TDA literature. Based on this connection, we have proposed two

Mapper-based pooling operators: a simple one that scores nodes using PageRank and a differentiable one that uses RBF kernels to simulate the cover. Our experiments show that both layers yield architectures competitive with several state-of-the-art methods on graph classification benchmarks.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding authors.

AUTHOR CONTRIBUTIONS

CB and CC have contributed equally in designing the model. CB has performed the visualization experiments and proved the theoretical results. CC has performed the pooling experiments. PL is the senior author.

FUNDING

CC is supported by EPSRC NRAG/465 NERC CDT Dream (grant no. NE/M009009/1). PL is funded by EPSRC.

ACKNOWLEDGMENTS

We would like to thank both reviewers for their constructive feedback and useful iterations on the manuscript. We would like to thank Petar Veličković, Ben Day, Felix Opolka, Simeon Spasov, Alessandro Di Stefano, Duo Wang, Jacob Deasy, Ramon Viñas, Alex Dumitru and Teodora Reu for their constructive comments. We are also grateful to Teo Stoleru for helping with the diagrams.

REFERENCES

Batagelj, V., Didimo, W., Liotta, G., Palladino, P., and Patrignani, M. (2010). Visual Analysis of Large Graphs Using. x.Y)-Clustering and Hybrid Visualizations. In (2010). IEEE Pacific Visualization Symposium (PacificVis). 209–216.

Beck, F., Burch, M., Diehl, S., and Weiskopf, D. (2017). A Taxonomy and Survey of Dynamic Graph Visualization. *Comp. Graphics Forum* 36, 133–159. doi:10.1111/cgf.12791

Bianchi, F. M., Grattarola, D., and Alippi, C. (2019). *Mincut Pooling in Graph Neural Networks* (arXiv preprint arXiv:1907.00481.

Bodnar, C., Frasca, F., Wang, Y. G., Otter, N., Montúfar, G., Liò, P., et al. (2021). *Weisfeiler and lehman Go Topological: Message Passing Simplicial Networks* (arXiv preprint arXiv:2103.03212.

Bruna, J., Zaremba, W., Szlam, A., and Lecun, Y. (2014). Spectral Networks and Locally Connected Networks on Graphs. *ICLR*.

Cangea, C., Veličković, P., Jovanović, N., Kipf, T., and Liò, P. (2018). *Towards Sparse Hierarchical Graph Classifiers* (arXiv preprint arXiv:1811.01287.

Carriere, M., Michel, B., and Oudot, S. (2018). Statistical Analysis and Parameter Selection for Mapper. *J. Machine Learn. Res.* 19, 478–516.

Carrière, M., and Oudot, S. (2018). Structure and Stability of the One-Dimensional Mapper. *Found. Comput. Math.* 18, 1333–1396. doi:10.1007/s10208-017-9370-z

Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., and Murphy, K. (2020). *Machine Learning on Graphs: A Model and Comprehensive Taxonomy* (ArXiv abs/2005.03675.

Chazal, F., and Michel, B. (2017). *An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists* (arXiv preprint arXiv:1710.04019.

Demmel, J. (1995). *UC Berkeley CS267 - Lecture 20: Partitioning Graphs without Coordinate Information II*.

Dey, T. K., Mémoli, F., and Wang, Y. (2017). Topological Analysis of Nerves, Reeb Spaces, Mappers, and Multiscale Mappers. In Symposium on Computational Geometry.

Dunne, C., and Shneiderman, B. (2013). Motif Simplification. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA: Association for Computing Machinery), CHI '13, 3247–3256. doi:10.1145/2470654.2466444

Dwyer, T., Riche, N. H., Marriott, K., and Mears, C. (2013). Edge Compression Techniques for Visualization of Dense Directed Graphs. *IEEE Trans. Vis. Comput. Graphics* 19, 2596–2605. doi:10.1109/TVCG.2013.151

Fiedler, M. (1973). Algebraic Connectivity of Graphs. *Czech. Math. J.* 23, 298–305. doi:10.21136/cmj.1973.101168

Gansner, E. R., and North, S. C. (2000). An Open Graph Visualization System and its Applications to Software Engineering. *Softw. Pract. Exper.* 30, 1203–1233. doi:10.1002/1097-024x(200009)3011<1203::aid-spe338>3.0.co;2-n

Gao, H., and Ji, S. (2019). Graph U-Nets. In International Conference on Machine Learning, 2083–2092.

Goller, C., and Kuchler, A. (1996). Learning Task-dependent Distributed Representations by Backpropagation through Structure. *ICNN*.

Gori, M., Monfardini, G., and Scarselli, F. (2005). A New Model for Learning in Graph Domains. *ICNN*.

Hajij, M., Rosen, P., and Wang, B. (2018). *Mapper on Graphs for Network Visualization*.

Huang, J., Li, Z., Li, N., Liu, S., and Li, G. (2019). AttPool: Towards Hierarchical Feature Representation in Graph Convolutional Networks via Attention Mechanism. In Proceedings of the IEEE International Conference on Computer Vision. 6480–6489.

Kersting, K., Kriege, N. M., Morris, C., Mutzel, P., and Neumann, M. (2016). *Benchmark Data Sets for Graph Kernels*.

Kingma, D. P., and Ba, J. (2014). *Adam: A Method for Stochastic Optimization* (arXiv preprint arXiv:1412.6980.

Kipf, T. N., and Welling, M. (2016). *Semi-Supervised Classification with Graph Convolutional Networks* (arXiv preprint arXiv:1609.02907.

Lee, J., Lee, I., and Kang, J. (2019). Self-Attention Graph Pooling. In International Conference on Machine Learning. 3734–3743.

Leskovec, J. (2016). *CS224W: Social and Information Network Analysis - Graph Clustering*.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. (2015). *Gated Graph Sequence Neural Networks* (arXiv:1511.05493.

Luzhnica, E., Day, B., and Lio, P. (2019). *Clique Pooling for Graph Classification* (arXiv preprint arXiv:1904.00374.

Ma, Y., Wang, S., Aggarwal, C. C., and Tang, J. (2019). Graph Convolutional Networks with EigenPooling. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 723–731.

McInnes, L., Healy, J., and Melville, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. ArXiv e-prints.

Nobre, C., Meyer, M., Streit, M., and Lex, A. (2019). The State of the Art in Visualizing Multivariate Networks. *Comp. Graphics Forum* 38, 807–832. doi:10.1111/cgf.13728

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The PageRank Citation Ranking: Bringing Order To the Web. *Tech. Rep.* Stanford InfoLab.

Ranjan, E., Sanyal, S., and Talukdar, P. P. (2019). *ASAP: Adaptive Structure Aware Pooling for Learning Hierarchical Graph Representations* (arXiv preprint arXiv:1911.07979.

Scarselli, F., Gori, M., Ah Chung Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). Computational Capabilities of Graph Neural Networks. *IEEE Trans. Neural Netw.* 20, 81–102. doi:10.1109/TNN.2008.2005141

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. (2008). Collective Classification in Network Data. *AIMag* 29, 93. doi:10.1609/aimag.v29i3.2157

Shervashidze, N., Schweitzer, P., Van Leeuwen, E. J., Mehlhorn, K., and Borgwardt, K. M. (2011). Weisfeiler-lehman Graph Kernels. *J. Machine Learn. Res.* 12, 2539–2561.

Singh, G., Mémoli, F., and Carlsson, G. E. (2007). Topological Methods for the Analysis of High Dimensional Data Sets and 3d Object Recognition. *SPBG* 91, 100.

Sperduti, A. (1994). Encoding Labeled Graphs by Labeling Raam. In NIPS.

Tenenbaum, J. B., Silva, V. d., and Langford, J. C. (2000). A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290, 2319–2323. doi:10.1126/science.290.5500.2319

van den Elzen, S., and van Wijk, J. J. (2014). Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations. *IEEE Trans. Vis. Comput. Graphics* 20, 2310–2319. doi:10.1109/tvcg.2014.2346441

van der Maaten, L., and Hinton, G. (2008). Visualizing Data Using T-SNE. *J. Machine Learn. Res.* 9, 2579–2605.

Veličković, P., Fedus, W., Hamilton, W. L., Liò, P., Bengio, Y., and Hjelm, R. D. (2018). *Deep Graph Infomax* (arXiv preprint arXiv:1809.10341.

von Landesberger, T., Kuijper, A., Schreck, T., Kohlhammer, J., van Wijk, J. J., Fekete, J.-D., et al. (2011). Visual Analysis of Large Graphs: State-Of-The-Art and Future Research Challenges. *Comp. Graphics Forum* 30, 1719–1749. doi:10.1111/j.1467-8659.2011.01898.x

Wattenberg, M. (2006). Visual Exploration of Multivariate Graphs. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY, USA: Association for Computing Machinery), CHI '06, 811–819. doi:10.1145/1124772.1124891

Yang, Z., Cohen, W. W., and Salakhutdinov, R. (2016). Revisiting Semi-supervised Learning with Graph Embeddings. *ICML*.

Ying, Z., You, J., Morris, C., Ren, X., Hamilton, W., and Leskovec, J. (2018). Hierarchical Graph Representation Learning with Differentiable Pooling. In Advances in Neural Information Processing Systems. 4800–4810.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

APPENDIX

A Model Architecture and Hyperparameters.

We additionally performed a hyperparameter search for DiffPool on hidden sizes 32, 64, 128 and for DGM, over the following sets of possible values:

- all datasets: cover sizes {[40, 10], [20, 5]}, interval overlap {10%, 25%};
- D&D: learning rate $\{5e^{-4}, 1e^{-3}\}$;
- Proteins: learning rate $\{2e^{-4}, 5e^{-4}, 1e^{-3}\}$, cover sizes {[24, 6], [16, 4], [12, 3], [8, 2]}, hidden sizes {64, 128}.

frontiers in Artificial Intelligence

ORIGINAL RESEARCH
published: 29 June 2021
doi: 10.3389/frai.2021.668395

Topological Data Analysis of *C. elegans* Locomotion and Behavior

Ashleigh Thomas[1]*, *Kathleen Bates*[1], *Alex Elchesen*[2†], *Iryna Hartsock*[2†], *Hang Lu*[1] and *Peter Bubenik*[2]

[1]School of Chemical and Biomolecular Engineering, Georgia Institute of Technology, Atlanta, GA, United States, [2]Department of Mathematics, University of Florida, Gainesville, FL, United States

We apply topological data analysis to the behavior of *C. elegans*, a widely studied model organism in biology. In particular, we use topology to produce a quantitative summary of complex behavior which may be applied to high-throughput data. Our methods allow us to distinguish and classify videos from various environmental conditions and we analyze the trade-off between accuracy and interpretability. Furthermore, we present a novel technique for visualizing the outputs of our analysis in terms of the input. Specifically, we use representative cycles of persistent homology to produce synthetic videos of stereotypical behaviors.

Keywords: persistent homology, topological data analysis, delay embedding, sliding window embedding, *C. elegans*, behavior phenotyping

OPEN ACCESS

Edited by:
Kathryn Hess,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Jose Andres Perea,
Michigan State University, United States
Samir Chowdhury,
Stanford University, United States

***Correspondence:**
Ashleigh Thomas
althomas41@gmail.com

†*These authors have contributed equally to this work*

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *16 February 2021*
Accepted: *12 May 2021*
Published: *29 June 2021*

Citation:
Thomas A, Bates K, Elchesen A, Hartsock I, Lu H and Bubenik P (2021) Topological Data Analysis of C. elegans Locomotion and Behavior. Front. Artif. Intell. 4:668395. doi: 10.3389/frai.2021.668395

1 INTRODUCTION

Model organisms are indispensable in understanding basic principles of biology. Studies of model organisms have played a major role in discoveries of disease mechanisms, disease treatment, and neuroscience principles. The behavior of these model organisms can illuminate responses and phenotypes important for understanding the effects of experimental conditions on subjects. Behavior can be affected by neuron activity, external stimuli, and past experiences (learning), so being able to adequately measure and compare behaviors is a useful evaluation tool for a wide range of experiments.

We propose persistent homology as a new tool for assessing behavior of *Caenorhabditis elegans*, worms that are a widely used model organism. Persistence has been successfully used to study high-dimensional time series, especially those that exhibit some quasi-periodic behavior like the undulation of *C. elegans* (Tralie, 2016; Tralie and Perea, 2018). But to the authors' knowledge, persistent homology has not been previously used to analyze *C. elegans* behavior, though it and similar techniques have been used to study *C. elegans* neural data (Petri et al., 2013; Backholm et al., 2015; Sizemore et al., 2019; Helm et al., 2020; Lütgehetmann et al., 2020).

In this paper we use persistent homology to study the locomotion of *C. elegans* in two settings. In our initial study (**Section 3.1**), we follow one worm as it moves on the surface of an agar plate. Under these conditions there are no barriers to movement and the locomotion is both smooth and complex. We show that persistent homology is able to detect and differentiate between various characteristic behaviors such as forward crawling, backward crawling, and transitioning between the two. We also show a unique contribution of persistence: the synthesis of skeleton data of *C. elegans* performing stereotyped, periodic behaviors. This translates into videos of, for example, forward crawling that are smooth when looped (see **Supplementary Material** for an example). Furthermore, this mapping from persistence features to behavior gives a concrete and biologically relevant interpretation of

results: features of interest—such as a feature that is detected in one sample and not another—can be expressed as videos of synthetic behavior.

We also analyze data from an experiment of the effect of environment on *C. elegans* (**Section 3.2**). In this setting a more controlled environment is required, so the organisms are submerged in a solution and confined to wells in microfluidic devices. Our main results study *C. elegans*' locomotion in solutions that have various levels of viscosity. We show that we are able to use persistent homology—and average persistence landscapes in particular—to summarize *C. elegans* locomotion in a way that allows the classification of the viscosity of an animal's environment with a high level of accuracy, and in fact a much higher level of accuracy than simpler methods based on speed and variety of postures. Our results indicate that persistent homology is a promising tool for quantifying the impact of changes to genotype and environment on *C. elegans* locomotion.

1.1 Related Work

Caenorhabditis elegans is a free-living soil nematode that has been a workhorse genetic model system. The nematode's transparent tissue, simple anatomy, and fast reproduction contribute to both ease in culture and a literal window into the internal workings of a living organism. Its completely sequenced genome contains many genes that are homologous to human genes, and importantly the ability to manipulate genes with relative ease makes it an extremely attractive model system. For neuroscience in particular, *C. elegans* presents a unique opportunity with its simple nervous system (just 302 neurons) that is complex enough to exhibit many sensory modalities, including mechanosensation, chemosensation, and response to heat, osmolarity, and smell.

Behavior characterization in *C. elegans* was historically qualitative, mainly relying on experimentalists specifying end-point assessment (e.g. whether the worm chemotaxes to a particular source of odor within a certain amount of time), or experimentalists using heuristics to assess behavior (e.g. naming worms genes "*unc*" for uncoordinated). In the last decade, machine vision tools first replaced human identifications of worms from images and videos, which allows much larger dynamic datasets to be annotated and analyzed. In recent years, further development in quantitative behavior characterization tools such as tracking (Stirman et al., 2011; Swierczek et al., 2011; Husson et al., 2012; Yemini et al., 2013; Porto et al., 2019), eigenworms (Stephens et al., 2008), behavior "dictionaries" (Brown et al., 2013), and t-SNE (Berman et al., 2016; Liu et al., 2018) have moved the field away from merely describing the outcome to understanding the types of behavior the brain of this simple system can generate. While many of these techniques do well in quantitatively describing behavior and distinguishing differences in behavior, behavioral dynamics are rich and opportunities abound in exploring behavioral dynamics using other mathematical tools.

Persistent homology has been used to analyze time series data in many different settings. Some earlier work was theoretical and studied the interaction between persistence and sliding window embeddings—which we used in this research—as well as proposed possible applications (Firas and Elizabeth, 2015; Perea and Harer, 2015; Perea, 2016). Research into gene expression has used persistent homology to detect patterns or classify whether a signal is periodic (Dequéant et al., 2008; Perea et al., 2015). Frequently, persistence has been used to study neural data (Petri et al., 2013; Backholm et al., 2015; Stolz et al., 2017; Sizemore et al., 2019; Helm et al., 2020; Lütgehetmann et al., 2020), and in many cases neural data from *C. elegans*, but the analysis tends to rely on clique complexes as the topological space of interest instead of sliding window embeddings.

2 MATERIALS AND METHODS

In this section we describe the collection and preprocessing of experimental data (**Section 2.1**), mathematical background (**Sections 2.2** and **2.3**), and pipeline for using topological data analysis on *C. elegans* behavior data (**Section 2.4**).

2.1 Description of Data

C. elegans (N2 strain) were cultured at 20°C under standard conditions on agar plates seeded with OP50 *E. Coli*. Animals were age-synchronized via hatch-off and cultured on plate until they reached day 1 of adulthood. For behavior experiments on agar, animals were prepared, imaged, and tracked as previously described (Porto et al., 2019). For behavior experiments in methylcellulose media, synchronized populations were then washed off of culture plates with M9 buffer. Unless otherwise noted, video data was collected on a dissecting microscope (Leica MZ16) using a CMOS camera (Thorlabs DCC3240M), with a frame rate of 30 frames per second and a magnification of ×1.2.

Behavior data was collected with animals confined with microfluidic devices. In these devices the cavities in which worms are loaded have only slightly greater depth than the width of an adult worm, which restricts worms to the focal plane of the microscope and to almost entirely 2-dimensional behavior. Microfluidic devices were fabricated as described previously (Chung et al., 2011). Methylcellulose solutions were prepared at concentrations of 0.5%, 1%, 2%, and 3% weight in volume of M9 buffer. To ensure that single animals could be isolated in single chambers of the unbonded microchamber microfluidic device, we first picked animals onto a room-temperature, unseeded plate. To ensure that animals were fully immersed in methylcellulose mixture, we used a glass pipet to aspirate a small amount of methylcellulose solution, and then aspirated animals from the unseeded plate one at a time into the methylcellulose solution. Then, single animals surrounded by methylcellulose mixture were pipetted into individual chambers of an unbonded PDMS chamber device. The device could then be flipped over onto a sterile 10 cm Petri dish and gently pressed down until the individual chamber walls came into contact with the Petri dish, preventing animals from leaving their chambers. Animals were then imaged in devices for about 5 min at 30 frames per second, resulting in time series data with 10,665 points.

To extract midline data from videos, we first found masks for each frame to isolate the worm from the background using a combination of Otsu thresholding (Otsu, 1979), image smoothing

using a Gaussian kernel, and size filtration. Otsu thresholding is a thresholding algorithm based on the gray-level histogram of an image. The threshold is identified by the grayscale pixel value that minimizes the intra-class variance of background and foreground pixels. We then broadly followed the method used in Stephens et al. (Stephens et al., 2008) to represent the worm's posture in "worm-centric" coordinates. Briefly, we found the midline of the worm in each frame by thinning the mask to a single line and interpolating between pixels of this line such that the midline was represented by 101 evenly spaced points. We calculated the tangent angle between each pair of adjacent points along the midline so that the animal's posture could be represented as a vector of angles, and then transformed those vectors with PCA so users could balance accuracy requirements and resource limitations via truncation of the data. We replaced frames in which animals were self-occluded with the data from the most recent non-self-occluded frame. We used untruncated PCA data for most computations because it has the same persistence output as the raw angle data. We used truncated PCA data (the first five principal components) for the computations in **Section 3.1**.

The videos for this study were selected from a much larger set of data based on how well they could be segmented and skeletonized. Some videos have subsequences that are difficult to automatically skeletonize because the animals self-occlude, i.e. bend in such a way as to cross over themselves. Thus, this dataset is likely biased toward less complex behaviors like thrashing and there are some cases where there are multiple videos of the same animal. The resulting data set has 40 samples of 10,665 points each with 10 samples for each viscosity condition.

2.2 Sliding Window Embeddings

Sliding window embeddings turn time series data into point cloud data in a way that does not forget the temporal information of the time series. There are some additional benefits to sliding window embeddings, including that they "separate" points that intersect each other in a time series, such as in Examples 2.6 and 2.7.

Definition 2.1: A time series is a sequence of vectors $(x_t)_{t\in T} = (x_t, x_{t+1}, x_{t+2}, \ldots)$ where each x_t is in the same finite-dimensional vector space V and T is a totally ordered set.

Remark 2.2: The totally ordered set T, which indexes the time series, can be $\mathbb{Z}$, $\mathbb{N}$, or a finite set like $[N] = \{1, 2, \ldots, N\}$. For many applications including the ones in this paper, the indexing set is finite and will be omitted in notation for brevity, as in $(x_t)_t$.Given any time series we can construct a new time series called a sliding window embedding, which is also known as a time delay embedding with a lag or delay time of 1.

Definition 2.3: Given a time series $\tau = (x_t)_t$ with vectors $x_t \in V$, a sliding window embedding of window length l *of* τ is a new time series, $\tau^l = (\widetilde{x_t})_t$, with

$$\widetilde{x_t} = [x_t \quad x_{t+1} \quad \ldots \quad x_{t+l-1}] \in V^l,$$

where $[\cdot]$ is concatenation of vectors.

That is, the t^{th} vector in the new time series is the concatenation of l consecutive vectors in the original time series and has dimension equal to $l \cdot \dim(V)$.

Remark 2.4: If the original time series has N points, then the sliding window embedding of window length l has $N - l + 1$ points, as one can see in Example 2.5.

Example 2.5: Consider the time series $\tau = ([1,2],[3,4],[5,6],[7,8],[9,10])$ in $\mathbb{R}^2$. The sliding window embedding of τ of window length $l = 3$ is

$$\tau^3 = ([1,2,3,4,5,6],\ [3,4,5,6,7,8],[5,6,7,8,9,10]) \subseteq \mathbb{R}^6,$$

which has $5 - 3 + 1 = 3$ points.

We applied persistent homology (**Section 2.3**) to sliding window embeddings of *C. elegans* video data in order to quantify behavior. Degree one persistent homology detected cycles in these sliding window embeddings which we show correspond to particular behaviors.

The cycles that persistent homology detects may consist of collections of points that trace out a closed curve. A cycle is "large" or highly persistent if it encloses an area that could fit a large ball; a cycle that is tall and skinny has small persistence.

Below we see two examples where a time series exhibits a single periodic behavior but persistent homology will detect either two or zero non-trivial cycles. In contrast, the persistent homology of a sliding window embedding detects exactly one non-trivial cycle in both examples.

Example 2.6: **Figure 1A** displays one period of a periodic time series in $\mathbb{R}^2$ with the property that if successive points are connected by line segments then the path of the time series self-intersects. To discover this figure-eight-shaped loop, one might try to use persistent homology (**Section 2.3**). However, persistent homology would detect two distinct loops, each comprising half of the period. See **Figure 2D** for an illustration of these loops.

Figures 1B,C show two-dimensional PCA projections of sliding window embeddings of the figure-eight for $l = 10$ and $l = 20$, respectively, using the first and third principal components. In these point clouds the time series draw out simple closed curves, and in fact in each of these cases persistence detects a single loop.

Notice that as the window length increases, the "size" of the loop increases. This increase in the loop's persistence makes it easier for persistent homology to robustly detect it.

Example 2.7: **Figure 3A** shows a 1-dimensional time series that is a discretization of a sine wave. This periodic behavior creates no loops — in fact, because the points take values in $\mathbb{R}$, the time series cannot produce degree 1 homology. However, a sliding window embedding, in this case of window length 4, creates a loop that is detected by persistent homology. That loop in $\mathbb{R}^4$ is projected down to two dimensions in **Figure 3B**.

2.3 Persistent Homology

In this section we provide an overview of persistent homology and how it may be used to produce quantitative summaries of the shape of a collection of points such as the sliding window embedding discussed above.

Definition 2.9: A simplicial complex on a set of vertices V is a collection K of non-empty subsets of V such that if $\tau \in K$ and $\tau' \subset \tau$, then $\tau' \in K$. An element $\tau \in K$ is called a simplex. An n-simplex or simplex of dimension n is a simplex $\tau \in K$ with size

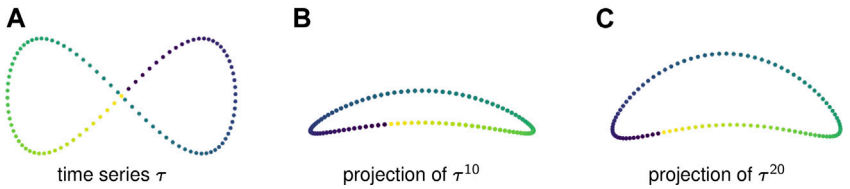


FIGURE 1 | (A) A time series in $\mathbb{R}^2$ determines a self-intersecting curve. **(B)** The sliding window embedding of window length 10 separates the previously intersecting segments of the curve. **(C)** A sliding window embedding with a higher window length separates the intersecting segments even further. With too small of a window length *l*, the resulting loop will be relatively flat and long and will therefore have small persistence and be difficult to differentiate from noise.

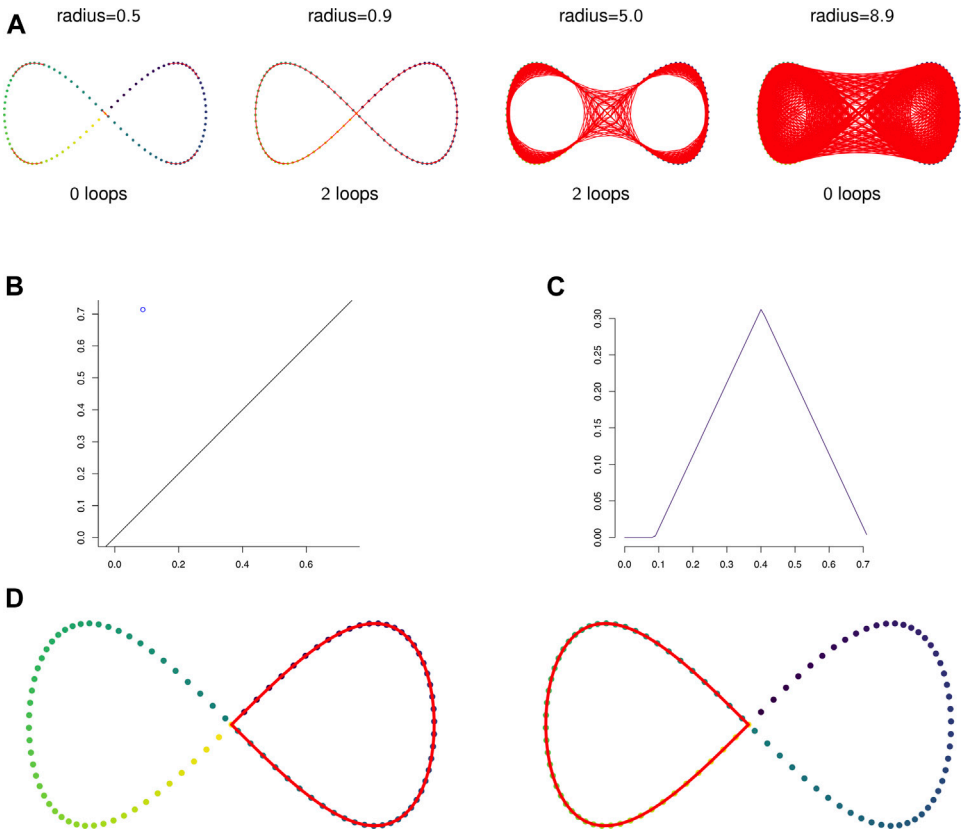


FIGURE 2 | (A) The 1-skeleton of the Vietoris-Rips filtered simplicial complex of a figure-eight-shaped point cloud at four scales. **(B)** The degree 1 persistence diagram of the figure eight in 2-dimensions. Note that the point has multiplicity 2. **(C)** The corresponding degree 1 persistence landscape. The first and second landscapes are nonzero and identical and all other landscapes are trivial. **(D)** The two loops that generate the homology of the Vietoris-Rips complex on the figure eight.

$|\tau| = n+1$. The 1-skeleton of a simplicial complex K is the set of simplices with dimension at most one. A filtered simplicial complex or filtration is a collection $\{K_r\}_{r\in S}$ of simplicial complexes K_r where $S \subseteq \mathbb{R}$ such that $K_r \subseteq K_s$ for all $r, s \in S$ with $r \leq s$.

Definition 2.10: Let $X \subset \mathbb{R}^d$ be a finite set and let $r \geq 0$. The Vietoris-Rips complex of X at scale r, denoted $\mathcal{R}_r(X)$, is the simplicial complex with vertex set X and whose simplices are given as follows. A subset $\{x_0, \ldots, x_n\} \subset X$ is an n-simplex in $\mathcal{R}_r(X)$ if and only if $|x_i - x_j| \leq r$ for all $i, j \in \{0, \ldots, n\}$.

Definition 2.11: The Vietoris-Rips filtration of a finite set $X \subset \mathbb{R}^d$ is the collection $\mathcal{R}(X) := \{\mathcal{R}_r(X)\}_{r\geq 0}$.Note that while the Vietoris-Rips complex of X is parameterized by the non-negative reals, the finiteness of X guarantees that $\mathcal{R}(X)$ consists of only finitely many distinct simplicial complexes.

Example 2.12: **Figure 2A** shows the 1-skeleton of the Vietoris-Rips complex of a pointcloud in $\mathbb{R}^2$ at four scales. Notice that each simplicial complex includes into the next.

The persistent homology of a Vietoris-Rips filtration can be represented by a multiset in $\mathbb{R}^2$ called a persistence diagram in

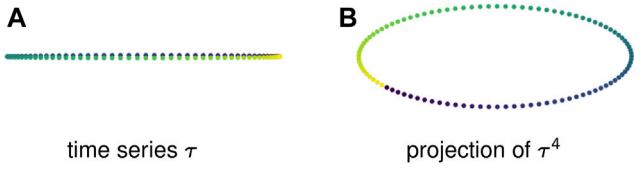


FIGURE 3 | Sliding window embeddings encode periodic behavior in the form of loops. **(A)** A periodic time series that has trivial first homology. **(B)** A sliding window embedding with $l = 4$ of the original time series that has nontrivial first homology.

which each point gives the scale of the appearance and disappearance of a topological feature (such as a loop) in the filtration.

Example 2.13: **Figures 2B,C** show the persistent homology in degree 1 of Example 2.12. Notice that both **Figure 2B**—the persistence diagram—and **Figure 2C**—the persistence landscape (see Definition 2.14)—show two cycles, but because they are born and die at exactly the same radius parameters they are plotted in the same place. The two cycles are shown in **Figure 2D**.

It is difficult to apply standard tools of statistics and machine learning directly to persistence diagrams, which, for example, need not have unique averages (Mileyko et al., 2011). A solution is to map persistence diagrams into a vector space or Hilbert space. One such mapping is the persistence landscape. See (Bubenik, 2015) for the following definitions and results.

Definition 2.14: For $a < b$ let $f_{a,b} : \mathbb{R} \to \mathbb{R}$ be the piecewise-linear function given by

$$f_{a,b}(t) = \begin{cases} t - a, \text{ if } a \le t \le \dfrac{a+b}{2} \\ b - t, \text{ if } \dfrac{a+b}{2} \le t \le b \\ 0, \text{ otherwise.} \end{cases}$$

Given a persistence diagram $\mathrm{Dgm}_p(\mathcal{K})$, the corresponding kth persistence landscape is the function $\lambda_k : \mathbb{R} \to \mathbb{R}$ given by defining $\lambda_k(t)$ to be the kth largest value of $f_{a,b}(t)$ over all points $(a, b) \in \mathrm{Dgm}_p(\mathcal{K})$. The persistence landscape is the sequence $(\lambda_k)_k$. The parameter k is called the depth of the persistence landscape. For a point cloud X, we will denote by PL (X) the persistence landscape obtained by applying degree 1 persistent homology to the Vietoris-Rips filtration of X.

Persistence landscapes have unique averages, satisfy the law of large numbers and central limit theorems, and can be discretized for computations. Because the sequence of functions that make up a landscape are nested, they can all be graphed on the same plot as in the right column of **Figure 4**.

While the persistence landscape is defined to be an object in a space of continuous functions, it can be discretized and turned into a finite-dimensional vector. Through discretization, each depth of the landscape transforms from a continuous function on $\mathbb{R}$ to a vector where the i^{th} entry in the vector corresponds to the function value at the i^{th} discrete parameter value. The vectors for each depth of the landscape are concatenated together to produce a single high-dimensional vector. These discrete landscapes can be computed directly, which we did for the computations outlined in **Section 2.4**.

This vector space (in fact, Hilbert space) setting lets us use linear algebra-based statistical and machine learning techniques such as principal component analysis (PCA). The principal components from PCA on discretized landscapes can be converted into a format much like a persistence landscape—a sequence of continuous functions on $\mathbb{R}$—but the principal components are not themselves persistence landscapes because the functions fail to be nonnegative.

2.4 Pipeline

In this section we give details for our analysis of *C. elegans* data. The input consist of piecewise linear midlines of *C. elegans* from video recordings as described in **Section 2.1**. These midlines were parameterized by the 100 angles between adjacent segments and then were transformed using PCA, so each sample input to our system was a time series τ of 100-dimensional vectors measured in radians. See Figure 1 in (Stephens et al., 2008) and the accompanied description for more details on this parameterization of the *C. elegans* midlines or see our short summary of the procedure in **Section 2.1**.

The time domain of this time series was divided into overlapping patches of a given size called the patch length, resulting in a collection $\{\tau_i\}_i$ of smaller time series. For our experiments, a patch length of 300 was chosen, with adjacent patches overlapping by half of the patch length. The sliding window embeddings of the τ_i were then computed with window length parameter $l = 20$, resulting in a new collection $\{(\tau_i)^l\}_i$ of time series of length $300 - l + 1 = 281$. This analysis is not particularly sensitive to the choices of the hyperparameters patch length and window length; only extreme changes in either parameter lead to significant changes in results. The hyperparameter choices were motivated by the timescales at which *C. elegans* complete meaningful behaviors: for patch length, 150 frames of 30 fps video is 5 s of behavior; for window lengths, 20 frames is 0.67 s and corresponds to roughly one period of forward crawling in adult *C. elegans* submerged in the 0.5% methylcellulose environment. Strategies for choosing appropriate window lengths are described in (Perea and Harer, 2015). The method for cross validation of the choice of window length is described at the end of this section.

Persistence diagrams were computed for each of the patches $(\tau_i)^l$. This step accounts for the vast majority of the computational resources of the pipeline, and the computational costs are made worse by the concatenation of vectors in a sliding window embedding. This is where we greatly benefit from the preprocessing that turns video data, which is extremely high-dimensional (see (Tralie and Perea, 2018) Section 3.1), into a 100-dimensional time series. On a 2017 15-inch MacBook Pro with a 2.8 GHz Intel Core i7 processor and 16 GB of RAM, this step took 22588.632 s, or about 6 h and 15 min.

For each $(\tau_i)^l$, a (discretized) persistence landscape PL $((\tau_i)^l)$ was computed from the persistence diagram of the Vietoris-Rips complex $\mathcal{R}((\tau_i)^l)$. The grid of parameter values on which the

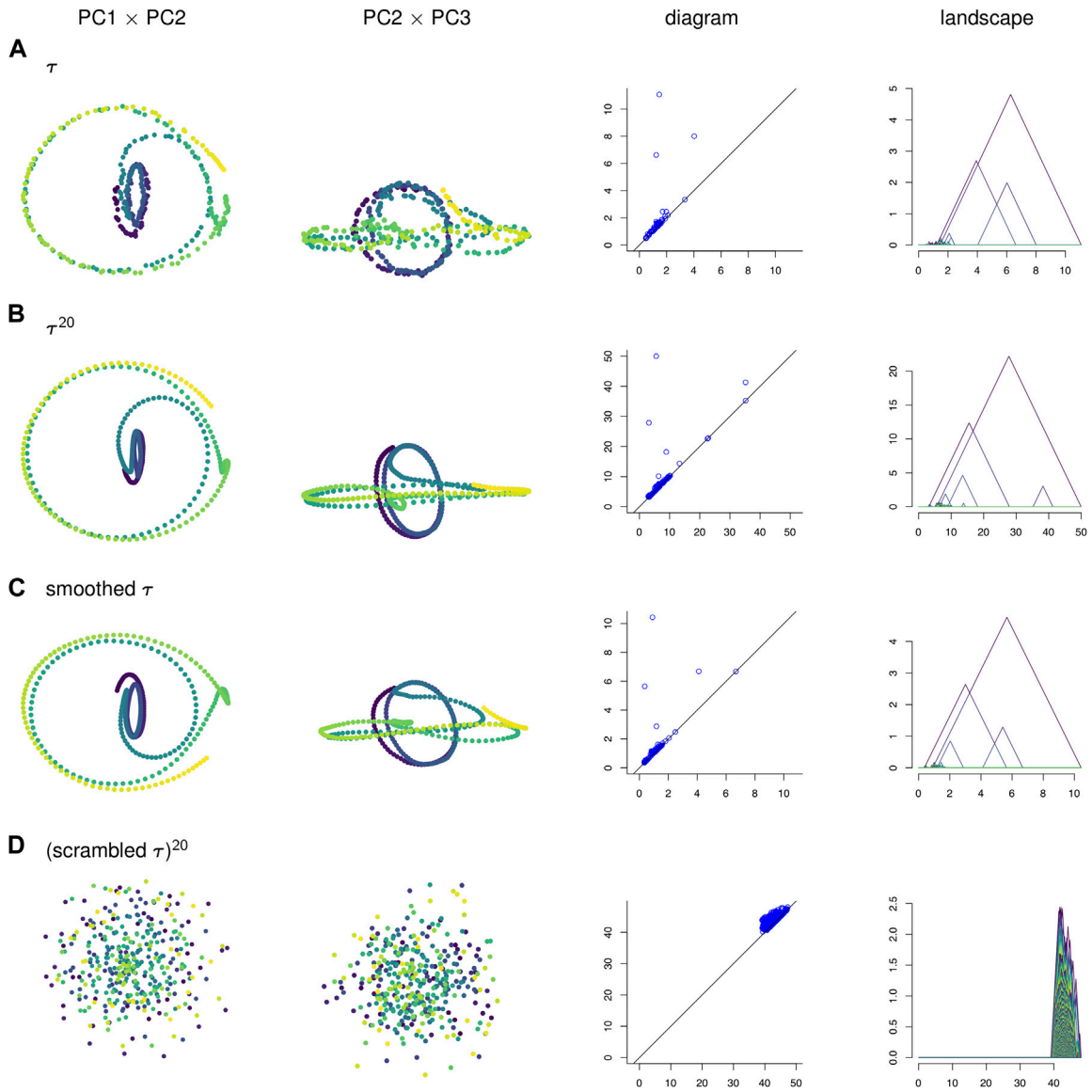


FIGURE 4 | (A) A time series τ, **(B)** its sliding window embedding of window length $l = 20$, **(C)** its smoothing by moving average filter of window length 20, and **(D)** the null model, all projected onto 2-dimensional axes of principal components. The corresponding persistence diagrams and persistence landscapes are to the right.

persistence landscape was evaluated to produce its discretization was chosen to include all of the bars of the barcode and to be sufficiently fine to produce nice visualizations. Since the persistence landscapes are piecewise linear with slope bounded by ± 1, the step size of this discretization bounds the error and there is eventually little to be gained from a finer discretization. The step size of the discretization we used was 0.1. The maximum depth was chosen to include all nonzero depths of the persistence landscapes.

The collection $\{\mathrm{PL}\,((\tau_i)^l)\}_i$ of persistence landscapes was then assembled into a single summary for a given video by averaging the persistence landscapes across patches to result in a single average persistence landscape associated to each video. These steps are summarized in **Figure 5**. For each environment viscosity, the average persistence landscapes for each video were averaged to give an average persistence landscape of the class.

The persistence landscapes for each class were used for analysis of the sliding window embedding as follows. Distances between each class' average persistence landscapes were computed using the usual Euclidean distance. The pairwise distances were visualized using multidimensional scaling to give a 2-dimensional visualization of the similarities between the classes; see **Figure 6A**.

Principal component analysis (PCA) was applied to the set of average persistence landscapes for each video and the first two principal components were plotted as sequences of functions. We

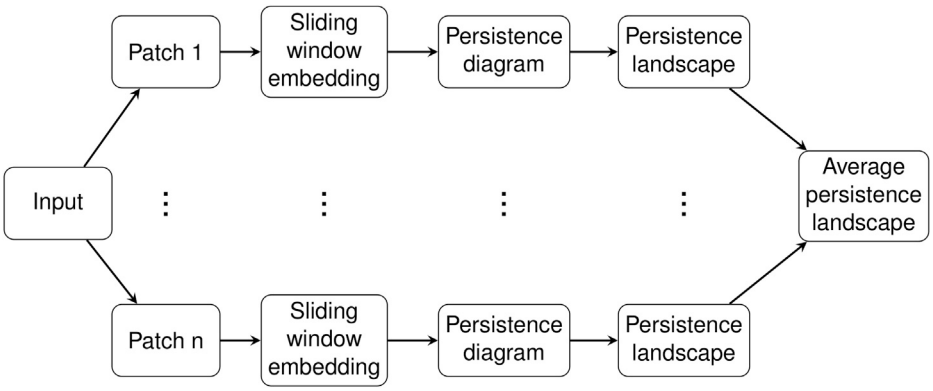


FIGURE 5 | Pipeline for a single worm.

plotted the PCA projection of these video average persistence landscapes together with the average of each class; see **Figure 7**. These plots visualize some of the similarity between classes and variation within classes.

Next, we further studied the variation within classes in two ways. First, the standard deviations of each coordinate were computed for the average persistence landscapes of the videos in each class. These were visualized as sequences of functions in **Figure 8B** to show the variation in different parts of the average persistence landscapes. Second, we applied PCA to the average persistence landscapes of the videos in each class and plotted the cumulative variances explained by the first n principal components for $n = 1, \ldots, 10$ (10 is the number of samples in each class). The first three principal components were also computed. See **Figure 9**.

We conducted a permutation test on the pairwise Euclidean distances between the average persistence landscapes of each videos. We used 10,000 permutations for each permutation test. The approximate p-value equals the percentage of cases in which the distance is at least as large as the observed distance. Results can be found in **Table 2A**.

We applied multiclass support vector machines (SVM) to classify samples according to viscosity of their environments. We use the ksvm function from the kernlab package in R on the average persistence landscapes of the videos. Accuracy was estimated using 10-fold cross validation with cost set to 10. Cross validation was repeated 20 times and the results were averaged. A confusion matrix for one instance of SVM with 10-fold cross validation was also computed and is shown in **Table 2B**.

We used support vector regression (SVR) to approximate viscosity of the worm environments given the worm's behavior data. The goal was to assess how predictive our techniques are. Accuracy was estimated by averaging 10 repetitions of 10-fold cross validation. Results are plotted in **Figure 9**.

As a final step we applied cross validation to the hyperparameter window length. We cross-validated the choice of window length by running the above pipeline for window lengths of 1, 10, 20, and 30 on a subset of the data. To reduce total computation time, we restricted to the first minute of each video, which corresponds to 1800 frames. We then compared the permutation test and multiclass SVM results to see which hyperparameter choice gave the best results. The results from cross validation of window length can be found in **Section 3.2.1**.

2.5 Validation

To help validate our pipeline, we compared our results to those obtained by applying the same computational procedure to a null model given by randomly permuting the frames in each video.

We also studied the effects of using a preprocessing step different from sliding window embeddings: moving average filters. A moving average filter of window length l of time series data creates a new time series. This time series has the same length as the corresponding sliding window embedding and each point in the time series is constructed using the same window of the original time series. The moving average filter, however, takes the average of the vectors in its window, instead of concatenating them.

Furthermore, we compared our results to the ones obtained from two simpler techniques. For the first technique we attempted to characterize *C. elegans* behavior using the speed of the worm. To do this, we computed 2-norms of the differences between two consecutive frames of angle data and then averaged all of those discrete derivatives, which resulted in a single value per sample. For the second technique, which could be described as measuring the variance of the worm's pose over a video, we computed standard deviations for the angle data coordinate-wize, which resulted in a vector of length 100 per sample. In each case we performed a permutation test and multiclass SVM on the resulting feature vectors. The results of these experiments appear in **Section 3.2.2**.

3 RESULTS

We present the results of a case study of a single sample of behavior data and an experiment on the effects of viscosity of the surrounding environment on *C. elegans* locomotion and

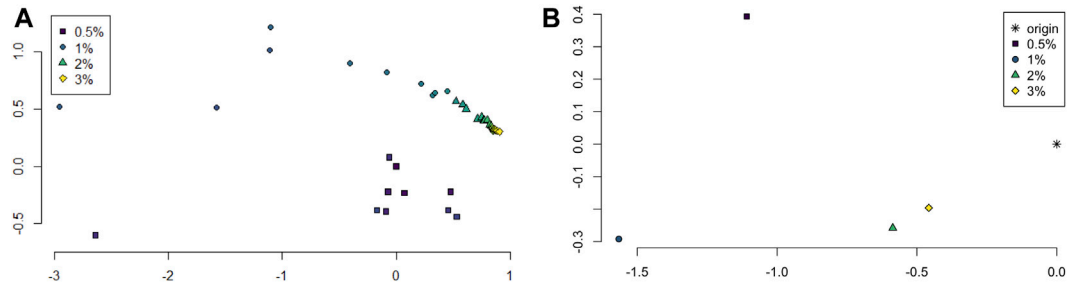


FIGURE 6 | (A) Multidimensional scaling of average persistence landscape of each sample. **(B)** Multidimensional scaling of average persistence landscapes of the classes and the origin.

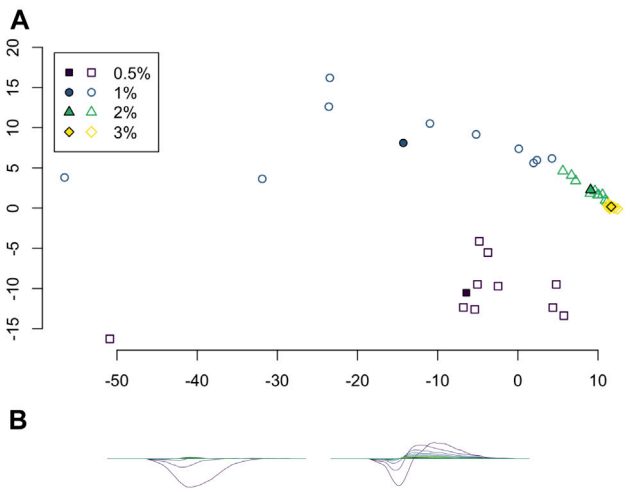


FIGURE 7 | (A) Projection of average persistence landscapes onto two principal components, with average persistence landscapes of videos given by outlined symbols and average persistence landscapes of classes given by solid symbols. **(B)** The first two principal components.

behavior. The case study assesses a significantly smaller data set and directly links topological features to specific behaviors. The experimental results in **Section 3.2** are more difficult to directly interpret in terms of specific behaviors but nonetheless we show the effectiveness of persistent homology in distinguishing variations in behaviors. We leave more explicit interpretation of average persistence landscapes in terms of specific behaviors for future work.

3.1 An Illustrative Case Study

The following results were obtained by carefully analyzing a sample of *C. elegans* behavior data from a video of a worm crawling on agar. The sample consists of 400 frames of a 30 frames per second video, so roughly 13 s of behavior. Having a solid surface to provide friction forces slower but more complicated behavior than we would see in an aqueous environment, and we take advantage of the resulting clarity of the data.

In the data we analyzed the subject exhibits the following behaviors in chronological order:

1. crawl forward,
2. crawl backward,
3. pause, and
4. crawl backward again.

Below we analyze the time series τ from this data, the corresponding sliding window embedding τ^{20}, the corresponding moving average filter, and the sliding window embedding of the corresponding null model. These comparisons illustrate various strengths of the sliding window embedding: it smooths the noise from the original time series; it retains more geometric data than the moving average filter; and it captures temporal data from the original time series that is destroyed in the null model. Then, using representative cycles we construct synthetic *C. elegans* midline data that produce a forward crawl and explain how this process gives concrete interpretations of persistence features in terms of synthetic behavior data. See **Section 3.1.1**.

To visualize the four point clouds on which we will compute persistence, we apply PCA and project onto the first few principal components. Some of these projections are shown in the two left-most columns of **Figure 4**. In contrast to the three other time series, the null model time series in **Figure 4D** has no discernible geometric structure. It appears that the corruption of temporal information has destroyed the interpretability of the visualizations of sliding window embeddings. Meanwhile, the original time series τ in **Figure 4A** has a similar shape to its sliding window embedding τ^{20} in **Figure 4B**, with the caveat that the sliding window embedding has the effect of smoothing the data and making it more robust to noise. The moving average filter in **Figure 4C** also has this smoothed property. Though the three time series in **Figures 4A–C** have similar shapes, persistence diagrams, and persistence landscapes, they vary in one important feature: the pause.

In **Figure 10A** the points in the sliding window embedding that correspond to frames where the worm is performing a specific behaviors are highlighted. The points corresponding to the pause behavior deviate from the path of points corresponding to crawling backwards. This deviation is small compared to the noisiness of the original time series, so the pause deviation does

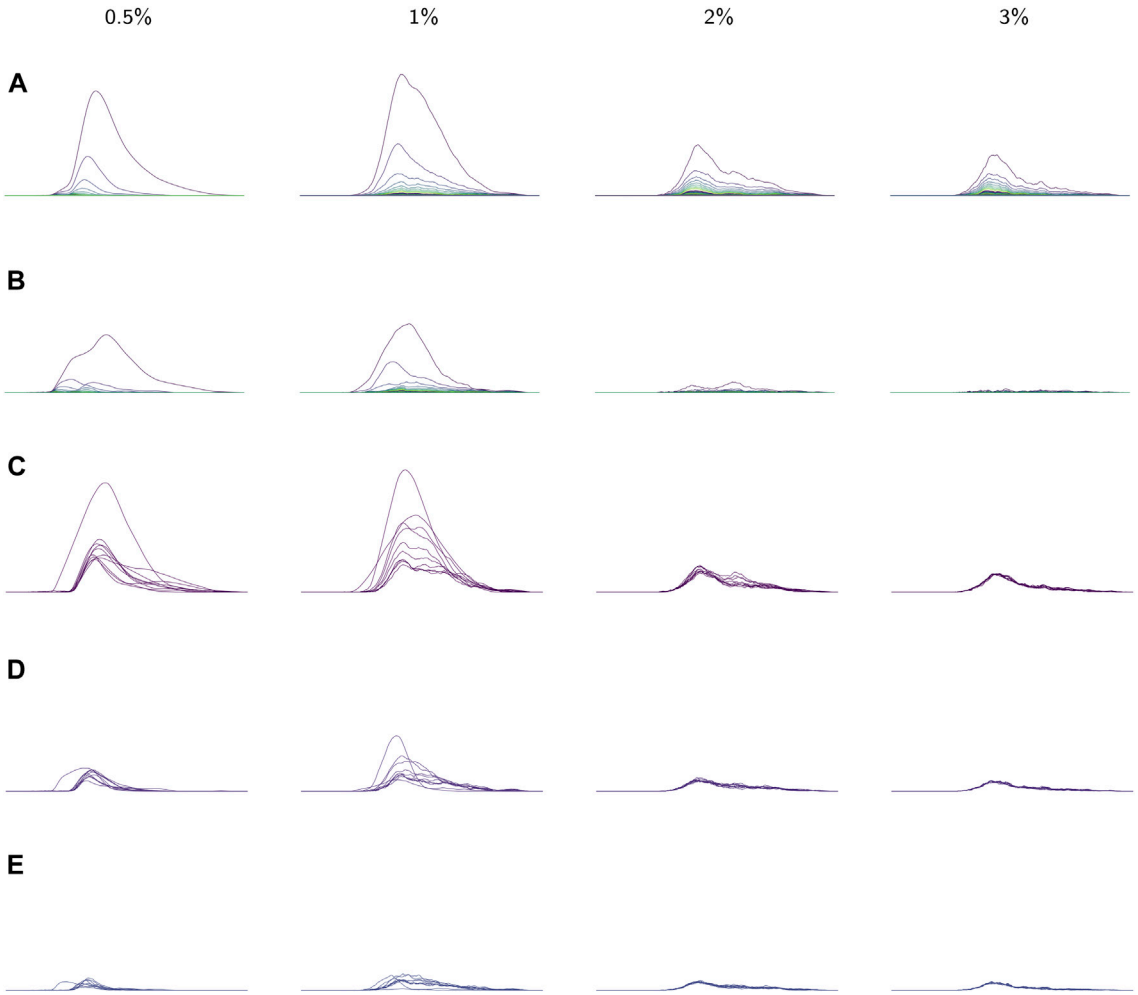


FIGURE 8 | (A) The average landscapes for each environmental viscosity, distinguished by percentage of methylcellulose present. The averages here are taken over the average persistence landscape of videos (i.e. individuals) in a given viscosity class. **(B)** Standard deviations of each coordinate in the average persistence landscapes for each class. **(C)** The first landscapes of each sample landscape, organized by class. These concurrently plotted first landscapes show the variation in the samples for each class. **(D,E)** The second and third landscapes, respectively, for each sample according to its class. All plots share the same *x*-axis; the groups of plots in **(A)**, **(B)**, and **(C–E)** each have their own *y*-axis scale.

not create a large topological feature in the graph of the original time series. This is reflected in the persistence diagrams and landscapes of **Figure 4** as well; the original time series diagram and landscape **Figure 4A** show only three significant topological features, while the diagram and landscape of the sliding window embedding **Figure 4B** and moving average filter **Figure 4C** show four.

The sliding window embedding and moving average filter both smooth the input data and detect the pause behavior, but they are qualitatively different. One piece of geometric information that the sliding window embedding retains that the moving average filter does not is the direction of the time series. Consider plotting a time series and the corresponding reverse-chronological-order time series in the same ambient space. The points of the original time series would align exactly with its reverse, so the two corresponding point clouds are the same. This is also true of a moving average filter on that time series. The sliding window embedding, however, can have distinct point clouds.

Retaining the direction of time in a time series is particularly important for data that has certain types of symmetry. A natural occurrence of such data is the sine wave data in Example 2.7. Because the data in this time series follows a path and then backtracks along that path, it never produces a loop with any significant persistence. There is no loop in the moving average filter of the data, either.

3.1.1 Interpretability: Mapping Persistence Features to (Synthetic) Behaviors

We computed representative cycles for persistent homology classes for each of the longest-persisting topological features in the sliding window embedding using Dionysus (Morozov, 2017). These are shown in **Figure 10B**. The homology classes that correspond to each of these representative

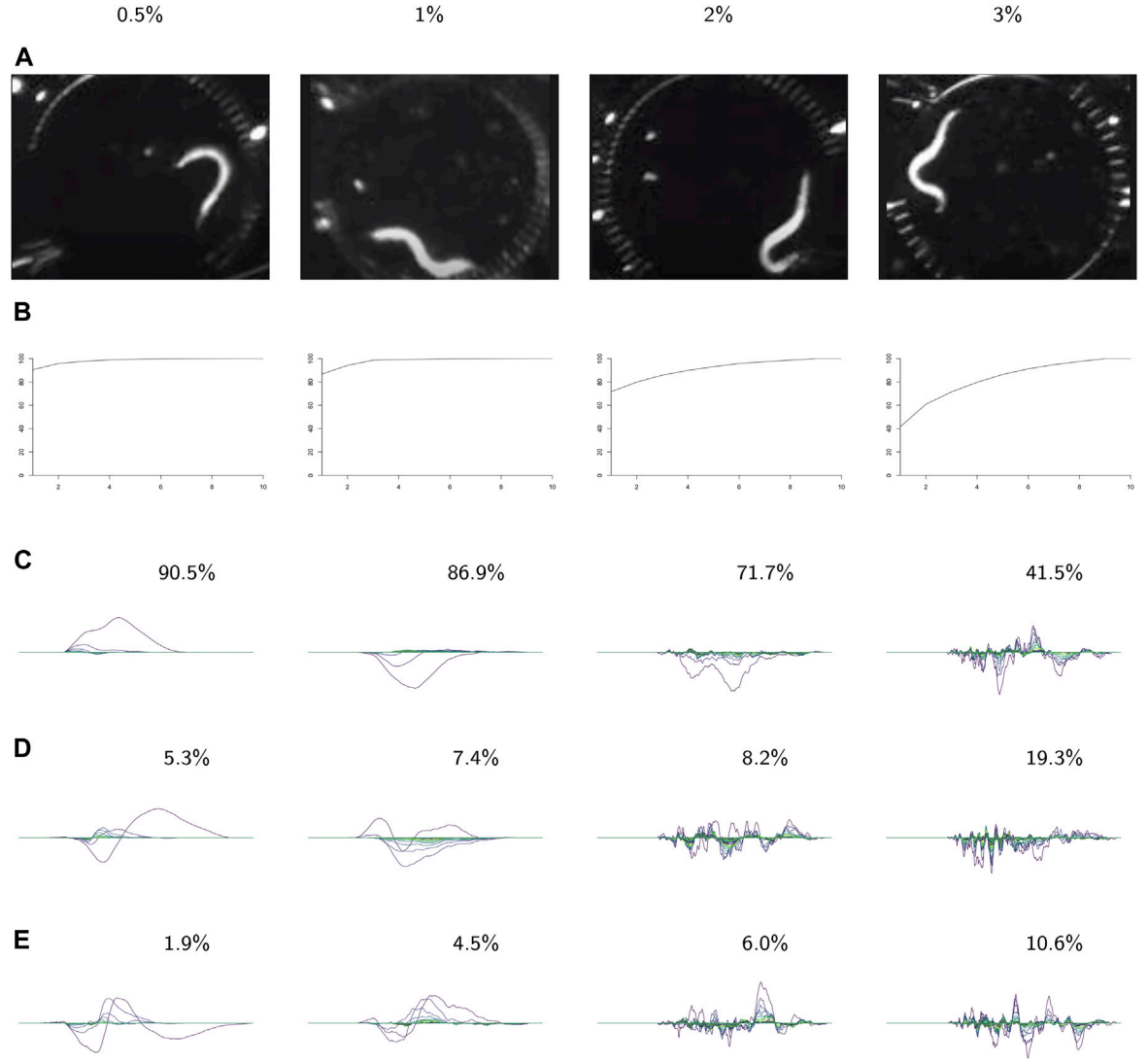


FIGURE 9 | PCA on the videos in each class. For each class: **(A)** video frames showing representative postures. **(B)** The cumulative variance of the first *n* principal components. **(C–E)** The first three principal components, labeled with the percent of the variance described by that component. These results show increasing complexity of shape and behavior as the environment becomes more viscous.

cycles are highlighted in **Figure 10C**. We remark that instead of the representative cycles produced by Dionysus one may want to use (approximate) shortest cycle representatives (Jeff Erickson, 2012; Dey et al., 2018; Obayashi, 2018; Day et al., 2019).

One of the benefits of using persistence for behavior analysis is that these representative cycles give a direct translation from persistence back into *C. elegans* behavior. Each point in the cycle corresponds to *l* poses and these points have a defined sequence in the cycle. The cycle lacks a direction (which way is forward in time vs which way is backward) but in many cases a direction can be inferred by subsets of the sequence that correspond to contiguous sequences in the original time series. Given all this data and a way to combine *l* poses into one "average" pose, we can construct synthetic, periodic behavior data from representative cycles. An example of frames from such a video is shown in **Figure 10D** and the corresponding video is available in the **Supplementary Materials**.

TABLE 1 | Normalized pairwise distances between average persistence landscapes of each class, where distance is Euclidean distance between vectors in $\mathbb{R}^{255969}$ and the normalization is such that the average distance to the origin is 1.

	0.5%	1%	2%	3%
origin	1.1873106	1.5934976	0.6777196	0.5414722
0.5%		0.8330355	0.8380992	0.8809327
1%			0.9972502	1.1255496
2%				0.1758501

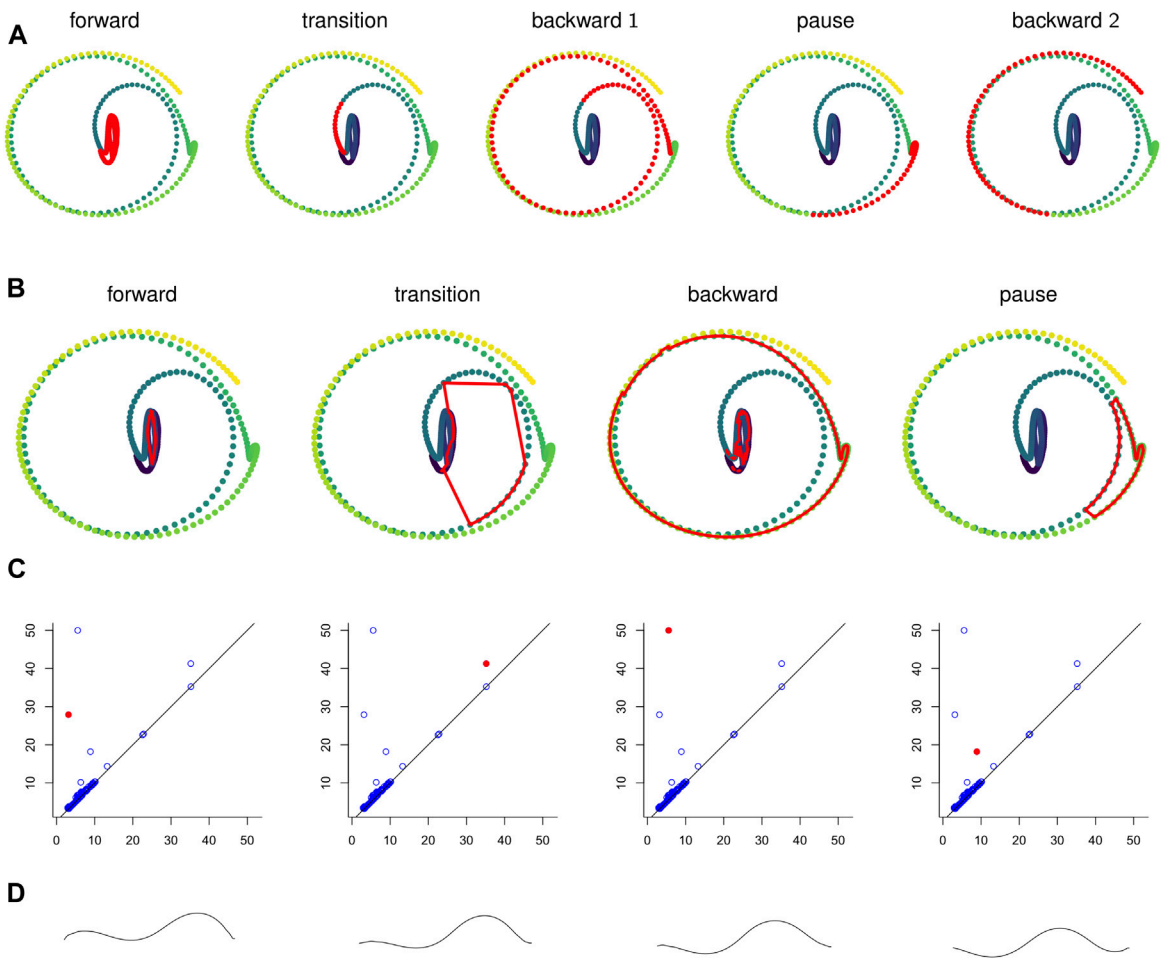


FIGURE 10 | (A) Points in the sliding window embedding τ^{20} that correspond to each of the labeled behaviors are highlighted. **(B)** The representative cycles with longest persistence from automated persistence software correspond to specific behaviors. **(C)** The persistence diagram with the homology class corresponding to the above cycle representative highlighted. **(D)** Still frames from a looping video of forward crawling data. The full video is in **Supplementary Materials**. This synthetic data was constructed from the forward representative cycle in **(B)**.

3.2 Experimental Results

We present our analysis of an experiment where *C. elegans* are submerged in solutions with varying viscosities. The viscosity of the solution is correlated with how much methylcellulose is added and experimental conditions are labeled with their methylcellulose content, usually in order from low to high methylcellulose and viscosity.

Average persistence landscapes for each class are shown in **Figure 8A**. The lower viscosity conditions allow for larger depth one landscapes (λ_1) but have relatively few non-zero higher-depth landscapes, which means there are a smaller number of larger loops detected in the sliding window embeddings. This indicates that in lower-viscosity environments, *C. elegans* exhibit behaviors of higher amplitude but either

TABLE 2 | Classification statistics. (A) Permutation test results. (B) Confusion matrix for one instance of SVM with 10-fold cross validation.

A

	1%	2%	3%
0.5%	0.0077	0.0001	0.0000
1%		0.0000	0.0000
2%			0.0000

B

	0.5%	1%	2%	3%
0.5%	10	0	0	0
1%	0	9	0	0
2%	0	1	9	0
3%	0	0	1	10

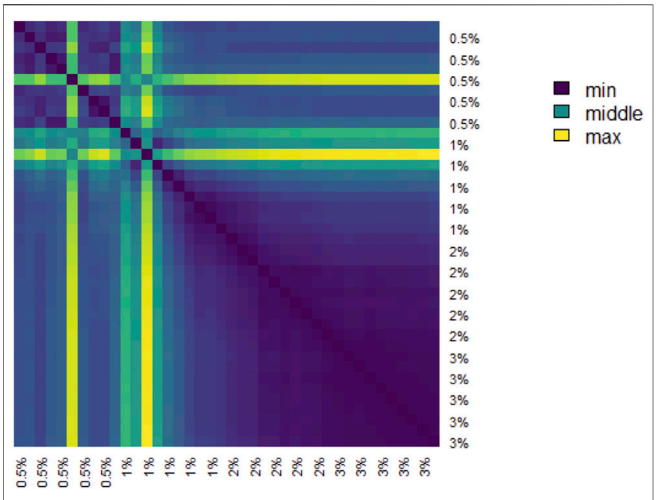


FIGURE 11 | Heatmap of the distance matrix of average persistence landscapes of samples.

demonstrate fewer distinct behaviors or have much less variation between repetitions of behaviors. Conversely, the high-viscosity classes show many more cycles in the sliding window embeddings, with each cycle being small compared to the cycles found in the low-viscosity environments. These observations suggest that at high-viscosity, behaviors do not involve large changes in posture and are more varied. From observing the raw video data, it is apparent that in higher-viscosity environments *C. elegans* can make smaller, tighter body bends, which is consistent with these results.

We also observed that as viscosity increases, the support of the persistence landscapes stretches further to the right and cycles are born at higher radius values. The worms seemed to exhibit less varied behaviors in lower-viscosity environments, so perhaps in such environments they continued "retracing their steps" through the sliding window embedding space which resulted in more densely sampled curves and thus homology classes formed at lower radii.

The pairwise distances between landscapes for each sample are visualized in **Figure 11**. The normalized pairwise distances between the average persistence landscapes for each class are shown in **Table 1**. We include the origin—the zero persistence landscape, i.e. the 0 vector—in these distance computations to complete the normalization. Normalization is such that the average distances between each class and the origin is 1. Multidimensional scaling on these distances visualizes the similarities between samples and classes, respectively, and are shown in **Figure 6**. From the raw distances and the multidimensional scaling of the distances, we can see that the high-viscosity classes (2% and 3% methylcellulose) are closest together and that this pair, the 0.5% class, and the 1% class are roughly equidistant from one another.

Principal component analysis on the average persistence landscapes for each sample gives the graphs in **Figure 7**. In **Figure 7A**, the projections of the average persistence landscapes of the samples onto the first two principal components are given by hollow symbols and projections of the average persistence landscapes of the classes are given by solid symbols. Here we see results similar to those from the multidimensional scaling in **Figure 6**: the low-viscosity class landscapes are far from each other and the high-viscosity classes, while the high-viscosity class landscapes are quite close. We can also see some of the variance within classes. The low-viscosity classes have much more variability than the high-viscosity classes, with the highest-viscosity class, 3% methylcellulose, having very little variation in these first two principal components.

These conclusions about variation in each of the classes are supported by the standard deviations of each coordinate in the average (discrete) persistence landscapes of each class. In **Figure 8B** the standard deviations of each coordinate are graphed as sequences of functions so that the standard deviations can be easily matched up with their corresponding locations on the average persistence landscapes. We conclude that there is little variation in the 3% class, slightly more in the 2% class, and much more in the 0.5% and 1% classes. The 1% class showed more variation in higher-depth landscapes than the 0.5% class, suggesting that *C. elegans* can produce slightly more complex behaviors in a slightly higher viscosity environments. The variances of the 0.5% and 1% classes also exhibit a distinct pattern; the 0.5% samples varied more toward the lower radius parameters (the left side of the graph), whereas the 1% samples varied more toward higher (more to the right) radius parameters.

In the following analysis we study the complexity of behavior expressed in each class. **Figure 9** shows results from using PCA on the videos in each class. The viscosity of the environment is negatively correlated with the percent of variance explained by the first principal component, which suggests that behaviors in low-viscosity environments are simpler than those in high-viscosity environments. Viscosity appears to be correlated with the number of nonzero landscapes, which also suggests that high-viscosity environments allow for more varied behaviors.

We conducted permutation tests between pairs of classes to determine how well average persistence landscapes can distinguish between samples from different classes. The *p*-values for these computations are shown in **Table 2A**. The permutation test gives strong evidence of statistical significance, i.e. that the topological summaries of samples from each class are significantly different.

We then used multiclass support vector machines (SVM) to build a classifier for the samples. The estimated accuracy of the classifier, computed by averaging accuracies across 20 instantiations of the multiclass SVM classifier, was 95.125%. This indicates that persistent homology is able to produce meaningful, distinguishing features from the *C. elegans* videos. A sample confusion matrix for one instance of SVM is shown in **Table 2B**. Finally, we used support vector regression to estimate the methylcellulose content in the environment for each sample. The results are plotted in **Figure 12**. There are two outliers on this graph which are estimated as having negative methylcellulose content. The two animals in these samples moved much more quickly than their peers, so we believe that the SVR is picking up on the strong negative correlation between viscosity of the

environment and speed, and based on these animals' fast speed, assigning a methylcellulose content that is so low that it is negative.

3.2.1 Cross Validation of Window Length

As was shown in Example 2.6, changes in the window length of the sliding window embedding can affect the conclusions drawn from our computational pipeline. In fact, sliding window embeddings have been critiqued for their need of this seemingly arbitrary parameter that can cause significant artifacts when dealing with volatile data (Lindquist et al., 2014). Our data is not particularly volatile since it is constrained by physical limitations of *C. elegans*, but we have still justified our choice of window length by conducting cross validation.

We assembled the results from running our computational pipeline with window lengths of 1, 10, 20, and 30. We found that different window lengths could be more useful for different tasks.

The permutation test and multiclass SVM results show that using a window length of 1 (i.e., not using a sliding window embedding) is the most predictive and that the smaller the window length, the better the accuracy. Window length 1 had the largest multiclass SVM accuracy at 95.500% compared to accuracies 91.750%, 83.375%, and 79.125% for window lengths 10, 20, and 30, respectively. The permutation test results were less definitive, with window length 1 showing slightly more separation between the 0.5% and 1% methylcellulose classes but all window lengths giving strong results. The runtimes of each computation differed significantly, with window length one data running for just over 5 min while window length 30 data took 90 min. This indicates that for predictive purposes and doing statistics on large data sets, using persistence on as close to the raw data as possible is best, but sliding window embeddings can still be useful if they are desirable for other reasons.

These statistical results contrast with the ease of visualization and interpretability of the analyses using the different window lengths. PCA projections of the cross validation data show much more differentiation between classes when the window length is 10, 20, or 30 compared to when it is 1. This is presumably because the first two principal components do not explain as much of the variation in the raw data, so separation between classes takes more principal components to describe. Meanwhile, sliding window embeddings consolidate variation in the samples into fewer principal components, so there are fewer significant principal components to visualize and interpret in terms of the original application. Essentially, sliding window embeddings give a slightly simplified but still predictive representation of the raw data.

We can also see from **Figure 4** that PCA projections of behavior data are easier to interpret when we use a sliding window embedding than when we look at just the raw time series. Recall that one of the behavioral features from the data—the pause—was not detectable over noise when looking at the PCA projection of the original time series but could be identified in both the PCA projection and the persistence landscape of the sliding window embedding. Because we were able to identify the topological feature we were also able to compute a corresponding representative cycle, which in turn allowed an estimate of the locomotion corresponding to the pause behavior to be constructed. Identification and construction of the corresponding locomotion of the pause behavior would have been more difficult using the original time series.

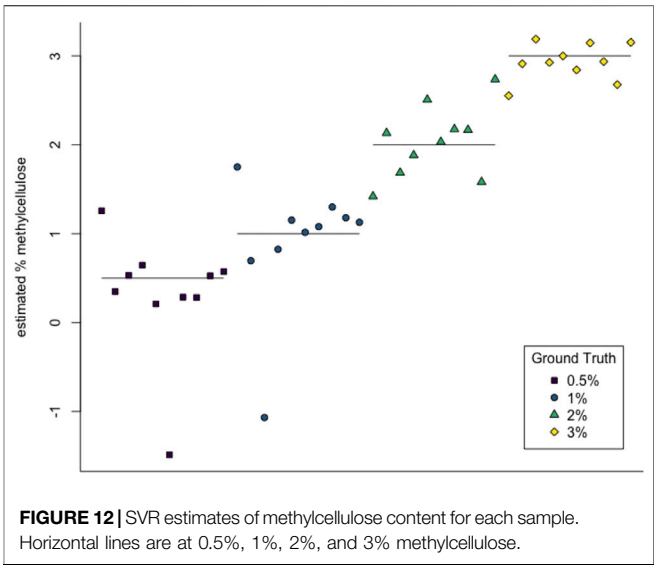


FIGURE 12 | SVR estimates of methylcellulose content for each sample. Horizontal lines are at 0.5%, 1%, 2%, and 3% methylcellulose.

3.2.2 Validation Results

We conducted permutation tests and applied multiclass SVM for data from two simple behavior quantification techniques which are described in **Section 2.5**. Results of the permutation test are listed in **Table 3**. For the method based on averages of 2-norms of consecutive frames of vector angles the cross validation error was 12.5% and training error was 7.5%. For the method based on standard deviations of vector angles the cross validation error was 70% and training error was 35%. Though more computationally taxing, persistence and sliding window embeddings produced much more accurate results than either of these simpler techniques.

Computations for the null model involve permuting the original time series of each sample and running our sliding window embedding and persistence techniques on that permuted data. Analysis of the null model showed that temporal information is necessary for our techniques to give good accuracy differentiating between samples taken in differing viscosity environments.

For the null model, the average error across 20 iterations of 10-fold cross validation on SVMs was 51.625%. The permutation tests showed that we could distinguish the 0.5% methylcellulose class without temporal information, but could not do as well differentiating between the three higher viscosity classes. It seems that the distribution of poses in the 0.5% methylcellulose class was different enough from the other classes to produce noticeably larger topological features that resulted in larger landscapes. This is probably because the lowest viscosity class had more extreme poses and thus the diameter of the space of poses for that class was significantly larger.

TABLE 3 | Permutation test results on simpler techniques which use rough measures of each worm's average movement speed and variation in posture. (A) Speed: averages of 2-norms of the difference between consecutive frames of vector angles. (B) Posture change: standard deviations of vectors angles.

A

	1%	2%	3%
0.5%	0.3497	0.0000	0.0000
1%		0.0492	0.0480
2%			0.1026

B

	1%	2%	3%
0.5%	0.0002	0.0000	0.0000
1%		0.9961	0.0625
2%			0.0411

4 DISCUSSION

We have demonstrated that persistent homology is a viable technique for studying *C. elegans* behavior and provides useful interpretations and visualizations. Our method consists of constructing sliding window embeddings of time series of piecewise linear *C. elegans* skeletons and using degree one persistent homology to create topological summaries for each patch of each video. These topological summaries, called persistence landscapes, are averaged over patches to produce a single average persistence for each video. These average persistence landscapes are our topological summary statistics and they are the statistics to which we apply further statistical analysis and machine learning techniques, such as principal component analysis, multidimensional scaling, permutation tests, and multiclass support vector machines. As far as we are aware, this is the first application of persistent homology to *C. elegans* behavior data.

Our analysis showed that persistence is able to detect variability in *C. elegans* behavior data, but also that it can provide interpretable conclusions and useful visualizations. The potential of persistence for interpretability and visualization results is demonstrated in the case study of **Section 3.1**, where topological features were connected directly to behavioral features and persistence was used to create synthetic behavior data corresponding to stereotyped behaviors such as forward crawling. Our analysis of experimental data shows that persistent homology can detect the variation of behavior induced by changes in the viscosity of the environment. It also suggests that persistence can measure complexity of behavior and that sliding window embeddings with low window lengths can be more predictive while sliding window embeddings with higher window lengths can be more useful for producing clear and interpretable visualizations, including video of synthetic data.

Persistent homology produces powerful summaries of the "shape" of data. However, using persistent homology in a way that is interpretable by experimentalists is a challenge and a topic of current research. We take a step in this direction by using representative cycles of the most persistent features of a sliding window embedding to produce synthetic videos of characteristic cyclic behaviors. At this time, there does not exist a straightforward way to similarly interpret our composite summaries, the average persistence landscapes. However, there is work in progress toward this goal (Bubenik and Wagner, 2018), and our pipeline would be able to incorporate such advances.

Our analysis has implications for future experimental design. We observed that low-viscosity environments allow for the detection of variation between samples, while high-viscosity environments may allow animals to perform more complex and varying behaviors. Tuning the viscosity of the environment for an experiment or performing experiments in multiple fluid environments with varying viscosities could allow for more easily assessing results regarding variations within populations or variations in behavior.

An extension to this experiment that could provide more validation for our techniques would be to include samples from two new environmental conditions: buffer, which would correspond to 0% methylcellulose and a lower viscosity than appears in our current data; and agar, which provides a solid surface for the worms to crawl on and surrounding air as opposed to an aqueous environment to be submerged and swim in. We would expect the new buffer class to allow for only fast, simple behaviors in line with the experiments already done, and the agar environment to allow more complex behaviors in the subjects.

The method that we have developed for applying topological data analysis to *C. elegans* locomotion data will facilitate the future study of biological phonemena such as aging. In particular, our rich quantitative summary of locomotion suggests that we may be able to measure not just lifespan, but "healthspan," the length of time an individual is healthy and physically capable. Many therapies and medicines for humans and other organisms have a goal of expanding healthspan, and therefore require a detailed measure of the ability to locomote, such as those provided by our methods.

DATA AVAILABILITY STATEMENT

Publicly available datasets were analyzed in this study. This data can be found here: https://www.youtube.com/playlist?list=PL5pzQyEKVlEjcmBWn9IVFivLJ4nqKKWC8.

AUTHOR CONTRIBUTIONS

KB collected and preprocessed the data under the guidance of HL. AT, AE, and IH analyzed the data under the guidance of PB. AT, HL, and PB regularly discussed the direction of the project from both biological and mathematical points of view.

FUNDING

This research was partially supported by the Southeast Center for Mathematics and Biology, an NSF-Simons Research Center for Mathematics of Complex Biological Systems, under National Science Foundation Grant No. DMS-1764406 and Simons Foundation Grant No. 594594. This material is based upon work supported by, or in part by, the Army Research Laboratory and the Army Research Office under contract/grant number W911NF-18-1-0307. Research reported in this publication was supported in part by the National Institutes of Health under award numbers R01NS096581, R01NS115484, and R01AG056436. The collection of the data used in this research was partially supported by the National Institutes of Health's Ruth L. Kirschstein NRSA award 1F31GM123662.

ACKNOWLEDGMENTS

The authors would like to thank the referees whose many comments considerably improved our manuscript. AT would also like to thank Kim Le for helpful conversations.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frai.2021.668395/full#supplementary-material

Supplementary Video 1 | Synthetic *C. elegans* behavior data of forward crawling constructed using data from **Section 3.1**. The data is constructed using a representative cycle from a persistent homology class in a sliding window embedding of real behavior data.

REFERENCES

Backholm, M., Kasper, A. K. S., Schulman, R. D., Ryu, W. S., and Dalnoki-Veress, K. (2015). "The Effects of Viscosity on the Undulatory Swimming Dynamics of *C. elegans*," in *Physics of Fluids*, 091901. 10897666. doi:10.1063/1.4931795

Berman, G. J., Bialek, W., and Shaevitz, J. W. (2016). "Predictability and Hierarchy in Drosophila Behavior," in *Proceedings of the National Academy of Sciences of the United States of America*, 11943–11948. 10916490. doi:10.1073/pnas.1607601113 url: https://pubmed.ncbi.nlm.nih.gov/27702892/.

Brown, A. E. X., Brown, E. I., Yemini, L. J., Jucikas, T., and Schafer, W. R. (2013). "A Dictionary of Behavioral Motifs Reveals Clusters of Genes Affecting *Caenorhabditis elegans* Locomotion," in *Proceedings of the Na- Tional Academy of Sciences of the United States of America*, 791–796. 00278424. doi:10.1073/pnas.1211447110 url: https://pubmed.ncbi.nlm.nih.gov/23267063/.

Bubenik, P. (2015). "Statistical Topological Data Analysis Using Persistence Landscapes," in *Journal of Machine Learning Research*, 77–102. 15337928. url: http://jmlr.org/papers/v16/bubenik15a.html.

Bubenik, P., and Wagner, A. (2018). "A Topological Heatmap," in preparation.

Chung, K., Zhan, M., Srinivasan, J., Sternberg, P. W., Gong, E., Schroeder, F. C., et al. (2011). "Microfluidic Chamber Arrays for Whole-Organism Behavior-Based Chemical Screening," in *Lab on a Chip*, 3689–3697. 14730189. doi:10.1039/c1lc20400aurl: https://pubmed.ncbi.nlm.nih.gov/21935539/.

Dequéant, M.-L., Ahnert, S., Edelsbrunner, H., Fink, T. M. A., Glynn, E. F., Hattem, G., et al. (2008). "Comparison of Pattern Detection Methods in Microarray Time Series of the Segmentation Clock," in *PLoS ONE* Editor R Khanin, e2856. 1932-6203. doi:10.1371/journal.pone.0002856

Dey, T. K., Hou, T., and Mandal, S. (2019). "Persistent 1-Cycles: Definition, Computation, and its Application," in *Computational Topology in Image Con-Text*. Editors R. Marfil, M. Calderón, F. del Río, P. Real, and A. Bandera (Cham: Springer International Publishing), 123–136. 978-3-030-10828-1.

Dey, T. K., Li, T., and Wang, Y. (2018). "Efficient Algorithms for Computing a Minimal Homology Basis," in *Lecture Notes in Computer Science (In- Cluding Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Springer-Verlag), 376–398. 9783319774039. doi:10.1007/978-3-319-77404-6_28

Firas, A. K., and Elizabeth, M. (2015). "Chatter Detection in Turning Using Persistent Homology," in *Mechanical Systems and Signal Processing*, 527–541. 10961216. doi:10.1016/j.ymssp.2015.09.046

Helm, A., Blevins, A. S., and Bassett, D. S. (2020). *The Growing Topology of the* C. elegans *Connectome*. arXiv: 2101.00065. url: http://arxiv.org/abs/2101.00065.

Husson, S. J., Wagner, S. C., Schmitt, C., and Gottschalk, A. (2012). *Keeping Track of Worm Trackers*. doi:10.1895/wormbook.1.156.1url: https://pubmed.ncbi.nlm.nih.gov/23436808/.

Jeff Erickson, J. (2012). "Combinatorial Optimization of Cycles and Bases," in *Advances in Applied and Computational Topology* (Providence, RI: Proc. Sympos. Appl. Math. Amer. Math. Soc.), 195–228. doi:10.1090/psapm/070/591

Lindquist, M. A., Xu, Y., Nebel, M. B., and Caffo, B. S. (20142014). "Evaluating Dynamic Bivariate Correlations in Resting-State fMRI: A Comparison Study and a New Approach," in *NeuroImage*, 531–546. 10959572. doi:10.1016/j.neuroimage.2014.06.052

Liu, M., Sharma, A. K., Shaevitz, J. W., and Leifer, A. M. (2018). "Temporal Processing and Context Dependency in caenorhabditis Elegans Response to Mechanosensation," in *eLife*. 2050084X. doi:10.7554/eLife.36419url: https://pubmed.ncbi.nlm.nih.gov/29943731/.

Lütgehetmann, D., Govc, D., Smith, J. P., and Levi, R. (2020). "Computing Persistent Homology of Directed Flag Complexes," in *Algorithms*, 19. 1999-4893. doi:10.3390/a13010019 url: https://www. mdpi.com/1999-4893/13/1/19.

Mileyko, Y., Mukherjee, S., and Harer, J. (2011). "Probability Measures on the Space of Persistence Diagrams," in *Inverse Problems*, 124007. 02665611. doi:10.1088/0266-5611/27/12/124007url: http://stacks.iop.org/0266-5611/27/i=12/a=124007?key=crossref. 95e8c511fc5be094303f6bde8cb2bafd.

Morozov, D. (2017). *Dionysus*. url: https ://www.mrzv.org/software/dionysus/.

Obayashi, I. (2018). "Volume-Optimal Cycle: Tightest Representative Cycle of a Generator in Persistent Homology," in *SIAM Journal on Applied Algebra and Geometry 2*, 508–534. 24706566. doi:10.1137/17M1159439url: http://www.siam.org/journals/siaga/2-4/M115943.html.

Otsu, N. (1979). "A Threshold Selection Method from gray-level Histograms," in *IEEE Transactions on Systems, Man, and Cybernetics*, 62–66. doi:10.1109/tsmc.1979.4310076

Perea, J. A., Deckard, A., Haase, S. B., and Harer, J. (2015). "SW1PerS: Sliding Windows and 1-persistence Scoring; Discovering Periodicity in Gene Expression Time Series Data," in *BMC Bioinformatics*, 257. 1471-2105. doi:10.1186/s12859-015-0645-6 url: http://bmcbioinformatics. biomedcentral.com/articles/10.1186/s12859-015-0645-6%20http://www.ncbi.nlm.nih.gov/pubmed/26277424%20 http://www.pubmedcentral. nih.gov/articlerender.fcgi?artid=PMC4537550.

Perea, J. A., and Harer, J. (2015). "Sliding Windows and Persistence: An Application of Topological Methods to Signal Analysis," in *Foundations of Computational Mathematics*, 799–838. doi:10.1007/s10208-014-9206-z url: https://link.springer.com/article/10. 1007/s10208 - 014 - 9206 - z % 20http ://link . springer . com/10 . 1007/s10208-014-9206-z.

Perea, J. A. (2016). "Persistent Homology of Toroidal Sliding Window Embeddings," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings* (Institute of Electrical and Electronics Engineers Inc.), 6435–6439. 9781479999880. doi:10.1109/ICASSP.2016.7472916

Petri, G., Scolamiero, M., Donato, I., and Vaccarino, F. (2013). "Topological Strata of Weighted Complex Networks," in *PLoS ONE*, e66506. 19326203. doi:10.1371/journal.pone.0066506 www.plosone.org.

Porto, D. A., Giblin, J., Zhao, Y., and Lu, H. (2019). "Reverse-Correlation Analysis of the Mechanosensation Circuit and Behavior in *C. elegans* Reveals Temporal and Spatial Encoding," in *Scientific Reports*, 1. 20452322. doi:10.1038/s41598-019-41349-0 url: https://pubmed.ncbi. nlm.nih.gov/30914655/.

Sizemore, A. E., Phillips-Cremins, J. E., Ghrist, R., and Bassett, D. S. (2019). "The Importance of the Whole: Topological Data Analysis for the Network Neuroscientist," in *Network Neuroscience*, 656–673. 24721751. doi:10.1162/netn_a_00073url: https://pubmed.ncbi.nlm.nih.gov/31410372/.

Stephens, G. J., Johnson-Kerner, B., Bialek, W., and Ryu, W. S. (2008). "Dimensionality and Dynamics in the Behavior of *C. elegans*," in *PLoS Computational Biology*. Editor O. Sporns, e1000028. 1553734X. doi:10.1371/journal.pcbi.1000028url: https://pubmed.ncbi.nlm.nih.gov/18389066/.

Stirman, J. N., Crane, M. M., Husson, S. J., Wabnig, S., Schultheis, C., Gottschalk, A., et al. (2011). "Real-time Multimodal Optical Control of Neurons and Muscles in Freely Behaving *Caenorhabditis elegans*," in *Nature Methods*, 153–158. 15487091. doi:10.1038/nmeth.1555url: https://pubmed.ncbi.nlm.nih.gov/21240278/.

Stolz, B. J., Harrington, H. A., and Porter, M. A. (2017). "Persistent Homology of Time-dependent Functional Networks Constructed from Coupled Time Series," in *Chaos*, 047410. 10541500. doi:10.1063/1.4978997

Swierczek, N. A., Giles, A. C., Rankin, C. H., and Kerr, R. A. (2011). "High-throughput Behavioral Analysis in *C. elegans*," in *Nature Methods*, 592–598. 15487091. doi:10.1038/nmeth.1625url: https://pubmed.ncbi.nlm.nih.gov/21642964/.

Tralie, C. J. (2016). "High Dimensional Geometry of Sliding Window Embeddings of Periodic Videos," in *Leibniz International Proceedings in Informat- Ics, LIPIcs*, 71.1–71.5. 18688969. doi:10.4230/LIPIcs.SoCG.2016.71

Tralie, C. J., and Perea, J. A. (2018). "(Quasi) Periodicity Quantification in Video Data, Using Topology," in *SIAM Journal on Imaging Sciences*, 1049–1077. 19364954. doi:10.1137/17M1150736url: https://github.com/ctralie/SlidingWindowVideoTDA.

Yemini, E., Jucikas, T., Grundy, L. J., Brown, A. E. X., and Schafer, W. R. (2013). "A Database of *Caenorhabditis elegans* Behavioral Phenotypes," in *Nature Methods*, 877–879. 15487091. doi:10.1038/nmeth.2560 url: pmc/articles/PMC3962822/%20/pmc/articles/PMC3962822/?report=abstract%20https://www.ncbi.nlm.nih. gov/pmc/articles/PMC3962822/.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Applied Mathematics and Statistics

ORIGINAL RESEARCH
published: 09 July 2021
doi: 10.3389/fams.2021.668046

Supervised Learning Using Homology Stable Rank Kernels

Jens Agerberg[1]*, *Ryan Ramanujam*[1,2], *Martina Scolamiero*[1†] and *Wojciech Chachólski*[1†]

[1]KTH Royal Institute of Technology, Mathematics Department, Stockholm, Sweden, [2]Department of Clinical Neuroscience, Karolinska Institutet, Stockholm, Sweden

Exciting recent developments in Topological Data Analysis have aimed at combining homology-based invariants with Machine Learning. In this article, we use hierarchical stabilization to bridge between persistence and kernel-based methods by introducing the so-called stable rank kernels. A fundamental property of the stable rank kernels is that they depend on metrics to compare persistence modules. We illustrate their use on artificial and real-world datasets and show that by varying the metric we can improve accuracy in classification tasks.

Keywords: topological data analysis, kernel methods, metrics, hierarchical stabilisation, persistent homology

OPEN ACCESS

Edited by:
Kathryn Hess,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Jonathan Scott,
Cleveland State University, United States
Ashleigh Thomas,
Georgia Institute of Technology, United States

****Correspondence:***
Jens Agerberg
jensag@kth.se

†These authors share last authorship

Specialty section:
This article was submitted to Mathematics of Computation and Data Science, a section of the journal Frontiers in Applied Mathematics and Statistics

Received: *15 February 2021*
Accepted: *02 June 2021*
Published: *09 July 2021*

Citation:
Agerberg J, Ramanujam R, Scolamiero M and Chachólski W (2021) Supervised Learning Using Homology Stable Rank Kernels. Front. Appl. Math. Stat. 7:668046. doi: 10.3389/fams.2021.668046

1 INTRODUCTION

Topological data analysis (TDA) is a framework for analyzing data which is mathematically well-founded and with roots in algebraic topology. Through the use of persistent homology, TDA proposes to analyze datasets, often being high-dimensional and unstructured, where each observation is an object encoding some notion of a distance. An example of such an object is a point cloud with Euclidean distance. A convenient way of encoding distance objects is *via* Vietoris-Rips complexes [1]. Persistent homology transforms these complexes into so-called persistence modules and diagrams [2, 3]. These modules and diagrams encode geometrical aspects of the distance objects captured by homology. We thus regard the obtained persistence diagrams as summaries encoding geometrical features of the considered distance objects. In recent applications, and in such varied fields as bioinformatics [4] and finance [5], it has been shown that these summaries encode valuable information which is often complementary to that derived from non-topological methods.

The discriminative information contained in the persistent homology summaries makes them interesting in the context of machine learning, for instance to serve as inputs in supervised learning problems. The space of persistence diagrams lacks however the structure of a Euclidean, or more generally Hilbert space, often required for the development of machine learning (ML) methods. Furthermore, for inference purposes we also need to be able to consider probability distributions over topological summaries. Since for persistence diagrams we only have Fréchet means at our disposal [6, 7] inference is difficult.

Our aim in this article is to present how persistent homology can be combined with machine learning algorithms within a framework called hierarchical stabilization [8–10]. We will use hierarchical stabilization to define new persistence-based kernels and illustrate them on artificial and real-world datasets. This article is based in part on some of the results described in Jens Agerberg's thesis [11].

Comparing and interpreting summaries produced by persistent homology should not just depend on their values but crucially also on the phenomena and the experiments that the considered datasets describe. Different phenomena might require different comparison criteria. It may not be optimal to

consider only Bottleneck or Wasserstein distances to compare outcomes of persistent homology of diverse datasets obtained from a variety of different experiments. The ability to choose distances that fit particular experiments is required. We do not, however, plan to use these distances to compare persistence modules directly. Instead, quite essentially we use a chosen distance to transform, *via* the hierarchical stabilization process, the space of persistence modules into the space $\mathcal{M}$ of (Lebesgue) measurable functions $[0,\infty) \to (-\infty,\infty)$ with the L_2 distance. Thus each distance d on persistence modules leads to a function denoted by the symbol $\widehat{\text{rank}}_d$: Persistence modules $\to \mathcal{M}$ called the stable rank. With the L_2 distance, $\mathcal{M}$ is a Hilbert space and its scalar product provides an effective tool to study geometrical aspects of the image of $\widehat{\text{rank}}_d$, particularly those captured by measuring length, angles, and exploring orthogonality. Thus kernel machine learning methods, which are based on scalar products, are effective tools in exploring such geometrical features. Illustrating the effectiveness of this strategy for modeling with stable ranks is the aim of this paper.

Since the stable rank is stable with respect to d, the kernel formed can be seen as a similarity measure associated to d, of practical importance in several machine learning methods. In this framework, supervised learning consists of identifying these distances d for which structural properties of the training data are reflected by the geometry of its image in $\mathcal{M}$ through the function $\widehat{\text{rank}}_d$. The strategy of looking for appropriate distances can only work if we are able to parametrize explicitly a rich subspace of distances on persistence modules. The hierarchical stabilization process builds on the discovery that such parametrization is possible using positive (Lebesgue) measurable functions $[0,\infty) \to (0,\infty)$ called densities. An organized search in the space of densities is beyond the scope of this article. The intention of this paper is to illustrate that by changing the density, the kernels can improve the accuracy in a supervised learning task.

Our method fits within the family of persistence based kernels [12], some of which also have parameters which can be optimized to fit a particular learning task [13]. However, a characteristic of our stable rank kernel is that it is defined on persistence modules rather than on persistence diagrams. A bar decomposition of the persistence modules is therefore useful but not essential for the definition of our kernel, which is readily generalisable to multi-parameter persistence.

2 MATERIALS AND METHODS

2.1 Homological Simplification: From Data to Persistence Modules

Recall that a distance on a set X is a function $d : X \times X \to [0,\infty)$ which is symmetric $d(x,y) = d(y,x)$ and reflexive $d(x,x) = 0$. It is a pseudometric if in addition it satisfies the triangular inequality $d(x,y) + d(y,z) \geq d(x,z)$. For example, by restricting a distance on the plane to a point cloud we obtain a finite distance space.

In this article we focus on data whose points are represented by finite distance spaces. This type of data is often the result of performing multiple measurements for each individual, representing these measurements as vectors, choosing a distance between the vectors, and representing each individual by a distance space. Encoding data points in this way reflects properties of the performed measurements accurately. That is an advantage but also a disadvantage as a lot of the complexity of the experiment is retained including possible noise, measurement inaccuracies, effects of external factors that might be irrelevant for the experiment but influence the measurements, etc. Because of this overwhelming complexity, to extract relevant information we need to simplify. Data analysis is a balancing act between simplifying, which amounts to ignoring some or often most of the information available, and retaining what might be meaningful for the particular task. In this article we study various simplifications based on homology.

The first step in extracting homology is to convert distance information into spatial information. We do that using so-called Vietoris-Rips complexes [1]. By definition the Vietoris-Rips complex $\text{VR}_\epsilon(X,d)$, at scale ϵ in $[0,\infty)$, is a simplicial complex whose simplices are given by the non-empty finite subsets $\sigma \subset X$ for which $d(x,y) \leq \epsilon$ for every x and y in σ. Vietoris-Rips complexes form an increasing filtration as $\text{VR}_\epsilon(X,d) \subset \text{VR}_\tau(X,d)$ when $\epsilon \leq \tau$. In the case X is finite, there is a finite sequence of parameters $0 \leq a_0 \leq \cdots \leq a_l$ such that $\text{VR}_\epsilon(X,d) \subset \text{VR}_\tau(X,d)$ may fail to be the equality only if $\epsilon < a_i \leq \tau$ for some i, i.e., the jumps in the Vietoris-Rips filtration can occur only when passing through some a_i. Such filtrations are called tame [8, 10].

The Vietoris-Rips filtration does not lose or add information about the distance space. It retains all the complexity of d. Thus, the purpose of this step is not to simplify, but rather to allow for the extraction of homology (see for example [14]). In this article we only consider reduced homology. The first step in extracting homology is to choose a field; for example, $\mathbb{F}_2$ with two elements. Homology in a given degree n, with coefficients in a chosen field F, converts a simplicial complex X into an F vector space $H_n(X)$. Homology is a functor which means that it also converts maps of simplicial complexes $f : X \to Y$ into linear functions $H_n(f) : H_n(X) \to H_n(Y)$ such that $H_n(\text{id}) = \text{id}$ and $H_n(gf) = H_n(g)H_n(f)$ for any composable maps f and g. Homology encodes certain geometric features of the simplicial complex, for example the dimension of $H_0(X)$ is one less than the number of connected components of X, as the considered homology is reduced.

By applying homology to Vietoris-Rips complexes, we obtain a vector space $H_n(\text{VR}_\epsilon(X,d))$ for every ϵ in $[0,\infty)$. By applying homology to the inclusions $\text{VR}_\epsilon(X,d) \subset \text{VR}_\tau(X,d)$, when $\epsilon \leq \tau$, we obtain linear functions $H_n(\text{VR}_\epsilon(X,d)) \to H_n(\text{VR}_\tau(X,d))$ (which may not be inclusions). These linear functions, for all $\epsilon \leq \tau$, form what is also called a persistence module [15]. Tameness of the Vietoris-Rips filtration implies tameness of the persistence module: there is a finite sequence of parameters $0 \leq a_0 \leq \cdots \leq a_l$ such that $H_n(\text{VR}_\epsilon(X,d)) \to H_n(\text{VR}_\tau(X,d))$ may fail to be an isomorphism only if $\epsilon < a_i \leq \tau$ for some i i.e., jumps occur only when passing through some a_i.

The described process of assigning a tame persistence module to a distance space is a simplification. This is because of a particularly simple structure theorem for tame persistence

modules [15, 16], which states that every tame persistence module is isomorphic to a direct sum of so-called bars. A bar, denoted by $b(s,e)$, is a tame persistence module determined by two real numbers $s<e$ in $[0,\infty)$, called the start and the end, such that: $b(s,e)_\epsilon$ is one dimensional in case $s\le\epsilon<e$ and 0 dimensional otherwise, and the linear function $b(s,e)_\epsilon \to b(s,e)_\tau$ is the identity for $s\le\epsilon\le\tau<e$. Tame persistence modules can therefore be parametrized by finite multisubsets [17] of $\Omega := \{(s,e)\in[0,\infty)^2 \mid s<e\}$. Such multisubsets are also called persistence diagrams. There exist several software implementations that compute persistence diagrams of distance spaces. Among them is Ripser [18] which we use for the persistent homology calculations presented in this paper.

In the rest of the article we explain and illustrate a framework for analyzing outcomes of persistence called hierarchical stabilization [8–10].

2.2 Hierarchical Stabilization: From Persistence Modules to Measurable Functions

The key ingredient in hierarchical stabilization is a choice of a pseudometric on persistence modules. It turns out that a pseudometric on persistence modules can be constructed for every action of the additive monoid of non negative reals $[0,\infty)$ on the poset of non negative reals $[0,\infty)$. Such an action is a function $C:[0,\infty)\times[0,\infty)\to[0,\infty)$ satisfying the following conditions $C(a,0)=a, C(C(a,\epsilon),\tau)=C(a,\epsilon+\tau)$, and $C(a,\epsilon)\le C(b,\tau)$ if $a\le b$ and $\epsilon\le\tau$. We refer to [8–10] for an explanation of how an action leads to a pseudometric. Here we recall how to construct a rich space of such actions.

We do that by associating actions to measurable functions with positive values $f:[0,\infty)\to(0,\infty)$ called densities. According to [8], a density leads to the following actions. One action $D_f:[0,\infty)\times[0,\infty)\to[0,\infty)$ is called of distance type and assigns to (a,ϵ) the unique number $D_f(a,\epsilon)$ for which $\int_a^{D_f(a,\epsilon)} f(x)dx=\epsilon$. Another action $S_f:[0,\infty)\times[0,\infty)\to[0,\infty)$ is called of shift type and is constructed as follows: choose y such that $a=\int_0^y f(x)dx$ and define $S_f(a,\epsilon):=\int_0^{y+\epsilon} f(x)dx$. For example, for the constant density with value 1, the two actions D_1 and S_1 coincide with the standard action $(a,\epsilon)\mapsto a+\epsilon$. We use the name the standard pseudometric to describe the pseudometric on persistence modules associated to this standard action. The standard pseudometric is equivalent to the Bottleneck distance [19] (see [10]).

Since densities form a rich space, then so do the pseudometrics on persistence modules they parametrize. By focusing on distance type actions defined by densities, in this paper we take advantage of the possibility of choosing a variety of pseudometrics on persistence modules. As already mentioned in the introduction, we are not going to use them to compare persistence modules directly. Instead we are going to use them to transform persistence modules into (Lebesgue) measurable functions $[0,\infty)\to(-\infty,\infty)$ called stable ranks. By definition, the stable rank $\widehat{\mathrm{rank}}_d(X)$ of a persistence module X, assigns to t in $[0,\infty)$ the following number: $\widehat{\mathrm{rank}}_d(X)(t) := \min\{\mathrm{rank}(Y) \mid d(Y,X)\le t\}$, where $\mathrm{rank}(Y)$ is the number of bars in a bar decomposition of Y. Thus $\widehat{\mathrm{rank}}_d(X)(t)$ is the minimal rank of the persistence modules that belong to the closed ball centered in X and of radius t with respect to the chosen pseudometric d. We refer to the stable rank associated to the standard pseudometric as standard stable rank. In the case the pseudometric d on persistence modules is associated with an action $C:[0,\infty)\times[0,\infty)\to[0,\infty)$, the stable rank $\widehat{\mathrm{rank}}_d(X)$ can be described directly in terms of C. Consider a bar decomposition $X\simeq\oplus_{i=0}^n b(s_i,e_i)$, then $\widehat{\mathrm{rank}}_d(X)(t)=\left|\{i \mid C(s_i,t)<e_i\}\right|$. Thus the values of the stable rank $\widehat{\mathrm{rank}}_d(X)$ are certain bar counts depending on C.

The key result states that the assignment $X\mapsto\widehat{\mathrm{rank}}_d(X)$ is a continuous function (in fact satisfying a certain Lipschitz condition [8]) with respect to the chosen pseudometric on persistence modules and the L_p metric on the space $\mathcal{M}$ of measurable functions $[0,\infty)\to(0,\infty)$. In this way we obtain a continuous function $\widehat{\mathrm{rank}}_d$: Persistence modules $\to\mathcal{M}$ into the space $\mathcal{M}$ in which geometrical, probabilistic and statistical methods are well developed. For example we can take averages and expected values of stable ranks assigned to various collections of persistence modules such as those given by homologies of Vietoris-Rips complexes obtained from a collection of distance spaces. In the case we choose the L_2 metric on $\mathcal{M}$, we can also use the Hilbert space structure on $\mathcal{M}$ and use the stable rank to construct a kernel on persistence modules. For persistence modules X and Y the stable rank kernel with respect to a pseudometric d is by definition given by $K_d(X,Y) := \int_0^\infty \widehat{\mathrm{rank}}_d(X)\widehat{\mathrm{rank}}_d(Y)dt$. The stable rank of a persistence module obtained as the reduced homology of Vietoris-Rips complexes is square integrable. Thus, for such persistence modules the stable rank kernel is finite.

In conjunction with various machine learning methods, the stable rank kernels for various densities can be used for classification purposes. Some of these possibilities are illustrated in the second half of this article where we use the stable rank kernels in conjunction with support vector machines (SVM).

2.3 Modeling: Determining Appropriate Distances on Persistence Modules

Supervised learning typically consists of fitting models to training data, and validating them on an appropriate testing set. Here, supervised persistence analysis takes the same form. We think about the function $\widehat{\mathrm{rank}}_d$: Persistence modules $\to\mathcal{M}$ as a model associated to a pseudometric d on persistence modules, for example the pseudometric given by the distance type action defined by a density. To fit such a model is to identify a parameter given by a pseudometric d (or a density leading to a pseudometric) for which structural properties of the data are reflected by the geometry of its image in $\mathcal{M}$ through the function $\widehat{\mathrm{rank}}_d$. Some of the aspects of this geometry are effectively encoded by the stable rank kernel.

There are two reasons why extracting information about persistence modules by exploring their stable ranks $\widehat{\mathrm{rank}}_d$ over varying pseudometrics d is effective. First, of practical importance for using kernel methods, is the fact that the stable rank $\widehat{\mathrm{rank}}_d$: Persistence modules $\to\mathcal{M}$ is not only a continuous function, it is

also continuous with respect to the changes of the pseudometric d or the density for which d is represented *via* either the action of distance type or the shift type [8]. Second, persistence modules are determined by their stable ranks: two tame persistence modules X and Y are isomorphic if and only if, for every density f, the functions $\widehat{\text{rank}}_{d_f}(X)$ and $\widehat{\text{rank}}_{d_f}(Y)$ coincide, where d_f is the pseudometric associated to the action D_f of distance type.

In the analysis presented in the next section we are going to use the following procedure for choosing a density. First we restrict ourselves to a simple family of densities: piecewise constant functions that are allowed to have at most four discontinuities, and the ratio of the maximum value divided by the minimum value is controlled. We sample 100 such densities and select the density corresponding to the optimal pseudometric by a procedure of cross-validation: first we split the dataset into a training set (60%), a validation set (20%) and a test set (20%). Next, new SVMs with the stable rank kernel corresponding to each of the 100 densities are fitted to the training set. The density leading to the best accuracy on the validation set is then selected. Last, the accuracy on the test set using the optimum density from the previous stage is evaluated and reported.

In our scheme to select an optimal density, we randomly sample piecewise constant functions. Both the family of densities on which the search is conducted and the search scheme can be varied. For example one can consider family of Gaussians as parametrized by their mean and standard deviation and proceed with a grid search for selecting optimal parameters. In our experience, in order to avoid overfitting, it is useful to restrict to functions that are constrained in their behavior.

3 RESULTS

In this section, two examples of analysis based on stable rank kernels are presented. In these examples, the objective is to correctly classify according to the categories, or labels, of each dataset. The focus of the first example is on certain finite subsets of the plane which are called plane figures. The plane figures considered have clear intuitive geometric meaning such as being a circle, rectangle, a triangle or an open path. Our aim is two-fold. First, we intend to illustrate that the stable rank kernel is applicable to the problem of differentiating between these geometrical shapes. Second, we will demonstrate how to enhance the discriminatory power by varying densities or by taking samplings of the data and averaging the associated stable ranks. We study the robustness of our method by altering the geometrical shapes with the addition of two types of noise and evaluating the accuracy of the stable rank kernels on these noisy figures.

The second example is concerned with activity monitoring data which is not simulated but consists of collected measurements. In PAMAP2 [20], seven subjects were asked to perform a number of physical activities (walking, ascending/descending stairs, etc.) while wearing the following sensors: a heart rate monitor and three units (placed on the arm, chest and ankle) containing an accelerometer, a magnetometer and a gyroscope. This resulted in a dataset of 28-dimensional time series labeled with subject and activity. In this example, we concentrate on distinguishing between data from ascending and descending stairs of different individuals.

3.1 Plane Figures: Dataset Generation

We consider four subsets of the plane: a circle, a rectangle, a triangle and an "M"-formed path: see the first row in **Figure 1** for the illustration. We refer to these subsets as shapes. The plane figures dataset is generated in the following way: 100 points are sampled uniformly from each of these subsets. Each point in this sampling is then perturbed by adding Gaussian noise (i.e., it is replaced by a point sampled from an isotropic Gaussian centered at the point). This is repeated 500 times for each shape. In this way we obtain 2000 subsets of 100 elements in the plane (500 for each shape). By considering the Euclidean distance to compare points on the plane, we can regard these subsets as finite distance spaces and call them plane figures. The collection of these 2000 plane figures of four classes is our first dataset. The elements in this dataset are labeled by the shapes. The objective of our analysis is to illustrate how to recover this labeling using the stable rank kernels.

3.2 Plane Figures: Analysis Based on Zero-th Homology

As a first exploratory step, for each plane figure we compute the Vietoris-Rips filtration, the corresponding 0-th homology persistence module and its stable rank with respect to the standard pseudometric. **Figure 2** shows four plane figures with different labels from our dataset (first row) and the corresponding stable ranks (second row). By plotting the average of all stable ranks for plane figures corresponding to each shape (**Figure 3**) we get an indication that indeed the 0-th homology analysis may not be very effective at distinguishing between plane figures labeled by different shapes. To confirm this, we formulate our problem in machine learning terms as classifying a given plane figure to the shape from which it was generated. The dataset is split into a training set (70%) and a test set (30%). A support vector machine (SVM) is fitted on the training set using the standard stable rank kernel and evaluated on the test set. We take advantage of the fact that we can generate the data and repeat the whole procedure 20 times. This results in a rather weak average classification accuracy of 35.0%. We suspect that the poor classification is due to the fact that the plane figures do not exhibit distinct clustering patterns. The stable rank, with respect to the standard pseudometric, is a fully discriminatory invariant of persistence modules resulting from the 0-th homology of Vietoris-Rips filtrations (this is a consequence of the fact that the stable rank is a certain bar count, see *Hierarchical Stabilization: From Persistence Modules to Measurable Functions*). Since this invariant completely describes our 0-th homology persistence modules, we do not expect that the classification accuracy can be noticeably improved by considering stable rank kernels associated with different pseudometrics.

3.3 Plane Figures: Analysis Based on First Homology

We repeat the same procedure for the 1-st homology persistence modules of the Vietoris-Rips filtrations. An indication that these

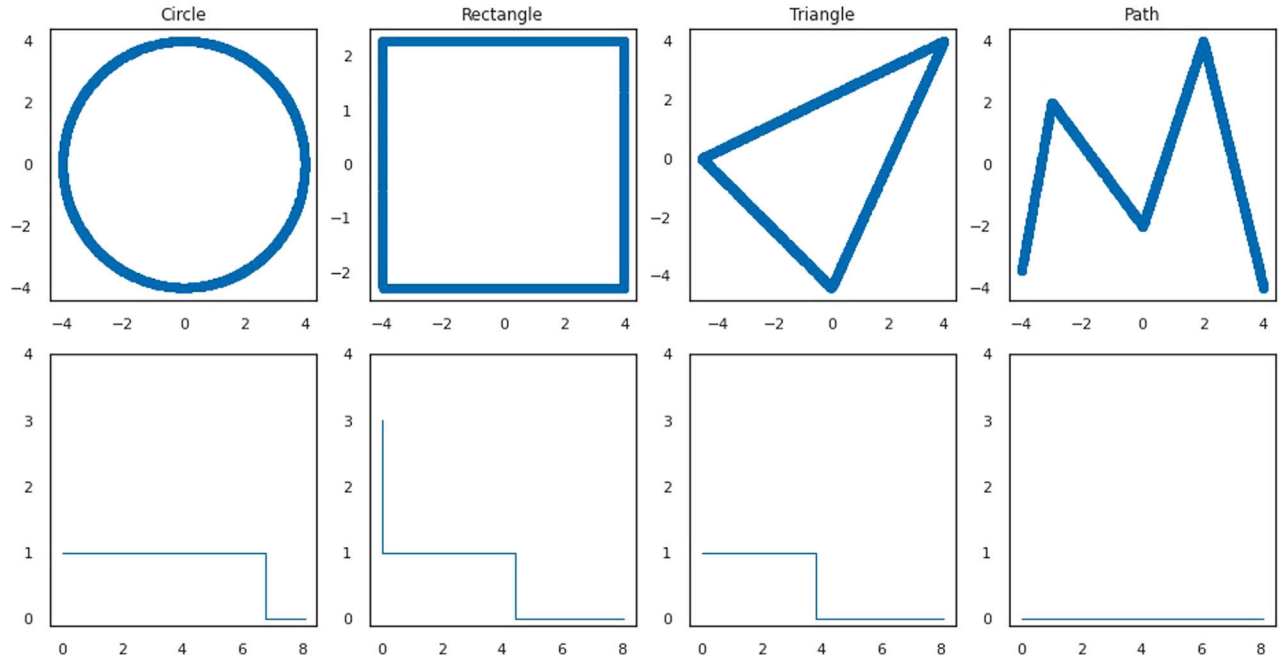


FIGURE 1 | First row: a dense point cloud sampling without noise from each of the shapes. **Second row:** the corresponding standard stable ranks of the persistence modules given by the 1-st homology of the Vietoris-Rips complexes of the distances given by the restriction of the Euclidean metric to the point clouds.

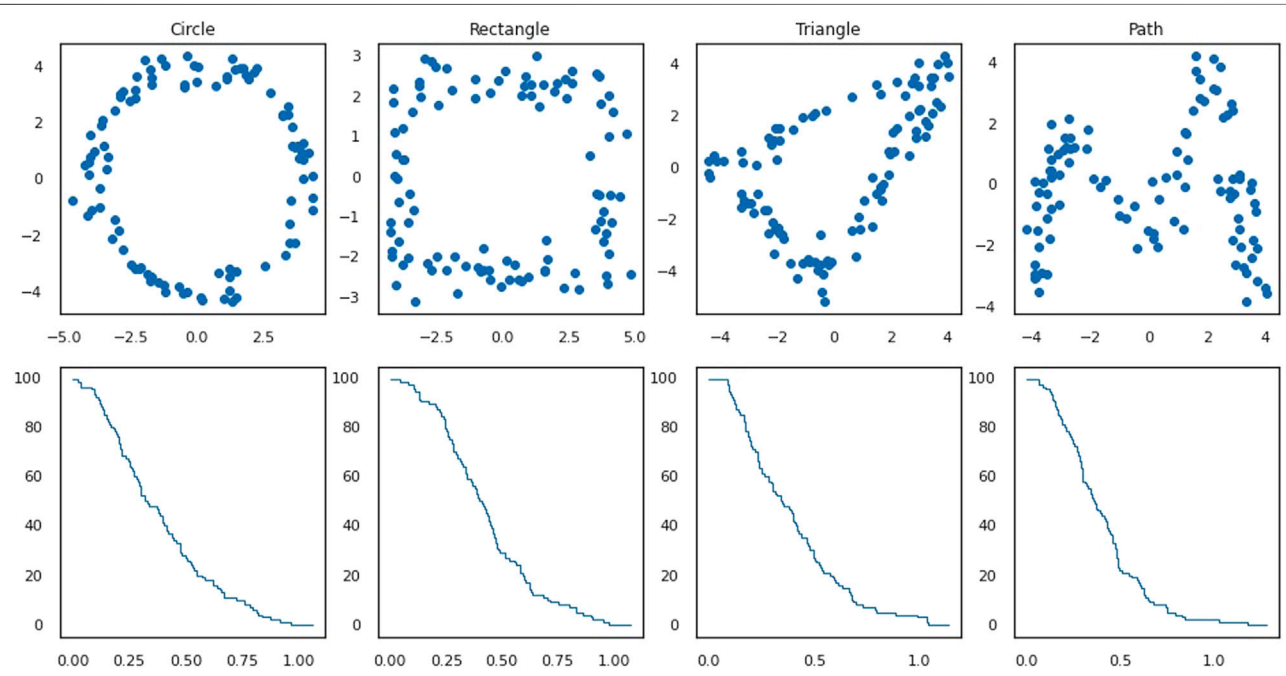


FIGURE 2 | First row: examples of point clouds representing four plane figures in the dataset, one for each of the four shapes. **Second row:** the corresponding standard stable ranks of the persistence modules given by the 0-th homology of the Vietoris-Rips complexes of the plane figures.

stable ranks may be more effective at distinguishing between shapes is given by observing their average per shape, plotted in (**Figure 4A**) together with the standard deviation. This intuition is confirmed when considering the corresponding classification problem, in which we now achieve 88.5% accuracy. Note that in comparison to the stable ranks of the shapes (**Figure 1**, second row), adding noise and averaging has the effect that the stable ranks of plane figures (**Figure 4A**) are smoother, and decrease more gradually. When

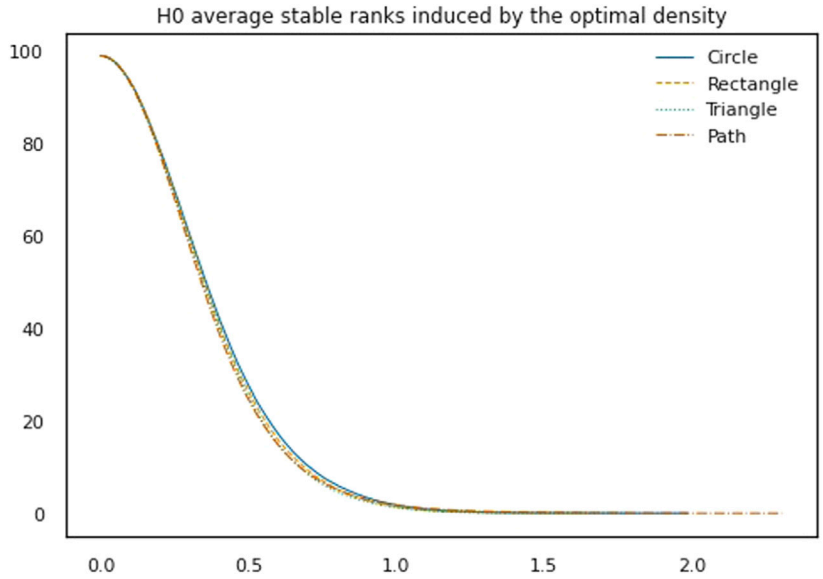


FIGURE 3 | Average 0-th homology standard stable ranks for each shape.

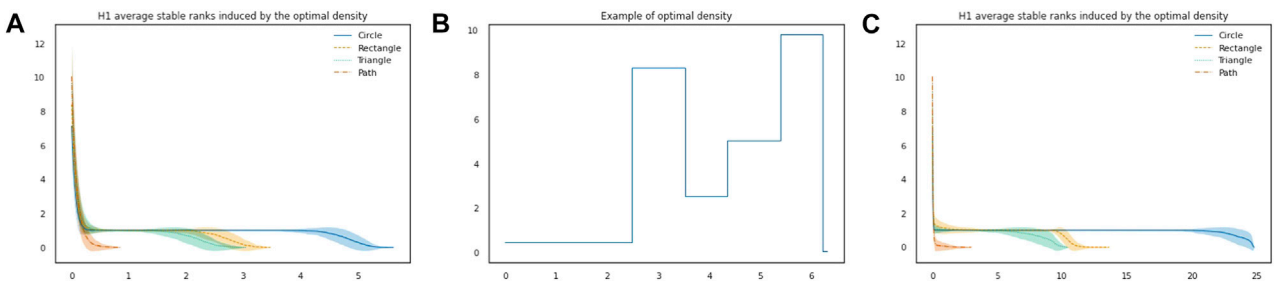


FIGURE 4 | (A): Average 1-st homology standard stable ranks for each shape (shaded area: standard deviation). **(B)**: Example of a density considered optimal during one run of the procedure. **(C)**: Average 1-st homology stable ranks with respect to the optimal pseudometric for each shape (shaded area: standard deviation).

considering persistence modules resulting from the 1-st homology of Vietoris-Rips filtrations, the stable rank associated to the standard pseudometric is no longer a fully discriminatory invariant. Therefore, to improve accuracy it might be a viable strategy to consider alternatives to the standard pseudometric. To generate additional pseudometrics, we will use actions of distance type, which we recall can be defined by means of densities (see *Hierarchical Stabilization: From Persistence Modules to Measurable Functions*). In **Figure 5**, we illustrate the effect of changing densities.

As explained in *Modeling: Determining Appropriate Distances on Persistence Modules*, our strategy to produce densities is to restrict to a simple class of piecewise constant functions. We randomly sample 100 such densities and select the density corresponding to the optimal pseudometric using a cross-validation procedure. The density leading to the best accuracy on the validation set is kept and finally the accuracy on the test set is evaluated and reported.

Again because the dataset is artificially generated we can repeat the procedure many times to get robust results. It appears that although sampling densities introduces another source of randomness, restricting the densities to a simple family allows the improvement to be consistent and outperform the standard action every time. On average, we obtain an accuracy of 94.75%. In **Figures 4B,C**, a density considered optimal during one run of the procedure is shown, together with the stable ranks with respect to that density.

In this case, a simple interpretation for why this density leads to better accuracy might be found by inspecting the average 1-st homology Betti curve per shape. We recall that the 1-st homology Betti curve measures, for t in the filtration scale, the number of bars in a bar decomposition of the 1-st homology persistence module which contain t. Further, averages and standard deviations can be computed per shape. As shown in **Figure 6** it appears that the rectangle and the triangle (which are the sources of confusion in the classification problem) are most easily distinguished by the 1-st homology Betti curve approximately in the interval [3, 4.5] of the filtration scale. Intuitively, the optimal density emphasizes this interval, leading to better accuracy. In

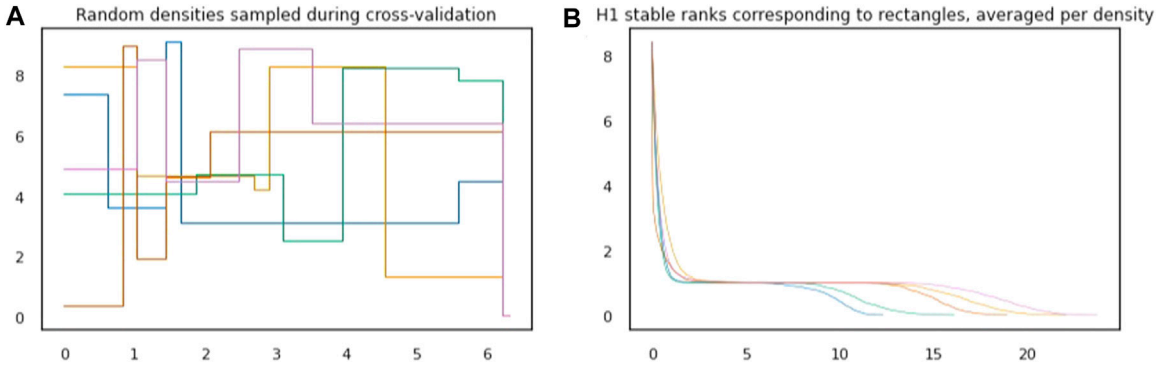


FIGURE 5 | (A): Examples of five densities randomly sampled during cross-validation. **(B)**: For plane figures in the dataset corresponding to the rectangle, stable ranks are computed under the five different densities from the left plot. Average stable ranks are plotted with the same color as the density under which they were computed.

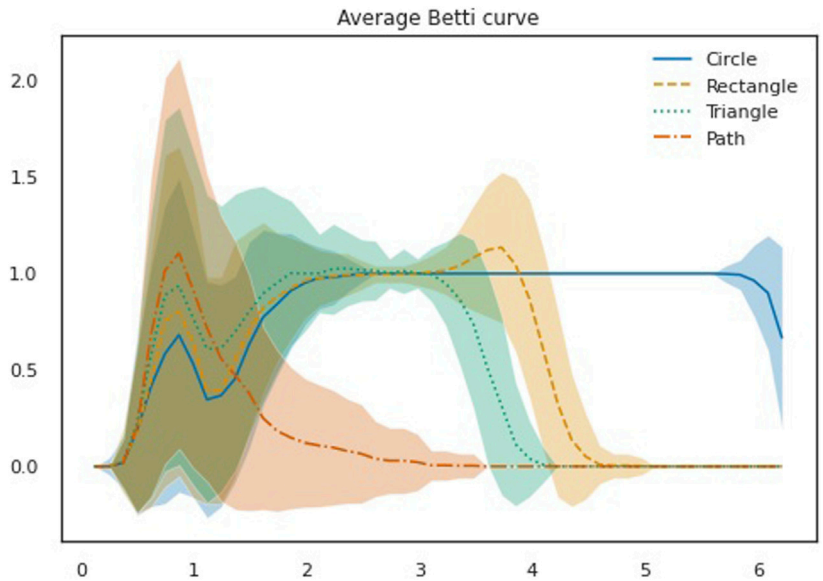


FIGURE 6 | Average 1-st homology Betti curve for each shape (shaded area: standard deviation).

particular the density with value one in the interval [3, 4.5] and value 0.05 elsewhere leads to comparable results as obtained by using an optimal density found through the cross-validation scheme.

3.4 Plane Figures: Analysis Based on Subsampling, Averaging, and First Homology

We now modify our dataset. To add ambient noise to the point clouds we generate: 30% of the 100 points that constitute each point cloud are now sampled uniformly from the $[-5, 5] \times [-5, 5]$ square. The remaining 70% of the points are sampled as described before (see *Plane Figures: Dataset Generation*). In this way we obtain a new dataset of 2000 distance spaces labeled again by four shapes. **Figure 7** shows four point clouds representing four plane figures with different labels in this new dataset. This figure also shows the corresponding standard stable ranks of the persistence modules given by the 1-st homology of the Vietoris-Rips complexes of the distances given by the restriction of the Euclidean metric to the point clouds. The addition of ambient noise has a substantial negative effect: patterns detected by persistent homology such as formation of clusters and voids are very sensitive to the insertion of even a small number of uniformly distributed points. This negative effect is well illustrated in **Figure 8A** where the average of 1-st homology standard stable ranks appear less distinctive for different shapes. However, a simple procedure of subsampling allows us to denoise the data, leading to an invariant which again can discriminate between the different shapes. For each point cloud, we now

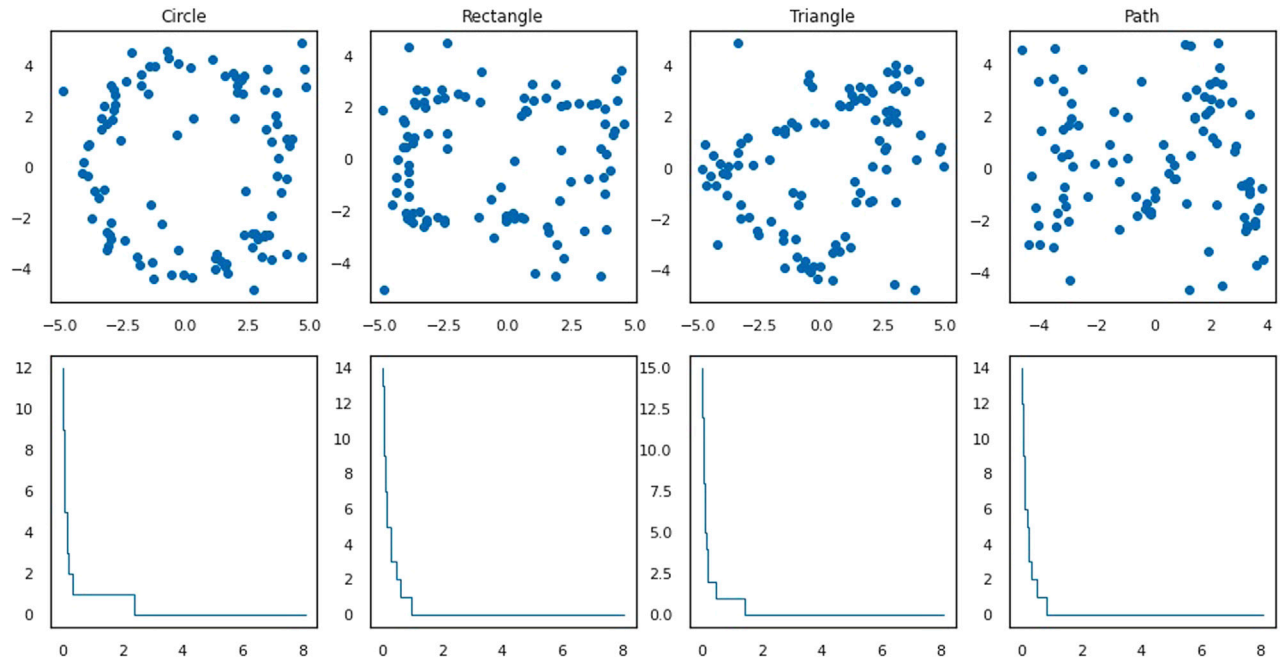


FIGURE 7 | First row: examples of point clouds representing four plane figures in the dataset, one for each of the four shapes, with 30% ambient noise. **Second row:** the corresponding standard stable ranks of the persistence modules given by the 1-st homology of the Vietoris-Rips complexes of the plane figures.

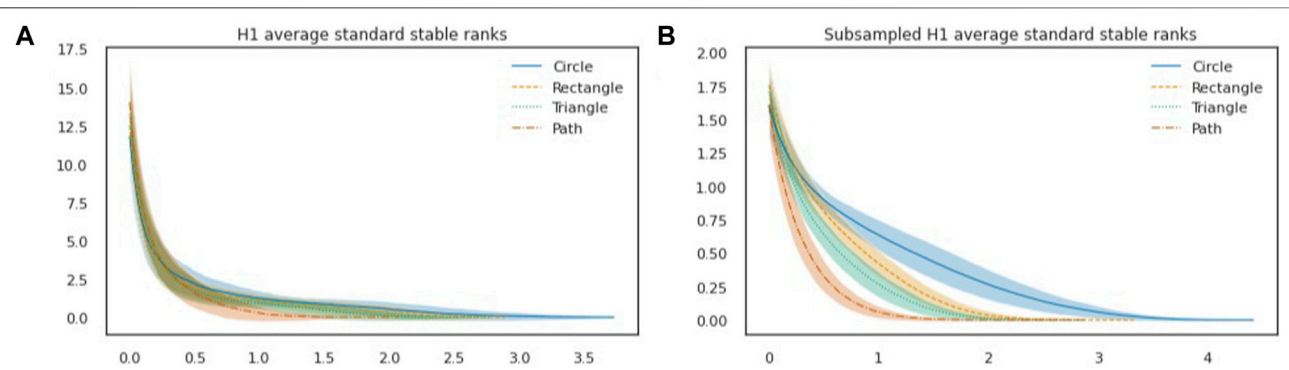


FIGURE 8 | Point clouds generated with ambient noise. **(A)**: 1-st homology average standard stable ranks (shaded area: standard deviation). **(B)**: plane figures are represented by average 1-st homology standard stable ranks of subsamplings. Averages of these representations per shape are illustrated (shaded area: standard deviation).

subsample 20% of its points and generate the corresponding 1-st homology standard stable rank. We repeat this 50 times and compute the average of these 50 stable ranks. Repeating this for all distance spaces in the dataset we obtain 2000 functions whose averages per shape are illustrated in **Figure 8B**. Using the same classification procedure as described in *Plane Figures: Analysis Based on First Homology*, we obtain a much higher shape detection accuracy of 86.25%.

Finally, instead of fixing the level of noise at 30% we now vary it by considering noise levels 0%, 10%, 20%, . . . , 90%, 100%. For each noise level the same process is repeated: generation of figures, subsampling, generation of stable ranks, averaging, and classification, resulting in an accuracy for each level. This procedure was performed both with and without subsampling, as shown in **Figure 9**. As expected, the results are similar in accuracy when there is 0% noise and also when there is 100% noise. However, in between the subsampling clearly leads to an improvement.

3.5 Activity Monitoring

As a real world dataset, we consider activity monitoring data from the PAMAP2 [20] dataset which consists of time series labeled per activity and per indivdual. On average, each time series has 13,872 time steps. The data was preprocessed as in [8]. We select

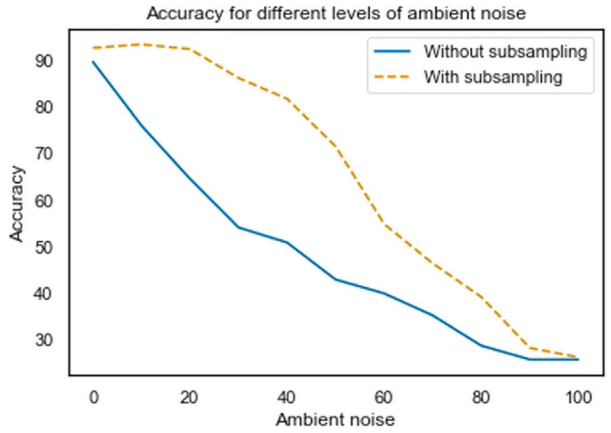


FIGURE 9 | For each level of ambient noise, classification is performed with standard stable ranks resulting from point clouds without subsampling (solid line) and with subsampling (dashed line) and classification accuracies are reported.

two activities (ascending and descending stairs) and seven individuals. By taking the Cartesian product of activities and individuals we thus obtain 14 classes. For each class, which thus represents temporal measurements of one activity performed by one individual we remove the time steps that had no reported heart rate. We also remove a number of columns suspected to contain invalid information. That resulted in 28-dimensional time series with 1,268 time steps on average per individual and per activity. We then sample uniformly without replacement 100 time steps from each of these time series independently. Using the Euclidean distance we obtain a metric space (of size 100) per individual and per activity. Computing 0-th-and 1-st persistent homologies of the Vietoris-Rips filtrations of these metric spaces results in persistence modules. By repeating this procedure 100 times we obtain a dataset consisting of 1,400 observations (100 for each class) where each observation is a pair of persistence modules. Stable ranks can then be computed, first with respect to the standard action. In **Figure 10** average standard stable ranks per class are plotted, both for 0-th and 1-st homologies. One can see that stable ranks allow to distinguish between individuals but even more so between activities.

The problem is formulated as classifying an out-of-sample pair of persistence modules within one of the 14 classes. In contrast with [8] but similarly to the previous experiment, we use an SVM classifier with the stable rank kernel. We construct two kernels, corresponding to 0-th and 1-st homologies respectively, using stable rank with respect to the standard action. For this experiment, however, both kernels appear to be informative and we wish to combine them to achieve better classification accuracy. Since a sum of kernels is also a kernel, we train our SVM with the sum of the kernels for the 0-th and 1-st homologies. There are also other ways to combine multiple kernels into a new one such as taking linear combinations or products [21], which might be useful for other experiments. We use random subsampling validation repeated 20 times with a 60/40 training/test set split. This results in a 68.2% accuracy, demonstrating an improvement over [8] where 60% accuracy was obtained.

Next we apply the same procedure of cross-validation as in the previous experiment to attempt to find a better density and corresponding pseudometric and kernel. We search for alternative densities for the 1-st homology stable rank kernel while keeping the standard action for 0-th homology. This leads to an accuracy of 71.7%, thus somewhat higher than with the standard action. We note that the densities found through this method are similar to the one used in [8] which also led to an improvement. The confusion matrix corresponding to this kernel is shown in **Figure 11**.

4 DISCUSSION

A common pipeline when working with persistent homology is to start with a unique distance on persistence modules (Bottleneck or Wasserstein), then analyze persistence diagrams and finally consider feature maps from persistence diagrams, in case machine learning algorithms are to be applied. Our aim in this article has been to illustrate an alternative pipeline, where we instead start with a vast choice of distances on persistence modules and then consider the induced stable rank which is a continuous mapping with respect to the chosen distance. Our approach is very flexible, distances can be derived from

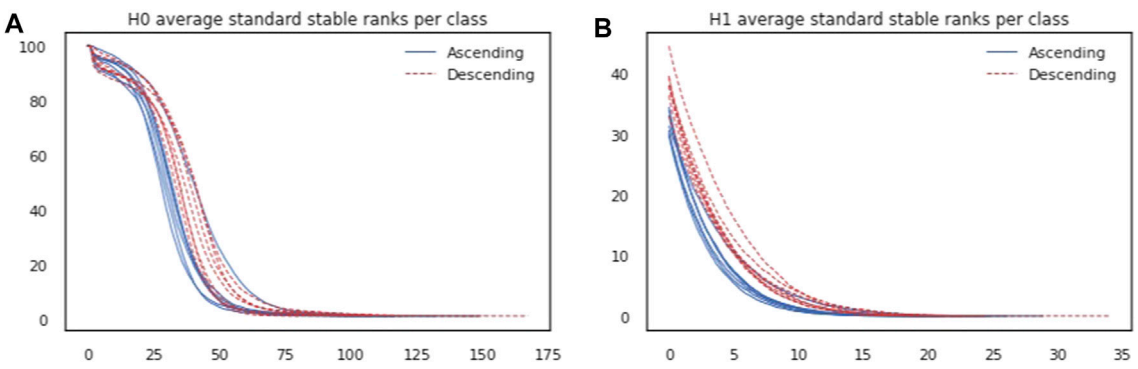


FIGURE 10 | (A): 0-th homology average standard stable ranks per class. **(B)**: 1-st average standard stable ranks per class.

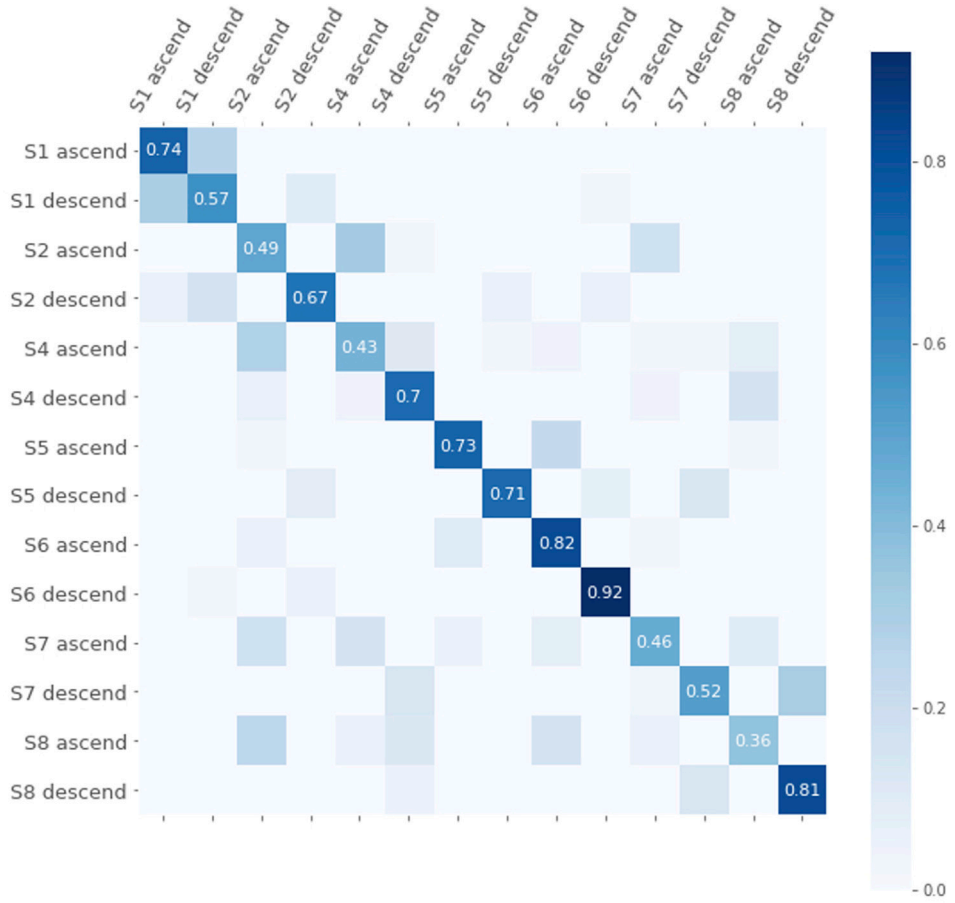


FIGURE 11 | Confusion matrix for the classification with the kernel based on the 0-th homology standard stable rank and the 1-st homology stable rank with respect to the optimal density.

parametrisations (densities) which often have an intuitive interpretation and can be found from simple search procedures, as we wanted to illustrate in the experiments. We believe that this simpler pipeline makes it natural to go between data analysis and machine learning, as stable ranks are amenable to both. For instance, we have presented how intuitions guided by the average of stable ranks then corresponded to classification accuracies through the stable rank kernel. Finally the simplicity of stable ranks makes them computationally efficient, something that is particularly useful for kernel methods.

A schematic situation where alternative distances can be useful, is when the bar decomposition of the classes is as follows. In the first, noisy, part of the filtration scale, bars are distributed randomly following the same distribution for both classes. In the second part of the filtration scale instead, bars are distributed according to two distinct distributions, one for each of the classes. Bottleneck distance is too sensitive to the noise, to utilize the signal, for instance in a classification problem. A distance defined by a density which has small values on the first part of the filtration scale and high values on the second part of the filtration scale, would instead extract the difference between the classes and result in a better classification when encoded in the stable rank kernel. In this case we could directly design a density which improves accuracy in a classification task. In other occasions, as we have shown for example in the plane figures dataset, Betti curves can give an indication of how to design appropriate densities. More generally, when the noise pattern becomes more complicated, we have proposed to randomly generate densities and then evaluate them in a cross-validation procedure. This method was particularly useful for the activity monitoring dataset, where given the difficulty of this classification problem, it was not possible to manually construct densities which improved classification. For the plane figure dataset instead we could still construct densities which perform as well if not slightly better than the optimal density among the randomly generated ones.

Learning algorithms for density optimization are an appealing alternative to this strategy, although we believe conceptual and algorithmic challenges are inherent to this problem. If for example density optimization is framed in terms of metric learning, the most difficult part is to identify a meaningful and well behaved objective function to optimize.

Preliminary work by O. Gävfert [22], highlights that the choice of basic objective functions do not lead to convex optimization problems. Here we circumvent the question of identifying an appropriate objective function by evaluating the performance of a density through the accuracy of the associated kernel in SVM.

To enhance analysis using our methods one should keep in mind that in most cases it is convenient to consider several distances at the same time. For example different degree homologies (e.g., 0-th and 1-st homologies) could, and possibly should, be treated independently. In other words, distances that are suitable for the 0-th homology might not be informative for the 1-st homology. Even for analysis involving only one degree homology one should not look for just one density and one kernel since stable ranks with respect to different densities might show different geometrical aspects of the data. In principle it is possible to fully recover persistence modules, by using stable ranks (see *Modeling: Determining Appropriate Distances on Persistence Modules* and [8]), however in practice the whole information of the persistence modules might be redundant, while with an appropriate number and choice of densities, we believe, one could be able to extract more valuable information for a classification task.

While the examples in this article concern classification problems based on one-dimensional persistence, a more general treatment of the kernel would be interesting, both in terms of multi-persistence (the stable rank kernel is a multi-persistence kernel but the barcode decomposition in the one-dimensional case allows to compute it very efficiently), and in terms of utilizing the kernel in other contexts, such as for statistical inference.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

This article is based on JA thesis [11]. JA, MS, WC, and RR conceived and developed the study. JA performed the data analyses. JA, MS, WC, and RR wrote the paper.

FUNDING

JA was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation. WC was partially supported by VR, the Wallenberg AI, Autonomous System and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation, and MultipleMS funded by the European Union under the Horizon 2020 program, grant agreement 733,161. RR was partially supported by MultipleMS funded by the European Union under the Horizon 2020 program, grant agreement 733,161. MS was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation, VR, and Brummer and Partners MathDataLab. Support of the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation was indispensable for conducting this research.

REFERENCES

1. Hausmann JC. On the Vietoris-Rips Complexes and a Cohomology Theory for Metric Spaces. Prospects in Topology (Princeton, NJ, 1994) (Princeton Univ. Press, Princeton, NJ). *Ann Math Stud* (1995) 138:175–88. doi:10.1515/9781400882588-013
2. Edelsbrunner H, Letscher D, and Zomorodian A. *Topological Persistence and Simplification. 41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA, 2000, Topological Persistence and Simplification)*. Los Alamitos, CA: IEEE Comput. Soc. Press (2000). p. 454–63. doi:10.1109/SFCS.2000.892133
3. Edelsbrunner H, Letscher D, and Zomorodian A. Topological Persistence and Simplification. *Discrete Comput Geom* (2002) 28:511–33. doi:10.1007/s00454-002-2885-2
4. Nielson JL, Paquette J, Liu AW, Guandique CF, Tovar CA, Inoue T, et al. Topological Data Analysis for Discovery in Preclinical Spinal Cord Injury and Traumatic Brain Injury. *Nat Commun* (2015) 6:8581. doi:10.1038/ncomms9581
5. Marian G, and Yuri K. Topological Data Analysis of Financial Time Series: Landscapes of Crashes. *Physica A: Stat Mech its Appl* (2018) 491:820–834. doi:10.1016/j.physa.2017.09.028
6. Mileyko Y, Mukherjee S, and Harer J. Probability Measures on the Space of Persistence Diagrams. *Inverse Probl* (2011) 27:124007. doi:10.1088/0266-5611/27/12/124007
7. Turner K, Mileyko Y, Mukherjee S, and Harer J. Fréchet Means for Distributions of Persistence Diagrams. *Discrete Comput Geom* (2014) 52:44–70. doi:10.1007/s00454-014-9604-7
8. Chachólski W, and Riihimäki H. Metrics and Stabilization in One Parameter Persistence. *SIAM J Appl Algebra Geometry* (2020) 4:69–98. doi:10.1137/19M1243932
9. Oliver G, and Wojciech C. Stable Invariants for Multidimensional Persistence. *arXiv* [Preprint] (2017). Available from: https://arxiv.org/abs/1703.03632v1
10. Scolamiero M, Chachólski W, Lundman A, Ramanujam R, and Öberg S. Multidimensional Persistence and Noise. *Found Comput Math* (2017) 17:1367–406. doi:10.1007/s10208-016-9323-y
11. Agerberg J. *Statistical Learning And Analysis On Homology-Based Features. Master's Thesis*. KTH Royal Institute of Technology, Stockholm (2020).
12. Reininghaus J, Huber S, Bauer U, and Kwitt R. A Stable Multi-Scale Kernel for Topological Machine Learning. *Proc IEEE Conf Comput Vis pattern recognition* (2015), 4741–4748. doi:10.1109/cvpr.2015.7299106
13. Zhao Q., and Wang Y. (2019). "Learning Metrics for Persistence-Based Summaries and Applications for Graph Classification," in *Advances in Neural Information Processing Systems*. Editors H. Wallach, H. Larochelle, and A. Beygelzimer (Red Hook, NY: Curran Associates, Inc.) 32.
14. Massey WS. A Basic Course in Algebraic Topology. *Graduate Texts Mathematics* (1991) 27:xvi+428. doi:10.1007/978-1-4939-9063-4
15. Zomorodian A, and Carlsson G. Computing Persistent Homology. *Discrete Comput Geom* (2005) 33:249–74. doi:10.1007/s00454-004-1146-y

16. Chazal F, de Silva V, Glisse M, and Oudot S. *The Structure and Stability of Persistence Modules*. Springer International Publishing (2016).
17. Anderson I. *Combinatorics of Finite Sets. Oxford Science Publications*. New York: The Clarendon Press, Oxford University Press (1987). p. xvi+250.
18. Bauer U. *Ripser: Efficient Computation of Vietoris-Rips Persistence Barcodes*. *arXiv* [Preprint] (2019). Available from: https://arxiv.org/abs/1908.02518v1
19. Chazal F, Cohen-Steiner D, Glisse M, Guibas LJ, and Oudot SY. Proximity of Persistence Modules and Their Diagrams. In: *Proceedings of the 25th Annual Symposium on Computational Geometry* ***SCG '09*** (2009). p. 237–46.
20. PAMAP. *Physical Activity Monitoring for Aging People*. Available from: www.pamap.org.
21. Shawe-Taylor J, and Cristianini N, *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge (2004).
22. Gävfert O. *Topology-based Metric Learning* (2018). Available from: https://people.kth.se/~oliverg/

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Artificial Intelligence

REVIEW
published: 28 July 2021
doi: 10.3389/frai.2021.681174

A Comparative Study of Machine Learning Methods for Persistence Diagrams

Danielle Barnes[1]*, *Luis Polanco*[1,2] and *Jose A. Perea*[1,2]

[1]Department of Computational Mathematics, Science and Engineering, Michigan State University, East Lansing, MI, United States, [2]Department of Mathematics, Michigan State University, East Lansing, MI, United States

Many and varied methods currently exist for featurization, which is the process of mapping persistence diagrams to Euclidean space, with the goal of maximally preserving structure. However, and to our knowledge, there are presently no methodical comparisons of existing approaches, nor a standardized collection of test data sets. This paper provides a comparative study of several such methods. In particular, we review, evaluate, and compare the stable multi-scale kernel, persistence landscapes, persistence images, the ring of algebraic functions, template functions, and adaptive template systems. Using these approaches for feature extraction, we apply and compare popular machine learning methods on five data sets: MNIST, Shape retrieval of non-rigid 3D Human Models (SHREC14), extracts from the Protein Classification Benchmark Collection (Protein), MPEG7 shape matching, and HAM10000 skin lesion data set. These data sets are commonly used in the above methods for featurization, and we use them to evaluate predictive utility in real-world applications.

Keywords: persistent homology, machine learning, topological data analysis, persistence diagrams, barcodes

OPEN ACCESS

Edited by:
Kathryn Hess,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Mathieu Carrière,
Institut National de Recherche en Informatique et en Automatique, France
Gregory Henselman-Petrusek,
University of Oxford, United Kingdom

***Correspondence:**
Danielle Barnes
barnesd8@msu.edu

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *16 March 2021*
Accepted: *11 June 2021*
Published: *28 July 2021*

Citation:
Barnes D, Polanco L and Perea JA (2021) A Comparative Study of Machine Learning Methods for Persistence Diagrams. Front. Artif. Intell. 4:681174. doi: 10.3389/frai.2021.681174

1 INTRODUCTION

Persistence diagrams are an increasingly useful shape descriptor from Topological Data Analysis. One of their more popular uses to date has been as features for machine learning tasks, with success in several applications to science and engineering. Though many methods and heuristics exist for performing learning with persistence diagrams, evaluating their relative merits is still largely unexplored. Our goal here is to contribute to the comparative analysis of machine learning methods with persistence diagrams.

Starting with topological descriptors of datasets, in the form of persistence diagrams, we provide examples and methodology to create features from these diagrams to be used in machine learning algorithms. We provide the necessary background and mathematical justification for six different methods (in chronological order): the Multi-Scale Kernel, Persistence Landscapes, Persistence Images, Adcock-Carlsson Coordinates, Template Systems, and Adaptive Template Systems. To thoroughly evaluate these methods, we have researched five different data sets and the relevant methods to compute persistence diagrams from them. The datasets, persistence diagrams and code to compute the persistence diagrams is readily available for academic use.

As part of this review, we also provide a user guide for these methods, including comparisons and evaluations across the different types of datasets. After computing the six types of features, we compared the predictive accuracy of a ridge regression, random forest, and support vector machine model to assess the type of featurization that is most useful in predictive models. The code developed

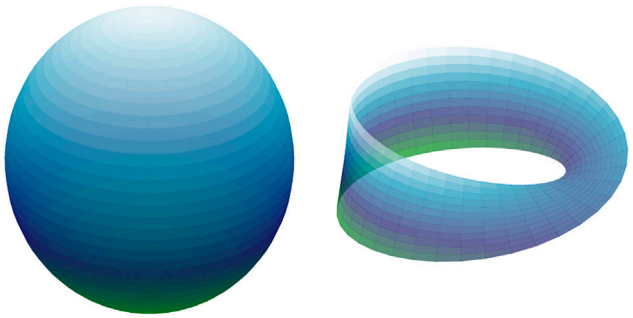


FIGURE 1 | Left: The 2-dimensional sphere $S^2 \subset \mathbb{R}^3$, right: the Möbius band $\mathcal{M}$.

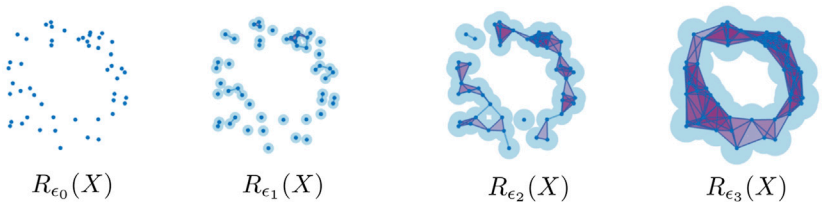


FIGURE 2 | The Rips complex on a point cloud $(X, \mathbf{d}_X)$ sampled around the unit circle, for four different scale choices.

for this analysis is available, with some functions developed specifically for use in machine learning applications, and easy-to-use jupyter notebooks showing examples of each function with multiple dataset types.

Of these methods, Persistence Landscapes, Adcock-Carlsson Coordinates, and Template Systems are quite accurate and create features for large datasets quickly. Adaptive Template Systems and Persistence Images took somewhat longer to run, however, the Adaptive Template Systems featurization method did improve accuracy over other methods. The Multi-Scale Kernel was the most computationally intensive, and during our evaluation we did not observe instances of it outperforming other methods.

2 BACKGROUND

Algebraic topology is the branch of mathematics concerned with the study of shape in abstract spaces. Its main goal is to quantify the presence of features which are invariant under continuous deformations; these include properties like the number of connected components in the space, the existence of holes and whether or not the space is orientable. As an example, **Figure 1** shows two spaces: the 2-dimensional sphere on the left, which is the set $S^2 = \{\mathbf{x} \in \mathbb{R}^3 : \mathbf{x} = 1\}$ of 3-dimensional vectors with unit norm, and the Möbius band $\mathcal{M} = [-1, 1] \times [-1, 1]/(-1, y) \sim (1, -y)$ on the right. The latter can be thought of as the result of gluing the right and left edges of the square $[-1, 1] \times [-1, 1]$ with opposite orientations.

The aforementioned properties of shape for these spaces are as follows. Both S^2 and $\mathcal{M}$ are connected, while S^2 is orientable but $\mathcal{M}$ is not. Moreover, any closed curve drawn on the surface of S^2 bounds a 2-dimensional spherical cap, and thus we say that the sphere has no 1-dimensional holes. The equator $\{(x, 0) : |x| \leq 1\}$ of the Möbius band, on the other hand, is a closed curve in $\mathcal{M}$ which is not the boundary of any 2-dimensional region, and therefore we say that $\mathcal{M}$ has one 1-dimensional hole. Finally, S^2 is itself a closed 2-dimensional surface bounding a 3-dimensional void—thus the sphere is said to have a 2-dimensional hole—but $\mathcal{M}$ has no such features.

The *homology* of a space is one way in which topologists have formalized measuring the presence of n-dimensional holes in a space (Hatcher, 2002). Indeed, for a space X (e.g., like the sphere or the Möbius band) an integer $n \geq 0$ and a field $\mathbb{F}$ (like the integers modulo a prime p, denoted $\mathbb{Z}_p$), the n-th homology of X with coefficients in $\mathbb{F}$ is a vector space over $\mathbb{F}$ denoted $H_n(X; \mathbb{F})$. The main point is that the dimension of this vector space corresponds roughly to the number of essentially distinct n-dimensional holes in X. Going back to the examples from **Figure 1**:

$$\begin{aligned} H_0(S^2; \mathbb{Z}_2) &= \mathbb{Z}_2, & H_0(\mathcal{M}; \mathbb{Z}_2) &= \mathbb{Z}_2 \\ H_1(S^2; \mathbb{Z}_2) &= \mathbf{0}, & H_1(\mathcal{M}; \mathbb{Z}_2) &= \mathbb{Z}_2 \\ H_2(S^2; \mathbb{Z}_2) &= \mathbb{Z}_2, & H_2(\mathcal{M}; \mathbb{Z}_2) &= \mathbf{0} \end{aligned}$$

where, again, the dimension of $H_0(X; \mathbb{F})$ corresponds to the number of connected components in X, the dimension of $H_1(X; \mathbb{F})$ represents the number of 1-dimensional holes, and so on for $H_n(X; \mathbb{F})$ and $n \geq 1$. It is entirely possible that different choices of $\mathbb{F}$ result in different dimensions for $H_n(X; \mathbb{F})$; this is an indication of intricate topological structure in X, but the metaphor of holes is still useful.

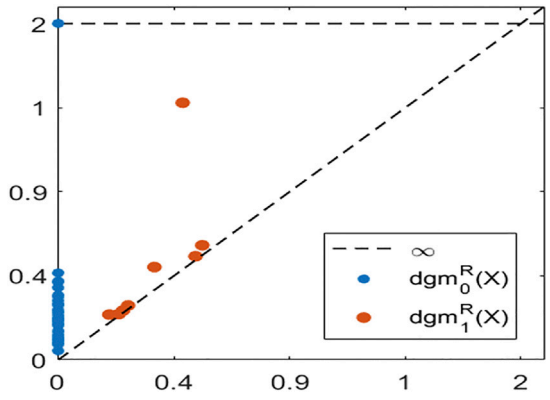


FIGURE 3 | The Rips persistence diagrams in dimensions 0 (blue) and 1 (orange), for a point cloud sampled around the unit circle.

2.1 Persistent Homology

There are several learning tasks where each point in a data set has shape or geometric information relevant to the problem at hand. Indeed, in shape retrieval, database elements are often 3D meshes discretizing physical objects, and the ensuing learning tasks are often related to pose-invariant classification (Pickup et al., 2014). In computational chemistry and drug design, databases of chemical compounds are mined in order to discover new targets with desirable functional properties. In this case, the shape of each molecule (i.e., of the collection of comprising atoms) is closely related to molecular function, and thus shape features can be useful in said data analysis tasks (Bai et al., 2009).

If homology is what topologists use to measure the shape of abstract spaces, then *persistent homology* is how the shape of a geometric data set can be quantified (Perea, 2019). Persistent homology takes as input an increasing sequence of spaces

$$\mathcal{X} : X_0 \subset X_1 \subset \cdots \subset X_L.$$

Any such sequence is called a *filtration*. The definition of persistent homology relies on two facts: first, that one can compute homology for each space separately, i.e., $H_n(X_\ell; \mathbb{F})$ for each $0 \le \ell \le L$, and second, that each inclusion $X_\ell \subset X_{\ell+1}$ induces a linear transformation $H_n(X_\ell; \mathbb{F}) \to H_n(X_{\ell+1}; \mathbb{F})$ between the corresponding vector spaces. The n-th persistent homology of the filtration $\mathcal{X}$ is the sequence

$$PH_n(\mathcal{X}; \mathbb{F}) : H_n(X_0; \mathbb{F}) \to H_n(X_1; \mathbb{F}) \to \cdots \to H_n(X_L; \mathbb{F})$$

of vector spaces and induced linear transformations.

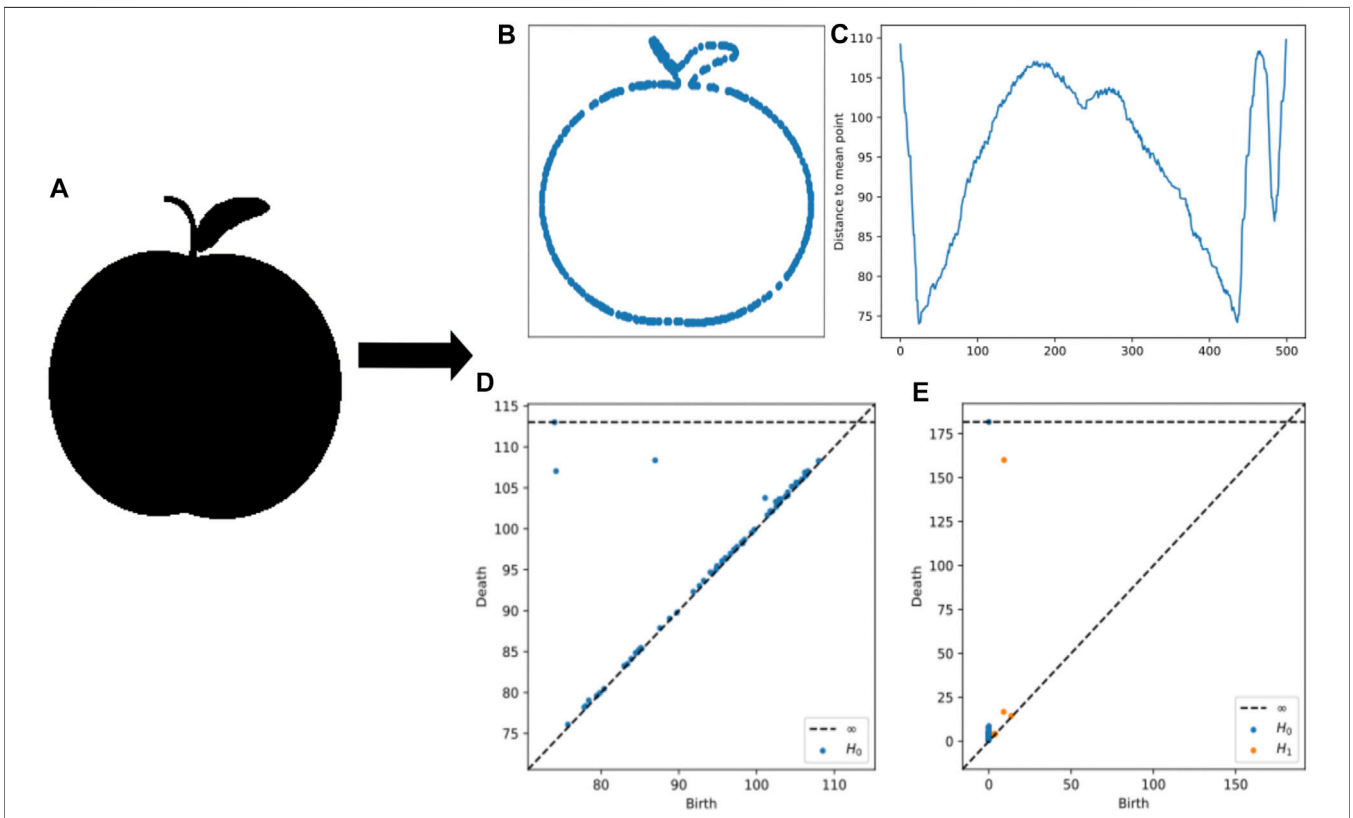


FIGURE 4 | (A) An example apple from MPEG7 data. **(B)** An example image contour used for MPEG7. **(C)** The distance to mean point calculation used for sublevel set persistence **(D)** Persistence diagrams from lower star persistence **(E)** Persistence diagrams from the contour.

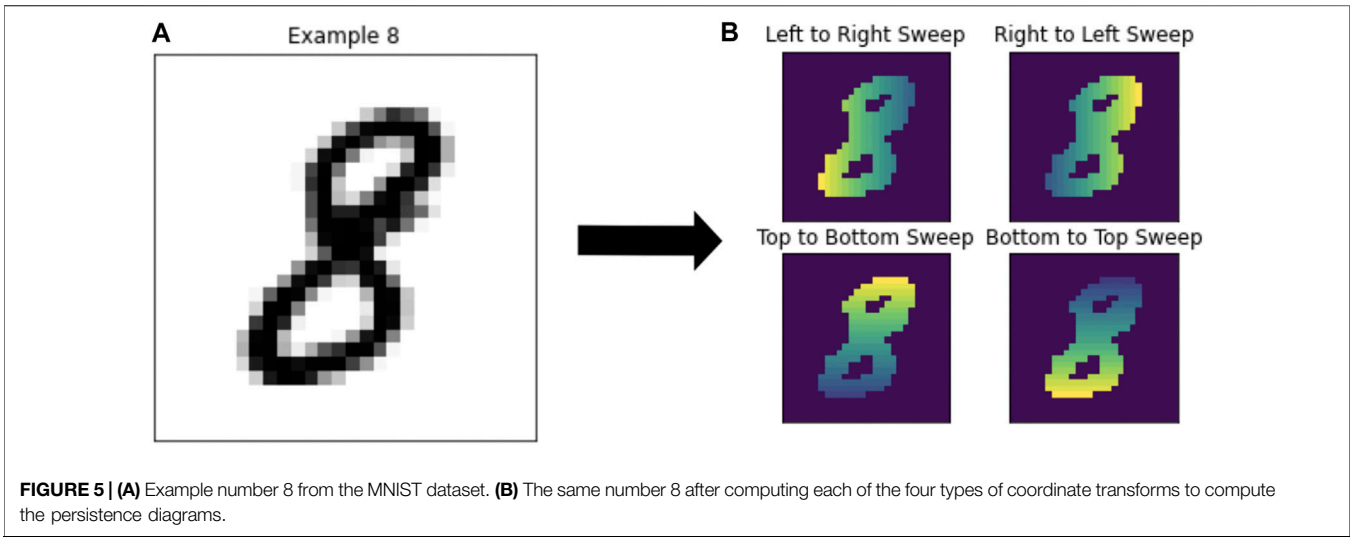


FIGURE 5 | (A) Example number 8 from the MNIST dataset. **(B)** The same number 8 after computing each of the four types of coordinate transforms to compute the persistence diagrams.

The evolution of features in $PH_n(\mathcal{X};\mathbb{F})$, which is the main point of interest, can be encoded and visualized through a *persistence diagram*. In a nutshell, if each $H_n(X_j;\mathbb{F})$ is finite dimensional and $\beta_n^{j,\ell}(\mathcal{X};\mathbb{F})$ denotes the rank of the linear transformation $H_n(X_j;\mathbb{F}) \to H_n(X_\ell;\mathbb{F})$ induced by the inclusion $X_j \subset X_\ell$, $j \le \ell$, then the persistence diagram of $PH_n(\mathcal{X};\mathbb{F})$, denoted $\mathrm{dgm}_n(\mathcal{X};\mathbb{F})$, is the collection of pairs (j,ℓ) with nonzero multiplicity (i.e., number of repeats).

$$\mu_n^{j,\ell} := \beta_n^{j,\ell-1}(\mathcal{X};\mathbb{F}) - \beta_n^{j-1,\ell-1}(\mathcal{X};\mathbb{F}) - \beta_n^{j,\ell}(\mathcal{X};\mathbb{F}) + \beta_n^{j-1,\ell}(\mathcal{X};\mathbb{F}),$$
$$0 \le j < \ell \le L.$$

See section VII.1 of Edelsbrunner and Harer (2010) for more details. In other words, $\mathrm{dgm}_n(\mathcal{X};\mathbb{F})$ is a multiset (i.e., a set whose elements appear with multiplicity) of pairs, where each $(j,\ell) \in \mathrm{dgm}_n(\mathcal{X};\mathbb{F})$ encodes $\mu_n^{j,\ell}$ homological features of the filtration $\mathcal{X}$ which appear at X_j (i.e., j is the *birth time*) and disappear entering X_ℓ (ℓ is the *death time*). The *persistence* $\ell - j$ of (j,ℓ) is often used as a measure of prominence across the filtration $\mathcal{X}$, but short-lived features can be quite informative for learning purposes as well [see for instance Bendich et al. (2016)].

2.1.1 Filtrations From Point Cloud Data

There are several ways of constructing filtrations from geometric data. Indeed, let X be a set and $\mathbf{d}_X$ a measure of distance between its elements. The pair $(X,\mathbf{d}_X)$ is often referred to as point cloud data, and the running hypothesis is that it is the result of sampling X from an unknown continuous space. The ensuing inference problem in Topological Data Analysis is to use $(X,\mathbf{d}_X)$ to estimate shape/homological features of the unknown underlying space. A popular strategy is to compute the *Vietoris-Rips complex*

$$R_\epsilon(X) := \left\{ \{x_0,\ldots,x_m\} \subset X \,\middle|\, \max_{0\le j,k\le m} \mathbf{d}_X(x_j,x_k) \le \epsilon, m \in \mathbb{N} \right\} \quad (1)$$

where $\epsilon \ge 0$, a singleton $\{x\}$ is thought of as a vertex at x, a set with two elements $\{x_0,x_1\}$ represents an edge between x_0 and x_1, a set $\{x_0,x_1,x_2\}$ spans a triangle, and so on. This construction is motivated by the fact that $R_\epsilon(X)$ is known to approximate the topology of the underlying space from which X was sampled under various conditions on X and ϵ (Latschev, 2001). In practice, however, an optimal choice of scale $\epsilon \ge 0$ is unclear at best, so one instead considers the *Vietoris-Rips filtration*

$$\mathcal{R}(X) : R_{\epsilon_0}(X) \subset R_{\epsilon_1}(X) \subset \cdots \subset R_{\epsilon_L}(X) \quad (2)$$

for $0 \le \epsilon_0 < \epsilon_1 < \cdots < \epsilon_L$. The ϵ_ℓ's can be chosen, for instance, to be the different values of the distance function $\mathbf{d}_X$. **Figure 2** shows an example of this construction for $X \subset \mathbb{C}$ sampled around the unit circle $S^1 = \{z \in \mathbb{C} : |z| = 1\}$, and four scales $\epsilon \ge 0$.

The persistent homology of the Vietoris-Rips filtration, i.e., $PH_n(\mathcal{R}(X);\mathbb{F})$, can then be used to measure the shape of the underlying shape of the point cloud. An important point is that even though homology is invariant under continuous deformations, the Vietoris-Rips complex is a metric-based construction. Thus, the resulting Vietoris-Rips persistence diagrams

$$\mathrm{dgm}_n^{\mathcal{R}}(X) = \left\{ \left(\epsilon_j,\epsilon_\ell\right) \text{ with multiplicity } \mu_n^{j,\ell} > 0 \right\}$$

often encode features such as density and curvature, in addition to the presence of holes and voids (Bubenik et al., 2020). **Figure 3** shows the Vietoris-Rips persistence diagrams in dimensions $n = 0, 1$ for the data sampled around the unit circle in **Figure 2**. The persistence of a point in a persistence diagram can be visualized as its vertical distance to the diagonal. This measures how likely it is for said feature to correspond to one of the underlying space, instead of being a reflection of sampling artifacts [see for instance Theorem 5.3 in Oudot (2015)]. The fact that there is one highly persistent point for $n = 0$ indicates that the data has one cluster (i.e., one connected component), while the presence of one highly persistent point for $n = 1$ indicates that there is a strong 1-dimensional hole in the data. Both are consistent with, and suggest, that the circle is the underlying space.

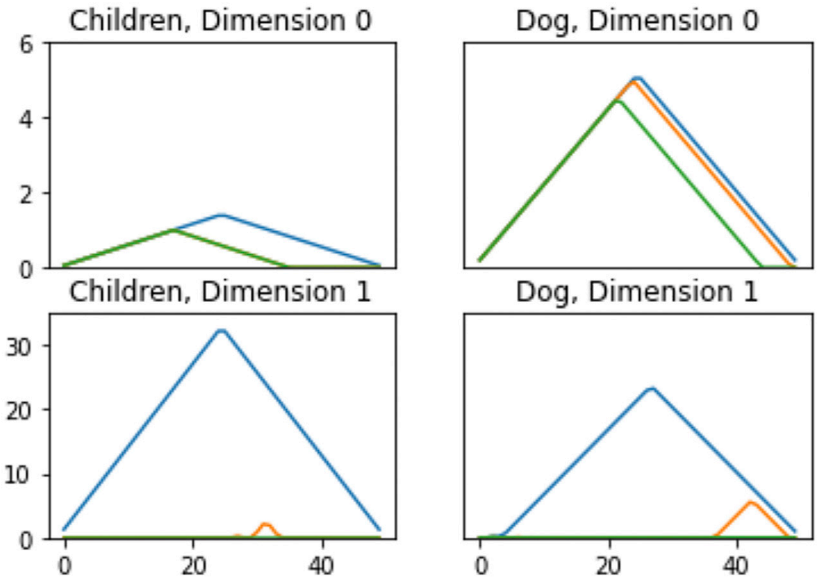


FIGURE 6 | Persistence Landscapes from the MPEG7 dataset to show differences in features. Each color corresponds to a different landscape, i.e., λ_k for $k = 1, 2, 3$.

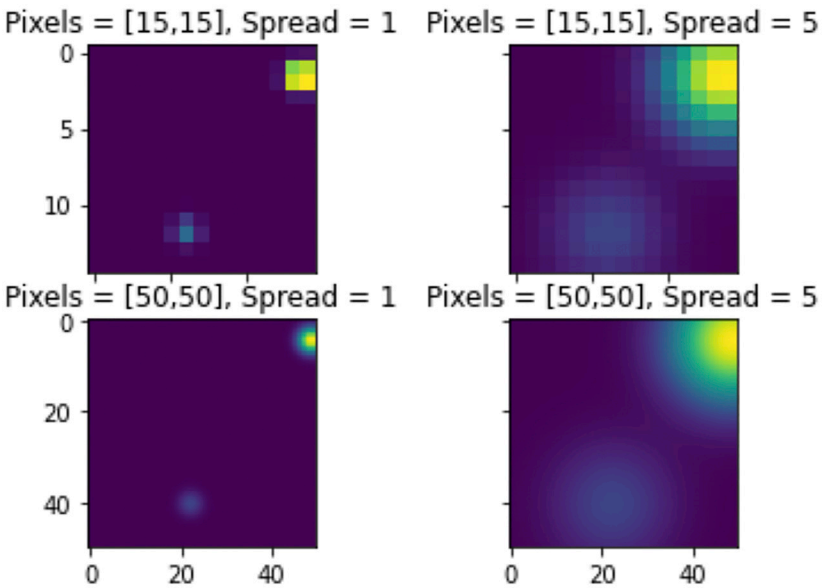


FIGURE 7 | Persistence Images of a 5 from the MNIST set in dimension 0.

2.1.2 Filtrations From Scalar Functions and Image Data

If $\mathbb{X}$ is a topological space and $f : \mathbb{X} \to \mathbb{R}$ is a function, then the sublevel sets

$$\mathbb{X}_a = f^{-1}(-\infty, a], \quad a \in \mathbb{R}$$

define the so called *sublevel set filtration* of $\mathbb{X}$. If $\mathbb{X}$ is a 3D mesh, for example, then one can compute estimates of curvature at every vertex, and then extend said function linearly (*via* barycentric coordinates) to the triangular faces. The persistent homology of the sublevel set filtration is often called *sublevel set persistence*, and it is useful in quantifying shape properties of geometric objects which are endowed with scalar functions. See **Figure 4** for an application of this idea. The corresponding persistence diagrams are denoted $\mathrm{dgm}_n(f)$.

Images provide another data modality where sublevel set persistence can be useful. Indeed, an image can be thought of as a function on a grid of pixels; if the image is in grey scale,

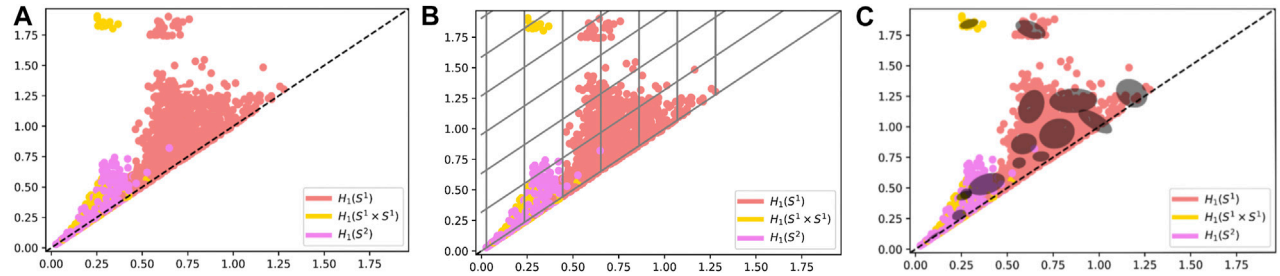


FIGURE 8 | (A) Persistence Diagrams **(B)** Persistence Diagrams with boundaries of the support function for Tent Features **(C)** Persistence Diagrams with boundaries of the support for Adaptive Templates.

then we have a single scalar valued function, and if the image is multi-channel (like RGB color representations) then each channel is analyzed independently. The grid yields a triangulated space *via* a Freudenthal triangulation of the plane, and the values of pixel intensity in each channel can be extended *via* convex combinations to the faces [see Lemma 1 of Kuhn (1960)]. We will apply this methodology later on to the MNIST hand written digit data base (**Figure 5**). This approach to computing persistent homology from images is not unique in the literature; other popular methods such as cubical homology (Kaczynski et al., 2004) have been used for this same purpose. This work, however, deals exclusively with simplicial homology as it is the standard approach in many applications.

2.2 The Space of Persistence Diagrams

Persistence diagrams have shown to be a powerful tool for quantifying shape in geometric data (Carlsson, 2014). Moreover, one of their key properties is their stability with respect to perturbations in the input, which is crucial when dealing with noisy measurements. Indeed, two persistence diagrams D and D' are said to be δ-matched, $\delta > 0$, if there exists a bijection $m : A \to A'$ of multisets $A \subset D$ and $A' \subset D'$ with.

$\|\mathbf{x} - m(\mathbf{x})\|_\infty < \delta$ for every $\mathbf{x} \in A$, where $\|\cdot\|_\infty$ is the maximum metric in $\mathbb{R}^2$.
If $(a, b) \in (D \backslash A) \cup (D' \backslash A')$, then $b - a < 2\delta$.

The *bottleneck distance* $d_B(D, D')$ is the infimum over all $\delta > 0$ so that D and D' are δ-matched; this defines a metric on the set $\mathcal{D}_0$ of all finite persistence diagrams. The *stability* theorem of Cohen-Steiner et al. (2007) for sublevel set persistence contends that if $\mathbb{X}$ is a finitely triangulated space and $f, g : \mathbb{X} \to \mathbb{R}$ are *tame* and continuous, then

$$d_B(\mathrm{dgm}_n(f), \mathrm{dgm}_n(g)) \leq \|f - g\|_\infty$$

for every integer $n \geq 0$. We note that the theorem is still true if continuous is replaced by piecewise linear. Similarly, if $(X, \mathbf{d}_X)$ and $(Y, \mathbf{d}_Y)$ are finite metric spaces, then the stability of Rips persistent homology (Chazal et al., 2014, Theorem 5.2) says that

$$d_B(\mathrm{dgm}_n^{\mathcal{R}}(X), \mathrm{dgm}_n^{\mathcal{R}}(Y)) \leq 2 d_{GH}(X, Y)$$

where $d_{GH}(\cdot, \cdot)$ denotes the Gromov-Hausdorff distance (Gromov, 2007).

In order to develop the mathematical foundations needed for doing machine learning with persistence diagrams, it has been informative to first study the structure of the space they form. Indeed, if $\mathcal{D}_0$ denotes the space of finite persistence diagrams, then we will let $\mathcal{D}$ denote its metric completion with respect to the bottleneck distance d_B. It readily follows that d_B extends to a metric on $\mathcal{D}$. See Blumberg et al. (2014) for an explicit description of what the elements of $\mathcal{D}$ are. In addition to the bottleneck distance, the *Wasserstein metric* from optimal transport suggests another way of measuring similarity between persistence diagrams. Indeed, for each integer $p \geq 1$ and $D, D' \in \mathcal{D}$, their p-th Wasserstein distance is

$$d_{W_p}(D, D') := \inf_m \left(\sum_{x \in A} \|x - m(x)\|_\infty^p + \sum_{(a,b) \in (D \backslash A) \cup (D' \backslash A')} \left(\frac{b-a}{2} \right)^p \right)^{1/p}$$

where the infimum runs over all multiset bijections $m : A \to A'$, for $A \subset D$ and $A' \subset D'$. One can show that d_{W_p} defines a metric on the set

$$\mathcal{D}_p := \left\{ D \in \mathcal{D} \middle| d_{W_p}(D, \varnothing) < \infty \right\}$$

and that $(\mathcal{D}_p, d_{W_p})$ is a complete separable metric space (Mileyko et al., 2011) with $d_{W_p} \to d_B$ as $p \to \infty$.

Doing statistics and machine learning directly on the space of persistence diagrams turns out to be quite difficult. Indeed, $(\mathcal{D}, d_B)$ does not have unique geodesics, and thus the Fréchet mean of general collections of persistence diagrams is not unique (Turner et al., 2014). Since computing averages, and in general, doing linear algebra on persistence diagrams is not available, then several authors have proposed mapping $(\mathcal{D}, d_B)$ to topological vector spaces where further analysis can be done. These methods are the main focus of this review. The theory of vectorization of persistence diagrams is an active area of research, with recent results showing the impossibility of full embeddability. Indeed, even though the space of persistence diagrams with exactly n points can be coarsely embedded in a Hilbert space (Mitra and Virk,

TABLE 1 | Results from the Shrec14 Dataset using the average model classification accuracy ± standard deviation over 100 trials.

Full results for SHREC14 dataset			
Method	**Train**	**Test**	**Model**
Multi-scale kernel (sigma = .5, sum of kernels)	.8942 ± .0142	.8938 ± .0464	Kernel SVM
Persistence landscapes (n = 5, r = 200)	**.9968 ± .0037**	**.9312 ± .0336**	**Ridge regression**
	.9302 ± .0098	.9186 ± .0417	SVM (RBF, c = 10)
	.9739 ± .0190	.9114 ± .0441	Random forest
Persistent images (p = 40, s = .5)	.7243 ± .0387	.7048 ± .0588	Ridge regression
	.9067 ± .0147	.8876 ± .0479	SVM (RBF, c = 1)
	.9855 ± .0092	.865 ± .0764	Random forest
Adcock-carlsson coordinates	.85 ± .0199	.7124 ± .0814	Ridge regression
	.8671 ± .0183	.6928 ± .0599	SVM (RBF, c = 50)
	.9147 ± .0299	.6976 ± .0899	Random forest
Template systems (d = 12, p = 1.1)	.9442 ± .0087	.9100 ± .0405	Ridge regression
	.9350 ± .0079	.9159 ± .0383	SVM (RBF, c = 1)
	.9483 ± .0214	.8874 ± .0481	Random forest
Adaptive template systems (CDER)	**.9937 ± .0078**	**.9169 ± .0395**	**Ridge regression**
	.9929 ± .0083	.9064 ± .0397	SVM (RBF, c = 10)
	.9729 ± .0200	.9164 ± .0422	Random forest

TABLE 2 | Results from the Protein Dataset using the average model classification accuracy ± standard deviation over 54 trials corresponding to the predefined indices of the dataset.

Full results for protein dataset			
Method	**Train**	**Test**	**Model**
Multi-scale kernel (sum of kernels)	.8294 ± .1063	.8803 ± .0702	Kernel SVM
Persistence landscapes	.9108 ± .0615	.9620 ± .0204	Ridge regression
	.9012 ± .0682	.9782 ± .0151	SVM (RBF)
	.9011 ± .0686	.9782 ± .0152	Random forest
Persistent images	.9011 ± .0682	.9758 ± .0165	Ridge regression
	.9007 ± .0684	.9782 ± .0151	SVM (RBF)
	.9008 ± .0685	.9782 ± .0151	Random forest
Adcock-carlsson coordinates	.9008 ± .0685	.9780 ± .0151	Ridge regression
	.9009 ± .0685	.9782 ± .0151	SVM (RBF)
	.9015 ± .0677	.9779 ± .0151	Random forest
Template systems	.9008 ± .0684	.9780 ± .0151	Ridge regression
	.9020 ± .0678	.9782 ± .0151	SVM (RBF)
	.9016 ± .0678	.9775 ± .0152	Random forest
Adaptive template systems	.9008 ± .0685	.9782 ± .0151	Ridge regression (CDER)
	.9007 ± .0684	.9782 ± .0151	SVM (CDER) (HDB)
	.9100 ± .0685	.9800 ± .0151	Random forest

2021), this ceases to be true if the number of points is allowed do vary (Wagner, 2019; Bubenik and Wagner, 2020). That said, partial featurization is still useful as we will demonstrate here.

3 FEATURIZATION METHODS

For each of the methods below, we start with a collection of persistence diagrams. A persistence diagram can be represented in either the birth-death plane or birth-lifetime plane—some methods will require birth-death coordinates and others will require birth-lifetime coordinates. The **birth-death plane** is the representation pair (x, y) where x is the time of birth, and y is the time of death of the feature in the persistence diagram. The **birth-lifetime plane** can be defined as the collection of points $(x, y - x)$, where (x, y) is in birth-death coordinates. In this manner, we define lifetime as the persistence $y - x$ of a feature (x, y). The persistence diagrams of a particular geometric object can be calculated in a variety of ways, which will be made explicit for each dataset at time of evaluation.

3.1 Multi-Scale Kernel

The Multi-Scale Kernel of Reininghaus et al. (2015) defines a Kernel over the space of persistence diagrams, which can then be used in various types of kernel learning methods. In general, a kernel k is by definition a symmetric and positive definite function of two variables. Mathematically, from Reininghaus et al. (2015), given a set X, a function $k : X \times X \to \mathbb{R}$ is a *kernel* if there exists a Hilbert space H, called the *feature space*, and a map $\Phi : X \to H$, called the *feature map*, such that $k(x, y) = \langle \Phi(x), \Phi(y) \rangle_H$ for all $x, y \in X$. The kernel induces a distance on X defined as

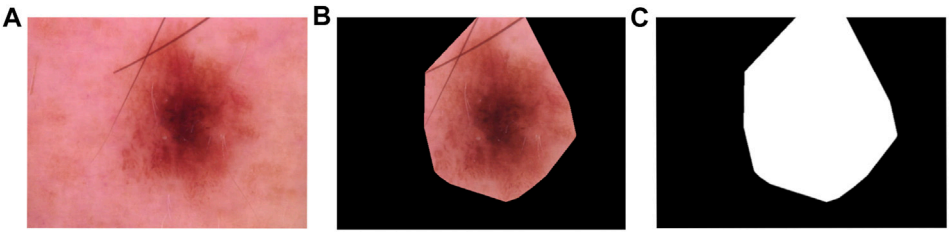


FIGURE 9 | (A) Example of skin lesion in HAM10000 **(B)** Skin lesion with mask **(C)** Mask only dataset.

TABLE 3 | Characteristics of each dataset. The column headings can be explained as such: Observations—number of observations in the dataset, Diagrams—the number of homological types used to compute persistence diagrams, Average Pairs—the average number of birth/death pairs across the set of persistence diagrams for a single observation in the original dataset, and Min/Max Pairs—the minimum and maximum number of birth/death pairs across the set of persistence diagrams for a single observation in the original dataset.

Dataset characteristics				
Dataset	**Observations**	**Diagrams**	**Average pairs**	**Min/Max pairs**
MNIST	70,000	8	1.15	0/7
SHREC14	300	2	14	1/29
Protein	1,357	2	346	3/500
MPEG7	1,400	2	205	1/500
HAM10000	10,000	18	5,783	13/32610

$$d_k(x,y) = \left(k(x,x) + k(y,y) - 2k(x,y)\right)^{\frac{1}{2}} = \left\|\Phi(x) - \Phi(y)\right\|_H.$$

Reininghaus et al. (2015) propose a multi-scale kernel on $\mathcal{D}$ as follows. Given $F, G \in \mathcal{D}$, the persistence scale space kernel k_σ is

$$k_\sigma(F,G) = \langle \Phi_\sigma(F), \Phi_\sigma(G) \rangle_{L^2(\Omega)} \tag{3}$$

where $\Phi_\sigma : \mathcal{D} \to L^2(\Omega)$ is the associated feature map, and $\Omega \subset \mathbb{R}^2$ is the closed half-plane above the diagonal. Deriving the solution of a distribution-analogue of the Heat equation with boundary conditions in Definition 1 of Reininghaus et al. (2015), the closed form expression of the multi-scale kernel is:

$$k_\sigma(F,G) = \frac{1}{8\pi\sigma} \sum_{p \in F, q \in G} e^{-\frac{||p-q||^2}{8\sigma}} - e^{-\frac{p-\overline{q}^2}{8\sigma}}$$

where if $q = (a,b)$, then $\overline{q} = (b,a)$.

The multi-scale kernel is shown to be stable w.r.t the 1-Wasserstein distance by Theorem 2 of Reininghaus et al. (2015), which is a desirable property for classification algorithms. However, by Theorem 3 of Reininghaus et al. (2015), the multi-scale kernel is not stable in the Wasserstein sense for $1 < p \leq \infty$.

3.2 Persistence Landscapes

Persistence landscapes are a mapping of persistence diagrams into a function space that is either a Banach space or Hilbert space (Bubenik, 2020). Advantages of persistence landscapes are that they are invertible, stable, parameter-free, and nonlinear. Persistence landscapes can be computed from a persistence diagram as follows.

From Bubenik (2020), for a persistence diagram $D = (a_i, b_i)_{i \in I}$, and for $a < b$, let

$$f_{(a,b)}(t) = \max(0, \min(a + t, b - t)) \tag{4}$$

and

$$\lambda_k(t) = \mathrm{kmax}\left\{f_{(a_i,b_i)}(t)\right\}_{i \in I} \tag{5}$$

with kmax as the kth largest element.

The *persistence landscape* is the sequence of piecewise linear functions, $\lambda_1, \lambda_2, \ldots : \mathbb{R} \to \mathbb{R}$. Bubenik shows desirable properties for working with persistence landscapes in statistical modeling, in particular that even if unique means do not exist in the set of persistence diagrams, persistence landscapes do have unique means and the mean landscape converges to the expected persistence landscape. **Figure 6** shows an example of persistence landscapes from the MPEG7 dataset, described in the data section.

3.3 Persistence Images

From Adams et al. (2015), persistence images are a mapping sending a persistence diagram to an integrable function, called a persistence surface. Fixing a grid on $\mathbb{R}^2$, the integral over this grid yields pixel values forming the persistence image. Advantages of persistence images include a representation in $\mathbb{R}^n$, stability, and ease of computation. When calculating the persistence image, a resolution, a distribution, and a weighting function are required as parameters. It is worth noting that the resolution (i.e., number of pixels) determines the number of features computed by the persistence image.

TABLE 4 | Results from the MNIST Dataset using the average model classification accuracy ± standard deviation over 10 trials.

Full results for MNIST dataset			
Topological method	**Training accuracy**	**Testing accuracy**	**Model type**
Multi-scale kernel (sum of kernels for 12,000 observations)	.6895 ± .0035	.6932 ± .0117	SVM
Persistence landscapes	.8844 ± .0004	.8786 ± .0019	Ridge regression
	.9231 ± .0004	.9180 ± .0018	SVM (RBF)
	.5814 ± .0098	.5828 ± .0098	Random forest
Persistent images	.8997 ± .0005	.8934 ± .0021	Ridge regression
	.9368 ± .0004	**.9199 ± .0023**	**SVM (RBF)**
	.6889 ± .0036	.6953 ± .0123	Random forest
Adcock-carlsson coordinates	.8590 ± .0010	.8547 ± .0030	Ridge regression
	.9525 ± .0004	**.9356 ± .0018**	**SVM (RBF)**
	.7214 ± .0092	.7170 ± .0097	Random forest
Template systems	.896 ± .0005	.8959 ± .0017	Ridge regression
	.9638 ± .0003	**.9477 ± .0015**	**SVM (RBF)**
	.6967 ± .0035	.6973 ± .0031	Random forest
Adaptive template systems	.8819 ± .0016	.8817 ± .0027	Ridge regression (GMM)
	.9515 ± .0021	**.9363 ± .0021**	**SVM (RBF) (GMM)**
	.6914 ± .0188	.6932 ± .0209	Random forest

More explicitly, let D be a persistence diagram in birth-lifetime coordinates. We take $\phi_u : \mathbb{R}^2 \to \mathbb{R}$ to be a differentiable probability distribution. Using, for instance, the Gaussian Distribution with mean u and variance σ^2 we have

$$\phi_u(x,y) = \frac{1}{2\pi\sigma^2} e^{-\left[(x-u_x)^2+(y-u_y)^2\right]/2\sigma^2}$$

The persistence surface $\rho_D : \mathbb{R}^2 \to \mathbb{R}$ is the function

$$\rho_D(z) = \sum_{u=(x,y-x)\in D} f(u)\phi_u(z)$$

with $f : \mathbb{R}^2 \to \mathbb{R}$, a nonnegative weighting function that is zero along the horizontal axis, continuous, and piecewise differentiable. The **persistence image** is then $I(\rho_D)p = \iint_p \rho_D dydx$, where integration is over the fixed grid on $\mathbb{R}^2$. This creates an image depicting high and low density areas in the defined grid, that are represented as a high-dimensional vector for use in machine learning algorithms. An example is shown in **Figure 7** taken from the MNIST dataset.

3.4 Adcock-Carlsson Coordinates: The Ring of Algebraic Functions on Persistence Diagrams

This method is explored by Adcock et al. (2016) where the authors highlight the fact that any persistence diagram with exactly n points can be described by a vector of the form $(x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$ where x_i denotes the birth of the i-th class and y_i the corresponding death time. Since this specific representation imposes an arbitrary ordering of the elements in the persistence diagram, one can more precisely identify the set of persistence diagrams with exactly n points with elements of the n-symmetric product of $\mathbb{R}^2$, denoted $Sp^n(\mathbb{R}^2)$.

The inclusions $Sp^n(\mathbb{R}^2) \hookrightarrow Sp^{n+1}(\mathbb{R}^2)$ thus produce an inverse system of affine coordinate rings

$$\cdots \to A\left[Sp^{n+1}(\mathbb{R}^2)\right] \to A\left[Sp^n(\mathbb{R}^2)\right] \to \cdots$$

which provide the basis for studying algebraic functions on the space of persistence diagrams.

With this setting in mind, the main goal of Adcock et al. (2016) is to determine free generating sets for the subalgebra of $A[Sp^\infty(\mathbb{R}^2)]$ comprised of elements which are invariant under adjoining a point of zero persistence to a persistence diagram. The following theorem is an answer to this question (see Theorem 1 Adcock et al. (2016)).

Theorem 1 The subalgebra of 2-multisymmetric functions invariant under adding points with zero persistence, is freely generated over $\mathbb{R}$ by the set of elements of the form

$$p_{a,b} = \sum_i (x_i + y_i)^a (y_i - x_i)^b$$

for integers $a \geq 0$ and $b \geq 1$.

These are the features we call *Adcock-Carlsson coordinates*.

Using this method we chose the following features for both the 0-dimensional and 1-dimensional persistence diagrams, as suggested in Adcock et al. (2016) when analyzing the MNIST data set: $\sum_i x_i(y_i - x_i)$, $\sum_i (y_{max} - y_i)(y_i - x_i)$, $\sum_i x_i^2 (y_i - x_i)^4$, $\sum_i (y_{max} - y_i)^2 (y_i - x_i)^4$.

3.5 Template Systems

The goal of this method is to find features for persistence diagrams by finding dense subsets of $C(\mathcal{D}, \mathbb{R})$. To accomplish this we will rely on the fact that given a persistence diagram $D \in \mathcal{D}$, and a continuous and compactly supported real-valued function on $\mathbb{W} = \{(x,y) \in \mathbb{R}^2 : 0 \leq x < y\}$, i.e. for $f \in C_c(\mathbb{W})$, we can define a continuous [see Theorem 26 Perea et al. (2019)] map $\nu(D) : C_c(\mathbb{W}) \to \mathbb{R}$ given by

$$\nu(D, f) := \sum_{\mathbf{x}\in D} f(\mathbf{x}).$$

The function $D \mapsto \nu(D, \cdot)$ defines a continuous injection $\mathcal{D} \hookrightarrow C_c(\mathbb{W})'$ into the topological dual of $C_c(\mathbb{W})$. The specific topology in the codomain is chosen so that ν is in fact continuous.

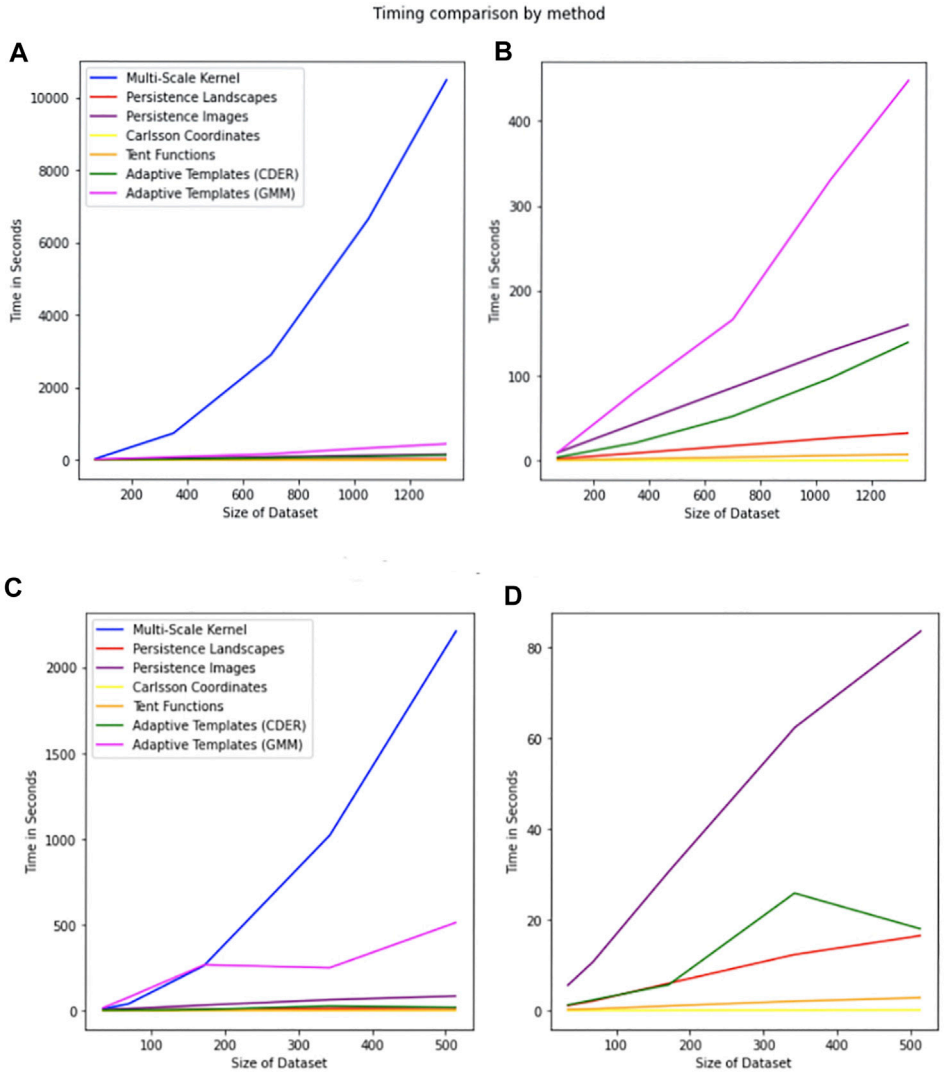


FIGURE 10 | Comparison of timing by method. The legend is the same for all plots. The x-axis represents the size of the dataset, and the y-axis represents the time in seconds required for calculation of all of the persistence diagrams associated with the dataset of the given size. **(A)** Timings for the MPEG7 Dataset including the Multi-Scale Kernel. **(B)** Timings for the MPEG7 Dataset excluding the Multi-Scale Kernel. **(C)** Timings for the Protein Dataset including all features. **(D)** Timings for the Protein Dataset excluding Multi-Scale Kernel and Adaptive Templates (GMM).

This injective featurization allows us to define a **template system** for $\mathcal{D}$ as a collection $\mathcal{T} \in C_c(\mathbb{W})$ such that $\mathcal{F}_{\mathcal{T}} := \{\nu(\cdot, f) : \mathcal{D} \to \mathbb{R} | f \in \mathcal{T}\}$ separates points. That is, if $D, D^{'} \in \mathcal{D}$ are distinct, then there exists $f \in \mathcal{T}$ for which $\nu(D, f) \neq \nu(D^{'}, f)$.

The advantage of working with these template systems is that they can be used to approximate real-valued functions on the space of persistence diagrams as proven by the following theorem [see Theorem 29 Perea et al. (2019)].

Theorem 2 Let $\mathcal{T} \subset C_c(\mathbb{W})$ Be a Template System for $\mathcal{D}$, let $\mathcal{C} \subset \mathcal{D}$ Be Compact, and let $F : \mathcal{C} \to \mathbb{R}$ Be Continuous. Then for Every $\epsilon > 0$ There Exist $N \in \mathbb{N}$, a Polynomial $p \in \mathbb{R}[x_1, \ldots, x_N]$ and Template Functions $f_1, \ldots, f_N \in \mathcal{T}$ so That

$$\left| p(\nu(D, f_1), \ldots, \nu(D, f_N)) - F(D) \right| < \varepsilon$$

for every $D \in \mathcal{C}$.

That is, the collection of functions of the form $D \to p(\nu(D, f_1), \ldots, \nu(D, f_N))$, is dense in $C(\mathcal{D}, \mathbb{R})$ with respect to the compact-open topology.

Even though this theorem provides the theoretical underpinnings to guarantee the existence of solutions to supervised machine learning problems, it does not provide the specifics for doing so. In particular, one question to answer is how to choose suitable families of template functions. In our evaluations we will explore both prescribed families for template systems, as well as data-driven or adaptive ones.

In the prescribed front we have the tent functions described below. See also **Figure 8**. In the birth-lifetime plane, and given a point $\mathbf{x} = (a, b) \in \mathbb{W}$ and a discretization scale $0 < \delta < b$, the associated tent function on $\mathbb{W}$ is given by

TABLE 5 | Results from the HAM10000 Dataset using the average model classification accuracy ± standard deviation over 10 trials.

Results for HAM10000 dataset			
Topological method	**Training accuracy**	**Testing accuracy**	**Model type**
Multi-scale kernel		Did not run	
Persistence landscapes	.8347 ± .0022	.6881 ± .0074	Ridge regression
	.6695 ± 0	.6692 ± 1.2*e* − 16	SVM (RBF, c = 1)
	.6695 ± 0	.6692 ± 1.2*e* − 16	Random forest
Persistent images (pixels = 20, spread = 1)	.7417 ± .0017	.6371 ± .0671	Ridge regression
	.7122 ± .0012	.6895 ± .0031	SVM (RBF, c = 1)
	.6695 ± 0	.6692 ± 1.2*e* − 16	Random forest
Adcock-carlsson coordinates	.6719 ± .0007	.6696 ± .0025	Ridge regression
	.6801 ± .0009	.6710 ± .0019	SVM (RBF)
	.6695 ± 0	.6692 ± 1.12*e* − 16	Random forest
Template systems (d = 10, p = 1.5)	.7193 ± .0015	.6987 ± .0041	Ridge regression
	.7830 ± .0024	.7303 ± .0054	SVM (RBF, c = 5)
	.6695 ± 0	.6692 ± 1.2*e* − 16	Random forest
Adaptive template systems		Did not run	

$$g_{\mathbf{x},\delta}(x,y) = \left|1 - \frac{1}{\delta}\max\{|x-a|, |y-b|\}\right|_+$$

where $|r|_+ = \max\{0, r\}$. As $\delta < b$, this function has support in the compact box $[a-\delta, a+\delta] \times [b-\delta, b+\delta] \subset \mathbb{W}$. Given a persistence diagram $D \in \mathcal{D}$ in birth-death coordinates, the value of the tent function is

$$G_{\mathbf{x},\delta}(D) = \sum_{(x,y)\in D} g_{x,\delta}(x, y-x).$$

3.6 Adaptive Template Systems

The Adaptive Template Systems methodology of Polanco and Perea (2019a) concerns itself with improving and furthering some of the work presented in Perea et al. (2019). The goal is to produce template systems that are attuned or adaptive to the input data set and the supervised classification problem at hand. One shortcoming of template systems, like tent functions, when applied to Theorem 2 is that without prior knowledge about the compact set $\mathcal{C} \subset \mathcal{D}$, the number of template functions that carry no information relevant to the problem can be high. By reducing this overhead, adaptive templates improve the computation times and accuracy in some specific problems.

The relationship between template systems and adaptive template systems is demonstrated in **Figure 4**, showing the adaptive template systems depend on density of data. To do so, given a compact set $\mathcal{C} \subset \mathcal{D}$ we consider the set $S = \bigcup_{D\in\mathcal{C}} D \subset \mathbb{W}$ along with different algorithms such as Gaussian mixture models (GMM) (Reynolds, 2009), Hierarchical density-based spatial clustering of applications with noise (HDBSCAN) (Campello et al. ,2013) and Cover-Tree Entropy Reduction (CDER) (Smith et al., 2017) to define a family of ellipsoidal domains $\{\mathbf{z} \in \mathbb{R}^2 : (\mathbf{z}-\mathbf{x})^* A (\mathbf{z}-\mathbf{x}) \le 1\}$ in $\mathbb{W}$, fitting the density distribution of S. Here A is a 2×2 symmetric matrix and $\mathbf{x} \in \mathbb{R}^2$.

Once this family of ellipsoidal domains is computed, we use them to define the following adaptive template functions

$$f_A(\mathbf{z}) = \begin{cases} 1 - (\mathbf{z}-\mathbf{x})^* A (\mathbf{z}-\mathbf{x}) & \text{if } (\mathbf{z}-\mathbf{x})^* A (\mathbf{z}-\mathbf{x}) < 1 \\ 0 & \text{if } (\mathbf{z}-\mathbf{x})^* A (\mathbf{z}-\mathbf{x}) \ge 1 \end{cases}$$

3.7 Other Approaches

The featurization methods presented in this section are by no means an exhaustive list of what is available in the literature. Here are some others that the interested reader may find useful:

- The **Persistent homology rank functions** of Robins and Turner (2016) are similar in spirit to persistent landscapes, in that they provide an injective inclusion of $\mathcal{D}_0$ into a Hilbert space of functions where techniques like functional Principal Component Analysis are available. Indeed, for a filtration $\mathcal{X}$, its n-th persistent rank function is defined as

$$\begin{array}{ccc} \mathbb{W} & \to & \mathbb{R} \\ (a,b) & \mapsto & \beta_n^{a,b}(\mathcal{X}) = \operatorname{rank}(H_n(X_a) \to H_n(X_b)). \end{array}$$

This is equivalent, for a persistence diagram $D \in \mathcal{D}_0$, to defining the function

$$\begin{array}{ccc} \mathbb{W} & \to & \mathbb{R} \\ (a,b) & \mapsto & \#\{(x,y) \in D : x \le a \text{ and } y > b\} \end{array}$$

where # is multiset cardinality. The Hilbert space in question is the weighted L^2-space $L^2(\mathbb{W}, \phi)$. Here $\phi : [0,\infty) \to [0,\infty)$ satisfies $\int_0^\infty \phi(t)dt < \infty$, and the inner product of rank functions is

$$\langle f, g \rangle_\phi = \int_{\mathbb{W}} f(x,y) g(x,y) \phi(y-x) dx dy.$$

This approach has shown to be effective in analyzing point processes, and sphere packing patterns.

The **Persistent curve** (Chung et al., 2018; Giusti et al., 2015) provides another functional summary closely related to persistent rank functions. Specifically, for a persistence diagram $D \in \mathcal{D}_0$, its persistence curve (Chung et al., 2018) is the function

$$\begin{array}{ccc} [0,\infty) & \to & [0,\infty) \\ t & \mapsto & \#\{(x,y) \in D : x \le t < y\}. \end{array}$$

Discretizations of these curves have been useful in computer vision tasks (Chung and Lawson, 2020), as well as in neuroscience applications (Giusti et al., 2015).

Other **kernel methods**, besides the Multi-Scale kernel of Reininghaus et al. (2015), have appeared in the literature. They correspond to the following choices of kernel function $k : \mathcal{D}_0 \times \mathcal{D}_0 \to \mathbb{R}$. The *Persistence Weighted Gaussian Kernel* of Kusano et al. (2016) is defined as

$$k_{PWG}(D, D') = \sum_{\substack{\mathbf{x}=(x,y)\in D \\ \mathbf{x}'=(x',y')\in D'}} \arctan\left(C|y-x|^p\right) \cdot \arctan\left(C|y'-x'|^p\right) \cdot e^{-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2\sigma^2}}$$

for parameters $C, p, \sigma > 0$, while the *Sliced Wassertein Kernel* of Carriere et al. (2017) takes the form

$$k_{SW}\left(D, D'\right) = \exp^{\frac{-d_{SW}\left(D,D'\right)}{2\sigma^2}}$$

where $d_{SW}(\cdot,\cdot)$ is the so-called sliced Wasserstein distance between persistence diagrams [see Eq. 2 and Definition 3.1 of Carriere et al. (2017)]. If instead one uses the Fisher Information metric $d_{FIM}(\cdot,\cdot)$ [see Eq. 3 of Le and Yamada (2018)], then the result is the *Persistence Fisher Kernel*

$$k_{PF}\left(D, D'\right) = e^{-td_{FIM}\left(D,D'\right)}, t > 0.$$

Persistence diagrams as features for **deep neural networks** have also been studied recently. In particular, the *PersLay* framework of Carrière et al. (2020) leverages the Deep Sets architecture of Zaheer et al. (2017) to implement layers that can process persistence diagrams. Specifically, layers of the form:

$$D \mapsto \mathrm{op}\left(\{\omega(\mathbf{x})\phi(\mathbf{x})\}_{\mathbf{x}\in D}\right)$$

where $\mathrm{op}(\cdot)$ is a permutation invariant operator (e.g., max, min, sum, etc), $\omega : \mathbb{R}^2 \to \mathbb{R}$ is a weight function, and $\phi : \mathbb{R}^2 \to \mathbb{R}^q$ is a representation function. By optimizing ω and ϕ in a parametric family—i.e., $\omega = \omega_u$ and $\phi = \phi_v$—the training of the network can lead to vectorizations attuned to specific learning tasks.

4 DATASETS

The five different datasets considered in this work were chosen from a collection of experiments presented in the literature of topological methods for machine learning. We acknowledge that this selection is inherently bias towards datasets with favorable performance with regards to specific topological methods. Nevertheless, we counterbalance this by applying all the evaluated featurization methods to all the data sets here considered and compare the classification results across all the presented methodologies. This comparative work showcases how the variation between methods results in the need for the user to find suitable combination of featurization methods and parameter tuning to obtain optimal results in a given dataset. As such, readers should view this as a resource for their own analysis, and not as a recommendation for specific techniques.

For all datasets and methods, parameter tuning was done using a grid search method on a subset of data that was not used to report final results, and parameters were chosen based on performance of a ridge regression model, random forest and support-vector machine (SVM) model. It is worth noting a weakness of the analysis in that the same parameters were used in the feature set calculation for all reported models, and run with a single split. This was due to time required for feature calculation.

The ridge regression and random forest classifier were run with default parameters, and the support-vector machine was run using the radial basis function (RBF) with some tuning on the cost parameter (C). The exception is for the Multi-Scale Kernel feature set—we only fit a support-vector machine model. It is important to highlight that results regarding ridge regression with (polynomial and radial basis function) kernel methods are not included in this work as they produce increased computational times while the classification results do not improve significantly compared to the one presented here. Each dataset was sampled for a 10 or 100 trials depending on size, with the exception of the Protein Classification Dataset, which included indices for predefined problems.

Random forest classifiers as presented in Breiman (2001) are used to solve the same classification problems presented for each data set. Parameters such as number of trees in each forest and the size of each tree are chosen based on performance and tuned on the testing set.

4.1 MNIST

The MNIST dataset from LeCun and Cortes (1999) is a database of 70,000 handwritten single digits of numbers zero to nine. An example image from the MNIST database is shown in **Figure 7**.

The calculation of persistence diagrams for the MNIST dataset is as in Adcock et al. (2016). This method creates a base of 8 different persistence diagrams to use in the application of methods. The persistence diagrams are calculated using a "sweeping" motion in one of four directions: left to right, right to left, top to bottom, or bottom to top, corresponding to the 0-dimensional and 1-dimensional persistence diagrams. To compute this filtration, pixels are converted to a binary response with intensity calculated based on position. This has the effect that depending on the direction of sweep, features will have different birth and death times, providing distinct information for each direction. The number of topological features available for model fitting is dependent on the method. For the Persistent Images, Persistence Landscapes, and Template Systems there are eight features each. The Multi-Scale Kernel produces eight different kernel matrices, and for Adcock-Carlsson Coordinates, 32 different features were computed from these persistence diagrams.

Figure 5 shows the various calculations of persistence diagrams for an example number eight. Both 0-dimensional and 1-dimensional persistence diagrams were used for the MNIST dataset, noting that

some observations did not have 1-dimensional persistence diagrams, so these observations were filled with a single diagram of birth-death coordinate of [0,.01].

For the MNIST dataset, a random sample of 1,000 images was used to tune parameters, with 80% used for the training portion, and 20% used for the testing portion. We used the set of 60,000 images corresponding to the training set of MNIST to create our own training and testing sets for model fitting and evaluation. For this set of 60,000, 10 trials were run with an 80% training and 20% testing split to determine model performance.

4.2 SHREC14

We evaluated the SHREC 2014 dataset (Pickup et al., 2014) in the same manner as the authors of Polanco and Perea (2019a). To compute the topological features, the authors of Reininghaus et al. (2015) describe using a heat kernel signature to compute persistence diagrams for both the 0-dimensional and 1-dimensional persistence diagrams. The dataset consists of 15 different labels, corresponding to five different poses for the three classes of male, female, and child figures.

As noted in Polanco and Perea (2019a), parameters in the dataset define different problems due to a different calculation of the heat kernel signature, and for this evaluation we focused on the problem with the highest accuracy as reported in Polanco and Perea (2019a).

For the SHREC14 dataset, a random sample of 90 images (30% of the data) was used to tune the model and determine appropriate model parameters. The remaining 210 observations were split into 80% training and 20% testing for 100 trials to report final model fit. Persistence diagrams for 0-dimensional homology and 1-dimensional homology were computed for this dataset.

Table 1 shows complete results for the SHREC 2014 dataset.

4.3 Protein Classification

We use the Protein Classification Benchmark dataset PCB00019 Sonego et al. (2007) as another type of data to evaluate the topological methods above. This specific set contains information for 1,357 proteins corresponding to 55 classification problems, and we reported on 54 of the problems using one to tune parameters. The training and testing index were provided, and the mean accuracy was reported for both training and testing sets using these indices. **Table 2** shows results from our experiments using the training and testing indices provided in the original dataset.

Persistence diagrams for this dataset were computed for each protein by considering the 3-D structure [provided in wwPDB consortium (2018)] as a point cloud in $\mathbb{R}^3$. This point cloud was built using the x, y and z position of each atom in the molecule at hand. With this information the persistent 0-dimensional and 1-dimensional homology is computed using Ripser from Tralie et al. (2018).

4.4 MPEG7

The mpeg-7 dataset from Bai et al. (2009) is a database of object shapes in black and white, with 1,400 shapes in 70 classes. An example from the original dataset is shown in **Figure 4** along with the contour as described below.

To compute persistence diagrams, first the image contour is computed by placing observations from the point cloud into a sequence. The distance curve is computed as the distance from the center of the sequence. Sublevel set persistence is taken using the computed distance curves as point cloud data. Persistence diagrams for both 0-dimensional and 1-dimensional homology were computed for this dataset.

We used this dataset for a timing comparison of featurization methods from persistence diagrams. We do not report on the results of this dataset. An example notebook of MPEG7 is provided using only four shapes—apple, children, dog, and bone. This approach is due to the initial difficulty in getting accurate models for the full dataset. Due to the small number of samples (80 total) and lack of repeated sampling, the estimates provided for this dataset are not stable and are not reported.

4.5 HAM10000

The HAM10000 dataset provided by Philipp Tschandl et al. (2018) is a collection of 10,000 images of skin lesions with one 7 potential classifications: Actinic Keratoses and Intraepithelial Carcionma, Basal cell carcinoma, Benign keratosis, Dermatofibroma, Melanocytic nevi, Melanoma, Vascular skin lesions. A total of 18 persistence diagrams for this set were calculated using the methods outlined in Chung et al. (2018), 9 corresponding to the 0-dimensional homology and 9 corresponding to the 1-dimensional homology.

To obtain such diagrams, first a mask is computed by implementing the methodology proposed in Chung et al. (2018). In general terms, this method creates a filtration of binary images obtained from different thresholds to convert the gray scale image into a binary one. Once this binary filtration is obtained, the center most region of the image is computed using the "persistence" of each point in the binary filtration. An example image and this process is shown in **Figure 9**.

Once the mask is computed it is applied to the original image and then it is converted into three different color models: RGB, HSV , and XYZ. Each color model is split into their corresponding channels, and for each channel we use sublevel set filtration to obtain 0-dimensional and 1-dimensional persistence diagrams. In total, for each image on the data set we obtain 18 persistence diagrams, 9 in homological dimension 0 and 9 for homological dimension 1.

To tune the models, a random sample of 250 images were taken a ridge regression, random forest, and support vector machine model were fit to determine parameters. The remaining 9,750 images were split into an 80% training and 20% testing set to report final results.

To evaluate the HAM10000 dataset, due to the large number of birth and death pairs in each persistence diagram, subsampling of persistent features was required. Each observation in a data set, for example an image, will yield 18 persistence diagrams corresponding to homological features in that observation. In the HAM10000 dataset, there was an average of 5,783 birth-death pairs in each persistence diagram. This was an issue to complete computation for the vectorization methods, even for adaptive templates, so each persistence diagram was subsampled as follows.

The method of subsampling is two steps: Highly persistent features were always included, and a uniform random sampling method (without replacement) was used to sample the remaining

points. The threshold for feature lifetime and number of points to sample was determined by using parameters that preserve the distribution of points in each persistence diagram. As a result, features in each persistence diagram with a lifetime of five or more were automatically included, and 5% of the rest of the points were also included. This resulted in sampled persistence diagrams with an average of 290 points each (**Table 3**).

5 USER GUIDE

5.1 Available Functions

As part of the available code, a function for each method is included. Each function requires two sets of persistence diagrams, a training set and a testing set, and parameters specific to the function. The function returns two feature sets for that method, corresponding to the training and test set respectively. Each function also prints the time in seconds taken at the end of each run. In this section of the user guide each function is described, along with the required parameters for the function.

The **Multi-Scale Kernel** feature matrix can be computed using the function kernel_features or fast_kernel_features. It is recommended to use fast_kernel_features due to computation time. Both functions require a parameter sigma, denoted as s in the function with a default value of 4. In Reininghaus et al. (2015) this parameter is referred to as the scale parameter. From the closed form distribution of the Multi-Scale Kernel

$$k_\sigma(F,G) = \frac{1}{8\pi\sigma}\sum_{p\in Fq\in G} e^{-\left|\left|p-q\right|\right|^2_{8\sigma-e^{-\left|\left|p-\overline{q}\right|\right|^2_{8\sigma}}}} \tag{6}$$

we note that as sigma, σ, increases the function decreases. Increasing sigma results in a less diffuse kernel matrix, while decreasing sigma results in a more diffuse kernel matrix.

Due to time required for the Multi-Scale Kernel, there are two additional sets of functions that use Numba (Siu Kwan Lam and Seibert, 2015) for significantly faster computation. In the current implementation, these are not able to be combined with multi-core processing (MPI for example), and have a different format than the other functions included. These functions are provided in the github repository for this project, and were used to compute results for the Multi-Scale Kernel for the MNIST dataset.

The **Persistence Landscapes** features can be computed using the function landscape_features. The Multi-Scale Kernel function, landscape_features requires two parameters: the number of landscapes, n and resolution, r. The number of landscapes parameter, n, controls the number of landscapes used, and the resolution, r, controls the number of samples used to compute the landscapes. The default parameters for n is 5 and r is 100.

The **Persistence Images** can be computed using the function persistence_image_features. The persistence_image_features function requires two parameters, pixels and spread. The pixels, p is a pair that defines the number of pixels along each axis. The spread, s, is the standard deviation of the gaussian kernel used to generate the persistence image. It is worth noting that the implementation here uses the gaussian kernel, however, other distributions could be chosen so that s would correspond to parameters specific to the chosen distribution. Additionally, the weighting function is constant for this implementation. Increasing spread increases the variance for each distribution, resulting in larger "hot spots". Increasing pixels provides a smoother distribution, whereas decreasing pixels yields a less smooth distribution. Note that increasing pixels increases computation time. This is demonstrated in **Figure 7** in the methods section.

The **Adcock-Carlsson Coordinates** features can be computed using the function carlsson_coordinates, does not require any parameters. This function returns four different features for every type of persistence diagram provided. So for datasets that have persistence diagrams corresponding to 0-dimensional and 1-dimensional homology, 8 features are returned for machine learning. The features returned correspond to the four coordinates calculated in Adcock et al. (2016), and are:

$$\sum_i x_i(y_i - x_i),$$
$$\sum_i (y_{max} - y_i)(y_i - x_i)$$
$$\sum_i x_i^2(y_i - x_i)^4,$$
$$\sum_i (y_{max} - y_i)^2(y_i - x_i)^4$$

The **Template Systems** features can be computed using the function tent_features, and has a choice of two parameters: d, which defines the number of bins in each axis and padding, which controls the padding around the collection of persistence diagrams. This function returns a training and testing set. This function computes the tent features from Perea et al. (2019).

The **Adaptive Template Systems** features can be called with the function adaptive_features, and requires the labels for the training set. Users can choose three different types of Adaptive Templates: Gaussian Mixture Modeling (GMM), Cover-Tree Entropy Reduction (CDER), and Hierarchical density-based clustering of applications with noise (HDBSCAN). The parameter d refers to the number of components when using the GMM model type. This would be minimally the number of classes in your data, and ideally represents closer to the number of distributions in the data that correspond to each observation. Details on these methods can be found in Polanco and Perea (2019a), as well as the original references linked in the methods section. During this evaluation, we evaluated adaptive templates using both GMM and CDER methods, but did not formally evaluate HDBSCAN. HDBSCAN was difficult to formally assess as we had difficulty with completion of the algorithm for some datasets. For those datasets we were able to complete, we did not notice an improvement over other adaptive methods.

6 RESULTS

One consideration we must make before analysing the results comes from the computation of **Multi-Scale Kernel** features. As explained for each dataset in Section 4, more often than not we will compute

multiple persistent diagrams per data point in a given data set. Such persistent diagram correspond to 0-dimensional, 1-dimensional, and in some cases 2-dimensional persistent homology (see details in **Section 2**). To compute Multi-Scale Kernel as given by Eq. 6 we require pairs of persistent diagrams. Since this multi-scale kernel provides a notion of similarity between persistent diagrams (Reininghaus et al., 2015) we require it to be computed between diagrams corresponding to the same dimension homology and method type. For example, the kernel matrix that corresponds to the 0-dimensional homology of a data set is computed using the persistence scale space kernel between two sets of persistence diagrams that represent the 0-dimensional homology. This means that for a dataset that has sets of 0-dimensional homology persistence diagrams and 1-dimensional homology persistence diagrams, two kernel matrices were returned (one per each dimension).

The kernel matrix used in our models is the sum of available kernels, and differs based on the persistence diagrams available for each dataset. While this does improve accuracy significantly over individual kernel matrices, other methods of combining kernel features were not explored in this paper, but is available in Gönen and Alpaydin (2011) for the interested reader. The available parameter, sigma, is consistent across all types of diagrams for our evaluation.

For each of the other methods, **Persistence Landscapes**, **Persistence Images**, **Adcock-Carlsson Coordinates**, **Template Systems**, and **Adaptive Template Systems**, each feature matrix was constructed for the relevant set of diagrams, and all topological types were used in fitting the same model.

The datasets used in this analysis were of varying size, both in terms of observations and the size of sets of persistence diagrams. As noted in the descriptions of data, the types of persistence diagrams calculated also differs. A summary of characteristics for each dataset is included in **Table 3**.

6.1 MNIST

The Multi-Scale Kernel features calculated yielded eight different kernel matrices, and the final kernel matrix was calculated using the unweighted summation of these kernels as in Gönen and Alpaydin (2011). Due to the time needed for computation of the Multi-Scale Kernel, a smaller set of 12,000 observations was used to report final results and a version of the kernel computation using Numba with a gpu target was necessary.

Table 4 shows complete results for the MNIST analysis. Four different methods (highlighted on the table) provided similar results for the MNIST dataset, and we note the SVM model had higher accuracy in each case. This table, and all subsequent results tables, include the method used to construct topological features, training and test accuracy, and model and parameters used for evaluation.

6.2 SHREC14

Results are reported in **Table 1**. Adaptive Template Systems and Persistence Landscapes were the two methods with highest classification accuracy on the test dataset, with Template Systems and the Multi-Scale Kernel performing nearly as well.

6.3 Protein Classification

Nearly all of the topological methods in this paper provided similar classification accuracy for this dataset. We observe the testing accuracy as higher than the training accuracy for this dataset, and the results are similar to those in Polanco and Perea (2019b). The Multi-Scale Kernel though did not perform as well and as shown in **Figure 10** is the most computationally intensive. Results are reported in **Table 2**.

6.4 HAM10000

Due to run time for the large number of points in each persistence diagram, even after subsampling, results were not reported for the Multi-Scale Kernel or Adaptive Template Systems.

Results are listed in **Table 5**. The HAM10000 dataset presented the largest computational challenge during this review, and is a continued area of research.

7 COMPUTATION TIME OF FEATURES

Formal timings were captured for all features for the 0-dimensional persistence diagrams for the MPEG7 and Protein Datasets. A comparison of timings is in **Figure 10**. The timing reported is for the generation of features from one type of persistence diagram for a dataset of that size. This means when computing a training feature set and testing feature set for multiple types of persistence diagrams, the expected time to generate features can be significantly longer. For example, in the MNIST dataset we compute four different types of persistence diagrams with both 0-dimensional and 1-dimensional homology, giving eight sets of features that can be generated for the sets of persistence diagrams for that dataset. Specific to the multi-scale kernel method, the timing reported is for a symmetric feature matrix that is nxn, where n is the number of observations in the dataset. This means the training feature set requires less computation time than a testing feature set of comparable size.

Additionally, during the review of these methods, we did not encounter significant issues with model fitting, hence formal timings were not recorded for this portion of the analysis. Conclusions from these timings are addressed in the discussion section.

7.1 Data Availability

The datasets, persistence diagrams (or code to compute the diagrams), and all other associated code for this study can be found in the machine learning methods for persistence diagrams github repository https://github.com/barnesd8/machine_learning_for_persistence.

For each of the five datasets, the following code is available:

- A jupyter notebook that loads and formats the persistence diagrams including images and does a preliminary model fitting on a subset of the data
- A python script that calculates the persistence diagrams from the original dataset - some of these are written using MPI depending on the size of the dataset
- A python script that fits models for random samples of the data to get mean estimates of accuracy for both the training and test dataset

These scripts reference modules included in the github repository, including a persistence methods script that calculates features from persistence diagrams for a training and test dataset. This uses a combination of other available functions and functions written specifically for this paper.

The **Template Systems** and **Adaptive Template Systems** methods use functions from https://github.com/lucho8908/adaptive_template_systems, which is the corresponding code repository for Polanco and Perea (2019a). The available methods in our extension include Tent Functions and Adaptive Template Functions (GMM, CDER, and HDBScan methods).

The **Adcock-Carlsson Coordinates** method is a function developed specifically for this paper, and includes the calculation of the 4 different features used in our analysis. The **Persistence Landscape** method uses the persistence landscape calculation from the Gudhi Package Dlotko (2021). The **Multi-Scale Kernel Method** has two included implementations, one is from Nathaniel Saul (2019) and is slower to compute, while the other is a faster implementation that can be used on larger datasets. All of the results in this paper were reported using the implementation written specifically for this paper. The **Persistence Images** features are also from Nathaniel Saul (2019). Additionally, many functions from Pedregosa et al. (2011) are used throughout.

8 DISCUSSION

Adcock-Carlsson Coordinates, Tent Functions, and Persistence Landscapes scale well, and perform well even for large datasets. It is of note though that parameter choice will affect computation time. This was especially notable in the Template Features (Tent Functions) computation time. As the number of tent functions is increased, the time to compute features also increases. We observed a superlinear increase, however, even with this increase computation time was not a barrier for analysis.

Persistence Images and Adaptive Template Functions do not scale or perform as well, however, do provide good featurizations for accurate models and should be considered depending on the dataset. Specifically, the Adaptive Template Functions was not completed for the full HAM10000 dataset due to computation time.

When using these methods, it should be of note that the Multi-Scale Kernel method is computationally intensive, and does not scale well. Additionally, the accuracy achieved is not better than other methods for the datasets in this paper.

9 CONCLUSION

This paper reviews six methods for topological feature extraction for use in machine-learning algorithms. Persistence Landscapes, Adcock-Carlsson Coordinates, Template Systems, and Adaptive Template Systems perform consistently with minimal differences between datasets and types of persistence diagrams. These methods are also less expensive in terms of execution time. A main contribution of this paper is the availability of datasets, persistence diagrams, and code for others to use and contribute to the research community.

AUTHOR CONTRIBUTIONS

JP coordinated the experimental framework and co-wrote the paper with significant contributions to the background sections. LP curated datasets, contributed to the software code and co-wrote the paper with significant contributions to the methods sections. DB curated datasets, ran the experimental results, developed the accompanying software and co-wrote the paper with significant contributions in most areas. All authors reviewed results and the manuscript, with DB approving and reviewing the final version.

FUNDING

This work was partially supported by the National Science Foundation through grants CCF-2006661, and CAREER award DMS-1943758.

ACKNOWLEDGMENTS

This work was supported in part by Michigan State University through computational resources provided by the Institute for Cyber-Enabled Research. Special thanks to Elizabeth Munch for providing background information for computing persistence diagrams using a directional transform. We also thank the reviewers for critical feedback that helped improved our review.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frai.2021.681174/full#supplementary-material

REFERENCES

Adams, H., Chepushtanova, S., Emerson, T., Hanson, E., Kirby, M., Motta, F., et al. (2015). *Persistence Images: A Stable Vector Representation of Persistent Homology*. [Dataset]. http://arxiv.org/abs/1507.06217.

Adcock, A., Carlsson, E., and Carlsson, G. (2016). The Ring of Algebraic Functions on Persistence Bar Codes. *Homology, Homotopy Appl.* 18, 381–402. doi:10.4310/hha.2016.v18.n1.a21

Bai, X., Yang, X., Latecki, L. J., Liu, W., and Tu, Z. (2009). Learning Context Sensitive Shape Similarity by Graph Transduction. *IEEE Trans. Pattern Anal. Mach. Intell.* 32, 861–74. doi:10.1109/TPAMI.2009.85

Bendich, P., Marron, J. S., Miller, E., Pieloch, A., and Skwerer, S. (2016). Persistent Homology Analysis of Brain Artery Trees. *Ann. Appl. Stat.* 10, 198–218. doi:10.1214/15-AOAS886

Blumberg, A. J., Gal, I., Mandell, M. A., and Pancia, M. (2014). Robust Statistics, Hypothesis Testing, and Confidence Intervals for Persistent Homology on Metric Measure Spaces. *Found. Comput. Math.* 14, 745–789. doi:10.1007/s10208-014-9201-4

Breiman, L. (2001). Random Forests. *Machine Learn.* 45, 5–32. doi:10.1023/A:1010933404324

Bubenik, P., Hull, M., Patel, D., and Whittle, B. (2020). Persistent Homology Detects Curvature. *Inverse Probl.* 36, 025008. doi:10.1088/1361-6420/ab4ac0

Bubenik, P. (2020). The Persistence Landscape and Some of its Properties. *Topological Data Anal.* 15, 77–102. doi:10.1007/978-3-030-43408-3_4

Bubenik, P., and Wagner, A. (2020). Embeddings of Persistence Diagrams into hilbert Spaces. *J. Appl. Comput. Topology* 4, 339–351. doi:10.1007/s41468-020-00056-w

Campello, R. J. G. B., Moulavi, D., and Sander, J. (2013). "Density-based Clustering Based on Hierarchical Density Estimates," in *Advances in Knowledge Discovery and Data Mining*. Editors J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu (Berlin, Heidelberg: Springer Berlin Heidelberg), 160–172. doi:10.1007/978-3-642-37456-2_14

Carlsson, G. (2014). Topological Pattern Recognition for point Cloud Data. *Acta Numerica* 23, 289–368. doi:10.1017/s0962492914000051

Carrière, M., Chazal, F., Ike, Y., Lacombe, T., Royer, M., and Umeda, Y. (2020). "Perslay: a Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures," in International Conference on Artificial Intelligence and Statistics (PMLR), 2786–2796.

Carriere, M., Cuturi, M., and Oudot, S. (2017). "Sliced Wasserstein Kernel for Persistence Diagrams," in International Conference on Machine Learning PMLR), 664–673.

Chazal, F., De Silva, V., and Oudot, S. (2014). Persistence Stability for Geometric Complexes. *Geom Dedicata* 173, 193–214. doi:10.1007/s10711-013-9937-z

Chung, Y.-M., and Lawson, A. (2020). *Persistence Curves: A Canonical Framework for Summarizing Persistence Diagrams*.[Dataset]. http://arxiv.org/abs/1904.07768.

Chung, Y., Hu, C., Lawson, A., and Smyth, C. (2018). "Topological Approaches to Skin Disease Image Analysis," .in 2018 IEEE International Conference on Big Data (Big Data), 100–105.

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2007). Stability of Persistence Diagrams. *Discrete Comput. Geom* 37, 103–120. doi:10.1007/s00454-006-1276-5

Dlotko, P. (2021). "Persistence Representations," in *GUDHI User and Reference Manual*. 1 edn (GUDHI Editorial Board), 3.4.

Edelsbrunner, H., and Harer, J. (2010). *Computational Topology: An Introduction*, Vol. 69. Providence, RI: American Mathematical Soc.

Giusti, C., Pastalkova, E., Curto, C., and Itskov, V. (2015). Clique Topology Reveals Intrinsic Geometric Structure in Neural Correlations. *Proc. Natl. Acad. Sci. USA* 112, 13455–13460. doi:10.1073/pnas.1506407112

Gönen, M., and Alpaydin, E. (2011). Multiple Kernel Learning Algorithms. *J. Machine Learn. Res.*, 2211–2268.

Gromov, M. (2007). *Metric Structures for Riemannian and Non-riemannian Spaces*. Boston: Springer Science & Business Media.

Hatcher, A. (2002). *Algebraic Topology*. Cambridge University Press.

Kaczynski, T., Mischaikow, K., and Mrozek, M. (2004). *Cubical Homology*. New York, NY: Springer New York, 39–92. doi:10.1007/0-387-21597-2_2

Kuhn, H. W. (1960). Some Combinatorial Lemmas in Topology. *IBM J. Res. Dev.* 4, 518–524. doi:10.1147/rd.45.0518

Kusano, G., Hiraoka, Y., and Fukumizu, K. (2016). "Persistence Weighted Gaussian Kernel for Topological Data Analysis," in International Conference on Machine Learning (PMLR), 2004–2013.

Latschev, J. (2001). Vietoris-rips Complexes of Metric Spaces Near a Closed Riemannian Manifold. *Arch. Math.* 77, 522–528. doi:10.1007/pl00000526

Le, T., and Yamada, M. (2018). "Persistence fisher Kernel: a Riemannian Manifold Kernel for Persistence Diagrams," in Proceedings of the 32nd International Conference on Neural Information Processing Systems, 10028–10039.

LeCun, Y., and Cortes, C. (1999). *The Mnist Database*. [Dataset]. http://yann.lecun.com/exdb/mnist/.

Mileyko, Y., Mukherjee, S., and Harer, J. (2011). Probability Measures on the Space of Persistence Diagrams. *Inverse Probl.* 27, 124007. doi:10.1088/0266-5611/27/12/124007

Mitra, A., and Virk, Ż. (2021). The Space of Persistence Diagrams on n Points Coarsely Embeds into Hilbert Space. *Proc. Amer. Math. Soc.* 149, 2693–2703. doi:10.1090/proc/15363

Nathaniel Saul, C. T. (2019). *Scikit-tda: Topological Data Analysis for python*. doi:10.5281/zenodo.2533369 [Dataset].

Oudot, S. Y. (2015). *Persistence Theory: From Quiver Representations to Data Analysis*, Vol. 209. Providence, RI: American Mathematical Society Providence.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine Learning in Python. *J. Machine Learn. Res.* 12, 2825–2830.

Perea, J. A. (2019). A Brief History of Persistence. *Morfismos* 23, 1–16.

Perea, J. A., Munch, A., and Khasawneh, F. A. (2019). *Approximating Continuous Functions on Persistence Diagrams Using Template Functions.CoRR* abs/1902.07190

Pickup, D., Sun, X., Rosin, P. L., Martin, R. R., Cheng, Z., Lian, Z., et al. (2014). "SHREC'14 Track: Shape Retrieval of Non-rigid 3d Human Models," in Proceedings of the 7th Eurographics workshop on 3D Object Retrieval (Eurographics Association), 1–10. EG 3DOR'14

Polanco, L., and Perea, J. A. (2019a). Adaptive template systems: Data-driven feature selection for learning with persistence diagrams. Available at: http://arxiv.org/abs/1910.06741.

Polanco, L., and Perea, J. A. (2019b). "Coordinatizing Data with Lens Spaces and Persistent Cohomology," in Proceedings of the 31 st Canadian Conference on Computational Geometry (CCCG), 49–57.

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. (2015). "A Stable Multi-Scale Kernel for Topological Machine Learning," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4741–4748.

Reynolds, D. (2009). *Gaussian Mixture Models*. Boston, MA: Springer US, 659–663. doi:10.1007/978-0-387-73003-5_196

Robins, V., and Turner, K. (2016). Principal Component Analysis of Persistent Homology Rank Functions with Case Studies of Spatial point Patterns, Sphere Packing and Colloids. *Physica D: Nonlinear Phenomena* 334, 99–117. doi:10.1016/j.physd.2016.03.007

Siu Kwan Lam, A. P., and Seibert, S. (2015). "Numba: a Llvm-Based python Jit Compiler," in LLVM '15: Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, 1–6.

Smith, A., Bendich, P., Harer, J., and Hineman, J. (2017). *Supervised Learning of Labeled Pointcloud Differences* via *Cover-Tree Entropy Reduction*. *CoRR* Abs/1702.07959.

Sonego, P., Pacurar, M., Dhir, S., Kertész-Farkas, A., Kocsor, A., Gáspári, Z., et al. (2007). A Protein Classification Benchmark Collection for Machine Learning. *Nucleic Acids Res.* 35, D232–D236. doi:10.1093/nar/gkl812

Tralie, C., Saul, N., and Bar-On, R. (2018). Ripser.py: A Lean Persistent Homology Library for python. *Joss* 3, 925. doi:10.21105/joss.00925

Tschandl, P., Rosendahl, C., and Kittler, H. (2018). The Ham10000 Dataset, a Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions. *Sci. Data* 5, 180161. doi:10.1038/sdata.2018.161

Turner, K., Mileyko, Y., Mukherjee, S., and Harer, J. (2014). Fréchet Means for Distributions of Persistence Diagrams. *Discrete Comput. Geom* 52, 44–70. doi:10.1007/s00454-014-9604-7

Wagner, A. (2019). *Nonembeddability of Persistence Diagrams with Wasserstein Metric*. arXiv preprint arXiv:1910.13935

wwPDB consortium (2018). Protein Data Bank: the Single Global Archive for 3D Macromolecular Structure Data. *Nucleic Acids Res.* 47, D520–D528. doi:10.1093/nar/gky949

Zaheer, M., Kottur, S., Ravanbhakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J. (2017). "Deep Sets," in Proceedings of the 31st International Conference on Neural Information Processing Systems, 3394–3404.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers in Artificial Intelligence

ORIGINAL RESEARCH
published: 18 August 2021
doi: 10.3389/frai.2021.680564

Topological Data Analysis Highlights Novel Geographical Signatures of the Human Gut Microbiome

Eva Lymberopoulos[1,2], *Giorgia Isabella Gentili*[1], *Muhannad Alomari*[1,3] *and Nikhil Sharma*[1,4]*

[1]Department of Clinical and Movement Neurosciences, Institute of Neurology, University College London, London, United Kingdom, [2]CDT AI-Enabled Healthcare Systems, Institute of Health Informatics, University College London, London, United Kingdom, [3]R^2 Data Labs, Rolls-Royce Ltd, Derby, United Kingdom, [4]National Hospital for Neurology and Neurosurgery, University College London Hospitals NHS Foundation Trust, London, United Kingdom

Background: There is growing interest in the connection between the gut microbiome and human health and disease. Conventional approaches to analyse microbiome data typically entail dimensionality reduction and assume linearity of the observed relationships, however, the microbiome is a highly complex ecosystem marked by non-linear relationships. In this study, we use topological data analysis (TDA) to explore differences and similarities between the gut microbiome across several countries.

Methods: We used curated adult microbiome data at the genus level from the GMrepo database. The dataset contains OTU and demographical data of over 4,400 samples from 19 studies, spanning 12 countries. We analysed the data with *tmap*, an integrative framework for TDA specifically designed for stratification and enrichment analysis of population-based gut microbiome datasets.

Results: We find associations between specific microbial genera and groups of countries. Specifically, both the USA and UK were significantly co-enriched with the proinflammatory genera *Lachnoclostridium* and *Ruminiclostridium*, while France and New Zealand were co-enriched with other, butyrate-producing, taxa of the order Clostridiales.

Conclusion: The TDA approach demonstrates the overlap and distinctions of microbiome composition between and within countries. This yields unique insights into complex associations in the dataset, a finding not possible with conventional approaches. It highlights the potential utility of TDA as a complementary tool in microbiome research, particularly for large population-scale datasets, and suggests further analysis on the effects of diet and other regionally varying factors.

Keywords: gut microbiome, human microbiome, population health, global variation, topological data analysis (TDA)

OPEN ACCESS

Edited by:
Umberto Lupo,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
William (Kuang-Wei) Chang,
Albert Einstein College of Medicine, United States
Javier Arsuaga,
University of California, Davis, United States

***Correspondence:**
Nikhil Sharma
nikhil.sharma@ucl.ac.uk

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *14 March 2021*
Accepted: *28 July 2021*
Published: *18 August 2021*

Citation:
Lymberopoulos E, Gentili GI, Alomari M and Sharma N (2021) Topological Data Analysis Highlights Novel Geographical Signatures of the Human Gut Microbiome. Front. Artif. Intell. 4:680564. doi: 10.3389/frai.2021.680564

INTRODUCTION

In recent years, there has been a rapidly growing interest in the connection between the gut microbiome and disease. This area spans detailed exploration of the gut microbiome in small specific clinical disease phenotypes to larger population-level studies. In parallel, there have been advances in the analytics approaches the microbiome field has adopted to test the different hypotheses. Conventional approaches employ dimensionality reduction and typically assume linearity. Here

we use topological data analysis (TDA) exploring the difference and similarities between the gut microbiome across several countries. We highlight unique insights that are made possible with the use of TDA.

The Human Gut Microbiome

The gut microbiome is a diverse community of an estimated 100 billion to trillion microorganisms - bacteria, viruses, and fungi - inhabiting the intestine and gut (Sender et al., 2016). So far, 1,952 species have been classified - however, the majority of the microbiome remains unreferenced (Almeida et al., 2019). Unsurprisingly, the relationships between the different microbiome species are highly complex, dynamic, and nonlinear (Shoaie et al., 2013). Depletion of one species below a specific threshold can lead to the so-called blooming of others. Some species also exist only in either very low or very high abundance with specific tipping points (Lahti et al., 2014). Species can even change their phenotype based on the concentration in the gut, environmental, or genetic context; in other words, harmless bacteria can become pathogenic under specific circumstances (Casadevall, 2017). These species have so-called high pathogenic potential, are also known as pathobionts, and are usually kept under control by a healthy microbial community (Kamada et al., 2013). If this ecosystem is disrupted, pathobionts and external pathogens can bloom, affecting host health. It is important to note that a healthy composition of the microbiome is highly individual but based around a proposed universal "core microbiome" (Rinninella et al., 2019).

The microbiome coevolved with humans in a commensal, perhaps even symbiotic, way (Bäckhed et al., 2005; Shapira, 2016). The microbiome appears to play a central role in host immunity, metabolism, behaviour, and cognition through yet unclear pathways. Specifically, it is thought that a disturbed microbiome, also called gut dysbiosis, can set off inflammatory cascades. Disease, lifestyle changes, or environmental influences can disturb the delicate balance of the microbiome, leading to loss of seemingly beneficial microbes and a simultaneous blooming of bacterial taxa detrimental to the host (Petersen and Round, 2014). This can lead to the breakdown of the epithelial cells lining the gut, increasing gut permeability which can cause pro-inflammatory bacterial metabolites or products to leak out, triggering further inflammatory cascades in the host (Rooks and Garrett, 2016; Thevaranjan et al., 2017). Changes in the gut microbiome have increasingly been linked to a range of diseases, such as colitis, diabetes, neurodegenerative diseases, and autism (for a review *see* Ghaisas et al., 2016)). Additionally, the gut microbiome influences the efficacy and bioavailability of oral medication (*see* e.g. Enright et al., 2016; Wilson and Nicholson, 2017; Clarke et al., 2019). For instance, the interaction between drugs and microbiome appears important in Parkinson's disease (Rekdal et al., 2019), arthritis (Scher et al., 2020), schizophrenia (Seeman, 2021), and bipolar disorder (Flowers et al., 2020). Analysing the gut microbiome and illuminating the subtle relationships driving it has significant translational value for population health, particularly as it is an easily accessible and scalable potential therapeutic target.

Variation in the Gut Microbiome

Several factors affect the gut microbiome, which can be broadly distinguished into lifestyle, medical, and environmental factors. Perhaps the most prominent lifestyle factor is diet. A high-fat diet can induce dysbiosis in the gut microbiome (Vaughn et al., 2017), while a diet high in resistant starches and complex carbohydrates (such as the Mediterranean diet) increases beneficial species (Garcia-Mantrana et al., 2018). This includes Firmicutes that produce short-chain fatty acids (SCFA), which have anti-inflammatory properties and maintain the integrity of the epithelial layer in the intestine (Morrison and Preston, 2016; Levy et al., 2017). Similarly, moderate alcohol consumption seems to increase anti-inflammatory species (Quesada-Molina et al., 2019), and exercise is also associated with a beneficial effect (Monda et al., 2017). Travel has also been shown to negatively alter the microbiome by decreasing diversity (Riddle and Connor, 2016; Langelier et al., 2019). Crucially, hygiene is a non-negligible factor - while inadequate sanitation can increase the likelihood of bacterial infection, excessive hygiene as practised in some countries - even as a response to the COVID-19 pandemic - may lead to a reduction in the microbiome diversity (Schmidt et al., 2011; Burchill et al., 2021).

The use of oral medications, particularly antibiotics, is another important influence on the gut microbiome. It takes some microbial species up to 6 months to recover from a complete cycle of antibiotics (Dethlefsen et al., 2008). Non-antibiotic medication such as dopaminergic drugs (Hill-Burns et al., 2017), proton pump inhibitors, antipsychotic drugs, and opioids interact with the microbiome and can affect its composition (Le Bastard et al., 2018). As the microbiome is interlinked with metabolic pathways, chronic diseases such as diabetes are also associated with a disrupted gut microbiome, though the causal direction of this effect is unclear - the same holds true for obesity (Singer-Englar et al., 2019). Environmental factors are a crucial and sometimes overlooked part of the host-microbiome relationship. External pathogens such as viruses can induce changes to the gut microbiome, as can pesticides and other toxins (Li N. et al., 2019; Tu et al., 2020). Pollution has also been associated with changes to the microbiome, particularly air pollution (Vallès and Francino, 2018; Bailey et al., 2020). Crucially, there is evidence that the soil and drinking water microbiomes interact with the gut microbiome (Blum et al., 2019).

Geographical Variation of the Gut Microbiome

The factors influencing the microbiome vary regionally, leading to differences in the population microbiome across countries as has been observed in many past studies (e.g., Karlsson et al., 2014). One large review reports distinct geographical differences in the gut, oral, and skin microbiomes between non-industrialised and industrialised populations in addition to a conserved core microbiome (Gupta et al., 2017). More specifically, the review reported that while the non-industrialised gut microbiota include more species of the phyla Proteobacteria, Spirochaetes, order Clostridiales, and genera *Prevotella* or *Ruminobacter*, the

industrialised communities were more enriched with the Firmicutes phylum, and *Bacteroides* and *Bifidobacterium* genera.

Diet is one of the most intuitive drivers of these differences; while some countries consume large amounts of meat, others have a diet heavier in carbohydrates, or in fibre (Ritchie and Roser, 2019), which has been connected to observed differences in microbiome composition between countries (Riaz Rajoka et al., 2017). For example, one study compared gut microbiome signatures between children in urban Italy and rural Burkina Faso and found unique microbial genera in the African children that might be linked to differences in diet: the genera *Prevotella*, *Xylanibacter*, *Butyvibrio*, and *Treponema* are involved in cellulose and xylan hydrolysis which are fitting for the polysaccharide-rich diet of the African children which includes many whole grains, producing the beneficial SCFAs (De Filippo et al., 2010). These results are echoed in a later study comparing Egyptian and US-American teenagers, which found differences in the metabolic profiles consistent with the dominant diet of the respective region (Shankar et al., 2017).

Similarly, prescription patterns and access to antibiotics vary from country to country: while low- and low-middle-income countries count around 12 daily antibiotic doses per 1,000 citizens, high-income countries count around 25 per 1,000 (Klein et al., 2018). Pesticide use is another factor that varies starkly between countries, due to environmental regulations being less or more restrictive, as well as the importance of farming or industry for a country's economy and society (Handford et al., 2015). Accordingly, soil and water microbiome signatures vary between countries and regions, as demonstrated by the Earth Microbiome Project (Thompson et al., 2017). This is partly naturally caused, and partly due to external factors such as pesticide and fertiliser use (Gourmelon et al., 2016; Lupatini et al., 2017). In addition to these environmental factors, host genetics and the innate and adaptive immune systems can account for some of the human microbiome variation between populations, although the exact contributions of environmental and genetic factors, respectively, are unclear (Gupta et al., 2017).

Together, these factors could point to differences in the population microbiome which are important to health and disease. As some of the differences between populations described above include increased anti-inflammatory microbial products, this can affect inflammatory and disease processes in these regions. One example of this has been research into obesity: while obesity varies between countries and has been connected to the industrialisation level of a population, it has also been associated with a differential microbiome profile (Dugas et al., 2016). Mouse studies have even suggested causality: transplanting the gut microbiome of genetically modified obese mice into germ free mice led to weight gain (Turnbaugh et al., 2006). However, human data on whether shifts in the microbiome associated with geographical variations relate to geographical differences in obesity are rare. One recent study found that the gut microbiome of obese subjects in industrialised countries is more similar to that of other industrialised countries, even if these were geographically far apart, than to that of non-industrialised communities (Angelakis et al., 2019). Similar geographical insights could be relevant for non-communicable diseases that have been associated with deviations in the microbiome and that have differential prevalence in some countries over others, as has been observed for many gastrointestinal, neurodegenerative, psychiatric, or inflammatory diseases (GBD 2017 Disease and Injury Incidence and Prevalence Collaborators et al., 2018). Knowledge about what drives these differences could in turn inform improvements to existing medications or inspire novel treatment options through the gut microbiome.

Limitations of Standard Analysis

Traditional approaches to microbiome analysis comparing groups, even ones employing complex machine learning models, have many shared limitations, preventing reliability. Firstly, they rely on reduction in dimensionality to simplify the modelling of the ecosystem, which leads to loss of key information around the complex interplay of the microbiome. The binary output of these studies, namely which taxa are deemed to be beneficial or detrimental, is an oversimplification of the original problem. Attempting to address the highly complex and non-linear ecosystem of the gut microbiome with a simplistic linear approach introduces a range of errors to the results, such as precluding the real effect and leading to frequent false positives if not adequately addressed. Additionally, many human microbiome studies, including the ones on geographical variation, have very small sample sizes, particularly those comparing patients to healthy controls. Many also poorly control for potential confounders. There have been efforts to curate larger datasets to tackle some of these issues, leading to sample sizes of up to 12,000 in the American Gut Project (McDonald et al., 2018). However, the issues cannot be countered with an increase in sample size on its own - in fact, it can be argued that adding more data while maintaining the oversimplified, linear modelling approaches will add further noise to the results and lead to multiple comparison errors.

TDA and the Microbiome

Topological data analysis (TDA) can address many of these concerns. TDA is an analysis method coined by Gunnar Carlsson (2009) and was developed to analyse high-dimensional datasets. It uses principles from topology and differential geometry, specifically persistent homology. By doing so, TDA can represent the underlying geometric structure, or shape, of the data while accounting for its complexity. Additionally, TDA deals well with high-throughput biological data, such as the microarrays used to sequence the microbiome. It is therefore designed to detect subtle and non-linear relationships in the data and can deal with noisy or incomplete datasets. These factors support the use of TDA in microbiome research.

One previous study by another group has demonstrated the value of TDA for microbiome analysis by combining the well-known Mapper (Singh et al., 2007) with the Spatial Analysis of Functional Enrichment (SAFE) algorithm (Baryshnikova, 2016) to detect co-variance between metadata and microbiome taxa in the dataset (Liao et al., 2019). The authors report that *tmap* outperformed standard tools such as *envfit*, *adonis*, and ANOSIM

in a synthetic dataset, specifically in detecting non-linear, as well as mixed non-linear and linear associations within the data. They applied *tmap* to two population-based microbiome datasets, the Flemish Gut Flora Project (Falony et al., 2016) and the American Gut Project which further illustrates the potential to detect non-linear relationships, specifically associations with host-metadata. They report co-enrichment between two of the so-called enterotypes (Arumugam et al., 2011) and countries, specifically the USA with the Bacteroidetes enterotype and the UK with the Ruminococcaceae enterotype. Further analyses revealed co-enrichment of diet and medication, as well as other lifestyle factors, which were thus associated with both the countries and the enterotypes. TDA appears to be a promising tool to investigate the microbiome through large population-based datasets, specifically as it highlights the increased signal detection in noisy data. While an important and powerful proof of concept, a key scientific limitation of this study was the comparison of only two countries, limiting conclusions on geographical variation of the microbiome that can be drawn from this. Additionally, the authors did not investigate specific underlying microbiome taxa but focused instead on enterotypes, potentially missing more subtle relationships.

This Study

This present study aims to explore the relationship between a range of countries and specific microbiome signatures using TDA. To this end, we use a large repository of gut microbiome data spanning 12 countries with over 4,400 samples and apply the TDA pipeline *tmap* to investigate the co-enrichment of countries and specific microbiome taxa. To our knowledge, this is the first study using this analysis pipeline for this purpose on this data. We hypothesise that with this approach, we can find evidence for differences but also similarities in the gut microbiome signatures that have previously been overlooked by conventional microbiome approaches. This is important in developing our understanding of the microbiome not as a combination of singular taxa but as a rich, diverse, and interrelated ecosystem.

METHODS

Dataset

Microbiome data is obtained from stool samples that are metagenomically sequenced, and then taxonomically classified. The data is thus stored as operational taxonomic units (OTUs).

For this study, we used data from GMrepo, a database of curated gut microbiome metagenomes (Wu et al., 2020). Using the provided RESTful API, we obtained all run IDs associated with the "healthy" and adult phenotype (Mesh-ID D006262) and filtered for only those samples that passed quality control. We then used the run IDs to download the full metagenomic sequence at the genus level. Countries with less than 20 samples were excluded. Metadata of interest that were collected for the whole sample are age, sex, and BMI. BMI was coded into underweight (BMI below 18.5), normal (18.5–24.9), overweight (25–29.9), and obese (over 30) according to the criteria adopted by the WHO, NIH, and NHS.

Analysis Pipeline

Data analysis was conducted in Python 3.6, in a Jupyter notebook 6.0.2 environment. The scripts are available from thesharmalab.com GitHub repository.

A key aspect of TDA approaches is the production of the underlying shape and persistence of the structures. To explore this, we first produced a persistence diagram on the microbiome data as a point cloud with the giotto-tda package (Tauzin et al., 2021). This could then inform parameter tuning during subsequent steps. Then, TDA was conducted with the tmap analysis pipeline (Liao et al., 2019). The pipeline is an "integrative framework" based on TDA and is specifically designed for stratification and association analysis of population gut microbiome datasets. It utilises two established algorithms for TDA and stratification analysis, the Mapper and SAFE algorithms, respectively.

TDA With Mapper

The input to the Mapper algorithm is a point cloud of data points, in this case, each data point represents one stool sample. First, pairwise distances are calculated with the Bray-Curtis distance and these are then transformed to a square-form distance matrix. This matrix is filtered from the original high-dimensional space into a low-dimensional space using multidimensional scaling (MDS), a non-linear method of dimensionality reduction which translates pairwise distances among data points into the low-dimensional space (Mead, 1992). This filter was used as in the origination of the Mapper algorithm (Singh et al., 2007), and the components were set to two, as recommended by the developers of the *tmap* pipeline, with the "pre-computed" metric. Next, the low-dimensional space is partitioned into bins using overlapping covers with each cover including a subset of data points that overlap in some way. Within each cover, data points are then clustered based on the distances from each other in the original, high-dimensional space. These clusters are represented as a node in the TDA network. The shape of the network is a combination of distances in the low- and high-dimensional spaces. In other words, each node in the network is a group of samples with overlapping microbiome profiles and each link between the nodes indicates a shared sample between nodes. The clusterer used was the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) from scikit-learn, as is recommended in tmap documentation. To set an appropriate maximum distance between two data points (*eps*), we used the Mapper algorithm automated optimisation function (optimize_dbscan_eps) with a threshold of 95%, which specifies the percentage of samples for which to cover or cluster the surrounding neighbourhood, based on the distribution of nearest-neighbour distances. The minimum number of neighbours was set to 5.

To optimise the cover ratio, a measure of how many samples are retained during the clustering process, the resolution and overlap parameters were adjusted. Resolution is a measure of how

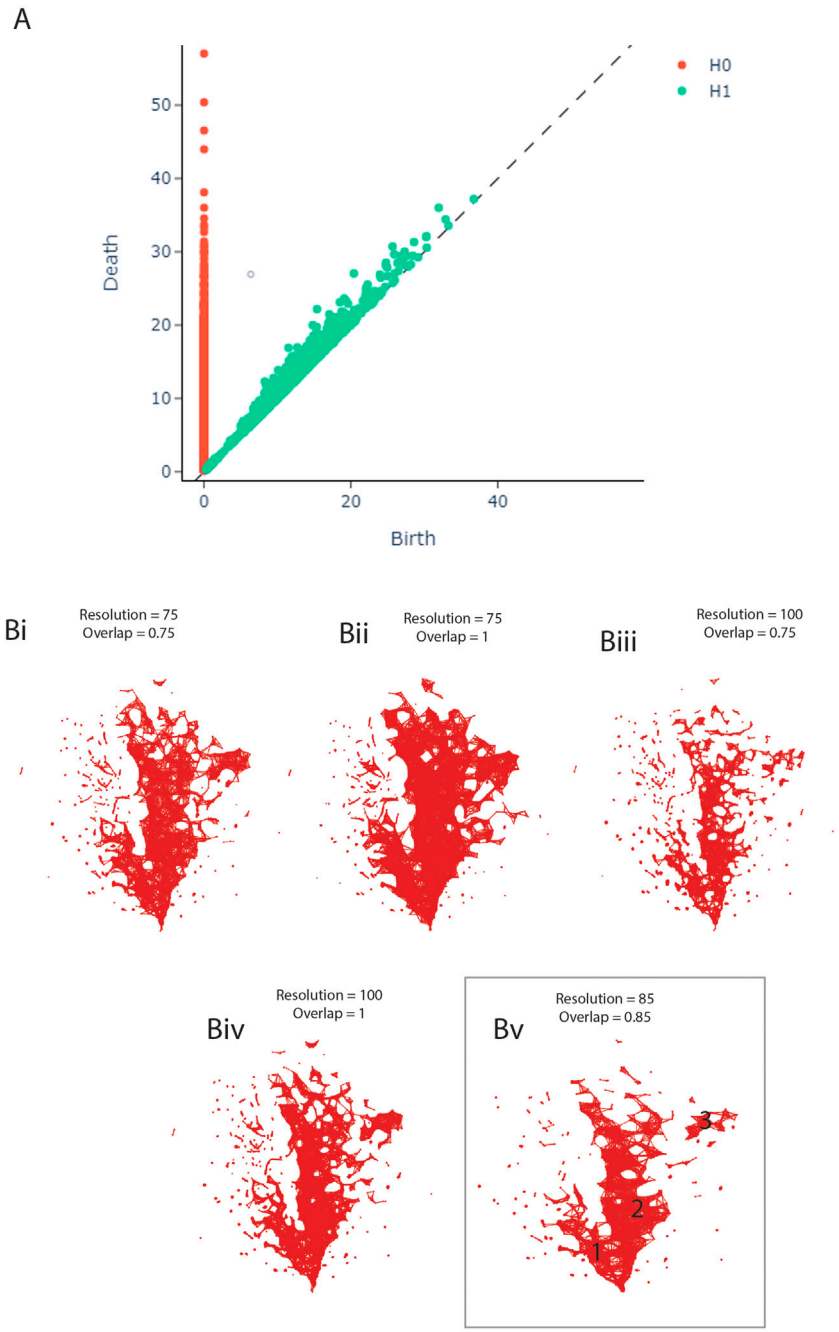


FIGURE 1 | (A) Persistence Diagram. **(B)** Adjusting the resolution and overlap parameters of the TDA network to achieve optimal cover ratio. Bv shows the final network, including the three clusters.

many bins the data is being split into, while overlap decides how big the overlap between adjacent bins needs in order to be considered overlapping. Resolution determines how sparse versus coarse the network will be and thereby how many nodes the network will have. Overlap, on the other hand, determines how densely connected the network will be and thereby how many edges the network will have. Both parameters were adjusted by hand and are shown in **Figure 1B**.

Enrichment With SAFE

The SAFE algorithm maps values of a variable onto the network, denoting enrichment of this variable. The algorithm uses the TDA network as input and then maps the values of a given variable onto the network as node attributes. For example, if the variable is age, then the SAFE algorithm maps the average age of each node (i.e., group of samples). This is called network enrichment.

Subsequently, each node is examined in subnetwork analyses while permutating a given number of times over the entire network which determines how significant the observed enrichment is. For this study, the number of network permutations during this step was set to 5,000 to maximise sensitivity. A subnetwork is identified as a local neighbourhood around each node, where constituent nodes are selected according to the maximum distance threshold. We kept this threshold at the 0.5th percentile of all pairwise node distances in the network. For each neighbourhood, the enrichment of observed values at the neighbour nodes is summed and then ranked to compare the observed with the permutated scores. The score is then log-transformed and normalised to yield a so-called SAFE score for each node of the network. To reiterate, the SAFE scores quantify the enrichment level of a variable in the nodes around a given node. These local scores can be filtered and summed to yield the SAFE enriched score, which represents the network-level association of a target variable. It can be used as analogous to an effect size, allowing comparison between variables in the form of ranking, as well as investigation of their co-enrichment of variables.

Stratification, meaning subgrouping of a population, can be conducted by analysing the enrichment of a host metadata variable across the network. For this, metadata and taxa were entered as covariates for the network enrichment analysis described above. Continuous data, such as age and the microbial taxa, yield stratification heat maps. These show the distribution of absolute values across the network (with the mean of each group of samples represented as one node value), as well as the distribution of enrichment across the network as represented with the SAFE score for each node. Note that dark blue corresponds to the number 0 in both cases. All countries, as well as the metadata variables sex and BMI, were dummy coded. The different levels, or groups, of each variable can be plotted against each other by comparing the SAFE scores of each level at a given node. This means that for each node, the visualisation shows which group was more enriched. If none of the groups show enrichment at a given node, it is grey. Additionally, the most enriched taxa can be found by identifying the most enriched taxon of each node and colouring that node accordingly. It is important to note that to assess significance of an enrichment, the SAFE algorithm depends on both sample size and distribution across the TDA network, affecting the SAFE score, number of significantly enriched nodes, and the SAFE enriched score. This means that for a metadata variable with few samples that are highly distributed across the network, the probability of permutation is very small, making assessment of the enrichment difficult. This affects interpretation of the results, especially for countries with low sample size. If these countries have low SAFE scores, this does not signify the absence of an effect; instead, it demonstrates an inability to detect the presence of an effect. This should be kept in mind when interpreting the results of the SAFE algorithm.

Finally, co-enrichment between variables can be determined, which describes relationships between host metadata and microbiome variations (Liao et al., 2019). While two variables can be considered co-enriched if they enrich in the same area of the network – suggesting that they account for the shape of the network in this area –, it is also possible to quantify this association. For this, we calculated the pairwise co-enrichment for all taxa and metadata, yielding the significance level of each pair. We then applied a threshold to the significance at the 0.5th percentile and binarized the data accordingly. This strict threshold was used to account for the large number of pair-wise tests and reduce the type I error rate. The binarization allowed us to easily find significant co-enrichment between variables. Specifically, we used this quantitative indicator to supplement visual indications of co-enrichment, such as enrichment in the same areas of the network, between the variables most highly enriched across the network.

RESULTS

Dataset

Based on our criteria, the final dataset includes 4,437 stool samples, 1,341 taxonomic units, as well as relevant study and host metadata. The data spans 12 countries from 19 studies, including both Amplicon and metagenomic data. Mean and standard deviations for age, as well as the distribution of sex and BMI for each country, are shown in **Table 1**.

Persistence Diagram

The persistence diagram (**Figure 1A**) shows four highly persistent structures in dimension 0 - representing clusters - and no high persistence in dimension 1 - representing loops. We thus expect to see two to four clusters and a relatively noisy network in the next step of our analysis.

Parameter Adjustment

During the construction of the TDA graph, the resolution and overlap parameters were adjusted by hand to obtain the optimal cover ratio that is representative of the persistence diagram, meaning two to four clusters and no loops. The panels in **Figure 1B** show the result of this adjustment. The final network was constructed with the resolution set to 85 and overlap to 0.85 (**Figure 1B**v).

TDA Network

The TDA network produced by tmap contains 1,435 nodes and 8,870 edges, based on 2,910 samples. 1,527 (65.58%) samples had to be dropped during the construction of the network, likely due to missing data as the individual studies did not measure the same taxa, leading to many OTUs being marked as 0 in each sample. As can be seen in **Figure 1B**v, the network has two central clusters, a smaller one on the left (1) and a larger one in the middle (2). There is also a small third cluster on the right (3). This pattern is broadly consistent with the persistence diagram (**Figure 1A**).

Geographical Enrichment

Enrichment of the countries across the TDA network is shown in **Figure 2A**, in which each node is coloured according to which country has the most enrichment at that local node. Additionally, larger nodes correspond to a larger number of samples in that node.

Most countries are either predominantly enriched in cluster 1 (e.g., Canada) or cluster 2 (e.g., the USA, the UK, Italy,

TABLE 1 | Demographic Data.

	Brazil	Canada	China	Denmark	France	Germany	Italy	New Zealand	Spain	Tanzania	UK	USA	Total
Age													
Mean	30.1	25.9	43.3	55.4	62.0	38.1	39.3	36.9	40.9	36.1	51.3	41.7	**40.0**
SD	5.0	5.1	12.4	8.1	10.5	8.3	13.6	12.6	14.5	13.3	13.2	16.7	**16.9**
Sex													
Female	18	659	81	73	249	0	26	82	31	8	122	847	**2196**
Male	2	610	90	34	216	70	14	49	16	14	149	965	**2229**
Missing	0	0	0	0	0	0	0	0	0	0	2	10	**12**
BMI													
Underweight	0	0	8	0	4	0	1	0	0	0	8	38	**59**
Normal	15	973	33	1	228	29	26	101	2	0	180	1059	**2647**
Overweight	4	296	32	0	182	29	2	30	0	0	69	495	**1139**
Obese	1	0	0	0	38	12	0	0	0	0	16	95	**162**
Missing	0	0	98	106	13	0	11	0	45	22	0	135	**430**
Total	**20**	**1269**	**171**	**107**	**465**	**70**	**40**	**131**	**47**	**22**	**273**	**1822**	**4437**

New Zealand). Interestingly, the USA and the UK are also significantly co-enriched. China is enriched at the junction of cluster 1 and 2, as well as in cluster 3, in which they are together with the USA and Canada. Brazil is enriched both in cluster 2, as well as the junction of the two big clusters. Samples from France are enriched in the same area of the graph, namely the top left, which appears sparse and disconnected from the rest of the network. As can be seen in **Figure 1B**, this holds true for all the observed parameter adjustments. Finally, Tanzania is not significantly enriched in the network at all. As mentioned above, this finding needs to be interpreted cautiously due to the low sample size of Tanzania.

Figure 3A shows all host metadata ranked according to their SAFE enriched scores. The USA and Canada stand out with scores of over 400 each, making them the two most enriched host metadata. The next most enriched country is the UK with a SAFE score of 190, and China with a score of 84. Together with France, these countries also have the most samples (*see* **Table 1**).

Other Metadata

Age, sex, and BMI were also investigated. Host age has a SAFE enriched score of 205 and is enriched mostly in cluster 2 (*see* **Figure 4A**). Host age is significantly co-enriched with the UK and France, which both have higher than average age compared to the other countries (*see* **Table 1**). Host sex was relatively highly enriched (SAFE enriched scores Male: 223, Female: 210), and the enrichment network shows that female sex is mostly enriched in cluster 1, while the enrichment of male sex is more distributed across the network (**Figure 4B**). Interestingly, both female and male sex are significantly co-enriched with Canada. Finally, BMI seems to be a relevant host variable, as normal BMI is the third most enriched metadata with a SAFE enriched score of 302. Most of the enrichment of the normal BMI appears in cluster 1, while cluster 2 is more enriched with non-normal BMI phenotypes (**Figure 4C**). This is reflected in a significant co-enrichment of normal BMI with Canada. Further, normal BMI is significantly co-enriched with male sex.

Taxa

We also explored the enrichment patterns of different taxa with host metadata. **Figure 2B** shows the taxa with the most enriched nodes in one figure, while **Figures 5**, **6** show network heat maps of the most relevant taxa. The top enriched taxa belong to the Bacteroidetes and Firmicutes phyla. **Figure 6B** shows a heatmap on the matrix of all co-enrichment pairs between host metadata and taxa of interest. Note that the significance threshold for co-enrichment was set to the 0.5th percentile of all scores, so that some of the seemingly lowest significances don't pass the threshold.

Of the Bacteroidetes, *Bacteroides* is the genus with the second most enriched nodes, has a SAFE enriched score of 256, and is enriched in cluster one (**Figure 2B**). Looking at its heat map, it also becomes apparent that it is enriched in the junction of the two large clusters (**Figure 5A**). Despite the co-enrichment observable in these figures, no co-enrichment passes the significance threshold. **Figures 2B**,**5B** show that the genus *Prevotella*, which has one of the highest numbers of enriched nodes despite a relatively low SAFE enriched score of 99, is highly enriched exclusively in cluster 3, while it is abundant across the network. Although the heat map indicates co-enrichment of this genus with many variables such as China, Canada, or the US, they don't pass the significance threshold. The genus with the most enriched nodes, *Paludibacater*, is the third most enriched taxon in the dataset, with a SAFE enriched score of 273. It is mainly enriched in the lower half of cluster 2 (**Figures 2B**,**5C**). Further, it is significantly co-enriched with the countries USA and Italy, as well as obese BMI, and *Lachnoclostridium*. *Alistipes* is exclusively enriched in the top left part of cluster 1 (**Figure 6A**). It is significantly co-enriched with Canada and normal BMI, which notably also co-enriched with each other.

The two genera with the most enriched nodes - *Lachnoclostridium* (SAFE enriched score 295) and *Ruminiclostridium* (284) - are both members of the *Clostridiales* order in the Firmicutes phylum and enriched predominantly in cluster 2 (**Figures 6B,C**), *Ruminiclostridium* particularly in the top half. Both are significantly co-enriched with the United States, United Kingdom, and Italy, *Ruminiclostridium* is further significantly co-enriched with host age. They are also both significantly co-enriched with *Faecalibacterium prausnitzii* and each other. Finally, the sparse and disconnected nodes in the top left of the network are most enriched by *Blautia* (SAFE enriched score 153) and *Faecalibacterium prausnitzii* (SAFE enriched score 223).

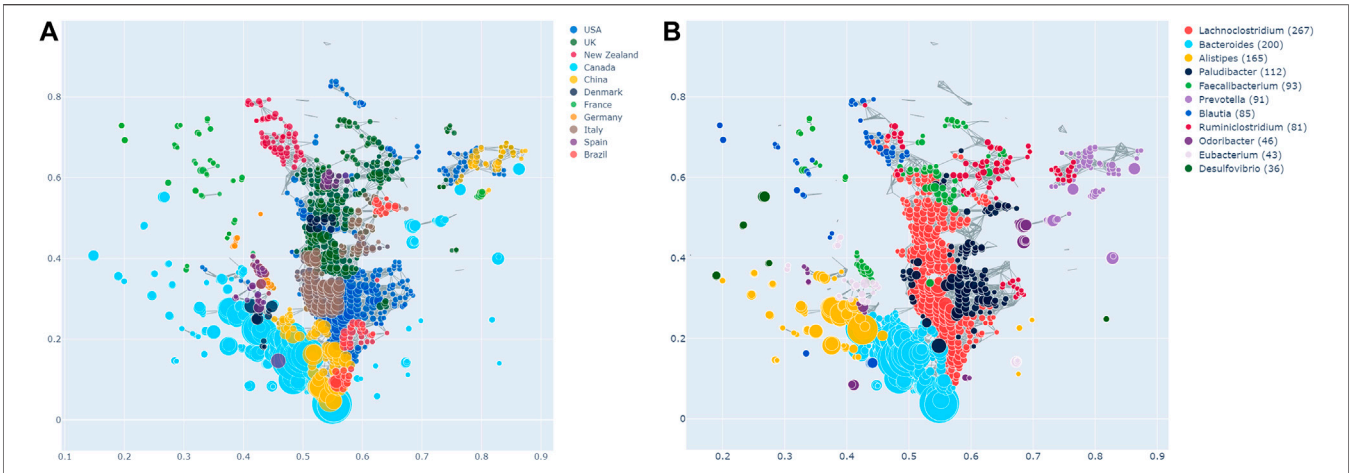


FIGURE 2 | (A) Network stratification of the countries in the dataset, showing the enriched nodes. Note that Tanzania is not included here, as it did not have any significantly enriched nodes. However, this does not necessarily imply the absence of an effect. **(B)** Network stratification of the taxa with the most enriched nodes.

Blautia is significantly co-enriched with New Zealand and *Ruminococcus*, while *F. prausnitzii* is significantly enriched with the UK and host age. Additionally, France, enriched across this sparse area, is significantly co-enriched with *Eubacterium, Ruminococcus,* as well as *Dorea* – a genus also significantly co-enriched with New Zealand.

DISCUSSION

Using TDA, we highlight novel differences and similarities of the gut microbiome across 12 countries. The TDA approach demonstrates the overlap between countries, a finding not possible with conventional approaches. We found distinct distributions of the countries across the TDA network that, through co-enrichment analysis, corresponded to the distribution of specific driver microbial genera, namely *Paludibacter*, *Bacteroides*, *Prevotella*, and *Alistipes* of the phylum Bacteroidetes, as well as *Lachnoclostridium* and *Ruminiclostridium*, as well as *Blautia, Faecalibacterium prausnitzii*, *Dorea*, *Eubacterium*, and *Ruminococcus* of the Firmicutes phylum. This highlights the potential utility of TDA as a complementary tool in microbiome research, and particularly of the library *tmap* as a helpful tool for implementing TDA in the microbiome space.

Geographical Co-enrichment of Taxa

Broadly, TDA shows the similarities between the countries within each cluster. For example, the first cluster (Cluster 1) shows the shared features of the gut microbiome of Canada, China, Denmark, and Spain. Likewise, several countries have shared features in cluster 2 (USA, UK, Italy, New Zealand, Germany, Brazil), and Cluster 3 (USA, China, Canada). Importantly, the distinct clusters are associated with differences in the gut microbiome composition between these regions. It is also notable that membership of the cluster is not exclusive. For instance, the gut microbiome from Canadian samples shares features with cluster 1 and cluster 3. In contrast, the UK appears only in cluster 2. It is noteworthy that countries in close geographical proximity such as the USA and Canada, or France and Germany, seem to have important differences in their microbiome composition, whereas countries that are separated by thousands of miles, such as the UK and USA, share many features, as evidence by their co-enrichment. Below, we explore these differences in more detail with a focus on the specific co-enriched taxa and potential underlying explanations and confounds.

Bacteroidetes

One of the most enriched genera this study identified is *Bacteroides* of the Bacteroidetes phylum. It was most enriched at the junction of clusters 1 and 2 and visually co-enriched with China, the USA, Denmark, and Brazil, although on individual testing these did not reach significance. The genus contains many pathogens and pathobionts and generally has a high virulence potential, as well as the highest antibiotic resistance of microbial genera (Wexler, 2007). It is further associated with diseases such as Irritable Bowel Disease (IBD; Walters et al., 2014) or the gut microbiome changes seen in ulcerative colitis carcinogenesis, as shown in a recent mouse study (Wang et al., 2019). Additionally, the genus *Bacteroides* is associated with obesity (Ppatil et al., 2012), which corresponds to the co-enrichment of an obese BMI score in cluster 1, which however was not statistically significant. Increased *Bacteroides* is associated with a long-term high-fat diet, specifically an omnivorous diet high in protein and animal fat (Wu et al., 2011; Zimmer et al., 2012; Ferrocino et al., 2015; Franco-De-Moraes et al., 2017). This diet is prevalent in high-income countries such as the USA and Canada (Ritchie and Roser, 2019), and becoming more common in middle-income countries, such as China and Brazil, as average income rises (Fu et al., 2012). Our results mirror the results of the tmap study by

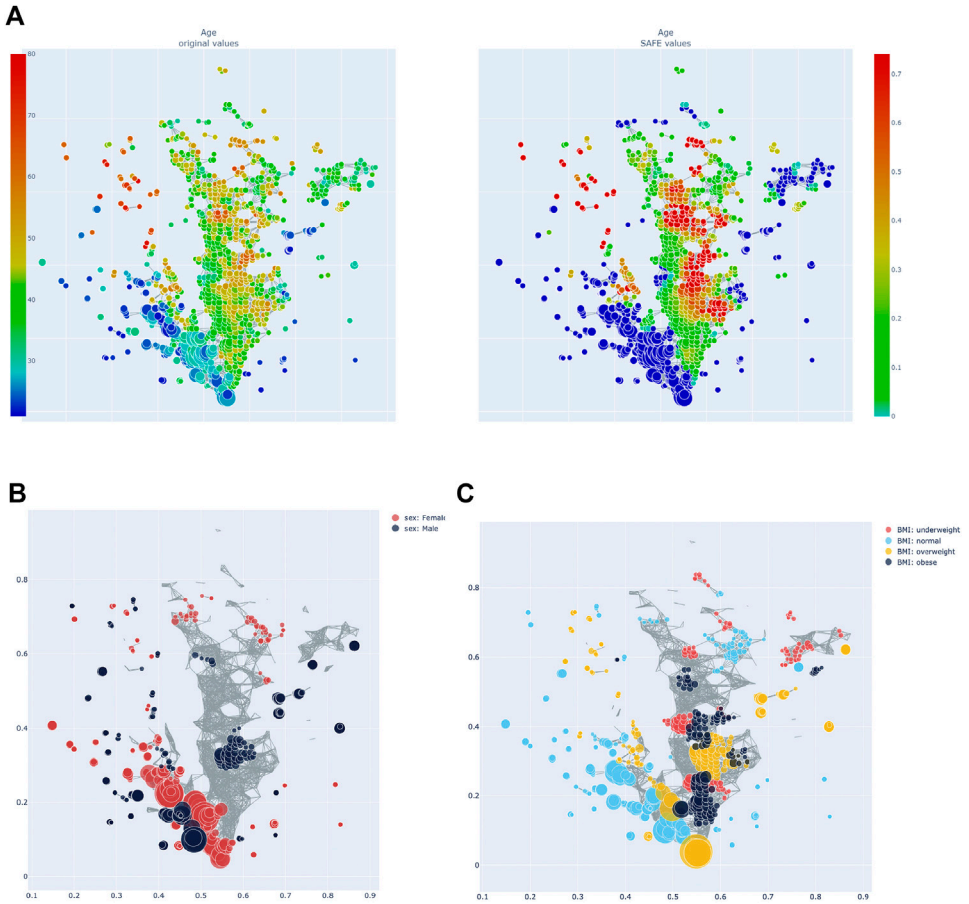


FIGURE 3 | (A) Heatmap of age: distribution of absolute value across the network on the left, enrichment on the right. **(B)** Network stratification of sex across the network. **(C)** Network stratification of BMI across the network.

Liao et al. observed (2019), as they also associated the USA with a *Bacteroides* enterotype. While this overlapping result may be explained by the inclusion of data from the American Gut Project (McDonald et al., 2018) in this study, the data here shows more complex associations, owing to the increased number of included countries in our study. *Bacteroides* has also been specifically associated with an "industrialised" diet: a study comparing children from the USA and Egypt associated the American children with a *Bacteroides* enterotype, meaning a microbiome profile dominated by *Bacteroides* (Shankar et al., 2017). The Egyptian children on the other hand, who ate a Mediterranean diet rich in plant-based foods and fibres, were associated with the *Prevotella* enterotype.

Other studies find similar results for *Prevotella*: it has been associated with a long-term diet high in carbohydrates (Wu et al., 2011), and is particularly abundant in vegans (Franco-De-Moraes et al., 2017). In this study, *Prevotella* is enriched in cluster 3 as the main driver taxon. The cluster is disconnected from the other clusters and highly enriched with samples from China, Canada, and the USA, although the visually observed co-enrichment with *Prevotella* does not reach significance. The influence of diet, particularly vegan versus omnivorous, needs to be addressed in future studies of population-level microbiome studies and may have specific impacts on disease phenotypes.

Paludibacter is a fermentative genus that includes species producing the SCFA propionate (Qiu et al., 2017). While there is a lack of literature exploring this genus in humans, one study has associated it with a high fibre diet as it consumes mostly polysaccharides and was found to be abundant in children from rural Burkina Faso (De Filippo et al., 2010). Its statistically significant co-enrichment with the USA and Italy, as well as with obese BMI, is thus surprising. However, it should be noted that the abundance of *Paludibacter* is zero for the entirety of cluster 1, implying that its high abundance and enrichment in cluster 2 could be an artefact of the genera sampled in the different studies.

Alistipes is another genus of the Bacteroidetes phylum, of the *Parabacteroides* family, that was among the most highly enriched taxa. Specifically, it was enriched in the top half of cluster 1, and significantly co-enriched with Canada and normal BMI, which also co-enrich with each other. This association is in line with previous research finding an association between *Alistipes* and a lower BMI (Aguirre et al., 2016; Lv et al., 2019). The association with Canada on the other hand could be a sampling artefact, as

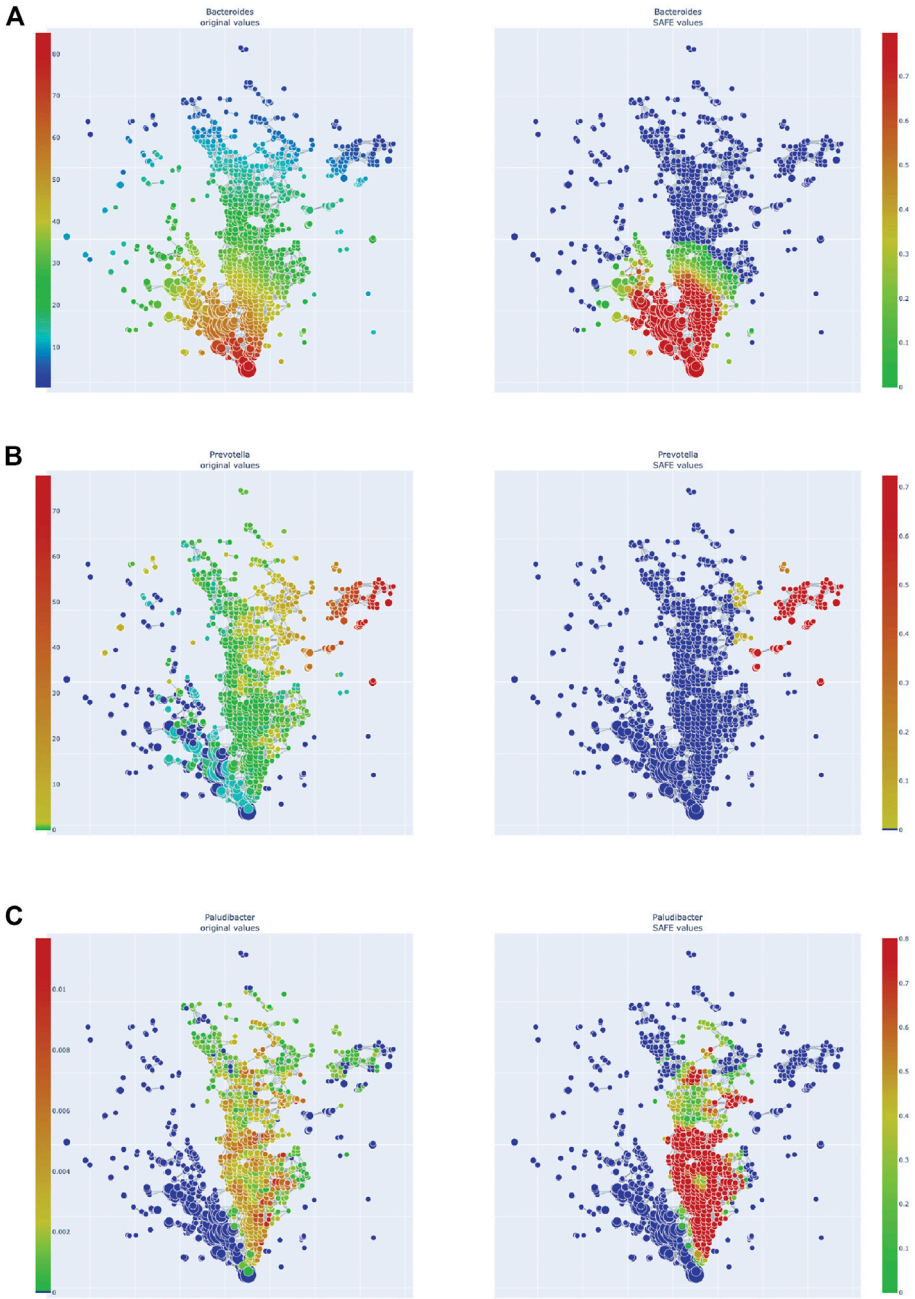


FIGURE 4 | Enrichment heatmaps of three Bacteroidetes genera, absolute abundance across the network on the left, enrichment on the right. **(A)** *Bacteroides*. **(B)** *Prevotella*. **(C)** *Paludibacter*.

Canada has a particularly high number of normal BMI samples compared to other countries in this study. Further, 77% of the Canadian sample used here are of normal BMI, while the Canadian adult population has an obesity rate of 64% (Government of Canada, 2017). Additionally, studies have found *Alistipes* to have both beneficial, as well as detrimental effects on the host: on the one hand, it has been found to attenuate colitis in mice (Dziarski et al., 2016), but on the other hand, it has consistently increased abundance in Parkinson's Disease (PD) patients (Barichella et al., 2016; Bedarf et al., 2017; Li C. et al., 2019). This could be further explored by applying our approach to the gut microbiome of populations with a differing prevalence of PD.

Firmicutes

Two of the top enriched taxa were Firmicutes of the order Clostridiales: *Lachnoclostridium* and *Ruminiclostridium*. While the phylum Firmicutes, and specifically the order Clostridiales, is often associated with beneficial effects for the host, as it contains

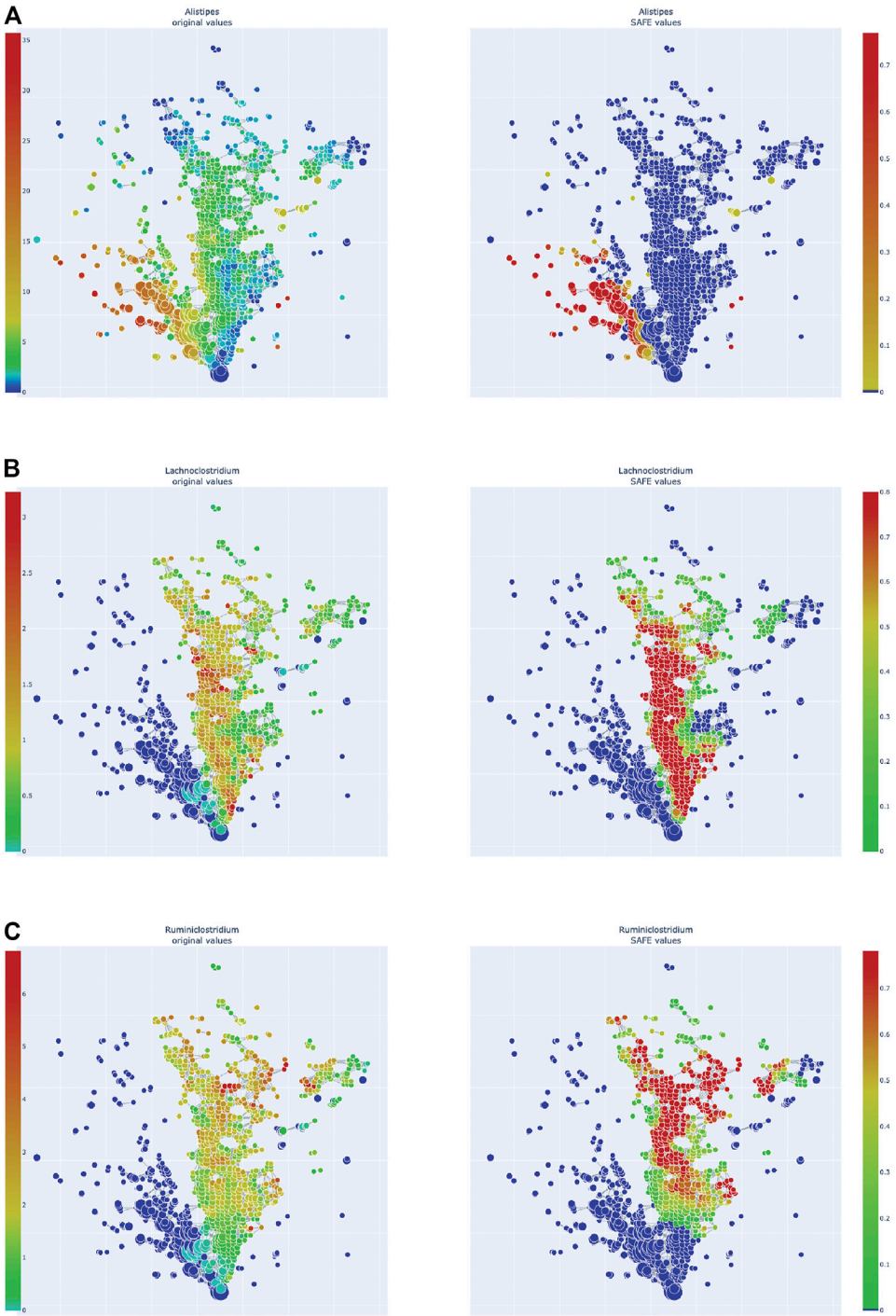


FIGURE 5 | Enrichment heatmaps of one Bacteroidetes genus and two Firmicute genera, absolute abundance across the network on the left, enrichment on the right. **(A)** *Alistipes, of Bacteroidetes phylum*. **(B)** *Lachnoclostridium*. **(C)** *Ruminiclostridium*.

many SCFA producers (Morrison and Preston, 2016; Levy et al., 2017), these two genera seem to fall out of this pattern.

Similar to the above *Bacteroides*, *Lachnoclostridium* has been associated with the changes in gut microbiome found in the carcinogenesis of ulcerative colitis in mice (Wang et al., 2019) - but recovered to a normal abundance after probiotic treatment. Additionally, a new *Lachnoclostridium* species has recently been found to contain a specific genetic marker that is enriched in people with colorectal adenoma, leading to it being suggested as a non-invasive diagnostic marker of the disease (Liang et al., 2020).

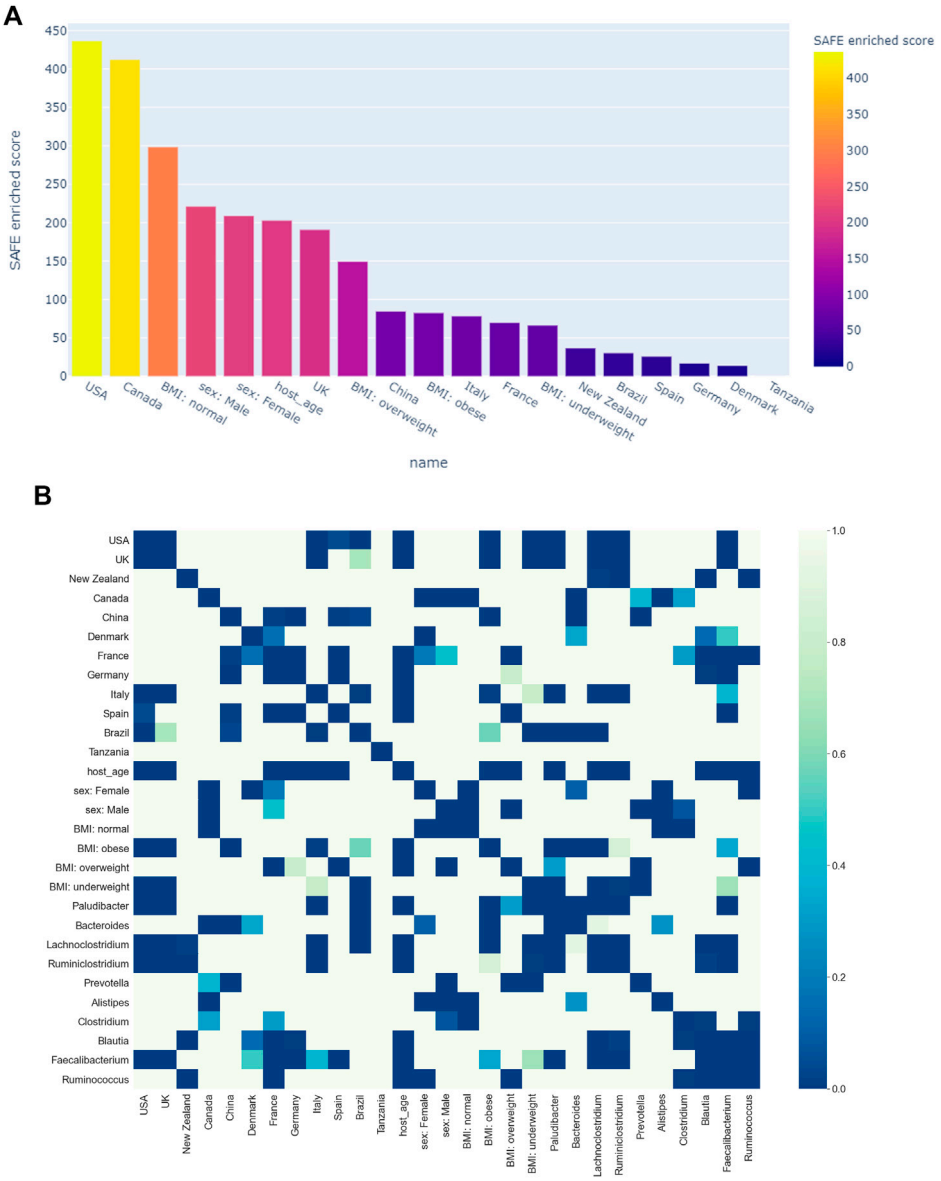


FIGURE 6 | (A) Ranking of host metadata by their SAFE enriched score. **(B)** Co-enrichment significance values of host metadata and the most enriched taxa, presented as a heatmap. The significance threshold was set at the 0.5th percentile threshold of all values.

Ruminiclostridium is a medium-chain fatty acid (MCFA) producer. MCFAs, such as Caproic acid (CA), are metabolites that are less studied than SCFAs. There is evidence that MCFAs antagonise the anti-inflammatory effects of SCFAs by enhancing TH1 and TH17 cell differentiation in a CNS autoimmune model (Haghikia et al., 2015). Additionally, CA was found to be augmented in Multiple Sclerosis patients while SCFAs were reduced, correlating with an immunological profile of an increase in TH1 and TH17 and a decrease in Treg lymphocytes (Saresella et al., 2020). *Ruminiclostridia* were also elevated in a mouse model of dysbiosis – and intriguingly also increased in aged mice (Liu et al., 2020). This is relevant as in this study, *Ruminiclostridium* was significantly co-enriched with host age.

It thus seems that these two Firmicutes are both associated with pro-inflammatory properties. These genera were highly enriched in cluster 2 and specifically co-enriched significantly with the USA and UK, suggesting an important role in those countries' microbiome profile that is distinct to that of other countries. This is further supported by the significant co-enrichment between the USA and UK, as well as between *Lachnoclostridium* and *Ruminiclostridium*. This finding is opposite to that of Liao and colleagues (2019) who found the two countries to have distinct microbiome signatures, specifically that only the UK but not the USA were co-enriched with the family Ruminococcaceae, which contains *Ruminiclostridium*.

Finally, the sparse and disconnected collection of nodes in the top left of the TDA network is highly enriched with France and five different genera of the order Clostridiales: *Blautia*, *Faecalibacterium prausnitzii*, *Dorea*, *Eubacterium*, and *Ruminococcus*, the last three of which were statistically significantly co-enriched with France. Some were also associated with the geographically separated New Zealand. The Clostridiales order has been identified as one of the main SCFA producers in the human gut and indeed all of these genera have been previously identified as butyrate producers (Venegas et al., 2019). However, the fact that the nodes of this network area do not cluster together complicates adequate interpretation of this finding, warranting further investigation by future studies.

Limitations

This study has several limitations. Firstly, the microbiome data is assumed to be representative of the country population, which may not be the case if sampling bias is present. The importance of this is highlighted in the unexpected co-enrichment between normal BMI and Canada, which is likely an artefact of sampling. Similarly, other countries such as Denmark, Spain, or Tanzania, are missing many data points on BMI, limiting the conclusions that can be drawn from BMI enrichment. Secondly, diet could not specifically be controlled for, given the dataset is a composite of many studies and diet was not collected for all samples. Similarly, the clear separation of enrichment between cluster 1 and 2 for the most highly enriched taxa might be an artefact of the way the data was curated as not all countries sequenced exactly the same taxa. However, this is unlikely to be a significant confounder, apart from the *Paludibacter* enrichment, as the heatmaps show taxa can be highly abundant but not highly enriched across the network. Another potential limitation is using microbiome data at the genus level. Each genus is composed of many species which in turn can be made up of various strains, all having potentially different effects. While not all strains constituting a genus are fully sequenced yet, analysis at species-level could still aid in making interpretation more precise. As species-level data is also available from the GMrepo database, this could be a future extension of the current study.

CONCLUSION

In summary, we find that TDA highlights novel insights into the differences and similarities between the gut microbiome at a population level, both between geographically separated countries and within single countries. This underscores the importance of accounting for factors such as geography or regionally varying factors such as diet when conducting microbiome studies. Further, the dimensionality preserving TDA approach may yield more depth and a richer understanding of the changes in the gut microbiome seen across several diseases and clinical phenotypes that would not be possible using conventional approaches. The python library *tmap* seems to serve as a valuable vehicle for such analyses, particularly due to the inclusion of co-enrichment analysis and network visualisation. TDA may be particularly beneficial for patient data, as current studies cannot account for non-linearity which is often present in such data. This would, however, require larger datasets on clinical phenotypes than are currently available in curated datasets such as GMrepo (Wu et al., 2020) or MGnify (Mitchell et al., 2020). Finally, there is the potential for TDA to be integrated with machine learning approaches, a novel avenue of research (Hensel et al., 2021). This may identify specific taxa for interventional therapeutics, though there are significant barriers to overcome before this may be feasible.

DATA AVAILABILITY STATEMENT

A publicly available dataset was analyzed in this study, which can be found here: https://gmrepo.humangut.info/home. The code used to obtain the results of this article can be found here: https://github.com/thesharmalab-team/tmap_geography.

AUTHOR CONTRIBUTIONS

EL and NS devised the idea, conducted data analysis, conceived the figures and table, and wrote and edited the manuscript. GG and MA contributed to the manuscript.

FUNDING

This research was supported and funded by a grant from the Reta Lila Weston Trust. EL is supported with a studentship by the EPSRC (training grant number EP/S021612/1). EL and NS are supported by the National Institute for Health Research University College London Hospitals Biomedical Research Centre.

REFERENCES

Aguirre, M., Bussolo de Souza, C., and Venema, K. (2016). The Gut Microbiota from Lean and Obese Subjects Contribute Differently to the Fermentation of Arabinogalactan and Inulin. *PLoS One* 11, e0159236. doi:10.1371/JOURNAL.PONE.0159236

Almeida, A., Mitchell, A. L., Boland, M., Forster, S. C., Gloor, G. B., Tarkowska, A., et al. (2019). A New Genomic Blueprint of the Human Gut Microbiota. *Nature* 568, 499–504. doi:10.1038/s41586-019-0965-1

Angelakis, E., Bachar, D., Yasir, M., Musso, D., Djossou, F., Melenotte, C., et al. (2019). Comparison of the Gut Microbiota of Obese Individuals from Different Geographic Origins. *New Microbes and New Infections* 27, 40–47. doi:10.1016/j.nmni.2018.11.005

Arumugam, M., Raes, J., Raes, J., Pelletier, E., Le Paslier, D., Yamada, T., et al. (2011). Enterotypes of the Human Gut Microbiome. *Nature* 473, 174–180. doi:10.1038/nature09944

Bäckhed, F., Ley, R. E., Sonnenburg, J. L., Peterson, D. A., and Gordon, J. I. (2005). Host-bacterial Mutualism in the Human Intestine. *Science* 307, 1915–1920. doi:10.1126/science.1104816

Bailey, M. J., Naik, N. N., Wild, L. E., Patterson, W. B., and Alderete, T. L. (2020). Exposure to Air Pollutants and the Gut Microbiota: A Potential Link Between Exposure, Obesity, and Type 2 Diabetes. *Gut Microbes* 11, 1188–1202. doi:10.1080/19490976.2020.1749754

Barichella, M., Pacchetti, C., Bolliri, C., Cassani, E., Iorio, L., Pusani, C., et al. (2016). Probiotics and Prebiotic Fiber for Constipation Associated with Parkinson Disease. *Neurology* 87, 1274–1280. doi:10.1212/WNL.0000000000003127

Baryshnikova, A. (2016). Systematic Functional Annotation and Visualization of Biological Networks. *Cel Syst.* 2, 412–421. doi:10.1016/j.cels.2016.04.014

Bedarf, J. R., Hildebrand, F., Coelho, L. P., Sunagawa, S., Bahram, M., Goeser, F., et al. (2017). Functional Implications of Microbial and Viral Gut Metagenome Changes in Early Stage L-DOPA-Naïve Parkinson's Disease Patients. *Genome Med.* 9, 1–13. doi:10.1186/s13073-017-0428-y

Blum, W. E. H., Zechmeister-Boltenstern, S., and Keiblinger, K. M. (2019). Does Soil Contribute to the Human Gut Microbiome? *Microorganisms* 7, 287. doi:10.3390/microorganisms7090287

Burchill, E., Lymberopoulos, E., Menozzi, E., Budhdeo, S., McIlroy, J. R., Macnaughtan, J., et al. (2021). The Unique Impact of COVID-19 on Human Gut Microbiome Research. *Front. Med.* 8, 267. doi:10.3389/FMED.2021.652464

Carlsson, G. (2009). Topology and Data. *Bull. Amer. Math. Soc.* 46, 255–308. doi:10.1090/S0273-0979-09-01249-X

Casadevall, A. (2017). The Pathogenic Potential of a Microbe. *mSphere* 2, e00015. doi:10.1128/mSphere.00015-17

Clarke, G., Sandhu, K. V., Griffin, B. T., Dinan, T. G., Cryan, J. F., and Hyland, N. P. (2019). Gut Reactions: Breaking Down Xenobiotic-Microbiome Interactions. *Pharmacol. Rev.* 71, 198–224. doi:10.1124/pr.118.015768

De Filippo, C., Cavalieri, D., Di Paola, M., Ramazzotti, M., Poullet, J. B., Massart, S., et al. (2010). Impact of Diet in Shaping Gut Microbiota Revealed by a Comparative Study in Children from Europe and Rural Africa. *Proc. Natl. Acad. Sci.* 107, 14691–14696. doi:10.1073/pnas.1005963107

Dethlefsen, L., Huse, S., Sogin, M. L., and Relman, D. A. (2008). The Pervasive Effects of an Antibiotic on the Human Gut Microbiota, as Revealed by Deep 16s rRNA Sequencing. *Plos Biol.* 6, e280–2400. doi:10.1371/journal.pbio.0060280

Dugas, L. R., Fuller, M., Gilbert, J., and Layden, B. T. (2016). The Obese Gut Microbiome Across the Epidemiologic Transition. *Emerg. Themes Epidemiol.* 13, 1–9. doi:10.1186/s12982-015-0044-5

Dziarski, R., Park, S. Y., Kashyap, D. R., Dowd, S. E., and Gupta, D. (2016). Pglyrp-Regulated Gut Microflora Prevotella Falsenii, Parabacteroides Distasonis and Bacteroides Eggerthii Enhance and Alistipes Finegoldii Attenuates Colitis in Mice. *PLoS One* 11, e0146162. doi:10.1371/journal.pone.0146162

Enright, E. F., Gahan, C. G., Joyce, S. A., and Griffin, B. T. (2016). The Impact of the Gut Microbiota on Drug Metabolism and Clinical Outcome. *Yale J. Biol. Med.* 89, 375–382.

Falony, G., Joossens, M., Vieira-Silva, S., Wang, J., Darzi, Y., Faust, K., et al. (2016). Population-level Analysis of Gut Microbiome Variation. *Science* 352, 560–564. doi:10.1126/science.aad3503

Ferrocino, I., Di Cagno, R., De Angelis, M., Turroni, S., Vannini, L., Bancalari, E., et al. (2015). Fecal Microbiota in Healthy Subjects Following Omnivore, Vegetarian and Vegan Diets: Culturable Populations and rRNA DGGE Profiling. *PLoS One* 10, e0128669. doi:10.1371/journal.pone.0128669

Flowers, S. A., Ward, K. M., and Clark, C. T. (2020). The Gut Microbiome in Bipolar Disorder and Pharmacotherapy Management. *Neuropsychobiology* 79, 43–49. doi:10.1159/000504496

Franco-De-Moraes, A. C., De Almeida-Pititto, B., Da Rocha Fernandes, G., Gomes, E. P., Da Costa Pereira, A., and Ferreira, S. R. G. (2017). Worse Inflammatory Profile in Omnivores Than in Vegetarians Associates with the Gut Microbiota Composition. *Diabetol. Metab. Syndr.* 9, 62. doi:10.1186/s13098-017-0261-x

Fu, W., Gandhi, V. P., Cao, L., Liu, H., and Zhou, Z. (2012). Rising Consumption of Animal Products in China and India: National and Global Implications. *China World Econ.* 20, 88–106. doi:10.1111/j.1749-124X.2012.01289.x

Garcia-Mantrana, I., Selma-Royo, M., Alcantara, C., and Collado, M. C. (2018). Shifts on Gut Microbiota Associated to Mediterranean Diet Adherence and Specific Dietary Intakes on General Adult Population. *Front. Microbiol.* 9, 890. doi:10.3389/fmicb.2018.00890

Ghaisas, S., Maher, J., and Kanthasamy, A. (2016). Gut Microbiome in Health and Disease: Linking the Microbiome-Gut-Brain Axis and Environmental Factors in the Pathogenesis of Systemic and Neurodegenerative Diseases. *Pharmacol. Ther.* 158, 52–62. doi:10.1016/j.pharmthera.2015.11.012

Gourmelon, V., Maggia, L., Powell, J. R., Gigante, S., Hortal, S., Gueunier, C., et al. (2016). Environmental and Geographical Factors Structure Soil Microbial Diversity in New Caledonian Ultramafic Substrates: A Metagenomic Approach. *PLoS One* 11, e0167405. doi:10.1371/journal.pone.0167405

Government of Canada (2017). Tackling Obesity in Canada: Obesity and Excess Weight Rates in Canadian Adults. Available at: https://www.canada.ca/en/public-health/services/publications/healthy-living/obesity-excess-weight-rates-canadian-adults.html (Accessed July 6, 2021).

Gupta, V. K., Paul, S., and Dutta, C. (2017). Geography, Ethnicity or Subsistence-specific Variations in Human Microbiome Composition and Diversity. *Front. Microbiol.* 8, 1162. doi:10.3389/fmicb.2017.01162

Haghikia, A., Jörg, S., Duscha, A., Berg, J., Manzel, A., Waschbisch, A., et al. (2015). Dietary Fatty Acids Directly Impact Central Nervous System Autoimmunity via the Small Intestine. *Immunity* 43, 817–829. doi:10.1016/j.immuni.2015.09.007

Handford, C. E., Elliott, C. T., and Campbell, K. (2015). A Review of the Global Pesticide Legislation and the Scale of Challenge in Reaching the Global Harmonization of Food Safety Standards. *Integr. Environ. Assess. Manag.* 11, 525–536. doi:10.1002/ieam.1635

Hensel, F., Moor, M., and Rieck, B. (2021). A Survey of Topological Machine Learning Methods. *Front. Artif. Intell.* 4, 681108. doi:10.3389/frai.2021.681108

Hill-Burns, E. M., Debelius, J. W., Morton, J. T., Wissemann, W. T., Lewis, M. R., Wallen, Z. D., et al. (2017). Parkinson's Disease and Parkinson's Disease Medications Have Distinct Signatures of the Gut Microbiome. *Mov Disord.* 32, 739–749. doi:10.1002/mds.26942

GBD 2017 Disease and Injury Incidence and Prevalence CollaboratorsJames, S. L., Abate, D., Abate, K. H., Abay, S. M., Abbafati, C., et al. (2018). Global, Regional, and National Incidence, Prevalence, and Years Lived with Disability for 354 Diseases and Injuries for 195 Countries and Territories, 1990–2017: A Systematic Analysis for the Global Burden of Disease Study 2017. *Lancet* 392, 1789–1858. doi:10.1016/S0140-6736(18)32279-7

Kamada, N., Chen, G. Y., Inohara, N., and Núñez, G. (2013). Control of Pathogens and Pathobionts by the Gut Microbiota. *Nat. Immunol.* 14, 685–690. doi:10.1038/ni.2608

Karlsson, F. H., Nookaew, I., and Nielsen, J. (2014). Metagenomic Data Utilization and Analysis (MEDUSA) and Construction of a Global Gut Microbial Gene Catalogue. *Plos Comput. Biol.* 10, e1003706. doi:10.1371/journal.pcbi.1003706

Klein, E. Y., Van Boeckel, T. P., Martinez, E. M., Pant, S., Gandra, S., Levin, S. A., et al. (2018). Global Increase and Geographic Convergence in Antibiotic Consumption Between 2000 and 2015. *Proc. Natl. Acad. Sci. USA* 115, E3463–E3470. doi:10.1073/pnas.1717295115

Lahti, L., Salojärvi, J., Salonen, A., Scheffer, M., and de Vos, W. M. (2014). Tipping Elements in the Human Intestinal Ecosystem. *Nat. Commun.* 5, 4344. doi:10.1038/ncomms5344

Langelier, C., Graves, M., Kalantar, K., Caldera, S., Durrant, R., Fisher, M., et al. (2019). Microbiome and Antimicrobial Resistance Gene Dynamics in International Travelers. *Emerg. Infect. Dis.* 25, 1380–1383. doi:10.3201/eid2507.181492

Le Bastard, Q., Al-Ghalith, G. A., Grégoire, M., Chapelet, G., Javaudin, F., Dailly, E., et al. (2018). Systematic Review: Human Gut Dysbiosis Induced by Non-antibiotic Prescription Medications. *Aliment. Pharmacol. Ther.* 47, 332–345. doi:10.1111/apt.14451

Levy, M., Blacher, E., and Elinav, E. (2017). Microbiome, Metabolites and Host Immunity. *Curr. Opin. Microbiol.* 35, 8–15. doi:10.1016/j.mib.2016.10.003

Li, C., Cui, L., Yang, Y., Miao, J., Zhao, X., Zhang, J., et al. (2019a). Gut Microbiota Differs Between Parkinson's Disease Patients and Healthy Controls in Northeast China. *Front. Mol. Neurosci.* 12, 171. doi:10.3389/fnmol.2019.00171

Li, N., Ma, W.-T., Pang, M., Fan, Q.-L., and Hua, J.-L. (2019b). The Commensal Microbiota and Viral Infection: A Comprehensive Review. *Front. Immunol.* 10, 1551. doi:10.3389/fimmu.2019.01551

Liang, J. Q., Li, T., Nakatsu, G., Chen, Y. X., Yau, T. O., Chu, E., et al. (2020). A Novel Faecal Lachnoclostridium Marker for the Non-invasive Diagnosis of Colorectal Adenoma and Cancer. *Gut* 69, 1248–1257. doi:10.1136/gutjnl-2019-318532

Liao, T., Wei, Y., Luo, M., Zhao, G.-P., and Zhou, H. (2019). Tmap: An Integrative Framework Based on Topological Data Analysis for Population-Scale Microbiome Stratification and Association Studies. *Genome Biol.* 20, 1–19. doi:10.1186/s13059-019-1871-4

Liu, A., Lv, H., Wang, H., Yang, H., Li, Y., and Qian, J. (2020). Aging increases the severity of colitis and the related changes to the gut barrier and gut microbiota in humans and mice. *J. Gerontol. Ser. A Biol. Sci. Med. Sci.* 75, 1284–1292. doi:10.1093/gerona/glz263

Lupatini, M., Korthals, G. W., de Hollander, M., Janssens, T. K. S., and Kuramae, E. E. (2017). Soil Microbiome Is More Heterogeneous in Organic Than in Conventional Farming System. *Front. Microbiol.* 7, 2064. doi:10.3389/fmicb.2016.02064

Lv, Y., Qin, X., Jia, H., Chen, S., Sun, W., and Wang, X. (2019). The Association Between Gut Microbiota Composition and BMI in Chinese Male College Students, as Analysed by Next-Generation Sequencing. *Br. J. Nutr.* 122, 986–995. doi:10.1017/S0007114519001909

Maini Rekdal, V., Bess, E. N., Bisanz, J. E., Turnbaugh, P. J., and Balskus, E. P. (2019). Discovery and Inhibition of an Interspecies Gut Bacterial Pathway for Levodopa Metabolism. *Science* 364, 364. doi:10.1126/science.aau6323

McDonald, D., Hyde, E., Debelius, J. W., Morton, J. T., Gonzalez, A., Ackermann, G., et al. (2018). American Gut: An Open Platform for Citizen Science Microbiome Research. *mSystems* 3, e00031. doi:10.1128/mSystems.00031-18

Mead, A. (1992). Review of the Development of Multidimensional Scaling Methods. *The Statistician* 41, 27. doi:10.2307/2348634

Mitchell, A. L., Almeida, A., Beracochea, M., Boland, M., Burgin, J., Cochrane, G., et al. (2020). MGnify: The Microbiome Analysis Resource in 2020. *Nucleic Acids Res.* 48, D570–D578. doi:10.1093/nar/gkz1035

Monda, V., Villano, I., Messina, A., Valenzano, A., Esposito, T., Moscatelli, F., et al. (2017). Exercise Modifies the Gut Microbiota with Positive Health Effects. *Oxidative Med. Cell Longevity* 2017, 3831972. doi:10.1155/2017/3831972

Morrison, D. J., and Preston, T. (2016). Formation of Short Chain Fatty Acids by the Gut Microbiota and Their Impact on Human Metabolism. *Gut Microbes* 7, 189–200. doi:10.1080/19490976.2015.1134082

Patil, D. P., Dhotre, D. P., Chavan, S. G., Sultan, A., Jain, D. S., Lanjekar, V. B., et al. (2012). Molecular Analysis of Gut Microbiota in Obesity Among Indian Individuals. *J. Biosci.* 37, 647–657. doi:10.1007/s12038-012-9244-0

Petersen, C., and Round, J. L. (2014). Defining Dysbiosis and its Influence on Host Immunity and Disease. *Cell. Microbiol.* 16, 1024–1033. doi:10.1111/cmi.12308

Qiu, Y.-L., Tourlousse, D. M., Matsuura, N., Ohashi, A., and Sekiguchi, Y. (2017). Draft Genome Sequence of Paludibacter Jiangxiensis NM7 T , A Propionate-Producing Fermentative Bacterium. *Genome Announc* 5, e00667. doi:10.1128/genomeA.00667-17

Quesada-Molina, M., Muñoz-Garach, A., Tinahones, F. J., and Moreno-Indias, I. (2019). A New Perspective on the Health Benefits of Moderate Beer Consumption: Involvement of the Gut Microbiota. *Metabolites* 9, 272. doi:10.3390/metabo9110272

Riaz Rajoka, M. S., Shi, J., Mehwish, H. M., Zhu, J., Li, Q., Shao, D., et al. (2017). Interaction Between Diet Composition and Gut Microbiota and its Impact on Gastrointestinal Tract Health. *Food Sci. Hum. Wellness* 6, 121–130. doi:10.1016/j.fshw.2017.07.003

Riddle, M. S., and Connor, B. A. (2016). The Traveling Microbiome. *Curr. Infect. Dis. Rep.* 18, 1–13. doi:10.1007/s11908-016-0536-7

Rinninella, E., Raoul, P., Cintoni, M., Franceschi, F., Miggiano, G., Gasbarrini, A., et al. (2019). What Is the Healthy Gut Microbiota Composition? A Changing Ecosystem Across Age, Environment, Diet, and Diseases. *Microorganisms* 7, 14. doi:10.3390/microorganisms7010014

Ritchie, H., and Roser, M. (2019). Meat and Dairy Production. *Our World Data*. Available at: https://ourworldindata.org/meat-production (Accessed March 14, 2021).

Rooks, M. G., and Garrett, W. S. (2016). Gut Microbiota, Metabolites and Host Immunity. *Nat. Rev. Immunol.* 16, 341–352. doi:10.1038/nri.2016.42

Saresella, M., Marventano, I., Barone, M., La Rosa, F., Piancone, F., Mendozzi, L., et al. (2020). Alterations in Circulating Fatty Acid Are Associated with Gut Microbiota Dysbiosis and Inflammation in Multiple Sclerosis. *Front. Immunol.* 11, 1390. doi:10.3389/fimmu.2020.01390

Scher, J. U., Nayak, R. R., Ubeda, C., Turnbaugh, P. J., and Abramson, S. B. (2020). Pharmacomicrobiomics in Inflammatory Arthritis: Gut Microbiome as Modulator of Therapeutic Response. *Nat. Rev. Rheumatol.* 16, 282–292. doi:10.1038/s41584-020-0395-3

Schmidt, B., Mulder, I. E., Musk, C. C., Aminov, R. I., Lewis, M., Stokes, C. R., et al. (2011). Establishment of Normal Gut Microbiota Is Compromised Under Excessive hygiene Conditions. *PLoS One* 6, e28284. doi:10.1371/journal.pone.0028284

Seeman, M. V. (2021). The Gut Microbiome and Antipsychotic Treatment Response. *Behav. Brain Res.* 396, 112886. doi:10.1016/j.bbr.2020.112886

Sender, R., Fuchs, S., and Milo, R. (2016). Revised Estimates for the Number of Human and Bacteria Cells in the Body. *Plos Biol.* 14, e1002533. doi:10.1371/journal.pbio.1002533

Shankar, V., Gouda, M., Moncivaiz, J., Gordon, A., Reo, N. V., Hussein, L., et al. (2017). Differences in Gut Metabolites and Microbial Composition and Functions Between Egyptian and U.S. Children Are Consistent with Their Diets. *mSystems* 2, e00169. doi:10.1128/msystems.00169-16

Shapira, M. (2016). Gut Microbiotas and Host Evolution: Scaling Up Symbiosis. *Trends Ecol. Evol.* 31, 539–549. doi:10.1016/j.tree.2016.03.006

Shoaie, S., Karlsson, F., Mardinoglu, A., Nookaew, I., Bordel, S., and Nielsen, J. (2013). Understanding the Interactions Between Bacteria in the Human Gut through Metabolic Modeling. *Sci. Rep.* 3, 2532. doi:10.1038/srep02532

Singer-Englar, T., Barlow, G., and Mathur, R. (2019). Obesity, Diabetes, and the Gut Microbiome: An Updated Review. *Expert Rev. Gastroenterol. Hepatol.* 13, 3–15. doi:10.1080/17474124.2019.1543023

Singh, G., Mémoli, F., and Carlsson, G. (2007). Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition. *Eurographics Symposium on Point-Based Graphics* (The Eurographics Association), 91–100. doi:10.2312/SPBG/SPBG07/091-100

Tauzin, G., Lupo, U., Tunstall, L., Burella Pérez, J., Caorsi, M., Medina-Mardones, A. M., et al. (2021). Giotto-Tda: A Topological Data Analysis Toolkit for Machine Learning and Data Exploration. *J. Mach. Learn. Res.* 22, 1–6. Available at: https://github.com/giotto-ai/pyflagser (Accessed March 12, 2021).

Thevaranjan, N., Puchta, A., Schulz, C., Naidoo, A., Szamosi, J. C., Verschoor, C. P., et al. (2017). Age-Associated Microbial Dysbiosis Promotes Intestinal Permeability, Systemic Inflammation, and Macrophage Dysfunction. *Cell Host & Microbe* 21, 455–466.e4. doi:10.1016/j.chom.2017.03.002

Thompson, L. R., Sanders, J. G., Sanders, J. G., McDonald, D., Amir, A., Ladau, J., et al. (2017). A Communal Catalogue Reveals Earth's Multiscale Microbial Diversity. *Nature* 551, 457–463. doi:10.1038/nature24621

Tu, P., Chi, L., Bodnar, W., Zhang, Z., Gao, B., Bian, X., et al. (2020). Gut Microbiome Toxicity: Connecting the Environment and Gut Microbiome-Associated Diseases. *Toxics* 8, 19. doi:10.3390/toxics8010019

Turnbaugh, P. J., Ley, R. E., Mahowald, M. A., Magrini, V., Mardis, E. R., and Gordon, J. I. (2006). An Obesity-Associated Gut Microbiome with Increased Capacity for Energy Harvest. *Nature* 444, 1027–1031. doi:10.1038/nature05414

Vallès, Y., and Francino, M. P. (2018). Air Pollution, Early Life Microbiome, and Development. *Curr. Envir Health Rpt* 5, 512–521. doi:10.1007/s40572-018-0215-y

Vaughn, A. C., Cooper, E. M., Dilorenzo, P. M., O'Loughlin, L. J., Konkel, M. E., Peters, J. H., et al. (2017). Energy-dense Diet Triggers Changes in Gut Microbiota, Reorganization of Gut-Brain Vagal Communication and Increases Body Fat Accumulation. *Acta Neurobiol. Exp. (Wars)* 77, 18–30. doi:10.21307/ane-2017-033

Venegas, D. P., De La Fuente, M. K., Landskron, G., González, M. J., Quera, R., Dijkstra, G., et al. (2019). Short Chain Fatty Acids (SCFAs)mediated Gut Epithelial and Immune Regulation and its Relevance for Inflammatory Bowel Diseases. *Front. Immunol.* 10, 277. doi:10.3389/fimmu.2019.00277

Walters, W. A., Xu, Z., and Knight, R. (2014). Meta-analyses of Human Gut Microbes Associated with Obesity and IBD. *FEBS Lett.* 588, 4223–4233. doi:10.1016/j.febslet.2014.09.039

Wang, C., Li, W., Wang, H., Ma, Y., Zhao, X., Zhang, X., et al. (2019). Saccharomyces Boulardii Alleviates Ulcerative Colitis Carcinogenesis in Mice by Reducing TNF-α and IL-6 Levels and Functions and by Rebalancing Intestinal Microbiota. *BMC Microbiol.* 19, 1–12. doi:10.1186/s12866-019-1610-8

Wexler, H. M. (2007). Bacteroides : The Good, the Bad, and the Nitty-Gritty. *Clin. Microbiol. Rev.* 20, 593–621. doi:10.1128/CMR.00008-07

Wilson, I. D., and Nicholson, J. K. (2017). Gut Microbiome Interactions with Drug Metabolism, Efficacy, and Toxicity. *Translational Res.* 179, 204–222. doi:10.1016/j.trsl.2016.08.002

Wu, G. D., Chen, J., Hoffmann, C., Bittinger, K., Chen, Y.-Y., Keilbaugh, S. A., et al. (2011). Linking Long-Term Dietary Patterns with Gut Microbial Enterotypes. *Science* 334, 105–108. doi:10.1126/science.1208344

Wu, S., Sun, C., Li, Y., Wang, T., Jia, L., Lai, S., et al. (2020). GMrepo: A Database of Curated and Consistently Annotated Human Gut Metagenomes. *Nucleic Acids Res.* 48, D545–D553. doi:10.1093/nar/gkz764

Zimmer, J., Lange, B., Frick, J.-S., Sauer, H., Zimmermann, K., Schwiertz, A., et al. (2012). A Vegan or Vegetarian Diet Substantially Alters the Human Colonic Faecal Microbiota. *Eur. J. Clin. Nutr.* 66, 53–60. doi:10.1038/ejcn.2011.141

Conflict of Interest: Authors MA and NS are co-founders of BioCorteX Ltd. Author MA was employed by the company Rolls-Royce Ltd.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers in Artificial Intelligence

ORIGINAL RESEARCH
published: 23 August 2021
doi: 10.3389/frai.2021.668163

On Topological Analysis of fs-LIMS Data. Implications for in Situ Planetary Mass Spectrometry

Rustam A. Lukmanov[1], Andreas Riedo[1], David Wacey[2], Niels F. W. Ligterink[1], Valentine Grimaudo[1], Marek Tulej[1], Coenraad de Koning[1], Anna Neubeck[3] and Peter Wurz[1]*

[1]Space Research and Planetary Sciences (WP), University of Bern, Bern, Switzerland, [2]Centre for Microscopy, Characterisation and Analysis, The University of Western Australia, Perth, WA, Australia, [3]Department of Earth Sciences, Uppsala University, Uppsala, Sweden

In this contribution, we present results of non-linear dimensionality reduction and classification of the fs laser ablation ionization mass spectrometry (LIMS) imaging dataset acquired from the Precambrian Gunflint chert (1.88 Ga) using a miniature time-of-flight mass spectrometer developed for *in situ* space applications. We discuss the data generation, processing, and analysis pipeline for the classification of the recorded fs-LIMS mass spectra. Further, we define topological biosignatures identified for Precambrian Gunflint microfossils by projecting the recorded fs-LIMS intensity space into low dimensions. Two distinct subtypes of microfossil-related spectra, a layer of organic contamination and inorganic quartz matrix were identified using the fs-LIMS data. The topological analysis applied to the fs-LIMS data allows to gain additional knowledge from large datasets, formulate hypotheses and quickly generate insights from spectral data. Our contribution illustrates the utility of applying spatially resolved mass spectrometry in combination with topology-based analytics in detecting signatures of early (primitive) life. Our results indicate that fs-LIMS, in combination with topological methods, provides a powerful analytical framework and could be applied to the study of other complex mineralogical samples.

Keywords: fs-LIMS, mass-spectrometry, UMAP (uniform manifold approximation and projection), mapper, microfossils, mars, Gunflint

OPEN ACCESS

Edited by:
Umberto Lupo,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Ricardo Arevalo,
University of Maryland, United States
Mathieu Carrière,
Institut National de Recherche en Informatique et en Automatique (INRIA), France

****Correspondence:***
Rustam A. Lukmanov
Rustam.lukmanov@space.unibe.ch

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *15 February 2021*
Accepted: *05 August 2021*
Published: *23 August 2021*

Citation:
Lukmanov RA, Riedo A, Wacey D, Ligterink NFW, Grimaudo V, Tulej M, de Koning C, Neubeck A and Wurz P (2021) On Topological Analysis of fs-LIMS Data. Implications for in Situ Planetary Mass Spectrometry. Front. Artif. Intell. 4:668163. doi: 10.3389/frai.2021.668163

INTRODUCTION

The current state of space exploration is on the verge of new frontiers, holding promise for discoveries on other planetary bodies through *in-situ* robotic exploration (Vago et al., 2015). For example, Mars and the icy moons of Jupiter and Saturn, once thought to be lifeless, have gained more attention from the scientific community in recent decades due to new data informing upon the potential habitability of these bodies (Priscu and Hand 2012; Garcia-Lopez and Cid 2017; McMahon et al., 2018). Thus, there is an ongoing need for sensitive and high-throughput space instrumentation providing precise analytical data on a microscale (Navarro-González et al., 2006; Goesmann et al., 2017). However, space-type instruments are usually small and provide only a fraction of the sensitivity and overall capability of their full-scale laboratory counterparts. Reduction in performance occurs due to the strict constraints on size, power consumption, and weight of the scientific payload. Therefore, the development of new miniature instruments and analytical methods with improved capabilities is a continuously pressing issue (Li et al., 2017; Arevalo et al., 2018; Stevens et al., 2019; Wurz et al., 2020).

Laser-based mass spectrometry (Laser Ablation Ionization and Desorption Mass Spectrometry–LIMS and LDMS) is a modern and compact analytical method that promises to greatly enhance the quality of chemical analysis on planetary bodies (Riedo et al., 2013b; Arevalo et al., 2018). The first LIMS instrument selected and built for a planetary lander was LASMA, developed for the Phobos-Grunt mission (Managadze et al., 2010). Recently, the second LIMS instrument was chosen for the upcoming ExoMars mission/Rosalind Franklin Rover (Goesmann et al., 2017), further facilitating developments in this field. Laser-based mass spectrometry, developed for *in-situ* planetary exploration, as a versatile method, can provide a description of molecular composition and element, isotope characterization of solids (Moreno-García et al., 2016; Arevalo et al., 2020; Tulej et al., 2020). The time-of-flight version of LDMS has been shown to be capable of measuring extremely low concentrations (fmole) of amino acids in the desorption mode (Ligterink et al., 2020). LIMS modification of this instrument has been reported to measure ppbw level trace elements and routinely measure fine chemistry from a variety of samples (Riedo et al., 2013a; Neuland et al., 2016; Wiesendanger et al., 2017). Moreover, a number of reports have indicated that LIMS, particularly fs-LIMS, might be applicable to the detection of faint signatures of life from microscopic inclusions (Tulej et al., 2015; Wiesendanger et al., 2018) and low-biomass Martian analogs (Stevens et al., 2019; Riedo et al., 2020). However, the field of study of early and primitive life remains profoundly complex (Brasier and Wacey 2012; Westall et al., 2015; Wacey et al., 2016) with no single chemical criterion that can be assigned as definitive proof of biogenicity. A number of authors have proposed a multi-criteria approach, where a multitude of methods needs to be applied before any conclusions can be drawn (Hofmann 2008; Brasier and Wacey 2012; Hand et al., 2017; Vago et al., 2017; Neveu et al., 2018; Chan et al., 2019). The multi-method approach enhances the size of parametric space and reduces the probability of false-positive detection. Therefore, any advancement within each of the applied methods can increase the overall confidence of the correct identification of signatures of life.

In this contribution, we hypothesize that on the basis of the full feature scale (mass range) present in the fs-LIMS spectral datasets, it is possible to identify minerals and compounds of specific chemistry using an unsupervised data-driven approach. We describe a topology-based analysis pipeline to define the complexity of the fs-LIMS imaging data in low dimensions and identify groups of spectra that share a significant degree of similarity. We apply the aforementioned method to 18,000 composite spectra acquired from the Gunflint chert (1.88 Ga), which contains populations of well-preserved Precambrian microfossils of proven biological origin (Wacey et al., 2013). The analysis of the data reveals four distinct populations of fs-LIMS spectra, which correspond to two groups of microfossils, the quartz matrix in which microfossils are entombed and organic surface contamination spectra. Moreover, we describe a fine transitional structure between classes and argue that low dimensional representations are of high utility in *in-situ* mass-spectrometry and space research. Further, we speculate that our approach is scalable to non-space instruments and may, therefore, prove useful in the field of Precambrian micropaleontology and high-dimensional analytical chemistry in general.

METHODS

In this study, we use laser ablation ionization mass spectrometry for the characterization of the chemical composition of the Gunflint sample and optical microscopy to identify morphological features. A thorough review of LIMS operation principles, current state-of-the-art, and application case studies can be found in a number of previous reports (Tulej et al., 2014; Wiesendanger et al., 2017; Grimaudo et al., 2020; Ligterink et al., 2020) and reviews (Grimaudo et al., 2019; Azov et al., 2020), and therefore, only a short description will be given here. In the simplest case, LIMS instruments include two main parts–a pulsed laser system to ablate and ionize materials and a mass analyzer to separate and register ions produced during the ablation and ionization process. The fs-LIMS is a successor of ns-LIMS, with the only difference that the mass analyzer is coupled to the fs laser system. Current commercial fs lasers can provide peak power fluences up to terawatt/cm^2, compressed to very short pulses of femto-second duration. Such high powers can ionize any material, thus, providing means for an isotope and element characterization of any solid with very small detection limits and reduced matrix effects (Riedo et al., 2013c). As an ion source, we have installed a Ti:Sapphire laser with chirped pulse amplification, which provides a stable IR-775 nm, ~190 fs laser radiation. Conversion of the fundamental wavelength from IR-775 nm to UV-258 nm was made using a commercially available third-harmonic generation module.

The fs-LIMS system used in this study consists of a miniature time-of-flight (TOF) mass analyzer (⌀ 60 × 160 mm) (*see* **Figure 1**) with an axially symmetric design and single unit mass resolution. The instrument was developed for *in-situ* space applications, and due to its miniature design it could be placed on a rover, lander, or even used as a handheld instrument (Wurz et al., 2020). In normal operation mode, fs-LIMS could identify major chemical composition along with ppmw-level concentrations of trace elements. As shown in **Figure 1**, the TOF mass analyzer consists of entrance ion optics (where ions are confined and accelerated), a drift tube (where ions experience mass/charge separation), a reflectron (ion mirror which uses an electric field), and a microchannel plate (MCP) detector system (Riedo et al., 2017) to register ion flux. The schematic illustration of the fs-LIMS sample analysis is shown in **Figure 1B**. The focused blue light indicates the fs-UV-258 nm laser radiation that passes through the instrument and ablates the small area of the sample with a diameter of the ablation spot of about 5 μm. The positioning of the ablation spots is determined by the internal microscopy system. The objective of the microscope is located at a fixed offset from the instrument. After ablation and ionization, positively charged ions are guided by an electric field of the instrument into a defined parabolic trajectory so that every ion that enters the instrument will land on the surface of the detector. Incoming time-separated ion flux launches an electron avalanche

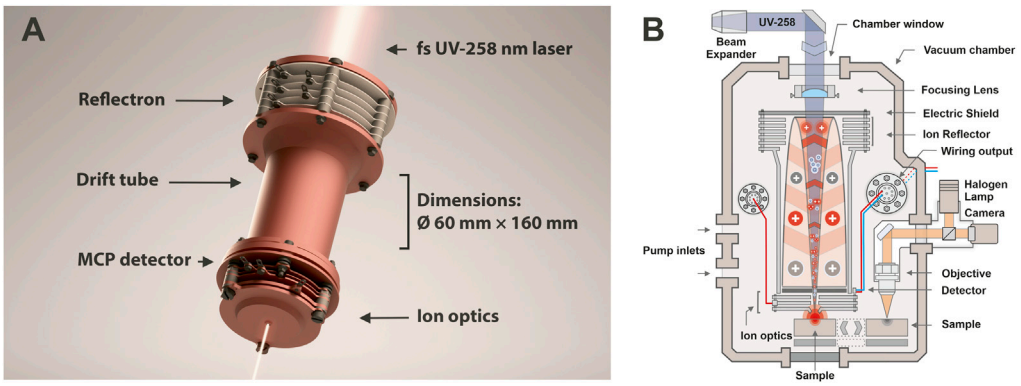


FIGURE 1 | (A) 3D render of our miniature time-of-flight mass analyzer. Location of the reflectron, drift tube, entrance ion optics, MCP detector, and dimensions of the instrument are denoted. The focusing fs-UV laser light shown on the top and the bottom and illustrates an axial design of the mass analyzer. Sample positioning is not shown. However, in the laboratory setting, the investigated sample is positioned in close vicinity to the entrance plate of the ion optical system of the mass analyzer, right in the position of the laser focus, to achieve ablation and subsequent ionization of target material. **(B)** Schematic illustration of the fs-LIMS. An fs-laser radiation (blue line) ablates and ionizes material from the sample. The positively charged ions are separated and detected using the time-of-flight mass spectrometer. The ablation position can be precisely located using an integrated microscopy system.

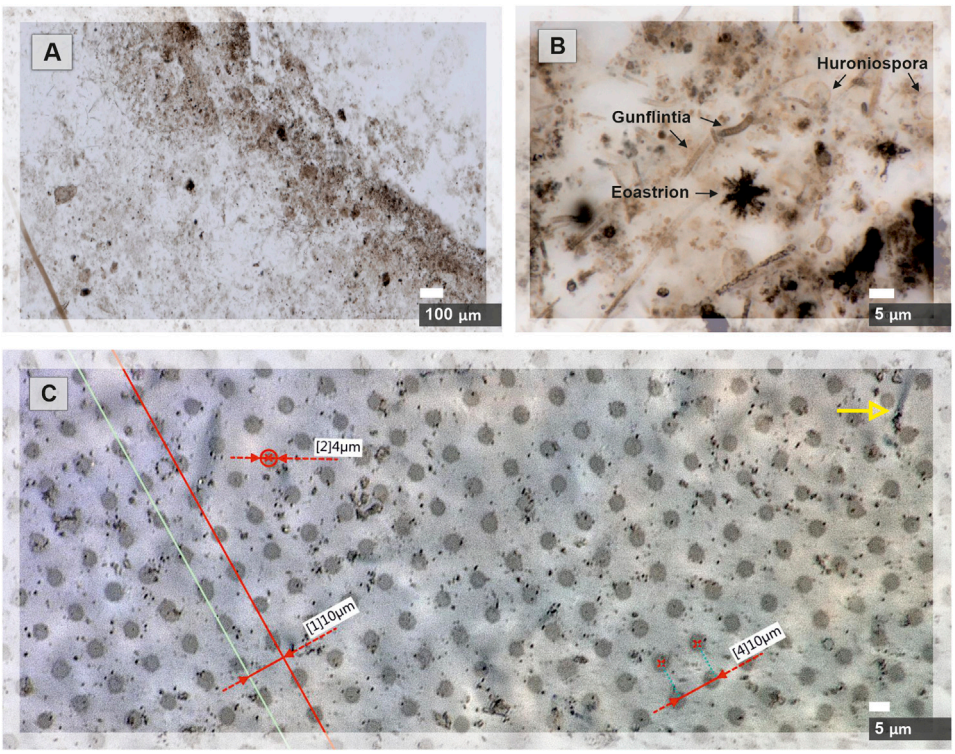


FIGURE 2 | Microscope images of the Gunflint chert before and after the fs-LIMS imaging campaign are shown. **(A)** Microscope image of the area (0.9*2 mm^2) chosen for the chemical imaging with our fs-LIMS system. The dark brown patches distributed through the sample and forming a diffuse layer in the middle of the picture represent a bio-lamination surface. **(B)** Close-up microscope picture of individual microfossils from the bio-lamination surface. Filamentous (Gunflintia), star-shaped (Eoastrion), and spherular microfossils (Huroniospora) can be seen. **(C)** Microscope picture of laser ablation craters (0.9*2 mm^2 area covered with 90*200 pixels–18,000 ablation positions) formed after the fs-LIMS imaging campaign. Red lines denote the accuracy of sample positioning (the gap between ablation craters is consistently 10 µm) and identify the ablation crater diameters. Individual craters range in diameter from 4 to 5 µm. Note, on the upper part of the image, the yellow arrow indicates an individual microfossil body. The size of the microfossil can be compared with the diameter of the analytical spot.

within the microchannels of the detector system and creates a measurable current on the output anodes. Thus, time-of-flight LIMS measures an output current per unit of time, which is correlated with the element and isotope abundances of an investigated spot. Note that the image of the fs-laser beam passing through the instrument (**Figure 1A**) is exaggerated - in

the laboratory setting, the laser focal point is located in close proximity to the entrance electrode of the mass analyzer (*see* **Figure 1B**).

The investigation of a 30 μm thick thin-section of Gunflint chert has been conducted with our miniature fs-LIMS system. The sample acquired from the Gunflint Formation (Schreiber beach locality, Ontario, Canada; Wacey et al., 2012, 2013) represents a finely polished thin slice of the original rock, glued to the glass substrate and mounted on a steel holder. Preliminary optical microscopy was performed on the sample to identify specific areas of microfossil aggregation (*see* **Figure 2A**). Matrix material in which microfossils are preserved was identified to be microcrystalline quartz. Chemical imaging of the rectangular area, containing a bio-lamination surface (aggregation of microfossils within a stromatolite) and a clear host area (quartz filled matrix) was done with the LIMS system using the fs UV-258 nm laser, which provides a flux of 4.8 eV UV photons, which is well suited for ionization of glasses and other non-conductive materials with low absorption coefficients.

Figure 1A. 3D rendering of our miniature time-of-flight mass analyzer. Location of the reflectron, drift tube, entrance ion optics, MCP detector, and dimensions of the instrument are denoted. The focusing fs-UV laser light shown on the top and the bottom and illustrates an axial design of the mass analyzer. Sample positioning is not shown. However, in the laboratory setting, the investigated sample is positioned in close vicinity to the entrance plate of the ion optical system of the mass analyzer, right in the position of the laser focus, to achieve ablation and subsequent ionization of target material. **Figure 1B**. Schematic illustration of the fs-LIMS. The fs-laser radiation (blue line) ablates and ionizes material from the sample. The positively charged ions are separated and detected using the time-of-flight mass spectrometer. The ablation position can be precisely located using the integrated microscopy system.

Data Acquisition

A rectangular area of 0.9 × 2 mm^2 was investigated using the fs-LIMS system (*see* **Figure 2A**). A relatively low number of laser pulses were applied to each surface position – 200 laser shots, to avoid material displacement and crater-to-crater cross-contamination. The spatially resolved measurements conducted on the Gunflint chert resulted in the collection of 18,000 composite spectra (collected from the grid - 90 by 200 position or ablation sites). A composite spectrum collected from the given position (or ablation site) resulted in the accumulation of 200 single-shot spectra, with 64,000 data points digitized per spectrum. Thus, the total number of registered shots resulted in 3.6 ×10^6 single-shot spectra. The laser energies applied to each position amounted to ~360 nJ/pulse (measured at the sample surface) using UV-258 nm laser. This energy was appropriate to produce the optimal quality for the mass-spectrometric signal, both from the microfossils and the quartz-filled host area. Analytical conditions were held constant during the data collection. The diameter of the average ablation crater was measured to be ~5 μm, and gaps between ablation craters were set to 10 μm (*see* **Figure 2C**). A custom-built software package was used to control the translation stage and the laser firing intervals. A fast data acquisition system from Keysight was used for digitizing current from the anodes, providing a 3.2 GSa/s sampling rate. An example of a single composite spectrum (representing a histogram of 200 individual single-shot spectra) registered from the Gunflint sample is shown in **Figure 3**. A single mass spectrum consists of 64,000 individual datapoints sampled with a digitizer, where each digitized data point corresponds to ~0.33 ns of a flight time. Thus, every recorded spectrum contains information about ~20 μs of a flight time, which provides a mass/charge (m/z) coverage of up to 800 m/z, providing a complete record of all stable isotopes and simple molecular compounds. Overall, 18,000 composite spectra were collected from the Gunflint sample, with a 10 μm gap between ablation craters. Additionally to the mass spectra collection, noise measurements were recorded, which allowed the enhancement of the recorded signal.

Figure 2. Microscope images of the Gunflint chert before and after the fs-LIMS imaging campaign are shown. A) Microscope image of the area (0.9*2 mm^2) chosen for the chemical imaging with our fs-LIMS system. A. The dark brown patches distributed through the sample and forming a diffuse layer in the middle of the picture represent a bio-lamination surface. B) Close-up microscope picture of individual microfossils from the bio-lamination surface. Filamentous (Gunflintia), star-shaped (Eoastrion), and spherular microfossils (Huroniospora) can be seen. C) Microscope picture of laser ablation craters (0.9*2 mm^2 area covered with 90*200 positions – 18,000 ablation sites) formed after the fs-LIMS imaging campaign. Red lines denote the accuracy of sample positioning (the gap between ablation craters is consistently 10 μm) and identify the ablation crater diameters. Individual craters range in diameter from 4 to 5 μm. Note, on the upper part of the image, the yellow arrow indicates an individual microfossil body. The size of the microfossil can be compared with the diameter of the analytical spot.

Data Preprocessing

The entire imaging dataset, which consists of ~50 GB of recorded composite mass spectra, was preprocessed before any analysis was applied to the data. A mass spectrometry preprocessing routine applied to the dataset consists of several typical steps that largely follow methods described in (Gil and Marco 2007) and (Meyer et al., 2017). The fs-LIMS preprocessing routine applied to the imaging data consisted of:

1) Noise removal for an improvement of the signal-to-noise ratio (SNR) of the signal. The noise signal (empty composite mass spectra) was recorded after the imaging campaign was completed. The recorded noise waveform was subtracted from the imaging observations.
2) Baseline subtraction. A filter function was applied to the noise-removed mass spectra to estimate varying baseline within multiple windows and regressed using spline approximation.
3) Jitter correction. Since materials within the analyzed sample might be of better or worse ionization efficiency (mainly due to topography), temporal variation of ion yields is expected to

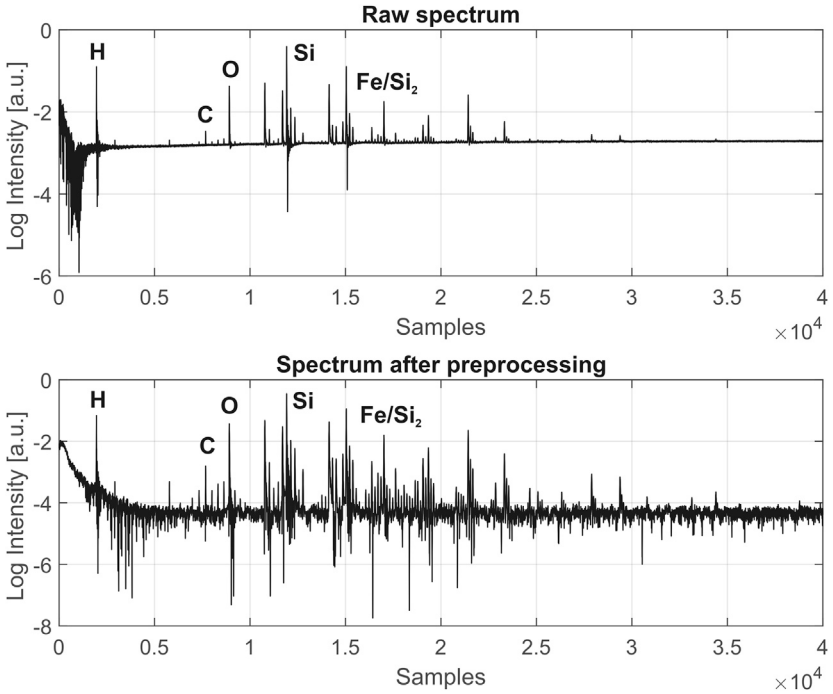


FIGURE 3 | Comparison of fs-LIMS spectra (composite spectrum of 200 laser shots, recorded from single position), before and after data preprocessing, acquired from the Gunflint chert sample. Each line in the spectrum represents a single unit mass. The increase of SNR to 10^4 and correction of the baseline can be noted. *See* the text for the full description of preprocessing procedures. Exemplary atomic lines are denoted on top of the spectrum.

occur so that times-of-flight of given ion packets might slightly vary. Typically, this effect is small and affects the peak shapes in a minor way. However, since we collected a relatively large dataset, a correction procedure has been applied. To correct for mismatch of times-of-flight, we have used an autocorrelation function described in (Gil and Marco 2007).

4) Low pass filtering. The low pass filter with normalized cutoff frequency at 0.13 πrad/sample and stopband attenuation of 60 dB was applied to each composite mass spectrum. This step removes the remaining high-frequency component from the recorded signal. Typically, it improves the SNR by two to five and does not alter the peak shapes.
5) Parametric peak preserving smoothing. The Savitzky–Golay filter function (Press and Teukolsky 1990) was applied to flatten the baseline further and increase the SNR.
6) Mass scale assignment. An average time-of-flight spectrum of all 18,000 spectra was recalculated for mass calibration purposes. A simple quadratic equation was used to calibrate the mass scale with the time-of-flight spectrum (Riedo et al., 2013a).
7) Single mass unit decomposition. An integration of consecutive 260 single unit masses, starting from ^{1}H, was achieved by recalculating the time-of-flight windows from the mass calibration equation and utilizing direct Simpson's integration (Meyer et al., 2017).

Figure 3 shows a typical raw spectrum (top panel) acquired from the Gunflint sample before any data preprocessing has been applied. The bottom panel shows a spectrum after preprocessing and reveals significantly improved SNR (10^4) and a flat baseline. After step number seven, multiple isotope maps were calculated using Kriging interpolation (further information in the text and *see* **Figure 4**) for an investigation of the distribution of major abundant elements. The imaging dataset was z-score normalized to remove the imbalanced scales. An assessment of the pairwise correlation factors was made, showing that approximately half of the dimensions (single unit masses) are empty or very weakly expressed.

The principal component analysis (PCA) reduction down to the first 60 principal components was applied to remove empty dimensions dominated by noise from the original dataset. The Uniform Manifold Approximation and Projection (UMAP) algorithm (McInnes et al., 2018) was used to further characterize non-linear dependencies present in the PCA reduced data matrix. The overall classification of the UMAP scores was made using a hierarchical density-based clustering algorithm (HDBSCAN) (Campello et al., 2013; McInnes et al., 2017). The specific spectra identified from the microfossils were further visualized using the Mapper algorithm (Singh et al., 2007). The identification of the modules present in the Mapper network was conducted using a greedy modularity optimization algorithm (Louvain) (Blondel et al., 2008).

Figure 3. Comparison of fs-LIMS spectra (composite spectrum - 200 laser shots, recorded from single pixel), before and after data preprocessing, acquired from the Gunflint chert sample. Each line in the spectrum represents a single unit mass.

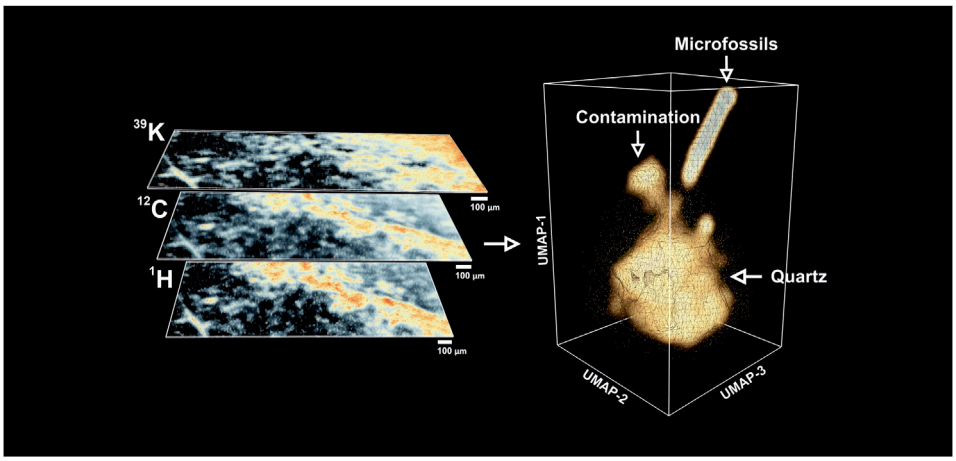


FIGURE 4 | Left panel–Exemplary isotope intensity maps (warmer colors indicate high concentrations) retrieved from the fs-LIMS mass spectra. The bio-lamination surface (aggregation of microfossils) could be identified from ^{12}C and ^{1}H maps (bright yellow to red areas), distribution of ^{39}K indicates the presence of the surface contamination (upper left corner, bright yellow to the red area). Dark areas on the isotope maps indicate low-intensity regions and correspond to the quartz matrix. To compare with an optical image, *see* **Figure 2A**. Right panel–Low dimensional structure of the imaging data cube revealed by UMAP. Triangulated mesh represents volumetric isodensity surface of UMAP scores calculated from the 18,000 fs-LIMS mass spectra. Three separate entities could be observed from the spectral neighborhood, namely quartz, contamination, and microfossils. The point cloud data plotted along with the density surface.

The increase of SNR to 10^4 and correction of the baseline can be noted. *See* the text for the full description of preprocessing procedures. Exemplary atomic lines are denoted on top of the spectrum.

RESULTS

We calculated the intensity maps of major (abundant) isotopes to understand a basic representation of the data. In **Figure 4**, the spatial distributions of ^{12}C, ^{1}H, and ^{39}K are illustrated and the chemical maps reveal specific areas where isotopes show elevated intensities. In comparison with the optical image of the same area (*see* **Figure 2A**), one can see that most of the dark brown patches identified from optical microscopy as microfossils preserved in the bio-lamination surface are spatially correlated with increased values of ^{12}C and ^{1}H. This observation is consistent with the fact that major elements within microfossil bodies are C and H. However, the intensity map of ^{39}K reveals different distribution. A top-right corner of the sample, which was previously identified as a clean matrix (milky quartz), reveals elevated concentrations of ^{39}K and relatively intense ion yields of ^{12}C. In fact, after a closer investigation of the mass spectra recorded from that region, we identified a full range of biorelevant elements (CHNOPS).

Additionally, a full range of Si isotopes, various silicon oxides, and small chains of hydrocarbon clusters were observed from that region. Considering that a particular location from optical microscopy does not show any distinct mineralogical association with described elements, we concluded that the identified area could belong to the organic contamination. From our previous studies of the Gunflint sample (Wiesendanger et al., 2018), particularly the chemical depth profiling of the neighboring region, it was identified that organic contamination is present as a thin surface layer and organic spectral features quickly decay with increasing depth. The organic contamination potentially comes from the sample handling and preparation procedures and likely represents a small layer of lipids finely distributed on the surface.

In general, the manually inspected mass spectra from various regions appeared to be somewhat similar. They contain the same elements with varying concentrations–Si, CHNOPS, and polyatomic molecules of similar composition. This observation makes it difficult to manually define compounds observed from the Gunflint sample since they seem to represent continually mixing variants. The borders between chemical classes are fused into each other. Thus, the deterministic classification solely based on isotope intensity maps cannot be made. However, we can further explore the chemical variations within different parts of the sample. For example, the spectral features from the top-right corner also show very close proximity to the chemical composition of the host mineral - Si, O, and various Si oxide chains indicate that ablation craters were deep enough to pass through the layer of organic contamination and probe the chemical composition of the original underlying mineral. Lower parts of the isotope maps, shown with black regions (**Figure 4**-left) after a closer investigation of the mass spectra, were proposed to be from quartz, showing previously described simple chemistry–Si, O, and minor amounts of Na, K, Al. The latter elements (Na, K, Al) could be found as impurities within the chert since they are relatively common in the seawater and could have precipitated together with Si during the time of the rock formation, or they could be from phyllosilicates (clay minerals) that can occasionally occur in the matrix of Gunflint Formation stromatolites, e.g., (Lepot et al., 2017). Since the ^{12}C and ^{1}H maps outline the structure of the bio-lamination surface, previously

identified from the optical microscopy, we can investigate the spectra from the lamination site. The spectra from that area can be characterized by the presence of the bio-relevant elements–with increased concentrations of CHNOPS and an additional minor contribution from Fe, Mn, and Cr. Another notable observation is that spectra from the lamination surface reveal relatively strong polyatomic molecules formation patterns. Various hydrocarbon molecules accompanied by Si oxide chains populate the mass spectra up to 200 m/z.

A dimensionality reduction algorithm was applied over the full mass range of the fs-LIMS imaging data (1–260 amu) to find similar spectra in the dataset. We used the UMAP algorithm (McInnes et al., 2018) to analyze our observations. The first six UMAP components were retrieved from the dataset precompressed with PCA. The UMAP scores were calculated using the Euclidean distance as a metric; every 15 nearest neighbors were used in the construction of the k-nearest neighborhood graph, with a small minimal distance (0.1), and iterated over 400 epochs. This particular set of hyperparameters were found to be appropriate for an approximation of the global structure of the manifold. In the right panel of **Figure 4**, a distribution of the first three UMAP components is shown. The spectral neighborhood appears to be relatively busy (see point cloud data). However, three main protrusions can be observed from the equal density surface of the UMAP scores. The composition of protruding clusters matches our previous interpretation of the data. The lower part of the plot represents a relatively large cluster of mass spectra acquired from the Quartz-filled matrix. A smaller cluster observed in the vicinity of the main body corresponds to the spectra measured from the area with signatures of organic contamination. It is noteworthy that the contamination cluster is more connected to the main quartz cluster and that the structure of the density surface indicates a smooth transition from pure quartz to the spectra from the surface contamination. The transition structure forms a narrow neck where the similarity of spectra gradually changes from one class to another. From the point cloud data, we could *see* that the contamination cluster is relatively fuzzy, and the fine structure of the transition could be observed on the isodensity surface.

Through the same transition pathway, a cluster of spectra that corresponds to the microfossils preserved within the bio-lamination layer could be observed. In comparison to the cluster of spectra with organic contamination, the density surface of the microfossils cluster forms a separate transition line. The cluster of microfossils forms a smooth identifiable shape, which gradually rises further apart from the quartz and contamination clusters. As one can see, the relative proximity of the spectra located closer to the transition "neck" indicates the ablation of small parts of microfossils. From the investigation of the individual spectra (*see* **Figure 3**), we have noted that almost all spectra from microfossils contain spectral features from the filling quartz mineral. This observation could be explained by the fact that bodies of microfossils represent partially collapsed and degraded cell walls. The thicknesses of the partially decayed cell walls vary from the first tens of nm to the first hundreds of nm, and these walls are all entombed in the silica matrix. By ablation of small portions of the microfossils and larger portions of the silica matrix, we can explain the smooth transition structure, where similarity of spectra transitions from the clean silica matrix. Thus, the end members of the microfossil cluster represent the best volumetric sampling of microfossils, as well as the best chemical composition of the fossils.

Overall, the volumetric density estimate of the UMAP scores provides a good overview of the spectral types and their transition structures. Also, it is possible to identify outliers (e.g., microscopic inclusions of other minerals) from this graph, for example, by recalculating the isolation forest scores (or any other outlier detection algorithm) – the data points that are weakly connected to the main clusters will have high values, thus, easily identifiable. In the fs-LIMS analysis, where fine chemistry is often of great interest, such information might be valuable because it allows the identification of detached spectra from the bulk of very similar ones.

Figure 4. Left panel–Exemplary isotope intensity maps (warmer colors indicate high concentrations) retrieved from the fs-LIMS mass spectra. The bio-lamination surface (aggregation of microfossils) could be identified from ^{12}C and ^{1}H maps (bright yellow to red areas), distribution of ^{39}K indicates the presence of the surface contamination (upper left corner, bright yellow to the red area). Dark areas on the isotope maps indicate low-intensity regions and correspond to the quartz matrix. To compare with an optical image, *see* **Figure 2A**. Right panel–Low dimensional structure of the imaging data cube revealed by UMAP. Triangulated mesh represents volumetric isodensity surface of UMAP scores calculated from the 18,000 fs-LIMS mass spectra. Three separate entities could be observed from the spectral neighborhood, namely quartz, contamination, and microfossils. The point cloud data plotted along with the density surface.

The UMAP isodensity estimate reveals the continuous structure of spectral similarities, and therefore it is not clear where to define a boundary between different classes. A density-based clustering approach was used to define discreet classes from the low dimensional UMAP scores. The six UMAP components were used to discretize distributions using a Hierarchical Density-Based Spatial Clustering (HDBSCAN) algorithm (Campello et al., 2013; McInnes et al., 2017). An HDBSCAN provides relatively conservative class assignments compared to other clustering algorithms and potentially more accurate in its predictions. An advantageous side of HDBSCAN over DBSCAN, for example, is that it can find clusters with varying densities, which is precisely the case with our data, where we have an oversampled data from the silicified matrix and a relatively small number of spectra from the microfossils. Moreover, it is possible to calculate the confidence probabilities of the assignment of each spectrum to the cluster, which makes troubleshooting of clustering results more intuitive and less bothersome. However, the downside of the conservative clustering is that some portions of the data might be classified as noise if they do not tightly belong to the densely packed cluster. In contrast to the previous interpretation of UMAP scores, the clustering algorithm found two microfossil populations, a cluster of surface contamination, and quartz from the matrix. The additional cluster of microfossils was hidden on the backside of the quartz-related spectra (*see* **Figure 4**). The Mapper networks were applied to the spectra registered from the

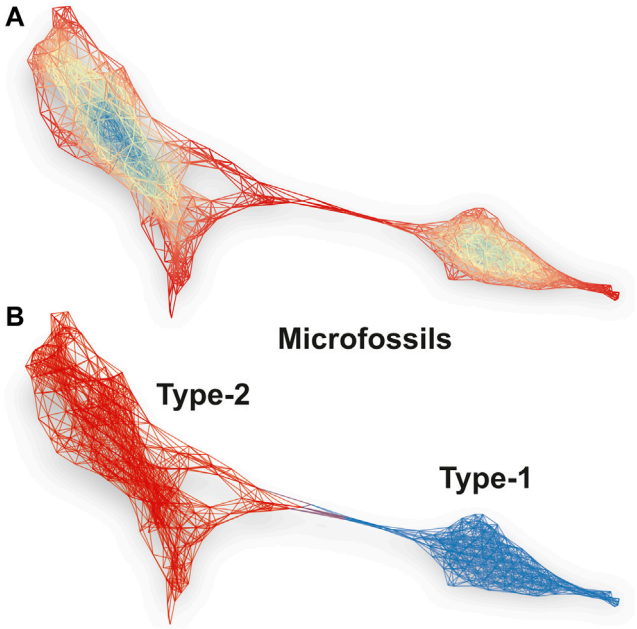


FIGURE 5 | (A) Spectral similarity network constructed from 1964 LIMS imaging spectra registered from the microfossils. Each node represents a single or a group of spectra with a significant similarity of intensity profiles. The edges connected with nodes indicate that nodes have one or more shared spectra. The network is colored according to the eigenvector centrality of nodes. A density-based clustering and first three UMAP components were used as a lens to project the data using the Mapper algorithm. The proximity of nodes in the network identifies groups of microfossils and transition structure between two classes. **(B)** The Louvain clustering of the spectral similarity network. The blue part of the network identifies type-1 microfossils, and the red part of the network illustrates spectra registered from the type-2 microfossils. *See* the text for more details.

microfossils to visualize the proximity structure between these two classes.

Figure 5A shows a spectral similarity network constructed from the fs-LIMS spectra registered from the microfossils, using the first three UMAP components as a lens. A python implementation - Kepler Mapper (Van Veen et al., 2019) of the Mapper algorithm (Singh et al., 2007) was used to calculate the similarity network of LIMS spectra. However, other open-source implementations exist - e.g., recently published Giotto-TDA (Tauzin et al., 2020). The density-based clustering was applied to identify clusters within overlapping filter function windows. In total, twenty windows were applied to construct the network with 40% overlap over three UMAP components, forming 8,000 sampling windows and resulting in a complex network with 417 nodes and 2,967 edges (from 1,964 composite spectra registered from the microfossils). Note that the number of filter dimensions is user-defined, and in principle, they might be defined as an n-dimensional hypercube, though two-dimensional filters provide the best interpretability. The nodes present in the network indicate groups of fs-LIMS spectra with a high degree of similarity. The nodes might contain one or hundreds of spectra, depending on the size of the filter function window. The edge between nodes is drawn if nodes share the same observations (it might be one or many more spectra). The coloring of the network is conducted according to the eigenvector centralities of the nodes. Blue parts of the network indicate the central nodes, and red parts indicate less connected network components.

The structure of the network identifies the presence of two connected communities. **Figure 5B** shows the same spectral similarity network as in **Figure 5A** but colored according to the Louvain modularity, calculated from the network topology. The red part of the network (nodes are not shown) categorizes the spectra identified from the type-2 microfossils, and the blue network indicates the type-1 microfossils. The type-2 microfossils can be characterized by increased proximity to the cluster of spectra registered from the quartz. In contrast to the spectra from type-2, type-1 microfossils are almost completely detached from other groups and form a community of highly connected nodes and correspond to the spectra in a linear protrusion in **Figure 4** (right panel). Note that the HDBSCAN and Louvain clustering provides mutually supportive clustering results, although the Mapper networks provide better tolerance to noise, thus allowing for improved clustering performance. In order to check that cluster assignments are not artifactual, we performed a clustering robustness analysis. The Rand Index (RI) metric was used to assess the clustering similarity between 10 random subsamples of the data registered from microfossils. In total, 75% of the data was used to generate random subsamples. The output UMAP subsamples were clustered using the Louvain community detection algorithm. The RI similarity matrix for Louvain clustering of random samples could be found in the supplementary information (*see* **Supplementary Table S1** and **Supplementary Figure S3**). Overall, 45 different clustering pairs revealed an average RI score of 92.5% with a standard deviation of 2%, which indicates that communities shown in **Figure 5** are not artefactual and that the cluster assignments are robust. Most of the clustering uncertainty can be attributed to the transition zone between two types of microfossils. The type-2 microfossils reveal more inhomogeneity (*see* **Supplementary Figures S1, S2**) in comparison to the type-1 microfossils and represent more intermixed with the host mineral material.

The spectral similarity network calculated from the first three UMAP components reveals a better visualization of internal structure and detects outliers and irregularities. Moreover, the force-directed layout (ForceAtlas2 (Jacomy et al., 2014)), applied to the network, exaggerates the positioning of weakly connected nodes, which makes them easier to detect. Moreover, interpretation of the low-dimensional embedding of fs-LIMS data can be easily achieved by coloring the network with original isotope intensities and synthetic features such as isotope ratios. Any other functions might be applied to the data (e.g., Kernel Density Estimate (KDE), Singular Value Decomposition (SVD), and Principal Component Analysis (PCA)), which makes Mapper networks a versatile and powerful tool for insight extraction and hypothesis generation. Furthermore, by reducing the large fs-LIMS intensity space down to a network, we can additionally define a multitude of secondary statistics that could be calculated from the graph topology. Metrics such as centrality, modularity (e.g., *see* **Figures 5A,B**), average degree, path length (e.g., between the host mineral and microfossils), and many more, can be applied to the specific

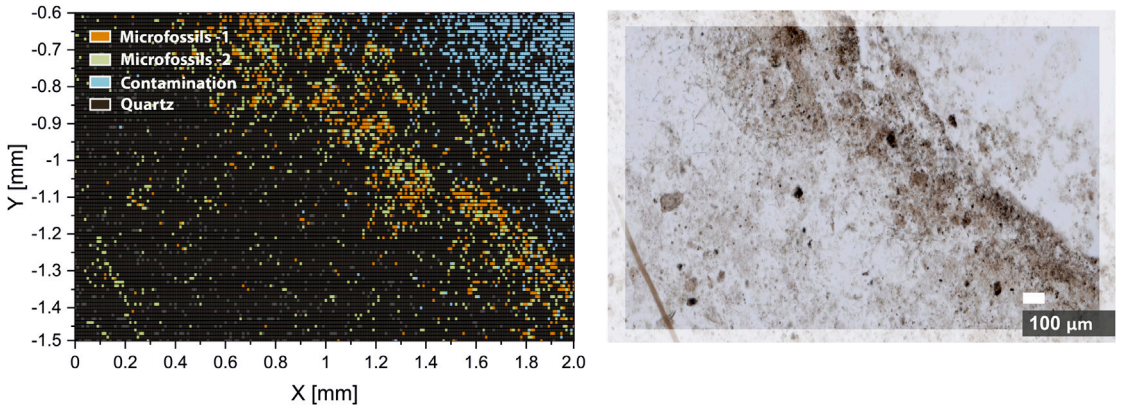


FIGURE 6 | Hierarchical density-based spatial clustering (HDBSCAN) of six UMAP components of the imaging dataset (left panel). The orange pixels represent spectra registered from the type-1 microfossils. The green pixels represent spectra registered from the type-2 microfossils. The blue pixels represent spectra registered from the surface contamination. Black and grey pixels–spectra registered from the quartz matrix of the Gunflint chert. Right panel–the optical microscopy image of the analyzed area. Note the aligned distribution of classified spectra with the bio-lamination surface crossing the image.

minerals and microfossils to define the multiparametric space further and enhance the potential for definitive identification.

Figure 5A. Spectral similarity network constructed from 1,964 LIMS imaging spectra registered from the microfossils. Each node represents a single or a group of spectra with a significant similarity of intensity profiles. The edges connected with nodes indicate that nodes have one or more shared spectra. The network is colored according to the eigenvector centrality of nodes. A density-based clustering and first three UMAP components were used as a lens to project the data using the Mapper algorithm. The proximity of nodes in the network identifies groups of microfossils and transition structure between two classes. **Figure 5B**. The Louvain clustering of the spectral similarity network. The blue part of the network identifies type-1 microfossils, and the red part of the network illustrates spectra registered from the type-2 microfossils. See the text for more details.

The overall results of the density-based clustering can be seen in **Figure 6**. Clustering results reveal a very close match with results of optical microscopy (*see* **Figure 6**, right panel) and conclusions from previous single isotope maps investigations. Moreover, we have identified two types of microfossils and a contamination zone, which were not acknowledged from the microscope image. The type–1 microfossils represent spectra obtained from the microfossils with the best microfossil over host (matrix mineral) sampling ratio. Thus, spectra from type-1 can be counted as the most representative of microfossils. On the other hand, type-2 represents the microfossils with an increased contribution from the host mineral, which is also shown in **Figure 6**. The chemical composition of type-1 microfossils can be characterized with increased content of carbon and oxygen (^{12}C, $^{12}C^{2+}$, and $^{16}O^{2+}$ peaks in the mass spectra), whereas type-2 microfossils contain less ^{12}C and more hydrocarbons, which indicates lower volumetric ablation and colder plasma temperatures, thus, more prevalent recombination processes. Higher plasma temperatures observed in the type-1 microfossils can be attributed to the higher volumetric contribution from absorptive kerogen. This observation also finds confirmation from the spatial distribution of microfossils. In **Figure 6**, the first type is mainly distributed in the densely populated area (*see* **Figure 6**, right panel), in contrast to type-2, which is largely distributed outside of the dense zone, and more likely to be sampled with larger portions of the host mineral. The identification of microfossils from the host mineral using fs-LIMS and low dimensional analysis provides topological biosignatures. As it was shown in **Figures 4**, **5**, the structure of spectral similarities identifies the positionings of spectra from different classes and provides means for identification, classification of large datasets, and has a potential for the prediction of spectral classes from previously unseen spectra, given that a sufficiently rich spectral library is provided.

Figure 6. Hierarchical density-based spatial clustering (HDBSCAN) of six UMAP components of the imaging dataset (left panel). The orange pixels represent spectra registered from the type-1 microfossils. The green pixels represent spectra registered from the type-2 microfossils. The blue pixels represent spectra registered from the surface contamination. Black and grey pixels–spectra registered from the quartz matrix of the Gunflint chert. Right panel–the optical microscopy image of the analyzed area. Note the aligned distribution of classified spectra with the bio-lamination surface crossing the image.

DISCUSSION

The identification and chemical characterization of minerals and prospective biosignatures from large spectral databases generated using fs-LIMS as well as other *in-situ* spectroscopic techniques is a longstanding problem that can be generalized to other analytical methods as well. For example, other important methods proposed for *in-situ* space exploration, such as Laser-Induced Breakdown Spectroscopy (LIBS) (e.g., ChemCam, currently operates on Mars as part of the Mars Science Laboratory), Raman spectroscopy (i.e., Raman Laser Spectrometer (RLS), one of the Pasteur Payload instruments from ExoMars), and a large variety of other techniques rely on harvesting large spectral information

from the analyte material. This spectral information is often hard to interpret due to the large dimensionality, complexity, and size of generated datasets. Outside of the context of space exploration, in the field of analytical chemistry, similar data analytical challenges are often encountered in the laboratory. For example, Secondary Ions Mass Spectrometry (SIMS) or Liquid Chromatography Mass Spectrometry (LC-MS), as high-throughput techniques, provide hundreds of mass lines per spectrum, and the output spectral dataset is not always easy to interpret. As was shown in this contribution, analysis of fs-LIMS data using topological methods reveals a fast and accurate description of spectral classes and provides a good understanding of transitional structures. In the low dimensional domain, it might be easier to generate insights and formulate a hypothesis, thus accelerating the extraction of knowledge from the given sample.

The analysis of data generated by using our fs-LIMS system might also be of use for future investigations of Precambrian rocks containing signatures of putative microfossils. The Gunflint sample is rare amongst Precambrian rocks as it exhibits an exceptional level of morphological and chemical preservation, so there is little argument over the biogenicity of the encased organic material (Barghoorn and Tyler 1965; Lepot et al., 2017; Wacey et al., 2012). However, traces of early life can be destroyed or heavily altered by heat, pressure, and time (diagenetic alteration and later metamorphism). As was briefly discussed before, the full mass range spectral proximity analysis provides a means for the classification of chemically similar entities. For example, organic contamination and microfossils - similar compounds (both contain CHNOPS and Si mass peaks), can be distinguished using topological methods (*see* **Figures 4–6**). A big challenge in the field of Precambrian micropaleontology surrounds the fact that altered and reduced carbon found in ancient rocks could potentially be of biological origin but could also have been created by abiotic processes. For example, Fischer-Tropsch type synthesis might be responsible for the presence of some abiotic hydrocarbons in Precambrian formations (Brasier et al., 2002). However, we speculate that synthetic products of Fischer-Tropsch-like reactions will have a distinct spectral profile (e.g., polyatomic plasma chemistry products might be different), and therefore corresponding topological positioning is expected to be distinguishable from *bona fide* microfossils. Thus, there is a hope that signs of life in controversial samples might be successfully identified using sensitive methods and full-feature-based topological representations.

The current state of space exploration also faces similar challenges in the field of *in-situ* chemical analysis of solids on planetary bodies. For example, the ns-LIMS instrument proposed for Europa (Ligterink et al., 2020) reported the identification of extremely low quantities of biological and abiotic amino acids from well-defined extracts at the fmole level. However, more complex molecules (e.g., proteins, polysaccharides, etc.) combined with various undefined matrices will likely form complex fragmentation patterns with hundreds of significant mass lines, thus, making the identification challenging. The topological representation, in this case, might provide a number of compounds present in the measured mixture and their similarity to the predefined classes.

The unsupervised identification of minerals from fs-LIMS chemical imaging datasets might also be of use in the determination of relative sensitivity coefficients (RSC's). The fs-LIMS is a quantitative method; however, it requires the establishment of RSC's, which are matrix dependent. With an introduction of fs-LIMS, some matrices have been reduced to unity (RSC = 1, no correction needed). However, non-absorptive samples such as glasses typically still require the determination of RSC's for quantitative measurements. In the case of exploratory analysis, where we do not know the sample (i.e., field exploration of Martian samples), if one would know the stoichiometry of the investigated mineral, it is possible to recalculate correction factors for major elements, and then through RSC's dependence on atomic orbital ionization energy recalculate concentrations of minor and trace elements (Tulej et al., 2021). The key component here is the identification of the mineral, and as we described above, topological methods provide a means to do that.

Here we also need to point out several caveats regarding the analysis of fs-LIMS data. First, at multiple stages, the data analysis procedures require a set of hyperparameters to be chosen. For example, in UMAP embedding and Mapper network construction, we used Euclidean and cosine distances, respectively, and defined the number of neighbors, number of clusters, and filter functions. However, a more rigorous study of the effect of hyperparameters needs to be assessed in future studies regarding the analysis of fs-LIMS data or data generated by other spectroscopic techniques. Nevertheless, a recent contribution by (Belchı et al., 2020) provides an insight into the numerical stability of Mapper-type algorithms. It was shown that reliable Mapper output could be identified as a local minimum of instability, regarded as a function of Mapper input parameters. Other statistical solutions were proposed to circumvent testing large parametric spaces and keep the most representative Mapper settings (Carriere et al., 2018). Furthermore, we have used UMAP scores as a lens in the construction of the similarity network; however, a large variety of other functions might be used, and their impact on visualizations needs to be assessed. It would also be valuable to implement into the analysis pipeline some domain-specific lenses for technical usage (i.e., mass resolution, mass accuracy, etc.), which will improve the extraction of quality metrics.

Overall, in addition to the account of topological descriptors of early life, we hope that our analysis will facilitate, in time, a predictive approach in the field of study of early life. The approach described here might be expanded to more powerful, state-of-the-art standalone laboratory instrumentation (e.g., high-resolution LIMS, SIMS, LA-ICP-MS), where data quality might provide a whole new quantification perspective.

CONCLUSION

Our contribution offers several important conclusions for *in-situ* space research. First, the miniature fs-LIMS system combined with topology-based data analysis demonstrates the utility and sensitivity to distinguish organically preserved microfossils from organic contamination and inorganic host mineralogy. Second,

the proposed approach might be extended to other complex samples with multimineral compositions and used with other high-resolution spectrometric or spectroscopic methods. Third, our approach - full spectral mass range convolution down to a similarity network for life detection stands out from multielement methods. It offers great flexibility and could be further expanded to study the chemical discrepancies between individual populations of microfossils. Furthermore, our analysis reveals fine transition structures between classes and the detection of outliers. Last, the fs-LIMS system, in combination with topological methods, enables faster data analysis, accelerates the formulation of hypotheses, and the generation of insights for mineralogical compositions of investigated samples.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the **Supplementary Materials**, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

RL performed the experiments and data analysis. RL wrote the main manuscript. All authors reviewed and revised the manuscript.

FUNDING

DW acknowledges support from the Australian Research Council *via* the Future Fellowship scheme (FT140100321). RL acknowledges support from Swiss National Science Foundation, Grant No. 200020-184657.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frai.2021.668163/full#supplementary-material

REFERENCES

Arevalo, R., Jr, Ni, Z., and Danell, R. M. (2020). Mass Spectrometry and Planetary Exploration: A Brief Review and Future Projection. *J. Mass. Spectrom.* 55, e4454. doi:10.1002/jms.4454

Arevalo, R., Jr, Selliez, L., Briois, C., Carrasco, N., Thirkell, L., Cherville, B., et al. (2018). An Orbitrap-Based Laser Desorption/Ablation Mass Spectrometer Designed for Spaceflight. *Rapid Commun. Mass. Spectrom.* 32, 1875–1886. doi:10.1002/rcm.8244

Azov, V. A., Mueller, L., and Makarov, A. A. (2020). Laser Ionization Mass Spectrometry at 55: Quo Vadis? *Mass Spectrom. Rev.* 1–52. doi:10.1002/mas.21669

Barghoorn, E. S., and Tyler, S. A. (1965). Microorganisms from the Gunflint Chert: These Structurally Preserved Precambrian Fossils from Ontario Are the Most Ancient Organisms Known. *Science* 147, 563–575. doi:10.1126/science.147.3658.563

Belchı, F., Brodzki, J., Burfitt, M., and Niranjan, M. (2020). A Numerical Measure of the Instability of Mapper-Type Algorithms. *J. Machine Learn. Res.* 21, 1–45.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast Unfolding of Communities in Large Networks. *J. Stat. Mech.* 2008, P10008. doi:10.1088/1742-5468/2008/10/p10008

Brasier, M. D., Green, O. R., Jephcoat, A. P., Kleppe, A. K., Van Kranendonk, M. J., Lindsay, J. F., et al. (2002). Questioning the Evidence for Earth's Oldest Fossils. *Nature* 416, 76–81. doi:10.1038/416076a

Brasier, M. D., and Wacey, D. (2012). Fossils and Astrobiology: New Protocols for Cell Evolution in Deep Time. *Int. J. Astrobiology* 11, 217–228. doi:10.1017/s1473550412000298

Campello, R. J., Moulavi, D., and Sander, J. (2013). "Density-Based Clustering Based on Hierarchical Density Estimates," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (Heidelberg: Springer). doi:10.1007/978-3-642-37456-2_14

Carriere, M., Michel, B., and Oudot, S. (2018). Statistical Analysis and Parameter Selection for Mapper. *J. Machine Learn. Res.* 19, 478–516.

Chan, M. A., Hinman, N. W., Potter-McIntyre, S. L., Schubert, K. E., Gillams, R. J., Awramik, S. M., et al. (2019). Deciphering Biosignatures in Planetary Contexts. *Astrobiology* 19, 1075–1102. doi:10.1089/ast.2018.1903

Garcia-Lopez, E., and Cid, C. (2017). Glaciers and Ice Sheets as Analog Environments of Potentially Habitable Icy Worlds. *Front. Microbiol.* 8, 1407. doi:10.3389/fmicb.2017.01407

Gil, A., and Marco, R. (2007). *Systems Bioinformatics: An Engineering Case-Based Approach*. Artech.

Goesmann, F., Brinckerhoff, W. B., Raulin, F., Goetz, W., Danell, R. M., Getty, S. A., et al. (2017). The Mars Organic Molecule Analyzer (MOMA) Instrument: Characterization of Organic Material in Martian Sediments. *Astrobiology* 17, 655–685. doi:10.1089/ast.2016.1551

Grimaudo, V., Tulej, M., Riedo, A., Lukmanov, R., Ligterink, N. F. W., de Koning, C., et al. (2020). UV Post-Ionization Laser Ablation Ionization Mass Spectrometry for Improved Nm-Depth Profiling Resolution on Cr/Ni Reference Standard. *Rapid Commun. Mass. Spectrom.* 34, e8803. doi:10.1002/rcm.8803

Grimaudo, V., Moreno-García, P., Riedo, A., López, A. C., Tulej, M., Wiesendanger, R., et al. (2019). Review-Laser Ablation Ionization Mass Spectrometry (LIMS) for Analysis of Electrodeposited Cu Interconnects. *J. Electrochem. Soc.* 166, D3190–D3199. doi:10.1149/2.0221901jes

Hand, K., Murray, A., Garvin, J., Brinckerhoff, W., Christner, B., Edgett, K., et al. (2017). *Europa Lander Study 2016 Report: Europa Lander Mission*. La Cañada Flintridge, CA: NASA Jet Propuls. Lab.

Hofmann, B. A. (2008). Morphological Biosignatures from Subsurface Environments: Recognition on Planetary Missions. *Space Sci. Rev.* 135, 245–254. doi:10.1007/s11214-007-9147-9

Jacomy, M., Venturini, T., Heymann, S., and Bastian, M. (2014). ForceAtlas2, A Continuous Graph Layout Algorithm for Handy Network Visualization Designed for the Gephi Software. *PloS one* 9, e98679. doi:10.1371/journal.pone.0098679

Lepot, K., Addad, A., Knoll, A. H., Wang, J., Troadec, D., Béché, A., et al. (2017). Iron Minerals within Specific Microfossil Morphospecies of the 1.88 Ga Gunflint Formation. *Nat. Commun.* 8, 14890. doi:10.1038/ncomms14890

Li, X., Danell, R. M., Pinnick, V. T., Grubisic, A., van Amerom, F., Arevalo, R. D., et al. (2017). Mars Organic Molecule Analyzer (MOMA) Laser Desorption/Ionization Source Design and Performance Characterization. *Int. J. Mass Spectrom.* 422, 177–187. doi:10.1016/j.ijms.2017.03.010

Ligterink, N. F. W., Grimaudo, V., Moreno-García, P., Lukmanov, R., Tulej, M., Leya, I., et al. (2020). ORIGIN: A Novel and Compact Laser Desorption - Mass Spectrometry System for Sensitive In Situ Detection of Amino Acids on Extraterrestrial Surfaces. *Sci. Rep.* 10, 9641. doi:10.1038/s41598-020-66240-1

Managadze, G. G., Wurz, P., Sagdeev, R. Z., Chumikov, A. E., Tuley, M., Yakovleva, M., et al. (2010). Study of the Main Geochemical Characteristics of Phobos' Regolith Using Laser Time-Of-Flight Mass Spectrometry. *Sol. Syst. Res.* 44, 376–384. doi:10.1134/s0038094610050047

McInnes, L., Healy, J., and Astels, S. (2017). Hdbscan: Hierarchical Density Based Clustering. *J. Open Source Softw.* 2, 205. doi:10.21105/joss.00205

McInnes, L., Healy, J., and Melville, J. (2018). Umap: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv [Preprint]. Available at: https://arxiv.org/abs/1802.03426 (Accessed September 18, 2020).

McMahon, S., Bosak, T., Grotzinger, J. P., Milliken, R. E., Summons, R. E., Daye, M., et al. (2018). A Field Guide to Finding Fossils on Mars. *J. Geophys. Res. Planets* 123, 1012–1040. doi:10.1029/2017je005478

Meyer, S., Riedo, A., Neuland, M. B., Tulej, M., and Wurz, P. (2017). Fully Automatic and Precise Data Analysis Developed for Time-Of-Flight Mass Spectrometry. *J. Mass. Spectrom.* 52, 580–590. doi:10.1002/jms.3964

Moreno-García, P., Grimaudo, V., Riedo, A., Tulej, M., Wurz, P., and Broekmann, P. (2016). Towards Matrix-Free Femtosecond-Laser Desorption Mass Spectrometry for In Situ Space Research. *Rapid Commun. Mass. Spectrom.* 30, 1031–1036. doi:10.1002/rcm.7533

Navarro-González, R., Navarro, K. F., Rosa, J. d. l., Iñiguez, E., Molina, P., Miranda, L. D., et al. (2006). The Limitations on Organic Detection in Mars-Like Soils by Thermal Volatilization-Gas Chromatography-MS and Their Implications for the Viking Results. *Proc. Natl. Acad. Sci.* 103, 16089–16094. doi:10.1073/pnas.0604210103

Neuland, M. B., Grimaudo, V., Mezger, K., Moreno-García, P., Riedo, A., Tulej, M., et al. (2016). Quantitative Measurement of the Chemical Composition of Geological Standards with a Miniature Laser Ablation/Ionization Mass Spectrometer Designed Forin Situapplication in Space Research. *Meas. Sci. Technol.* 27, 035904. doi:10.1088/0957-0233/27/3/035904

Neveu, M., Hays, L. E., Voytek, M. A., New, M. H., and Schulte, M. D. (2018). The Ladder of Life Detection. *Astrobiology* 18, 1375–1402. doi:10.1089/ast.2017.1773

Press, W. H., and Teukolsky, S. A. (1990). Savitzky-Golay Smoothing Filters. *Comput. Phys.* 4, 669–672. doi:10.1063/1.4822961

Priscu, J. C., and Hand, K. P. (2012). Microbial Habitability of Icy Worlds. *Microbe* 7, 167–172. doi:10.1128/microbe.7.167.1

Riedo, A., Bieler, A., Neuland, M., Tulej, M., and Wurz, P. (2013a). Performance Evaluation of a Miniature Laser Ablation Time-Of-Flight Mass Spectrometer Designed Forin Situinvestigations in Planetary Space Research. *J. Mass. Spectrom.* 48, 1–15. doi:10.1002/jms.3104

Riedo, A., Koning, C. d., Stevens, A., Cockell, C., McDonald, A., López, A. C., et al. (2020). The Detection of Elemental Signatures of Microbes in Martian Mudstone Analogs Using High Spatial Resolution Laser Ablation Ionization Mass Spectrometry. *Astrobiology* 20 (10), 1224–1235. doi:10.1089/ast.2019.2087

Riedo, A., Meyer, S., Heredia, B., Neuland, M. B., Bieler, A., Tulej, M., et al. (2013b). Highly Accurate Isotope Composition Measurements by a Miniature Laser Ablation Mass Spectrometer Designed for In Situ Investigations on Planetary Surfaces. *Planet. Space Sci.* 87, 1–13. doi:10.1016/j.pss.2013.09.007

Riedo, A., Neuland, M., Meyer, S., Tulej, M., and Wurz, P. (2013c). Coupling of LMS with a Fs-Laser Ablation Ion Source: Elemental and Isotope Composition Measurements. *J. Anal. Spectrom.* 28, 1256–1269. doi:10.1039/c3ja50117e

Riedo, A., Tulej, M., Rohner, U., and Wurz, P. (2017). High-Speed Microstrip Multi-Anode Multichannel Plate Detector System. *Rev. Scientific Instr.* 88, 045114. doi:10.1063/1.4981813

Singh, G., Mémoli, F., and Carlsson, G. E. (2007). "Topological Methods for the Analysis of High Dimensional Data Sets and 3d Object Recognition," in *Eurographics Symposium on Point-Based Graphics*. Editors M. Botsch, R. Pajarola, B. Chen, and M. Zwicker (The Eurographics Association), 91, 100. doi:10.2312/SPBG/SPBG07/091-100

Stevens, A. H., McDonald, A., de Koning, C., Riedo, A., Preston, L. J., Ehrenfreund, P., et al. (2019). Detectability of Biosignatures in a Low-Biomass Simulation of Martian Sediments. *Sci. Rep.* 9, 9706. doi:10.1038/s41598-019-46239-z

Tauzin, G., Lupo, U., Tunstall, L., Pérez, J. B., Caorsi, M., Medina-Mardones, A., et al. (2020). Giotto-Tda: A Topological Data Analysis Toolkit for Machine Learning and Data Exploration. arXiv [Preprint]. Available at: https://arxiv.org/abs/2004.02551 (Accessed March 5, 2021).

Tulej, M., Lukmanov, R., Grimaudo, V., Riedo, A., de Koning, C., Ligterink, N. F. W., et al. (2021). Determination of the Microscopic Mineralogy of Inclusion in an Amygdaloidal Pillow basalt by Fs-LIMS. *J. Anal. Spectrom.* 36, 80–91. doi:10.1039/d0ja00390e

Tulej, M., Neubeck, A., Ivarsson, M., Riedo, A., Neuland, M. B., Meyer, S., et al. (2015). Chemical Composition of Micrometer-Sized Filaments in an Aragonite Host by a Miniature Laser Ablation/Ionization Mass Spectrometer. *Astrobiology* 15, 669–682. doi:10.1089/ast.2015.1304

Tulej, M., Neubeck, A., Riedo, A., Lukmanov, R., Grimaudo, V., Ligterink, N. F. W., et al. (2020). Isotope Abundance Ratio Measurements Using Femtosecond Laser Ablation Ionization Mass Spectrometry. *J. Mass. Spectrom.* 55, e4660. doi:10.1002/jms.4660

Tulej, M., Riedo, A., Neuland, M. B., Meyer, S., Wurz, P., Thomas, N., et al. (2014). CAMAM: A Miniature Laser Ablation Ionisation Mass Spectrometer and Microscope-Camera System forIn SituInvestigation of the Composition and Morphology of Extraterrestrial Materials. *Geostand Geoanal Res.* 38, 441–466. doi:10.1111/j.1751-908x.2014.00302.x

Vago, J. L., Westall, F., Coates, A. J., Jaumann, R., Korablev, O., Ciarletti, V., et al. (2017). Habitability on Early Mars and the Search for Biosignatures with the ExoMars Rover. *Astrobiology* 17, 471–510. doi:10.1089/ast.2016.1533

Vago, J., Witasse, O., Svedhem, H., Baglioni, P., Haldemann, A., Gianfiglio, G., et al. (2015). ESA ExoMars Program: The Next Step in Exploring Mars. *Sol. Syst. Res.* 49, 518–528. doi:10.1134/s0038094615070199

Van Veen, H., Saul, N., Eargle, D., and Mangham, S. (2019). Kepler Mapper: A Flexible Python Implementation of the Mapper Algorithm. *J. Open Source Softw.* 4, 1315. doi:10.21105/joss.01315

Wacey, D., McLoughlin, N., Kilburn, M. R., Saunders, M., Cliff, J. B., Kong, C., et al. (2013). Nanoscale Analysis of Pyritized Microfossils Reveals Differential Heterotrophic Consumption in the ~1.9-Ga Gunflint Chert. *Proc. Natl. Acad. Sci. U S A.* 110, 8020–8024. doi:10.1073/pnas.1221965110

Wacey, D., Menon, S., Green, L., Gerstmann, D., Kong, C., McLoughlin, N., et al. (2012). Taphonomy of Very Ancient Microfossils from the ~3400Ma Strelley Pool Formation and ~1900Ma Gunflint Formation: New Insights Using a Focused Ion Beam. *Precambrian Res.* 220-221, 234–250. doi:10.1016/j.precamres.2012.08.005

Wacey, D., Saunders, M., Kong, C., Brasier, A., and Brasier, M. (2016). 3.46 Ga Apex Chert 'Microfossils' Reinterpreted as Mineral Artefacts Produced During Phyllosilicate Exfoliation. *Gondwana Res.* 36, 296–313. doi:10.1016/j.gr.2015.07.010

Westall, F., Foucher, F., Bost, N., Bertrand, M., Loizeau, D., Vago, J. L., et al. (2015). Biosignatures on Mars: What, Where, and How? Implications for the Search for Martian Life. *Astrobiology* 15, 998–1029. doi:10.1089/ast.2015.1374

Wiesendanger, R., Tulej, M., Riedo, A., Frey, S., Shea, H., and Wurz, P. (2017). Improved Detection Sensitivity for Heavy Trace Elements Using a Miniature Laser Ablation Ionisation Mass Spectrometer. *J. Anal. Spectrom.* 32, 2182–2188. doi:10.1039/c7ja00193b

Wiesendanger, R., Wacey, D., Tulej, M., Neubeck, A., Ivarsson, M., Grimaudo, V., et al. (2018). Chemical and Optical Identification of Micrometer-Sized 1.9 Billion-Year-Old Fossils by Combining a Miniature Laser Ablation Ionization Mass Spectrometry System with an Optical Microscope. *Astrobiology* 18, 1071–1080. doi:10.1089/ast.2017.1780

Wurz, P., Riedo, A., Tulej, M., Grimaudo, V., and Thomas, N. (2020). Investigation of the Surface Composition by Laser Ablation/Ionisation Mass Spectrometry. *LPI Contrib.* 2241, 5061. doi:10.1109/AERO50100.2021.9438486

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers in Artificial Intelligence

REVIEW
published: 29 September 2021
doi: 10.3389/frai.2021.667963

An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists

Frédéric Chazal[1] *and Bertrand Michel*[2]*

[1]Inria Saclay - Île-de-France Research Centre, Palaiseau, France, [2]Ecole Centrale de Nantes, Nantes, France

With the recent explosion in the amount, the variety, and the dimensionality of available data, identifying, extracting, and exploiting their underlying structure has become a problem of fundamental importance for data analysis and statistical learning. Topological data analysis (TDA) is a recent and fast-growing field providing a set of new topological and geometric tools to infer relevant features for possibly complex data. It proposes new well-founded mathematical theories and computational tools that can be used independently or in combination with other data analysis and statistical learning techniques. This article is a brief introduction, through a few selected topics, to basic fundamental and practical aspects of TDA for nonexperts.

Keywords: topological data analysis, machine learning, geometric inference, topological inference, statistic

OPEN ACCESS

Edited by:
Sriraam Natarajan,
The University of Texas at Dallas,
United States

Reviewed by:
Ajey Kumar,
Symbiosis International (Deemed University), India
Devendra Singh Dhami,
Darmstadt University of Technology, Germany

***Correspondence:**
Bertrand Michel
bertrand.michel@ec-nantes.fr

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: *15 February 2021*
Accepted: *16 July 2021*
Published: *29 September 2021*

Citation:
Chazal F and Michel B (2021) An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists. Front. Artif. Intell. 4:667963. doi: 10.3389/frai.2021.667963

1 INTRODUCTION AND MOTIVATION

Topological data analysis (TDA) is a recent field that emerged from various works in applied (algebraic) topology and computational geometry during the first decade of the century. Although one can trace back geometric approaches to data analysis quite far into the past, TDA really started as a field with the pioneering works of Edelsbrunner et al. (2002) and Zomorodian and Carlsson (2005) in persistent homology and was popularized in a landmark article in 2009 Carlsson (2009). TDA is mainly motivated by the idea that topology and geometry provide a powerful approach to infer robust qualitative, and sometimes quantitative, information about the structure of data [e.g., Chazal (2017)].

TDA aims at providing well-founded mathematical, statistical, and algorithmic methods to infer, analyze, and exploit the complex topological and geometric structures underlying data that are often represented as point clouds in Euclidean or more general metric spaces. During the last few years, a considerable effort has been made to provide robust and efficient data structures and algorithms for TDA that are now implemented and available and easy to use through standard libraries such as the GUDHI library[1] (C++ and Python) Maria et al. (2014) and its R software interface Fasy et al. (2014a), Dionysus[2], PHAT[3], DIPHA[4], or Giotto[5]. Although it is still rapidly evolving, TDA now provides a set of mature and efficient tools that can be used in combination with or complementarily to other data science tools.

[1]https://gudhi.inria.fr/
[2]http://www.mrzv.org/software/dionysus/
[3]https://bitbucket.org/phat-code/phat
[4]https://github.com/DIPHA/dipha
[5]https://giotto-ai.github.io/gtda-docs/0.4.0/library.html

The Topological Data Analysis Pipeline

TDA has recently known developments in various directions and application fields. There now exist a large variety of methods inspired by topological and geometric approaches. Providing a complete overview of all these existing approaches is beyond the scope of this introductory survey. However, many standard ones rely on the following basic pipeline that will serve as the backbone of this article:

1. The input is assumed to be a finite set of points coming with a notion of distance—or similarity—between them. This distance can be induced by the metric in the ambient space (e.g., the Euclidean metric when the data are embedded in $\mathbb{R}^d$) or comes as an intrinsic metric defined by a pairwise distance matrix. The definition of the metric on the data is usually given as an input or guided by the application. It is, however, important to notice that the choice of the metric may be critical to revealing interesting topological and geometric features of the data.
2. A "continuous" shape is built on the top of the data in order to highlight the underlying topology or geometry. This is often a simplicial complex or a nested family of simplicial complexes, called a filtration, which reflects the structure of the data on different scales. Simplicial complexes can be seen as higher-dimensional generalizations of neighboring graphs that are classically built on the top of data in many standard data analysis or learning algorithms. The challenge here is to define such structures as are proven to reflect relevant information about the structure of data and that can be effectively constructed and manipulated in practice.
3. Topological or geometric information is extracted from the structures built on the top of the data. This may either result in a full reconstruction, typically a triangulation, of the shape underlying the data from which topological/geometric features can be easily extracted or in crude summaries or approximations from which the extraction of relevant information requires specific methods, such as persistent homology. Beyond the identification of interesting topological/geometric information and its visualization and interpretation, the challenge at this step is to show its relevance, in particular its stability with respect to perturbations or the presence of noise in the input data. For that purpose, understanding the statistical behavior of the inferred features is also an important question.
4. The extracted topological and geometric information provides new families of features and descriptors of the data. They can be used to better understand the data—in particular, through visualization—or they can be combined with other kinds of features for further analysis and machine learning tasks. This information can also be used to design well-suited data analysis and machine learning models. Showing the added value and the complementarity (with respect to other features) of the information provided using TDA tools is an important question at this step.

Topological Data Analysis and Statistics

Until quite recently, the theoretical aspects of TDA and topological inference mostly relied on deterministic approaches. These deterministic approaches do not take into account the random nature of data and the intrinsic variability of the topological quantity they infer. Consequently, most of the corresponding methods remain exploratory, without being able to efficiently distinguish between information and what is sometimes called the "topological noise" (see **Section 6.2** further in the article).

A statistical approach to TDA means that we consider data as generated from an unknown distribution but also that the topological features inferred using TDA methods are seen as estimators of topological quantities describing an underlying object. Under this approach, the unknown object usually corresponds to the support of the data distribution (or part of it). The main goals of a statistical approach to topological data analysis can be summarized as the following list of problems:

Topic 1: proving consistency and studying the convergence rates of TDA methods.
Topic 2: providing confidence regions for topological features and discussing the significance of the estimated topological quantities.
Topic 3: selecting relevant scales on which the topological phenomenon should be considered, as a function of observed data.
Topic 4: dealing with outliers and providing robust methods for TDA.

Applications of Topological Data Analysis in Data Science

On the application side, many recent promising and successful results have demonstrated the interest in topological and geometric approaches in an increasing number of fields such as material science (Kramar et al., 2013; Nakamura et al., 2015; Pike et al., 2020), 3D shape analysis (Skraba et al., 2010; Turner et al., 2014b), image analysis (Qaiser et al., 2019; Rieck et al., 2020), multivariate time series analysis (Khasawneh and Munch, 2016; Seversky et al., 2016; Umeda, 2017), medicine (Dindin et al., 2020), biology (Yao et al., 2009), genomics (Carrière and Rabadán, 2020), chemistry (Lee et al., 2017; Smith et al., 2021), sensor networks De Silva and Ghrist (2007), or transportation (Li et al., 2019), to name a few. It is beyond our scope to give an exhaustive list of applications of TDA. On the other hand, most of the successes of TDA result from its combination with other analysis or learning techniques (see **Section 6.5** for a discussion and references). So, clarifying the position and complementarity of TDA with respect to other approaches and tools in data science is also an important question and an active research domain.

The overall objective of this survey article is two-fold. First, it intends to provide data scientists with a brief and comprehensive introduction to the mathematical and statistical foundations of TDA. For that purpose, the focus is put on a few selected, but fundamental, tools and topics, which are simplicial complexes (**Section 2**) and their use for exploratory topological data analysis (**Section 3**), geometric inference (**Section 4**), and persistent homology theory (**Section 5**), which play a central role in TDA. Second, this article also aims at demonstrating how, thanks to the recent progress of software, TDA tools can be easily applied in data science. In particular, we show how the Python version of the

GUDHI library allows us to easily implement and use the TDA tools presented in this article (**Section 7**). Our goal is to quickly provide the data scientist with a few basic keys—and relevant references—so that he can get a clear understanding of the basics of TDA and will be able to start to use TDA methods and software for his own problems and data.

Other reviews on TDA can be found in the literature, which are complementary to our work. Wasserman (2018) presented a statistical view on TDA, and it focused, in particular, on the connections between TDA and density clustering. Sizemore et al. (2019) proposed a survey about the application of TDA to neurosciences. Finally, Hensel et al. (2021) proposed a recent overview of applications of TDA to machine learning.

2 METRIC SPACES, COVERS, AND SIMPLICIAL COMPLEXES

As topological and geometric features are usually associated with continuous spaces, data represented as finite sets of observations do not directly reveal any topological information *per se*. A natural way to highlight some topological structure out of data is to "connect" data points that are close to each other in order to exhibit a global continuous shape underlying the data. Quantifying the notion of closeness between data points is usually done using a distance (or a dissimilarity measure), and it often turns out to be convenient in TDA to consider data sets as discrete metric spaces or as samples of metric spaces. This section introduces general concepts for geometric and topological inference; a more complete presentation of the topic is given in the study by Boissonnat et al. (2018).

Metric Spaces

Recall that a metric space (M, ρ) is a set M with a function $\rho: M \times M \to \mathbb{R}_+$, called a distance, such that for any $x, y, z \in M$, the following is the case:

i) $\rho(x, y) \geq 0$ and $\rho(x, y) = 0$ if and only if $x = y$,
ii) $\rho(x, y) = \rho(y, x)$, and
iii) $\rho(x, z) \leq \rho(x, y) + \rho(y, z)$.

Given a metric space (M, ρ), the set $\mathcal{K}(M)$ of its compact subsets can be endowed with the so-called Hausdorff distance; given two compact subsets $A, B \subseteq M$, the Hausdorff distance $d_H(A, B)$ between A and B is defined as the smallest nonnegative number δ such that for any $a \in A$, there exists $b \in B$ such that $\rho(a, b) \leq \delta$, and for any $b \in B$, there exists $a \in A$ such that $\rho(a, b) \leq \delta$ (see **Figure 1**). In other words, if for any compact subset $C \subseteq M$, we denote by $d(., C): M \to \mathbb{R}_+$ the distance function to C defined by $d(x, C) := \inf_{c \in C} \rho(x, c)$ for any $x \in M$, then one can prove that the Hausdorff distance between A and B is defined by any of the two following equalities:

$$d_H(A, B) = \max\left\{\sup_{b \in B} d(b, A), \sup_{a \in A} d(a, B)\right\} = \sup_{x \in M} |d(x, A) - d(x, B)| = \|d(., A) - d(., B)\|_\infty$$

It is a basic and classical result that the Hausdorff distance is indeed a distance on the set of compact subsets of a metric space. From a TDA perspective, it provides a convenient way to quantify the proximity between different data sets issued from the same ambient metric space. However, it sometimes occurs that one has to compare data sets that are not sampled from the same ambient space. Fortunately, the notion of the Hausdorff distance can be generalized to the comparison of any pair of compact metric spaces, giving rise to the notion of the Gromov–Hausdorff distance.

Two compact metric spaces, (M_1, ρ_1) and (M_2, ρ_2), are isometric if there exists a bijection $\phi: M_1 \to M_2$ that preserves distances, that is, $\rho_2(\phi(x), \phi(y)) = \rho_1(x, y)$ for any $x, y \in M_1$. The Gromov–Hausdorff distance measures how far two metric spaces are from being isometric.

Definition 1. *The Gromov–Hausdorff distance $d_{GH}(M_1, M_2)$ between two compact metric spaces is the infimum of the real numbers $r \geq 0$ such that there exists a metric space (M, ρ) and two compact subspaces C_1 and $C_2 \subset M$ that are isometric to M_1 and M_2 and such that $d_H(C_1, C_2) \leq r$.*

The Gromov–Hausdorff distance will be used later, in **Section 5**, for the study of stability properties and persistence diagrams.

Connecting pairs of nearby data points by edges leads to the standard notion of the neighboring graph from which the connectivity of the data can be analyzed, for example, using some clustering algorithms. To go beyond connectivity, a central idea in TDA is to build higher-dimensional equivalents of neighboring graphs using not only connecting pairs but also $(k + 1)$-uple of nearby data points. The resulting objects, called simplicial complexes, allow us to identify new topological features such as cycles, voids, and their higher-dimensional counterpart.

Geometric and Abstract Simplicial Complexes

Simplicial complexes can be seen as higher-dimensional generalization of graphs. They are mathematical objects that are both topological and combinatorial, a property making them particularly useful for TDA.

Given a set $\mathbb{X} = \{x_0, \ldots, x_k\} \subset \mathbb{R}^d$ of $k + 1$ affinely independent points, the k-dimensional simplex $\sigma = [x_0, \ldots, x_k]$ spanned by $\mathbb{X}$ is the convex hull of $\mathbb{X}$. The points of $\mathbb{X}$ are called the vertices of σ, and the simplices spanned by the subsets of $\mathbb{X}$ are called the faces of σ. A geometric simplicial complex K in $\mathbb{R}^d$ is a collection of simplices such that the following are the case:

i) any face of a simplex of K is a simplex of K and
ii) the intersection of any two simplices of K is either empty or a common face of both.

The union of the simplices of K is a subset of $\mathbb{R}^d$ called the underlying space of K that inherits from the topology of $\mathbb{R}^d$. So, K can also be seen as a topological space through its underlying space. Notice that once its vertices are known, K is fully characterized by the combinatorial description of a collection of simplices satisfying some incidence rules.

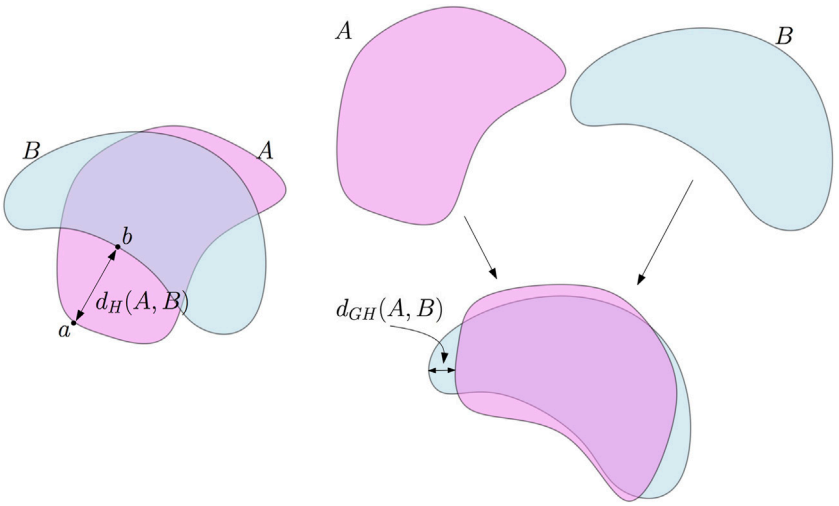


FIGURE 1 | Left: the Hausdorff distance between two subsets A and B of the plane. In this example, $d_H(A, B)$ is the distance between the point a in A which is the farthest from B and its nearest neighbor b on B. Right: the Gromov–Hausdorff distance between A and B. A can be rotated—this is an isometric embedding of A in the plane—to reduce its Hausdorff distance to B. As a consequence, $d_{GH}(A, B) \leq d_H(A, B)$.

Given a set V, an abstract simplicial complex with the vertex set V is a set $\tilde{K}$ of finite subsets of V such that the elements of V belong to $\tilde{K}$ and for any $\sigma \in \tilde{K}$, any subset of σ belongs to $\tilde{K}$. The elements of $\tilde{K}$ are called the faces or the simplices of $\tilde{K}$. The dimension of an abstract simplex is just its cardinality minus 1 and the dimension of $\tilde{K}$ is the largest dimension of its simplices. Notice that simplicial complexes of dimension 1 are graphs.

The combinatorial description of any geometric simplicial K obviously gives rise to an abstract simplicial complex $\tilde{K}$. The converse is also true; one can always associate with an abstract simplicial complex $\tilde{K}$ a topological space $|\tilde{K}|$ such that if K is a geometric complex whose combinatorial description is the same as $\tilde{K}$, the underlying space of K is homeomorphic to $|\tilde{K}|$. Such a K is called a geometric realization of $\tilde{K}$. As a consequence, abstract simplicial complexes can be seen as topological spaces and geometric complexes can be seen as geometric realizations of their underlying combinatorial structure. So, one can consider simplicial complexes at the same time as combinatorial objects that are well suited for effective computations and as topological spaces from which topological properties can be inferred.

Building Simplicial Complexes From Data

Given a data set, or more generally, a topological or metric space, there exist many ways to build simplicial complexes. We present here a few classical examples that are widely used in practice.

A first example is an immediate extension of the notion of the α-neighboring graph. Assume that we are given a set of points $\mathbb{X}$ in a metric space (M, ρ) and a real number $\alpha \geq 0$. The Vietoris–Rips complex $Rips_\alpha(\mathbb{X})$ is the set of simplices $[x_0, \ldots, x_k]$ such that $d_{\mathbb{X}}(x_i, x_j) \leq \alpha$ for all (i, j), see **Figure 2**. It follows immediately from the definition that this is an abstract simplicial complex. However, in general, even when $\mathbb{X}$ is a finite subset of $\mathbb{R}^d$, $Rips_\alpha(\mathbb{X})$ does not admit a geometric realization in $\mathbb{R}^d$; in particular, it can be of a dimension higher than d.

Closely related to the Vietoris–Rips complex is the Čech complex $Cech_\alpha(\mathbb{X})$ that is defined as the set of simplices $[x_0, \ldots, x_k]$ such that the $k + 1$ closed balls $B(x_i, \alpha)$ have a non-empty intersection, see **Figure 2**. Notice that these two complexes are related by

$$Rips_\alpha(\mathbb{X}) \subseteq Cech_\alpha(\mathbb{X}) \subseteq Rips_{2\alpha}(\mathbb{X})$$

and that if $\mathbb{X} \subset \mathbb{R}^d$, then $Cech_\alpha(\mathbb{X})$ and $Rips_{2\alpha}(\mathbb{X})$ have the same one-dimensional skeleton, that is, the same set of vertices and edges.

The Nerve Theorem

The Čech complex is a particular case of a family of complexes associated with covers. Given a cover $\mathcal{U} = (U_i)_{i \in I}$ of $\mathbb{M}$, that is, a family of sets U_i such that $\mathbb{M} = \cup_{i \in I} U_i$, the nerve of $\mathcal{U}$ is the abstract simplicial complex $C(\mathcal{U})$ whose vertices are the U_i's and such that

$$\sigma = [U_{i_0}, \ldots, U_{i_k}] \in C(\mathcal{U}) \text{ if and only if } \cap_{j=0}^{k} U_{i_j} \neq \emptyset.$$

Given a cover of a data set, where each set of the cover can be, for example, a local cluster or a grouping of data points sharing some common properties, its nerve provides a compact and global combinatorial description of the relationship between these sets through their intersection patterns (see **Figure 3**).

A fundamental theorem in algebraic topology relates, under some assumptions, the topology of the nerve of a cover to the topology of the union of the sets of the cover. To be formally stated, this result, known as the Nerve theorem, requires the introduction of a few notions.

Two topological spaces, X and Y, are usually considered as being the same from a topological point of view if they are

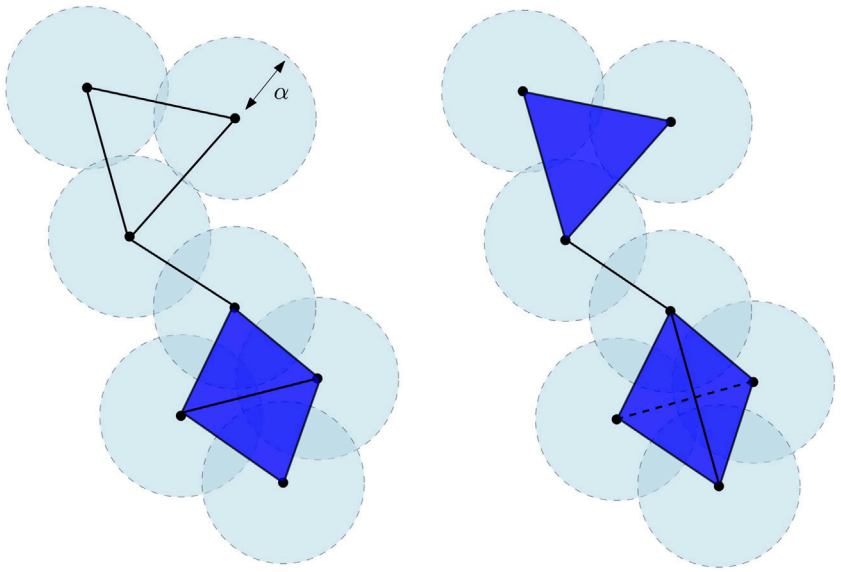


FIGURE 2 | Čech complex $Cech_\alpha(\mathbb{X})$ (left) and the Vietoris–Rips $Rips_{2\alpha}(\mathbb{X})$ (right) of a finite point cloud in the plane $\mathbb{R}^2$. The bottom part of $Cech_\alpha(\mathbb{X})$ is the union of two adjacent triangles, while the bottom part of $Rips_{2\alpha}(\mathbb{X})$ is the tetrahedron spanned by the four vertices and all its faces. The dimension of the Čech complex is 2. The dimension of the Vietoris–Rips complex is 3. Notice that this latter is thus not embedded in $\mathbb{R}^2$.

homeomorphic, that is, if there exist two continuous bijective maps f: $X \to Y$ and g: $Y \to X$ such that $f \circ g$ and $g \circ f$ are the identity map of Y and X, respectively. In many cases, asking X and Y to be homeomorphic turns out to be too strong a requirement to ensure that X and Y share the same topological features of interest for TDA. Two continuous maps f_0, f_1: $X \to Y$ are said to be homotopic if there exists a continuous map H: $X \times [0, 1] \to Y$ such that for any $x \in X$, $H(x, 0) = f_0(x)$ and $H(x, 1) = g(x)$. The spaces X and Y are then said to be homotopy equivalent if there exist two maps, f: $X \to Y$ and g: $Y \to X$, such that $f \circ g$ and $g \circ f$ are homotopic to the identity map of Y and X, respectively. The maps f and g are then called homotopy equivalent. The notion of homotopy equivalence is weaker than the notion of homeomorphism; if X and Y are homeomorphic, then they are obviously homotopy equivalent, but the converse is not true. However, spaces that are homotopy equivalent still share many topological invariants; in particular, they have the same homology (see **Section 4**).

A space is said to be contractible if it is homotopy equivalent to a point. Basic examples of contractible spaces are the balls and, more generally, the convex sets in $\mathbb{R}^d$. Open covers for whom all elements and their intersections are contractible have the remarkable following property.

Theorem 1 (Nerve theorem). *Let $\mathcal{U} = (U_i)_{i \in I}$ be a cover of a topological space X by open sets such that the intersection of any subcollection of the U_i's is either empty or contractible. Then, X and the nerve $C(\mathcal{U})$ are homotopy equivalent.*

It is easy to verify that convex subsets of Euclidean spaces are contractible. As a consequence, if $\mathcal{U} = (U_i)_{i \in I}$ is a collection of convex subsets of $\mathbb{R}^d$, then $C(\mathcal{U})$ and $\cup_{i \in I} U_i$ are homotopy equivalent. In particular, if $\mathbb{X}$ is a set of points in $\mathbb{R}^d$, then the Čech complex $Cech_\alpha(\mathbb{X})$ is homotopy equivalent to the union of balls $\cup_{x \in \mathbb{X}} B(x, \alpha)$.

The Nerve theorem plays a fundamental role in TDA; it provides a way to encode the topology of continuous spaces into abstract combinatorial structures that are well suited for the design of effective data structures and algorithms.

3 USING COVERS AND NERVES FOR EXPLORATORY DATA ANALYSIS AND VISUALIZATION: THE MAPPER ALGORITHM

Using the nerve of covers as a way to summarize, visualize, and explore data is a natural idea that was first proposed for TDA in the study by Singh et al. (2007), giving rise to the so-called Mapper algorithm.

Definition 2. *Let f: $X \to \mathbb{R}^d$, $d \geq 1$, be a continuous real valued function and let $\mathcal{U} = (U_i)_{i \in I}$ be a cover of $\mathbb{R}^d$. The pull-back cover of X induced by $(f, \mathcal{U})$ is the collection of open sets $(f^{-1}(U_i))_{i \in I}$. The refined pull-back is the collection of connected components of the open sets $f^{-1}(U_i)$, $i \in I$.*

The idea of the Mapper algorithm is, given a data set $\mathbb{X}$ and a well-chosen real-valued function f: $\mathbb{X} \to \mathbb{R}^d$, to summarize $\mathbb{X}$ through the nerve of the refined pull-back of a cover $\mathcal{U}$ of $f(\mathbb{X})$ (see **Figure 4A**). For well-chosen covers $\mathcal{U}$ (see below), this nerve is a graph providing an easy and convenient way to visualize the summary of the data. It is described in **Algorithm 1** and illustrated on a simple example in **Figure 4B**.

The Mapper algorithm is very simple (see **Algorithm 1**); but it raises several questions about the various choices that are left to the user and that we briefly discuss in the following.

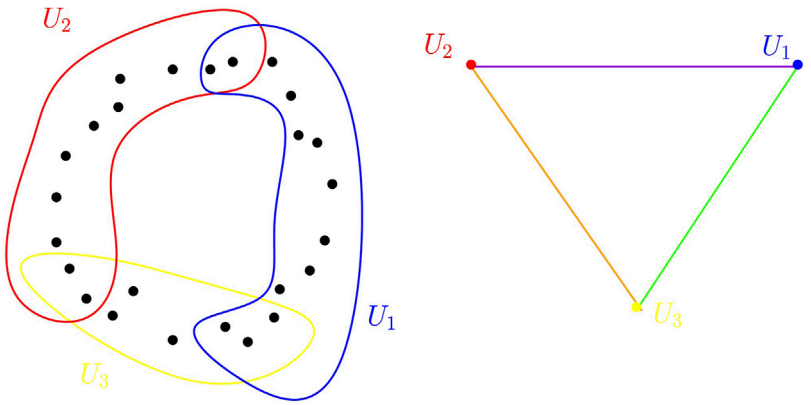


FIGURE 3 | Point cloud sampled in the plane and a cover of open sets for this point cloud (left). The nerve of this cover is a triangle (right). Edges correspond to a set of the cover whereas a vertex corresponds to a non-empty intersection between two sets of the cover.

Algorithm 1 | The Mapper algorithm

Input: a data set $\mathbb{X}$ with a metric or a dissimilarity measure between data points, a function f: $\mathbb{X} \to \mathbb{R}$ (or $\mathbb{R}^d$), and a cover $\mathcal{U}$ of $f(\mathbb{X})$
for each $U \in \mathcal{U}$ decompose $f^{-1}(U)$ into clusters $C_{U,1}, \ldots, C_{U,k_U}$.
Compute the nerve of the cover of X defined by the $C_{U,1}, \ldots, C_{U,k_U}$, $U \in \mathcal{U}$.
Output: a simplicial complex; the nerve (often a graph for well-chosen covers → easy to visualize) includes the following:
- a vertex $v_{U,i}$ for each cluster $C_{U,i}$ and
- an edge between $v_{U,i}$ and $v_{U',j}$ if $C_{U,i} \cap C_{U',j} \neq \emptyset$.

The Choice of *f*

The choice of the function f, sometimes called the filter or lens function, strongly depends on the features of the data that one expects to highlight. The following ones are among the ones more or less classically encountered in the literature:

-Density estimates: the Mapper complex may help to understand the structure and connectivity of high-density areas (clusters).
-PCA coordinates or coordinate functions obtained from a nonlinear dimensionality reduction (NLDR) technique, eigenfunctions of graph laplacians may help to reveal and understand some ambiguity in the use of nonlinear dimensionality reductions.
-The centrality function $f(x) = \sum_{y \in \mathbb{X}} d(x, y)$ and the eccentricity function $f(x) = \max_{y \in \mathbb{X}} d(x, y)$ sometimes appear to be good choices that do not require any specific knowledge about the data.
-For data that are sampled around one-dimensional filamentary structures, the distance function to a given point allows us to recover the underlying topology of the filamentary structures Chazal et al. (2015d).

The Choice of the Cover $\mathcal{U}$

When f is a real-valued function, a standard choice is to take $\mathcal{U}$ to be a set of regularly spaced intervals of equal length, $r > 0$, covering the set $f(\mathbb{X})$. The real r is sometimes called the resolution of the cover, and the percentage g of overlap between two consecutive intervals is called the gain of the cover. Note that if the gain g is chosen below 50%, then every point of the real line is covered by, at most, 2 open sets of $\mathcal{U}$, and the output nerve is a graph. It is important to notice that the output of Mapper is very sensitive to the choice of $\mathcal{U}$, and small changes in the resolution and gain parameters may result in very large changes in the output, making the method very unstable. A classical strategy consists in exploring some range of parameters and selecting the ones that turn out to provide the most informative output from the user perspective.

The Choice of the Clusters

The Mapper algorithm requires the clustering of the preimage of the open sets $U \in \mathcal{U}$. There are two strategies to compute the clusters. A first strategy consists in applying, for each $U \in \mathcal{U}$, a cluster algorithm, chosen by the user, to the preimage $f^{-1}(U)$. A second, more global, strategy consists in building a neighboring graph on the top of the data set $\mathbb{X}$, for example, a k-NN graph or a ε-graph, and, for each $U \in \mathcal{U}$, taking the connected components of the subgraph with the vertex set $f^{-1}(U)$.

Theoretical and Statistical Aspects of Mapper

Based on the results on stability and the structure of Mapper proposed in the study by Carrière and Oudot (2017), advances toward a statistically well-founded version of Mapper have been made recently in the study by Carriere et al. (2018). Unsurprisingly, the convergence of Mapper depends on both the sampling of the data and the regularity of the filter function. Moreover, subsampling strategies can be proposed to select a complex in a Rips filtration on a convenient scale, as well as the resolution and the gain for defining the Mapper graph. The case of stochastic and multivariate filters has also been studied by Carrière and Michel (2019). An alternative description of the probabilistic convergence of Mapper, in terms of categorification, has also been proposed in the study by Brown et al. (2020). Other approaches have been proposed to study and deal with the

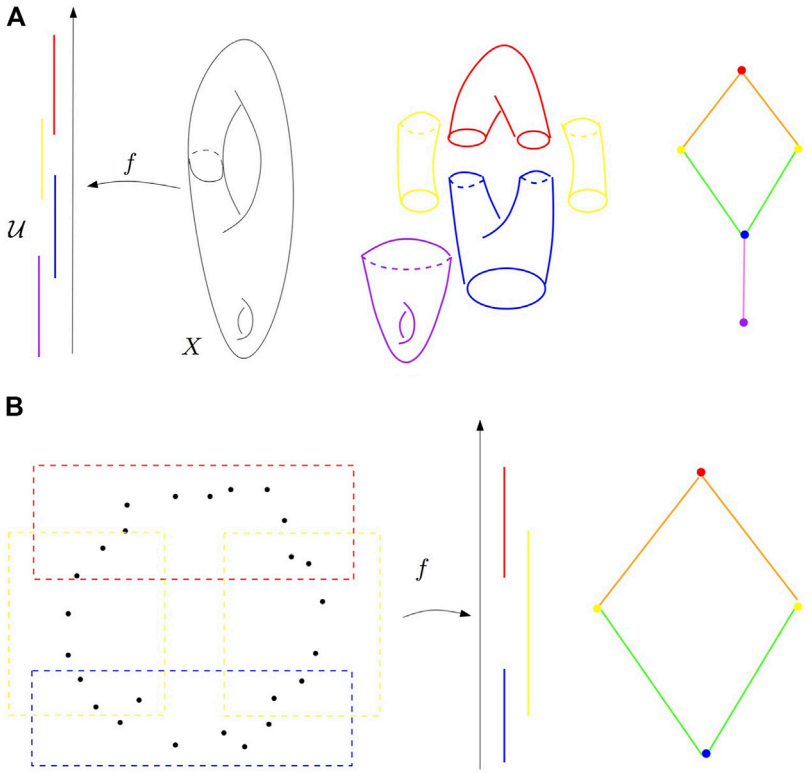


FIGURE 4 | (A) Refined pull-back cover of the height function on a surface in $\mathbb{R}^3$ and its nerve. **(B)** Mapper algorithm on a point cloud sampled around a circle and the height function. First, the pull-back cover of the height function defined on the point cloud is computed and refined (left). Second, the nerve of the refined pull-back is visualized as a graph (right).

instabilities of the Mapper algorithm in the works of Dey et al. (2016), Dey et al. (2017).

Data Analysis With Mapper

As an exploratory data analysis tool, Mapper has been successfully used for clustering and feature selection. The idea is to identify specific structures in the Mapper graph (or complex), in particular, loops and flares. These structures are then used to identify interesting clusters or to select features or variables that best discriminate the data in these structures. Applications on real data, illustrating these techniques, may be found, for example, in the studies by Carrière and Rabadán (2020), Lum et al. (2013), Yao et al. (2009).

4 GEOMETRIC RECONSTRUCTION AND HOMOLOGY INFERENCE

Another way to build covers and use their nerves to exhibit the topological structure of data is to consider the union of balls centered on the data points. In this section, we assume that $\mathbb{X}_n = \{x_0, \ldots, x_n\}$ is a subset of $\mathbb{R}^d$, sampled i. i. d. according to a probability measure μ with compact support $M \subset \mathbb{R}^d$. The general strategy to infer topological information about M from μ proceeds in two steps that are discussed in the following part of this section:

1. $\mathbb{X}_n$ is covered by a union of balls of a fixed radius centered on the x_i's. Under some regularity assumptions on M, one can relate the topology of this union of balls to the one of M and
2. from a practical and algorithmic perspective, topological features of M are inferred from the nerve of the union of balls, using the Nerve theorem.

In this framework, it is indeed possible to compare spaces through isotopy equivalence, a stronger notion than homeomorphism; $X \subseteq \mathbb{R}^d$ and $Y \subseteq \mathbb{R}^d$ are said to be (ambient) isotopic if there exists a continuous family of homeomorphisms $H\colon [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$, H continuous, such that for any $t \in [0,1]$, $H_t = H(t,.)\colon \mathbb{R}^d \to \mathbb{R}^d$ is a homeomorphism, H_0 is the identity map in $\mathbb{R}^d$, and $H_1(X) = Y$. Obviously, if X and Y are isotopic, then they are homeomorphic. The converse is not true; a knotted circle and an unknotted circle in $\mathbb{R}^3$ are not homeomorphic (notice that although this claim seems rather intuitive, its formal proof requires the use of some nonobvious algebraic topology tools).

4.1 Distance-Like Functions and Reconstruction

Given a compact subset K of $\mathbb{R}^d$ and a nonnegative real number r, the union of balls of radius r centered on K, $K^r = \cup_{x \in K} B(x, r)$, called the r-offset of K, is the r-sublevel set of the distance function $d_K\colon \mathbb{R}^d \to \mathbb{R}$ defined by $d_K(x) = \inf_{y \in K} \|x - y\|$; in

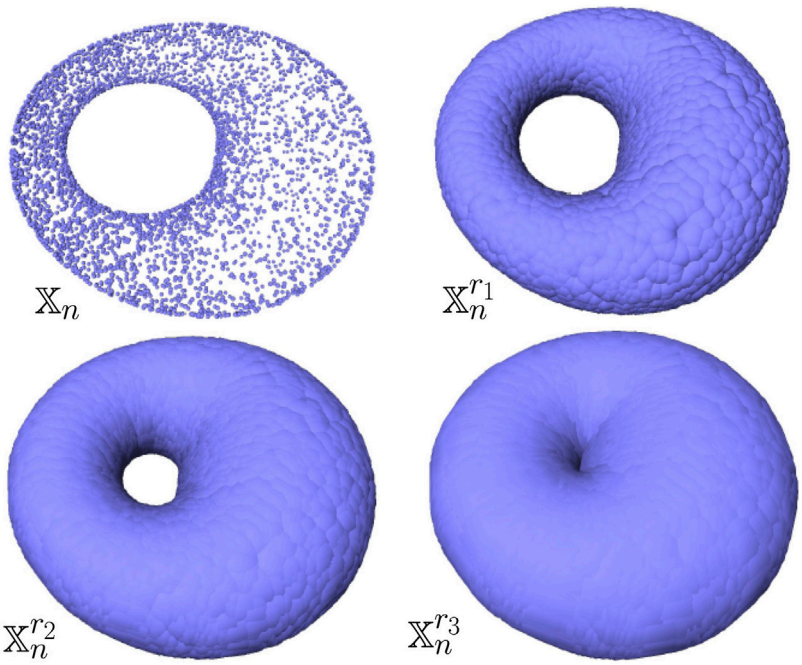


FIGURE 5 | Example of a point cloud $\mathbb{X}_n$ sampled on the surface of a torus in $\mathbb{R}^3$ (top left) and its offsets for different values of radii $r_1 < r_2 < r_3$. For well-chosen values of the radius (e.g., r_1 and r_2), the offsets are clearly homotopy equivalent to a torus.

other words, $K^r = d_k^{-1}([0, r])$. This remark allows us to use differential properties of distance functions and to compare the topology of the offsets of compact sets that are close to each other with respect to the Hausdorff distance.

Definition 3 (Hausdorff distance in $\mathbb{R}^d$). *The Hausdorff distance between two compact subsets K, K' of $\mathbb{R}^d$ is defined by*

$$d_H(K, K') = \|d_K - d_{K'}\|_\infty = \sup_{x \in \mathbb{R}^d} |d_K(x) - d_{K'}(x)|.$$

In our setting, the considered compact sets are the data set $\mathbb{X}_n$ and of the support M of the measure μ. When M is a smooth compact submanifold, under mild conditions on $d_H(\mathbb{X}_n, M)$, for some well-chosen r, the offsets of $\mathbb{X}_n$ are homotopy equivalent to M Chazal and Lieutier (2008), Niyogi et al. (2008) (see **Figure 5** for an illustration). These results extend to larger classes of compact sets and lead to stronger results on the inference of the isotopy type of the offsets of M Chazal et al. (2009c), Chazal et al. (2009d). They also lead to results on the estimation of other geometric and differential quantities such as normals Chazal et al. (2009c), curvatures Chazal et al. (2009e), or boundary measures Chazal et al. (2010) under assumptions on the Hausdorff distance between the underlying shape and the data sample.

These results rely on the one-semiconcavity of the squared distance function d_K^2, that is, the convexity of the function $x \to \|x\|^2 - d_K^2(x)$, and can be naturally stated in the following general framework.

Definition 4. *A function $\phi: \mathbb{R}^d \to \mathbb{R}_+$ is distance-like if it is proper (the preimage of any compact set in $\mathbb{R}$ is a compact set in $\mathbb{R}^d$) and $x \to \|x\|^2 - \phi^2(x)$ is convex.*

Thanks to its semiconcavity, a distance-like function ϕ has a well-defined, but not continuous, gradient $\nabla\phi: \mathbb{R}^d \to \mathbb{R}^d$ that can be integrated into a continuous flow (Petrunin, 2007) that allows us to track the evolution of the topology of its sublevel sets and to compare it to one of the sublevel sets of close distance-like functions.

Definition 5. *Let ϕ be a distance-like function and let $\phi^r = \phi^{-1}([0, r])$ be the r-sublevel set of ϕ.*

- *A point $x \in \mathbb{R}^d$ is called α-critical if $\|\nabla_x \phi\| \leq \alpha$. The corresponding value $r = \phi(x)$ is also said to be α-critical.*
- *The weak feature size of ϕ at r is the minimum $r' > 0$ such that ϕ does not have any critical value between r and $r + r'$. We denote it by $\mathrm{wfs}_\phi(r)$. For any $0 < \alpha < 1$, the α-reach of ϕ is the maximum r such that $\phi^{-1}((0, r])$ does not contain any α-critical point.*

The weak feature size $\mathrm{wfs}_\phi(r)$ (resp. α-reach) measures the regularity of ϕ around its r-level sets (resp. O-level set). When $\phi = d_K$ is the distance function to a compact set $K \subset \mathbb{R}^d$, the one-reach coincides with the classical reach from geometric measure theory Federer (1959). Its estimation from random samples has been studied by Aamari et al. (2019). An important property of a distance-like function ϕ is that the topology of their sublevel sets ϕ^r can only change when r crosses a 0-critical value.

Lemma 1 (isotopy lemma grove (1993)). *Let ϕ be a distance-like function and $r_1 < r_2$ be two positive numbers such that ϕ has no 0-critical point, that is, points x such that $\nabla\phi(x) = 0$, in the subset $\phi^{-1}([r_1, r_2])$. Then all the sublevel sets $\phi^{-1}([0, r])$ are isotopic for $r \in [r_1, r_2]$.*

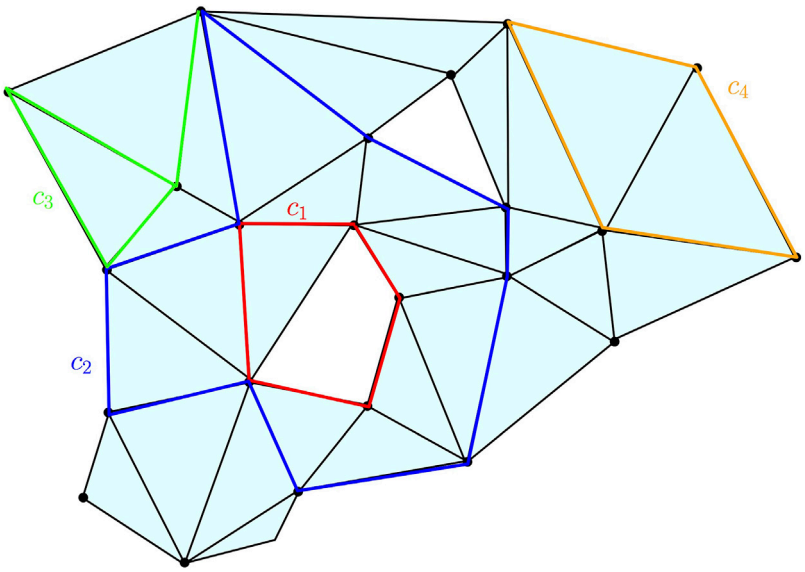


FIGURE 6 | Some examples of chains, cycles, and boundaries on a two-dimensional complex K: c_1, c_2, and c_4 are one-cycles; c_3 is a one-chain but not a one-cycle; c_4 is the one-boundary, namely, the boundary of the two-chain obtained as the sum of the two triangles surrounded by c_4. The cycles c_1 and c_2 span the same element in $H_1(K)$ as their difference is the two-chain represented by the union of the triangles surrounded by the union of c_1 and c_2.

As an immediate consequence of the isotopy lemma, all the sublevel sets of ϕ between r and $r + \mathrm{wfs}_\phi(r)$ have the same topology. Now the following reconstruction theorem from Chazal et al. (2011b) provides a connection between the topology of the sublevel sets of close distance-like functions.

Theorem 2 (Reconstruction theorem). *Let ϕ, ψ be two distance-like functions such that $\|\phi - \psi\|_\infty < \varepsilon$, with $\mathrm{reach}_\alpha(\phi) \geq R$ for some positive ε and α. Then, for every $r \in [4\varepsilon/\alpha^2, R - 3\varepsilon]$ and every $\eta \in (0, R)$, the sublevel sets ψ^r and ϕ^η are homotopy equivalent when*

$$\varepsilon \leq \frac{R}{5 + 4/\alpha^2}.$$

Under similar but slightly more technical conditions, the Reconstruction theorem can be extended to prove that the sublevel sets are indeed homeomorphic and even isotopic (Chazal et al., 2009c; Chazal et al., 2008).

Coming back to our setting and taking for $\phi = d_M$ and $\psi = d_{\mathbb{X}_n}$ the distance functions to the support M of the measure μ and to the data set $\mathbb{X}_n$, the condition $\mathrm{reach}_\alpha(d_M) \geq R$ can be interpreted as the regularity condition on M[6]. The Reconstruction theorem combined with the Nerve theorem tells that for well-chosen values of r, η and the η-offsets of M are homotopy equivalent to the nerve of the union of balls of radius r centered on $\mathbb{X}_n$, that is, the Cech complex $Cech_r(\mathbb{X}_n)$.

From a statistical perspective, the main advantage of these results involving the Hausdorff distance is that the estimation of the considered topological quantities boils down to support estimation questions that have been widely studied (see **Section 4.3**).

4.2 Homology Inference

The above results provide a mathematically well-founded framework to infer the topology of shapes from a simplicial complex built on the top of an approximating finite sample. However, from a more practical perspective, it raises two issues. First, the Reconstruction theorem requires a regularity assumption through the α-reach condition that may not always be satisfied and the choice of a radius r for the ball used to build the Čech complex $Cech_r(\mathbb{X}_n)$. Second, $Cech_r(\mathbb{X}_n)$ provides a topologically faithful summary of the data through a simplicial complex that is usually not well suited for further data processing. One often needs topological descriptors that are easier to handle, in particular numerical ones, which can be easily computed from the complex. This second issue is addressed by considering the homology of the considered simplicial complexes in the next paragraph, while the first issue will be addressed in the next section with the introduction of persistent homology.

Homology in a Nutshell

Homology is a classical concept in algebraic topology, providing a powerful tool to formalize and handle the notion of the topological features of a topological space or of a simplicial complex in an algebraic way. For any dimension k, the k-dimensional "holes" are represented by a vector space H_k, whose dimension is intuitively the number of such independent features. For example, the zero-dimensional homology group H_0 represents the connected components of the complex, the one-dimensional homology group H_1 represents

[6]As an example, if M is a smooth compact submanifold, then $\mathrm{reach}_0(\phi)$ is always positive and known as the reach of M Federer (1959).

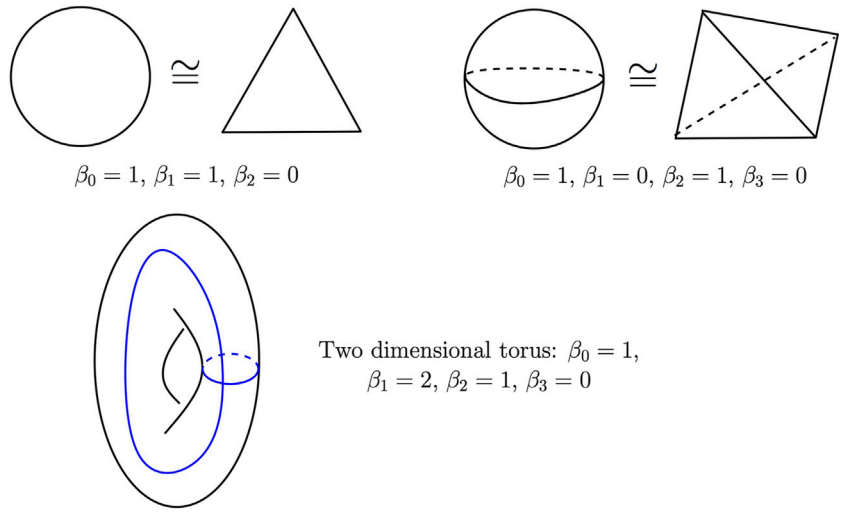


FIGURE 7 | Betti numbers of the circle (top left), the two-dimensional sphere (top right), and the two-dimensional torus (bottom). The blue curves on the torus represent two independent cycles whose homology class is a basis of its one-dimensional homology group.

the one-dimensional loops, the two-dimensional homology group H_2 represents the two-dimensional cavities, and so on.

To avoid technical subtleties and difficulties, we restrict the introduction of homology to the minimum that is necessary to understand its usage in the following of the article. In particular, we restrict our information to homology with coefficients in $\mathbb{Z}_2$, that is, the field with two elements, 0 and 1, such that $1 + 1 = 0$, which turns out to be geometrically a little bit more intuitive. However, all the notions and results presented in the sequel naturally extend to homology with coefficients in any field. We refer the reader to the study by Hatcher (2001) for a complete and comprehensible introduction to homology and to the study by Ghrist (2017) for a recent, concise, and very good introduction to applied algebraic topology and its connections to data analysis.

Let K be a (finite) simplicial complex and let k be a nonnegative integer. The space of k-chains on K, $C_k(K)$ is the set whose elements are the formal (finite) sums of k-simplices of K. More precisely, if $\{\sigma_1, \ldots, \sigma_p\}$ is the set of k-simplices of K, then any k-chain can be written as

$$c = \sum_{i=1}^{p} \varepsilon_i \sigma_i \quad \text{with} \quad \varepsilon_i \in \mathbb{Z}_2.$$

If $c' = \sum_{i=1}^{p} \varepsilon_i' \sigma_i$ is another k-chain and $\lambda \in \mathbb{Z}_2$, the sum $c + c'$ is defined as $c + c' = \sum_{i=1}^{p} (\varepsilon_i + \varepsilon_i')\sigma_i$ and the product $\lambda.c$ is defined as $\lambda.c = \sum_{i=1}^{p} (\lambda.\varepsilon_i)\sigma_i$, making $C_k(K)$ a vector space with coefficients in $\mathbb{Z}_2$. Since we are considering coefficients in $\mathbb{Z}_2$, geometrically, a k-chain can be seen as a finite collection of k-simplices and the sum of two k-chains as the symmetric difference of the two corresponding collections[7].

The boundary of a k-simplex $\sigma = [v_0, \ldots, v_k]$ is the $(k-1)$-chain

$$\partial_k(\sigma) = \sum_{i=0}^{k} (-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_k]$$

where $[v_0, \ldots, \hat{v}_i, \ldots, v_k]$ is the $(k-1)$-simplex spanned by all the vertices except v_i[8]. As the k-simplices form a basis of $C_k(K)$, ∂_k extends as a linear map from $C_k(K)$ to $C_{k-1}(K)$ called the boundary operator. The kernel $Z_k(K) = \{c \in C_k(K) : \partial_k = 0\}$ of ∂_k is called the space of k-cycles of K, and the image $B_k(K) = \{c \in C_k(K) : \exists c' \in C_{k+1}(K), \partial_{k+1}(c') = c\}$ of ∂_{k+1} is called the space of k-boundaries of K. The boundary operators satisfy the following fundamental property:

$$\partial_{k-1} \circ \partial_k \equiv 0 \quad \text{for any} \quad k \geq 1.$$

In other words, any k-boundary is a k-cycle, that is, $B_k(K) \subseteq Z_k(K) \subseteq C_k(K)$. These notions are illustrated in **Figure 6**.

Definition 6 (simplicial homology group and Betti numbers)**.** *The* kth *(simplicial) homology group of* K *is the quotient vector space*

$$H_k(K) = Z_k(K)/B_k(K).$$

The kth *Betti number of* K *is the dimension* $\beta_k(K) = \dim H_k(K)$ *of the vector space* $H_k(K)$.

Figure 7 gives the Betti numbers of several simple spaces. Two cycles, $c, c' \in Z_k(K)$, are said to be homologous if they differ by a boundary, that is, if there exists a $(k+1)$-chain d such that $c' = c + \partial_{k+1}(d)$. Two such cycles give rise to the same element of H_k. In other words, the elements of $H_k(K)$ are the equivalence classes of homologous cycles.

[7]Recall that the symmetric difference of two sets A and B is the set $A \Delta B = (A \setminus B) \cup (B \setminus A)$.

[8]Notice that as we are considering coefficients in $\mathbb{Z}_2$, here $-1 = 1$ and thus $(-1)^i = 1$ for any i.

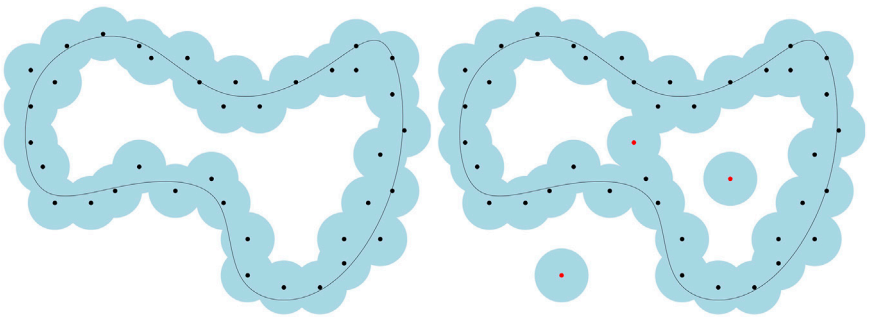


FIGURE 8 | Effect of outliers on the sublevel sets of distance functions. Adding just a few outliers to a point cloud may dramatically change its distance function and the topology of its offsets.

Simplicial homology groups and Betti numbers are topological invariants; if K, K' are two simplicial complexes whose geometric realizations are homotopy equivalent, then their homology groups are isomorphic and their Betti numbers are the same.

Singular homology is another notion of homology that allows us to consider larger classes of topological spaces. It is defined for any topological space X similarly to simplicial homology, except that the notion of the simplex is replaced by the notion of the singular simplex, which is just any continuous map $\sigma\colon \Delta_k \to X$ where Δ_k is the standard k-dimensional simplex. The space of k-chains is the vector space spanned by the k-dimensional singular simplices, and the boundary of a simplex σ is defined as the (alternated) sum of the restriction of σ to the $(k-1)$-dimensional faces of Δ_k. A remarkable fact about singular homology is that it coincides with simplicial homology whenever X is homeomorphic to the geometric realization of a simplicial complex. This allows us, in the sequel of this article, to indifferently talk about simplicial or singular homology for topological spaces and simplicial complexes.

Observing that if $f\colon X \to Y$ is a continuous map, then for any singular simplex $\sigma\colon \Delta_k \to X$ in X, $f\circ\sigma\colon \Delta_k \to Y$ is a singular simplex in Y, one easily deduces that continuous maps between topological spaces canonically induce homomorphisms between their homology groups. In particular, if f is a homeomorphism or a homotopy equivalence, then it induces an isomorphism between $H_k(X)$ and $H_k(Y)$ for any nonnegative integer k. As an example, it follows from the Nerve theorem that for any set of points $X \subset \mathbb{R}^d$ and any $r > 0$, the r-offset X^r and the Čech complex $Cech_r(X)$ have isomorphic homology groups and the same Betti numbers.

As a consequence, the Reconstruction theorem 2 leads to the following result on the estimation of Betti numbers.

Theorem 3. *Let $M \subset \mathbb{R}^d$ be a compact set such that $\mathrm{reach}_\alpha(d_M) \geq R > 0$ for some $\alpha \in (0,1)$ and let $\mathbb{X}$ be a finite set of points such that $d_H(M,\mathbb{X}) = \varepsilon < \frac{R}{5+4/\alpha^2}$. Then, for every $r \in [4\varepsilon/\alpha^2, R-3\varepsilon]$ and every $\eta \in (0,R)$, the Betti numbers of $Cech_r(\mathbb{X})$ are the same as the ones of M^η.*

In particular, if M is a smooth m-dimensional submanifold of $\mathbb{R}^d$, then $\beta_k(Cech_r(\mathbb{X})) = \beta_k(M)$ for any $k = 0, \ldots, m$.

From a practical perspective, this result raises three difficulties: first, the regularity assumption involving the α-reach of M may be too restrictive; second, the computation of the nerve of a union of balls requires the use of a tricky predicate testing the emptiness of a finite union of balls; third, the estimation of the Betti numbers relies on the scale parameter r, whose choice may be a problem.

To overcome these issues, Chazal and Oudot (2008) established the following result, which offers a solution to the first two problems.

Theorem 4. *Let $\mathbb{M} \subset \mathbb{R}^d$ be a compact set such that $\mathrm{wfs}(M) = \mathrm{wfs}_{d_M}(0) \geq R > 0$ and let $\mathbb{X}$ be a finite set of points such that $d_H(M,\mathbb{X}) = \varepsilon < \frac{1}{9}\mathrm{wfs}(M)$. Then for any $r \in [2\varepsilon, \frac{1}{4}(\mathrm{wfs}(M)-\varepsilon)]$ and any $\eta \in (0,R)$,*

$$\beta_k(X^\eta) = \mathrm{rk}\left(H_k\left(Rips_r(\mathbb{X})\right) \to H_k\left(Rips_{4r}(\mathbb{X})\right)\right)$$

where $rk(H_k(Rips_r(\mathbb{X})) \to H_k(Rips_{4r}(\mathbb{X})))$ denotes the rank of the homomorphism induced by the (continuous) canonical inclusion $Rips_r(\mathbb{X}) \hookrightarrow Rips_{4r}(\mathbb{X})$.

Although this result leaves the question of the choice of the scale parameter r open, it is proven in the study by Chazal and Oudot (2008) that a multiscale strategy whose description is beyond the scope of this article provides some help in identifying the relevant scales on which Theorem 4 can be applied.

4.3 Statistical Aspects of Homology Inference

According to the stability results presented in the previous section, a statistical approach to topological inference is strongly related to the problem of distribution support estimation and level sets estimation under the Hausdorff metric. A large number of methods and results are available for estimating the support of a distribution in statistics. For instance, the Devroye and Wise estimator (Devroye and Wise, 1980) defined on a sample $\mathbb{X}_n$ is also a particular offset of $\mathbb{X}_n$. The convergence rates of both $\mathbb{X}_n$ and the Devroye and Wise estimator to the support of the distribution for the Hausdorff distance were studied by Cuevas and Rodríguez-Casal (2004) in $\mathbb{R}^d$. More recently, the minimax rates of convergence of manifold estimation for the Hausdorff metric, which is particularly relevant for topological inference, has been studied by Genovese et al. (2012). There is also a large body of literature about level sets estimation in various metrics (see, for instance, Cadre, 2006; Polonik, 1995; Tsybakov, 1997) and, more particularly, for the Hausdorff metric Chen et al. (2017). All these works about

support and level sets estimation shed light on the statistical analysis of topological inference procedures.

In the study by Niyogi et al. (2008), it was shown that the homotopy type of Riemannian manifolds with a reach larger than a given constant can be recovered with high probability from offsets of a sample on (or close to) the manifold. This article was probably the first attempt to consider the topological inference problem in terms of probability. The result of the study by Niyogi et al. (2008) was derived from a retract contraction argument and was on tight bounds over the packing number of the manifold in order to control the Hausdorff distance between the manifold and the observed point cloud. The homology inference in the noisy case, in the sense that the distribution of the observation is concentrated around the manifold, was also studied by Niyogi et al. (2008), Niyogi et al. (2011). The assumption that the geometric object is a smooth Riemannian manifold is only used in the article to control in probability the Hausdorff distance between the sample and the manifold and is not actually necessary for the "topological part" of the result. Regarding the topological results, these are similar to those of the studies by Chazal et al. (2009d), Chazal and Lieutier (2008) in the particular framework of Riemannian manifolds. Starting from the result of the study by Niyogi et al. (2008), the minimax rates of convergence of the homology type have been studied by Balakrishna et al. (2012) under various models for Riemannian manifolds with a reach larger than a constant. In contrast, a statistical version of the work of Chazal et al. (2009d) has not yet been proposed.

More recently, following the ideas of Niyogi et al. (2008), Bobrowski et al. (2014) have proposed a robust homology estimator for the level sets of both density and regression functions, by considering the inclusion map between nested pairs of estimated level sets (in the spirit of Theorem 4 above) obtained using a plug-in approach from a kernel estimator.

4.4 Going Beyond Hausdorff Distance: Distance to Measure

It is well known that distance-based methods in TDA may fail completely in the presence of outliers. Indeed, adding even a single outlier to the point cloud can change the distance function dramatically (see **Figure 8** for an illustration). To answer this drawback, Chazal et al. (2011b) have introduced an alternative distance function which is robust to noise, the distance-to-measure.

Given a probability distribution P in $\mathbb{R}^d$ and a real parameter $0 \leq u \leq 1$, the notion of distance to the support of P may be generalized as the function

$$\delta_{P,u}\colon x \in \mathbb{R}^d \mapsto \inf\{t > 0\,;\, P(B(x,t)) \geq u\},$$

where $B(x, t)$ is the closed Euclidean ball of center x and radius t. To avoid issues due to discontinuities of the map $P \to \delta_{P,u}$, the distance-to-measure (DTM) function with parameter $m \in [0, 1]$ and power $r \geq 1$ is defined by

$$d_{P,m,r}(x)\colon x \in \mathbb{R}^d \mapsto \left(\frac{1}{m}\int_0^m \delta_{P,u}^r(x)\,du\right)^{1/r}. \tag{1}$$

A nice property of the DTM proved by Chazal et al. (2011b) is its stability with respect to perturbations of P in the Wasserstein metric. More precisely, the map $P \to d_{P,m,r}$ is $m^{-\frac{1}{r}}$-Lipschitz, that is, if P and $\tilde{P}$ are two probability distributions on $\mathbb{R}^d$, then

$$\|d_{P,m,r} - d_{\tilde{P},m,r}\|_\infty \leq m^{-\frac{1}{r}} W_r(P, \tilde{P}) \tag{2}$$

where W_r is the Wasserstein distance for the Euclidean metric on $\mathbb{R}^d$, with exponent r[9]. This property implies that the DTM associated with close distributions in the Wasserstein metric have close sublevel sets. Moreover, when $r = 2$, the function $d^2_{P,m,2}$ is semiconcave, ensuring strong regularity properties on the geometry of its sublevel sets. Using these properties, Chazal et al. (2011b) showed that under general assumptions, if $\tilde{P}$ is a probability distribution approximating P, then the sublevel sets of $d_{\tilde{P},m,2}$ provide a topologically correct approximation of the support of P.

In practice, the measure P is usually only known through a finite set of observations $\mathbb{X}_n = \{X_1, \ldots, X_n\}$ sampled from P, raising the question of the approximation of the DTM. A natural idea to estimate the DTM from $\mathbb{X}_n$ is to plug the empirical measure P_n instead of P into the definition of the DTM. This "plug-in strategy" corresponds to computing the distance to the empirical measure (DTEM). For $m = \frac{k}{n}$, the DTEM satisfies

$$d^r_{P_n,k/n,r}(x) := \frac{1}{k}\sum_{j=1}^{k} \|x - \mathbb{X}_n\|^r_{(j)},$$

where $\|x - \mathbb{X}_n\|_{(j)}$ denotes the distance between x and its jth neighbor in $\{X_1, \ldots, X_n\}$. This quantity can be easily computed in practice since it only requires the distances between x and the sample points. The convergence of the DTEM to the DTM has been studied by Chazal et al. (2017) and Chazal et al. (2016b).

The introduction of the DTM has motivated further works and applications in various directions such as topological data analysis (Buchet et al., 2015a), GPS trace analysis (Chazal et al., 2011a), density estimation (Biau et al., 2011), hypothesis testing Brécheteau (2019), and clustering (Chazal et al., 2013), just to name a few. Approximations, generalizations, and variants of the DTM have also been considered (Guibas et al., 2013; Phillips et al., 2014; Buchet et al., 2015b; Brécheteau and Levrard, 2020).

5 PERSISTENT HOMOLOGY

Persistent homology is a powerful tool used to efficiently compute, study, and encode multiscale topological features of nested families of simplicial complexes and topological spaces. It does not only provide efficient algorithms to compute the Betti numbers of each complex in the considered families, as

[9]See Villani (2003) for a definition of the Wasserstein distance

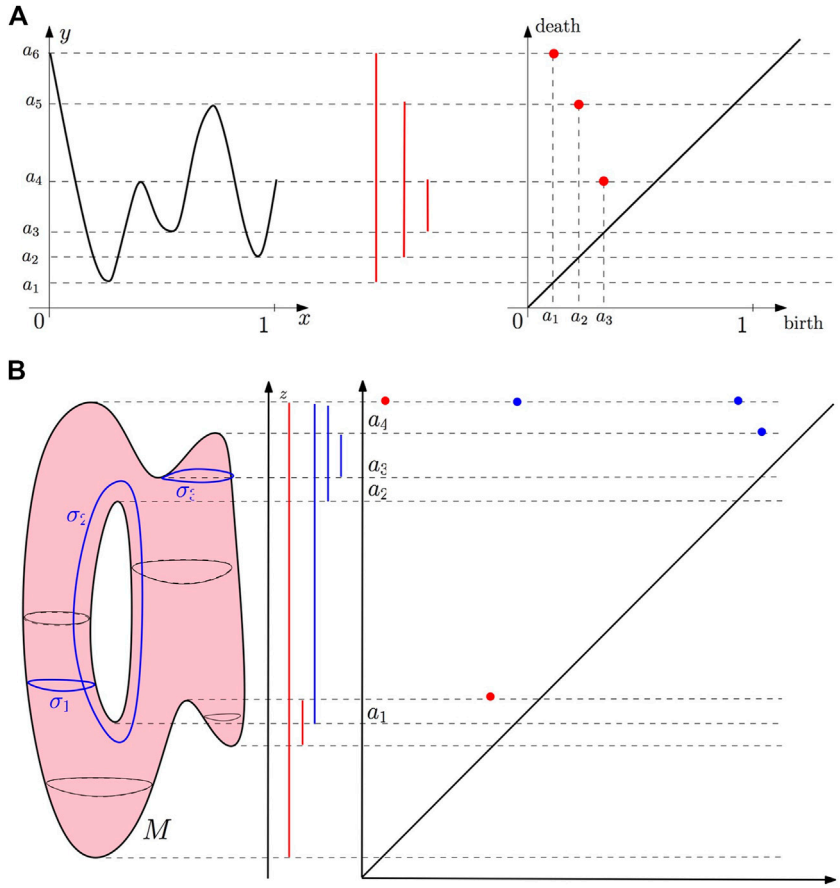


FIGURE 9 | (A) Example 1: the persistence barcode and the persistence diagram of a function f: $[0, 1] \to \mathbb{R}$. **(B)** Example 2: the persistence barcode and the persistence diagram of the height function (projection on the z-axis) defined on a surface in $\mathbb{R}^3$.

required for homology inference in the previous section, but also encodes the evolution of the homology groups of the nested complexes across the scales. Ideas and preliminary results underlying persistent homology theory can be traced back to the 20th century, in particular in the works of Barannikov (1994), Frosini (1992), Robins (1999). It started to know an important development in its modern form after the seminal works of Edelsbrunner et al. (2002) and Zomorodian and Carlsson (2005).

5.1 Filtrations

A filtration of a simplicial complex K is a nested family of subcomplexes $(K_r)_{r\in T}$, where $T \subseteq \mathbb{R}$, such that for any $r, r' \in T$, if $r \leq r'$ then $K_r \subseteq K_{r'}$ and $K = \cup_{r\in T} K_r$. The subset T may be either finite or infinite. More generally, a filtration of a topological space $\mathbb{M}$ is a nested family of subspaces $(M_r)_{r\in T}$, where $T \subseteq \mathbb{R}$, such that for any $r, r' \in T$, if $r \leq r'$ then $M_r \subseteq M_{r'}$ and $M = \cup_{r\in T} M_r$. For example, if f: $\mathbb{M} \to \mathbb{R}$ is a function, then the family $M_r = f^{-1}((-\infty, r])$, $r \in \mathbb{R}$ defines a filtration called the sublevel set filtration of f.

In practical situations, the parameter $r \in T$ can often be interpreted as a scale parameter, and filtrations classically used in TDA often belong to one of the two following families.

Filtrations Built on Top of Data

Given a subset $\mathbb{X}$ of a compact metric space (M, ρ), the families of Rips–Vietoris complexes $(Rips_r(\mathbb{X}))_{r\in\mathbb{R}}$ and Čech complexes $(Cech_r(\mathbb{X}))_{r\in\mathbb{R}}$ are filtrations[10]. Here, the parameter r can be interpreted as a resolution at which one considers the data set $\mathbb{X}$. For example, if $\mathbb{X}$ is a point cloud in $\mathbb{R}^d$, thanks to the Nerve theorem, the filtration $(Cech_r(\mathbb{X}))_{r\in\mathbb{R}}$ encodes the topology of the whole family of unions of balls $\mathbb{X}^r = \cup_{x\in\mathbb{X}} B(x, r)$, as r goes from 0 to $+\infty$. As the notion of filtration is quite flexible, many other filtrations have been considered in the literature and can be constructed on the top of data, such as the so-called witness complex popularized in TDA by De Silva and Carlsson (2004), the weighted Rips filtrations Buchet et al. (2015b), or the so-called DTM filtrations Anai et al. (2019) that allow us to handle data corrupted by noise and outliers.

Sublevel Sets Filtrations

Functions defined on the vertices of a simplicial complex give rise to another important example of filtration: let K be a simplicial complex

[10]We take here the convention that for r < 0, $Rips_r(\mathbb{X}) = Cech_r(\mathbb{X}) = \varnothing$

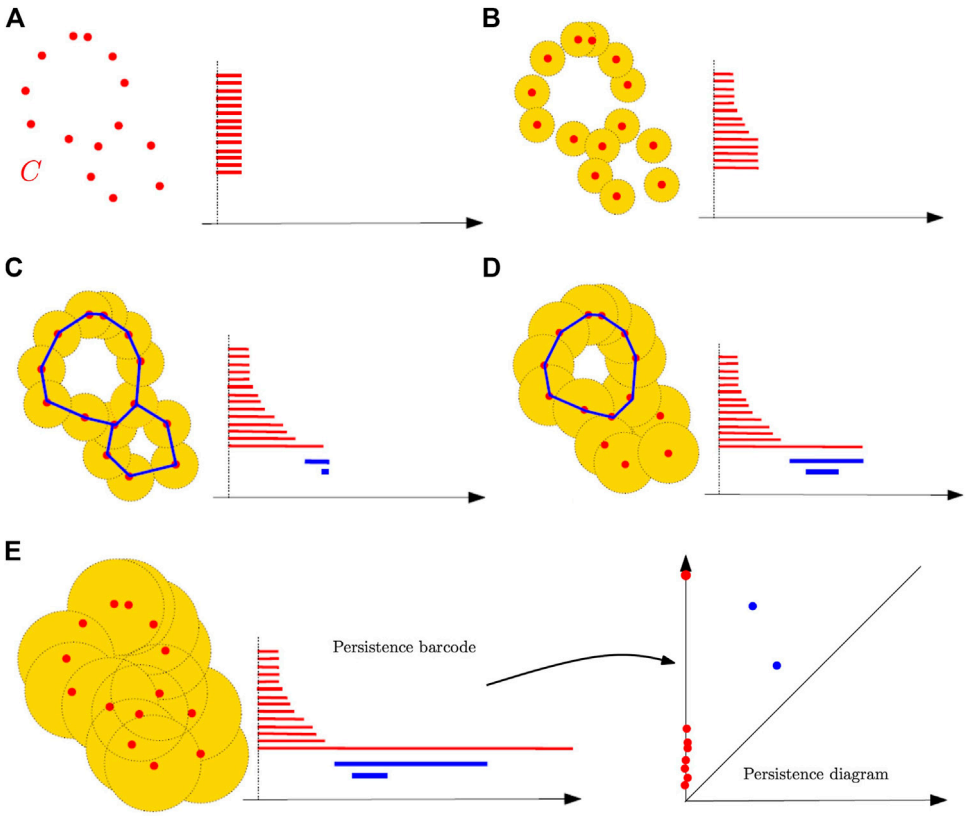


FIGURE 10 | The sublevel set filtration of the distance function to a point cloud and the construction of its persistence barcode as the radius of balls increases. The blue curves in the unions of balls represent one-cycles associated with the blue bars in the barcodes. The persistence diagram is finally defined from the persistence barcodes. **(A)** For the radius $r = 0$, the union of balls is reduced to the initial finite set of points, each of them corresponding to a zero-dimensional feature, that is, a connected component; an interval is created for the birth for each of these features at $r = 0$. **(B)** Some of the balls started to overlap, resulting in the death of some connected components that get merged together; the persistence diagram keeps track of these deaths, putting an end point to the corresponding intervals as they disappear. **(C)** New components have merged, giving rise to a single connected component and, so, all the intervals associated with a zero-dimensional feature have been ended, except the one corresponding to the remaining components; two new one-dimensional features have appeared, resulting in two new intervals (in blue) starting on their birth scale. **(D)** One of the two one-dimensional cycles has been filled, resulting in its death in the filtration and the end of the corresponding blue interval. **(E)** All the one-dimensional features have died; only the long (and never dying) red interval remains. As in the previous examples, the final barcode can also be equivalently represented as a persistence diagram where every interval (a, b) is represented by the point of coordinates (a, b) in $\mathbb{R}^2$. Intuitively, the longer an interval in the barcode or, equivalently, the farther from the diagonal the corresponding point in the diagram, the more persistent, and thus relevant, the corresponding homological feature across the filtration. Notice also that for a given radius r, the kth Betti number of the corresponding union of balls is equal to the number of persistence intervals corresponding to k-dimensional homological features and containing r. So, the persistence diagram can be seen as a multiscale topological signature encoding the homology of the union of balls for all radii as well as its evolution across the values of r.

with vertex set V and $f: V \to \mathbb{R}$. Then f can be extended to all simplices of K by $f([v_0, \ldots, v_k]) = \max\{f(v_i): i = 1, \ldots, k\}$ for any simplex $\sigma = [v_0, \ldots, v_k] \in K$ and the family of subcomplexes, $K_r = \{\sigma \in K: f(\sigma) \leq r\}$, defines a filtration called the sublevel set filtration of f. Similarly, one can define the upper-level set filtration of f.

In practice, even if the index set is infinite, all the considered filtrations are built on finite sets and are indeed finite. For example, when $\mathbb{X}$ is finite, the Vietoris–Rips complex $Rips_r(\mathbb{X})$ changes only at a finite number of indices, r. This allows us to easily handle them from an algorithmic perspective.

5.2 Starting With a Few Examples

Given a filtration $Filt = (F_r)_{r \in T}$ of a simplicial complex or a topological space, the homology of F_r changes as r increases; new connected components can appear, existing components can merge, loops and cavities can appear or be filled, etc. Persistent homology tracks these changes, identifies the appearing features, and associates a lifetime with them. The resulting information is encoded as a set of intervals called a barcode or, equivalently, as a multiset of points in $\mathbb{R}^2$ where the coordinate of each point is the starting and end point of the corresponding interval.

Before giving formal definitions, we introduce and illustrate persistent homology on a few simple examples.

Example 1

Let $f: [0, 1] \to \mathbb{R}$ be the function of **Figure 9A** and let $F_r = f^{-1}((-\infty, r))_{r \in \mathbb{R}}$ be the sublevel set filtration of f. All the sublevel sets of f are either empty or a union of intervals, so the only nontrivial topological information they carry is their zero-dimensional homology, that is, their number of connected components. For $r < a_1$, F_r is empty, but at $r = a_1$, a first connected component appears in F_{a_1}. Persistent homology thus registers a_1 as the birth time of a

connected component and starts to keep track of it by creating an interval starting at a_1. Then, F_r remains connected until r reaches the value a_2, where a second connected component appears. Persistent homology starts to keep track of this new connected component by creating a second interval starting at a_2. Similarly, when r reaches a_3, a new connected component appears and persistent homology creates a new interval starting at a_3. When r reaches a_4, the two connected components created at a_1 and a_3 merge together to give a single larger component. At this step, persistent homology follows the rule that it is the most recently appeared component in the filtration that dies; the interval started at a_3 is thus ended at a_4, and a first persistence interval encoding the life span of the component born at a_3 is created. When r reaches a_5, as in the previous case, the component born at a_2 dies, and the persistent interval (a_2, a_5) is created. The interval created at a_1 remains until the end of the filtration, giving rise to the persistent interval (a_1, a_6), if the filtration is stopped at a_6, or $(a_1, +\infty)$, if r goes to $+\infty$ (notice that in this latter case, the filtration remains constant for $r > a_6$). The obtained set of intervals encoding the life span of the different homological features encountered along the filtration is called the persistence barcode of f. Each interval (a, a') can be represented by the point of coordinates (a, a') in the $\mathbb{R}^2$ plane. The resulting set of points is called the persistence diagram of f. Notice that a function may have several copies of the same interval in its persistence barcode. As a consequence, the persistence diagram of f is indeed a multi-set where each point has an integer-valued multiplicity. Last, for technical reasons that will become clear in the next section, one adds to the persistence all the points of the diagonal $\Delta = \{(b, d): b = d\}$ with an infinite multiplicity.

Example 2

Let $f: M \to \mathbb{R}$ now be the function of **Figure 9B**, where M is a two-dimensional surface homeomorphic to a torus, and let $F_r = f^{-1}((-\infty, r))_{r\in\mathbb{R}}$ be the sublevel set filtration of f. The zero-dimensional persistent homology is computed as in the previous example, giving rise to the red bars in the barcode. Now, the sublevel sets also carry one-dimensional homological features. When r goes through the height a_1, the sublevel sets F_r that were homeomorphic to two discs become homeomorphic to the disjoint union of a disc and an annulus, creating a first cycle homologous to σ_1 in **Figure 9B**. An interval (in blue) representing the birth of this new one-cycle is thus started at a_1. Similarly, when r goes through the height a_2, a second cycle, homologous to σ_2, is created, giving rise to the start of a new persistent interval. These two created cycles are never filled (indeed, they span $H_1(M)$) and the corresponding intervals remain until the end of the filtration. When r reaches a_3, a new cycle is created that is filled and thus dies at a_4, giving rise to the persistence interval (a_3, a_4). So now, the sublevel set filtration of f gives rise to two barcodes, one for zero-dimensional homology (in red) and one for one-dimensional homology (in blue). As previously stated, these two barcodes can equivalently be represented as diagrams in the plane.

Example 3

In this last example, we consider the filtration given by a union of growing balls centered on the finite set of points C in **Figure 10**. Notice that this is the sublevel set filtration of the distance function to C, and thanks to the Nerve theorem, this filtration is homotopy equivalent to the Čech filtration built on the top of C. **Figure 10** shows several level sets of the filtration as follows:

a) For the radius $r = 0$, the union of balls is reduced to the initial finite set of points, each of them corresponding to a zero-dimensional feature, that is, a connected component; an interval is created for the birth for each of these features at $r = 0$.
b) Some of the balls started to overlap, resulting in the death of some connected components that get merged together; the persistence diagram keeps track of these deaths, putting an end point to the corresponding intervals as they disappear.
c) New components have merged, giving rise to a single connected component and, so, all the intervals associated with a zero-dimensional feature have been ended, except the one corresponding to the remaining components; two new one-dimensional features have appeared, resulting in two new intervals (in blue) starting on their birth scale.
d) One of the two one-dimensional cycles has been filled, resulting in its death in the filtration and the end of the corresponding blue interval.
e) All the one-dimensional features have died; only the long (and never dying) red interval remains. As in the previous examples, the final barcode can also be equivalently represented as a persistence diagram where every interval (a, b) is represented by the point of coordinates (a, b) in $\mathbb{R}^2$. Intuitively, the longer an interval in the barcode or, equivalently, the farther from the diagonal the corresponding point in the diagram, the more persistent, and thus relevant, the corresponding homological feature across the filtration. Notice also that for a given radius r, the kth Betti number of the corresponding union of balls is equal to the number of persistence intervals corresponding to k-dimensional homological features and containing r. So, the persistence diagram can be seen as a multiscale topological signature encoding the homology of the union of balls for all radii as well as its evolution across the values of r.

5.3 Persistent Modules and Persistence Diagrams

Persistent diagrams can be formally and rigorously defined in a purely algebraic way. This requires some care, and we only give the basic necessary notions here, leaving aside technical subtleties and difficulties. We refer the readers interested in a detailed exposition to Chazal et al. (2016a).

Let $Filt = (F_r)_{r\in T}$ be a filtration of a simplicial complex or a topological space. Given a nonnegative integer k and considering the homology groups $H_k(F_r)$, we obtain a sequence of vector spaces where the inclusions $F_r \subset F_{r'}$, $r \le r'$ induce linear maps between $H_k(F_r)$ and $H_k(F_{r'})$. Such a sequence of vector spaces together with the linear maps connecting them is called a persistence module.

Definition 7. *A persistence module $\mathbb{V}$ over a subset T of the real numbers $\mathbb{R}$ is an indexed family of vector spaces $(V_r | r \in T)$ and a doubly indexed family of linear maps $(v_s^r: V_r \to V_s \mid r \le s)$ which satisfy the composition law $v_t^s \circ v_s^r = v_t^r$ whenever $r \le s \le t$, and where v_r^r is the identity map on V_r.*

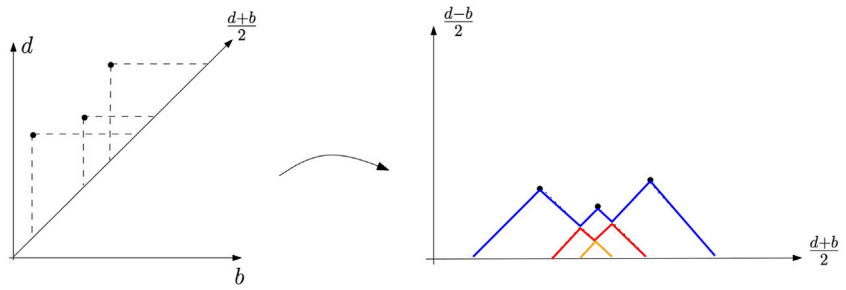


FIGURE 11 | Example of a persistence landscape (right) associated with a persistence diagram (left). The first landscape is in blue, the second one in red, and the last one in orange. All the other landscapes are zero.

In many cases, a persistence module can be decomposed into a direct sum of interval modules $\mathbb{I}_{(b,d)}$ of the form

$$\ldots, \to 0 \to \ldots, \to 0 \to \mathbb{Z}_2 \to \ldots, \to \mathbb{Z}_2 \to 0 \to \ldots$$

where the maps $\mathbb{Z}_2 \to \mathbb{Z}_2$ are identity maps while all the other maps are 0. Denoting b (resp. d), the infimum (resp. supremum) of the interval of indices corresponds to nonzero vector spaces; such a module can be interpreted as a feature that appears in the filtration at index b and disappears at index d. When a persistence module $\mathbb{V}$ can be decomposed as a direct sum of interval modules, one can show that this decomposition is unique up to reordering the intervals (see (Chazal et al., 2016a, Theorem 2.7)). As a consequence, the set of resulting intervals is independent of the decomposition of $\mathbb{V}$ and is called the persistence barcode of $\mathbb{V}$. As in the examples of the previous section, each interval (b, d) in the barcode can be represented as the point of coordinates (b, d) in the plane $\mathbb{R}^2$. The disjoint union of these points, together with the diagonal $\Delta = \{x = y\}$, is a multi-set called the persistence diagram of $\mathbb{V}$.

The following result, from (Chazal et al., 2016a, Theorem 2.8), gives some necessary conditions for a persistence module to be decomposable as a direct sum of interval modules.

Theorem 5. *Let $\mathbb{V}$ be a persistence module indexed by $T \subset \mathbb{R}$. If T is a finite set or if all the vector spaces V_r are finite-dimensional, then $\mathbb{V}$ is decomposable as a direct sum of interval modules. Moreover, for any s, $t \in T$, $s \leq t$, the number β_t^s of intervals starting before s and ending after t is equal to the rank of the linear map v_t^s and is called the (s, t)-persistent Betti number of the filtration.*

As both conditions above are satisfied for the persistent homology of filtrations of finite simplicial complexes, an immediate consequence of this result is that the persistence diagrams of such filtrations are always well defined.

Indeed, it is possible to show that persistence diagrams can be defined as soon as the following simple condition is satisfied.

Definition 8. *A persistence module $\mathbb{V}$ indexed by $T \subset \mathbb{R}$ is q-tame if for any $r < s$ in T, the rank of the linear map v_s^r: $V_r \to V_s$ is finite.*

Theorem 6 Chazal et al. (2009a), Chazal et al. (2016a). *If $\mathbb{V}$ is a q-tame persistence module, then it has a well-defined persistence diagram. Such a persistence diagram dgm($\mathbb{V}$) is the union of the points of the diagonal Δ of $\mathbb{R}^2$, counted with infinite multiplicity, and a multi-set above the diagonal in $\mathbb{R}^2$ that is locally finite. Here, by locally finite, we mean that for any rectangle R with sides parallel to the coordinate axes that does not intersect Δ, the number of points of dgm($\mathbb{V}$), counted with multiplicity, contained in R is finite. Also, the part of the diagram made of the points with the infinite second coordinate is called the essential part of the diagram.*

The construction of persistence diagrams of q-tame modules is beyond the scope of this article, but it gives rise to the same notion as in the case of decomposable modules. It can be done either by following the algebraic approach based upon the decomposability properties of modules or by adopting a measure theoretic approach that allows us to define diagrams as integer-valued measures on a space of rectangles in the plane. We refer the reader to Chazal et al. (2016a) for more information.

Although persistence modules encountered in practice are decomposable, the general framework of the q-tame persistence module plays a fundamental role in the mathematical and statistical analysis of persistent homology. In particular, it is needed to ensure the existence of limit diagrams when convergence properties are studied (see **Section 6**).

A filtration $Filt = (F_r)_{r \in T}$ of a simplicial complex or of a topological space is said to be tame if for any integer k, the persistence module $(H_k(F_r) | r \in T)$ is q-tame. Notice that the filtrations of finite simplicial complexes are always tame. As a consequence, for any integer k, a persistence diagram denoted $\mathrm{dgm}_k(\mathrm{Filt})$ is associated with the filtration Filt. When k is not explicitly specified and when there is no ambiguity, it is usual to drop the index k in the notation and to talk about "the" persistence diagram dgm(Filt) of the filtration Filt. This notation has to be understood as "dgm_k(Filt) for some k."

5.4 Persistence Landscapes

The persistence landscape introduced in the study by Bubenik (2015) is an alternative representation of persistence diagrams. This approach aims at representing the topological information encoded in persistence diagrams as elements of a Hilbert space, for which statistical learning methods can be directly applied. The persistence landscape is a collection of continuous, piecewise linear functions λ: $\mathbb{N} \times \mathbb{R} \to \mathbb{R}$ that summarizes a persistence diagram dgm.

A birth–death pair $p = (b, d) \in$ dgm is transformed into the point $(\frac{b+d}{2}, \frac{d-b}{2})$ (see **Figure 11**). Remember that the points with

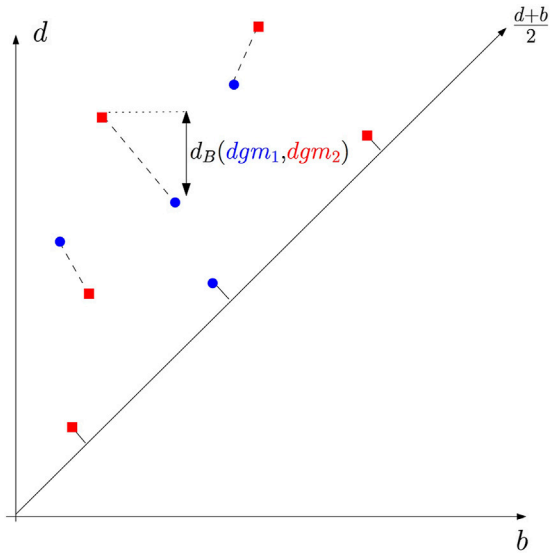


FIGURE 12 | Perfect matching and the bottleneck distance between a blue and a red diagram. Notice that some points of both diagrams are matched to points of the diagonal.

infinite persistence have been simply discarded in this definition. The landscape is then defined by considering the set of functions created by tenting the features of the rotated persistence diagram as follows:

$$\Lambda_p(t) = \begin{cases} t - b & t \in \left[b, \frac{b+d}{2}\right] \\ d - t & t \in \left(\frac{b+d}{2}, d\right] \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

The persistence landscape λ_{dgm} of dgm is a summary of the arrangement of piecewise linear curves obtained by overlaying the graphs of the functions $\{\Lambda_p\}_{p \in dgm}$. Formally, the persistence landscape of dgm is the collection of functions

$$\lambda_{dgm}(k, t) = \underset{r \in dgm}{\text{kmax}} \ \Lambda_r(t), \quad t \in [0, T], k \in \mathbb{N}, \tag{4}$$

where kmax is the kth largest value in the set; in particular, 1max is the usual maximum function. Given $k \in \mathbb{N}$, the function $\lambda_{dgm}(k, .)$: $\mathbb{R} \to \mathbb{R}$ is called the kth landscape of dgm. It is not difficult to see that the map that associates to each persistence diagram its corresponding landscape is injective. In other words, formally, no information is lost when a persistence diagram is represented through its persistence landscape.

The advantage of the persistence landscape representation is two-fold. First, persistence diagrams are mapped as elements of a functional space, opening the door to the use of a broad variety of statistical and data analysis tools for further processing of topological features see Bubenik (2015), Chazal et al. (2015c) and **Section 6.3.1**. Second, and fundamental from a theoretical perspective, the persistence landscapes share the same stability properties as those of persistence diagrams (see **Section 5.7**).

5.5 Linear Representations of Persistence Homology

A persistence diagram without its essential part can be represented as a discrete measure on $\Delta^+ = \{p = (b, d), b < d < \infty\}$. With a slight abuse of notation, we can write the following:

$$dgm = \sum_{p \in dgm} \delta_p,$$

where the features are counted with multiplicity and where $\delta_{(b,d)}$ denotes the Dirac measure in $p = (b, d)$. Most of the persistence-based descriptors that have been proposed to analyze persistence can be expressed as linear transformations of the persistence diagram, seen as a point process

$$\Psi(dgm) = \sum_{p \in dgm} f(p),$$

for some function f defined on Δ and taking values in a Banach space.

In most cases, we want these transformations to apply independently at each homological dimension. For $k \in \mathbb{N}$ a given homological dimension, we then consider some linear transformation of the persistence diagram, restricted to the topological features of dimension k as follows:

$$\Psi_k(dgm_k) = \sum_{p \in dgm_k} f_k(p), \tag{5}$$

where dgm_k is the persistence diagram of the topological features of dimension k and where f_k is defined on Δ and takes values in a Banach space.

Betti Curve

The simplest way to represent persistence homology is the Betti function or the Betti curve. The Betti curve of homological dimension k is defined as

$$\beta_k(t) = \sum_{(b,d) \in dgm} w(b, d) \mathbf{1}_{t \in [b,d]}$$

where w is a weight function defined on Δ. In other words, the Betti curve is the number of barcodes at time m. This descriptor is a linear representation of persistence homology by taking f in (5) such that $f(b, d)(t) = w(b, d)\mathbf{1}_{t \in [b,d]}$. A typical choice for the weigh function is an increasing function of the persistence $w(b, d) = \tilde{w}(d - b)$ where $\tilde{w}$ is an increasing function defined on $\mathbb{R}^+$. One of the first applications of Betti curves can be found in the study by Umeda (2017).

Persistence Surface

The persistence surface (also called persistence images) is obtained by making the convolution of a diagram with a kernel. It has been introduced in the study by Adams et al. (2017). For K: $\mathbb{R}^2 \to \mathbb{R}$, a kernel, and H, a 2×2 bandwidth matrix (e.g., a symmetric positive definite matrix), let for $u \in \mathbb{R}^2$

$$K_H(u) = \det(H)^{-1/2} K\left(H^{-1/2}u\right).$$

Let w: $\mathbb{R}^2 \to \mathbb{R}_+$ a weight function defined on Δ. One defines the persistence surface of homological dimension k associated with a diagram dgm, with kernel K and bandwidth matrix H by the following:

$$\forall u \in \mathbb{R}^2,\ \rho_k(dgm)(u) = \sum_{p \in dgm_k} w(r) K_H(u-p).$$

The persistence surface is obviously a linear representation of persistence homology. Typical weigh functions are increasing functions of the persistence.

Other Linear Representations of Persistence

Many other linear representations of persistence have been proposed in the literature, such as the persistence silhouette (Chazal et al., 2015b), the accumulated persistence function (Biscio and Møller, 2019), and variants of the persistence surface (Reininghaus et al., 2015; Kusano et al., 2016; Chen et al., 2017).

Considering persistence diagrams as discrete measures and their vectorizations as linear representation is an approach that has also proven fruitful to studying distributions of diagrams Divol and Chazal (2020) and the metric structure of the space of persistence diagrams Divol and Lacombe (2020) (see **Sections 5.6** and **Section 6.3**).

5.6 Metrics on the Space of Persistence Diagrams

To exploit the topological information and topological features inferred from persistent homology, one needs to be able to compare persistence diagrams, that is, to endow the space of persistence diagrams with a metric structure. Although several metrics can be considered, the most fundamental one is known as the bottleneck distance.

Recall that a persistence diagram is the union of a discrete multi-set in the half-plane above the diagonal Δ and, for technical reasons that will become clear below, of Δ where the point of Δ is counted with infinite multiplicity. A matching (see **Figure 12**) between two diagrams, dgm_1 and dgm_2, is a subset $m \subseteq \mathrm{dgm}_1 \times \mathrm{dgm}_2$ such that every point in $\mathrm{dgm}_1 \setminus \Delta$ and $\mathrm{dgm}_2 \setminus \Delta$ appears exactly once in m. In other words, for any $p \in \mathrm{dgm}_1 \setminus \Delta$ and for any $q \in \mathrm{dgm}_2 \setminus \Delta$, $(\{p\} \times \mathrm{dgm}_2) \cap m$ and $(\mathrm{dgm}_1 \times \{q\}) \cap m$ each contains a single pair. The bottleneck distance between dgm_1 and dgm_2 is then defined by

$$\mathrm{d_b}(dgm_1, dgm_2) = \inf_{\text{matching } m} \max_{(p,q)\in m} \|p-q\|_\infty.$$

The practical computation of the bottleneck distance boils down to the computation of a perfect matching in a bipartite graph for which classical algorithms can be used.

The bottleneck metric is an L_∞-like metric. It turns out to be the natural one to express stability properties of persistence diagrams presented in **Section 5.7**, but it suffers from the same drawbacks as the usual L_∞ norms, that is, it is completely determined by the largest distance among the pairs and does not take into account the closeness of the remaining pairs of points. A variant to overcome this issue, the so-called Wasserstein distance between diagrams, is sometimes considered. Given $p \geq 1$, it is defined by

$$W_p(dgm_1, dgm_2)^p = \inf_{\text{matching } m} \sum_{(p,q)\in m} \|p-q\|_\infty^p.$$

Useful stability results for persistence in the W_p metric exist among the literature, in particular the study by Cohen-Steiner et al. (2010), but they rely on assumptions that make them consequences of the stability results in the bottleneck metric. A general study of the space of persistence diagrams endowed with W_p metrics has been considered in the study by Divol and Lacombe (2020), where they proposed a general framework, based upon optimal partial transport, in which many important properties of persistence diagrams can be proven in a natural way.

5.7 Stability Properties of Persistence Diagrams

A fundamental property of persistence homology is that persistence diagrams of filtrations built on the top of data sets turn out to be very stable with respect to some perturbations of the data. To formalize and quantify such stability properties, we first need to be precise with regard to the notion of perturbation that is allowed.

Rather than working directly with filtrations built on the top of data sets, it turns out to be more convenient to define a notion of proximity between persistence modules, from which we will derive a general stability result for persistent homology. Then, most of the stability results for specific filtrations will appear as a consequence of this general theorem. To avoid technical discussions, from now on, we assume, without loss of generality, that the considered persistence modules are indexed by $\mathbb{R}$.

Definition 9. *Let $\mathbb{V}, \mathbb{W}$ be two persistence modules indexed by $\mathbb{R}$. Given $\delta \in \mathbb{R}$, a homomorphism of degree δ between $\mathbb{V}$ and $\mathbb{W}$ is a collection Φ of linear maps ϕ_r: $V_r \to W_{r+\delta}$, for all $r \in \mathbb{R}$ such that for any $r \leq s$, $\phi_s \circ v_s^r = w_{s+\delta}^{r+\delta} \circ \phi_r$.*

An important example of a homomorphism of degree δ is the shift endomorphism $1_{\mathbb{V}}^{\delta}$ which consists of the families of linear maps $(v_{r+\delta}^r)$. Notice also that homomorphisms of modules can naturally be composed; the composition of a homomorphism Ψ of degree δ between $\mathbb{U}$ and $\mathbb{V}$ and a homomorphism Φ of degree δ' between $\mathbb{V}$ and $\mathbb{W}$ naturally gives rise to a homomorphism $\Phi\Psi$ of degree $\delta + \delta'$ between $\mathbb{U}$ and $\mathbb{W}$.

Definition 10. *Let $\delta \geq 0$. Two persistence modules $\mathbb{V}, \mathbb{W}$ are δ-interleaved if there exist two homomorphisms of degree δ, Φ, from $\mathbb{V}$ to $\mathbb{W}$ and Ψ, from $\mathbb{W}$ to $\mathbb{V}$ such that $\Psi\Phi = 1_{\mathbb{V}}^{2\delta}$ and $\Phi\Psi = 1_{\mathbb{W}}^{2\delta}$.*

Although it does not define a metric on the space of persistence modules, the notion of closeness between two persistence modules may be defined as the smallest nonnegative δ such that they are δ-interleaved. Moreover, it allows us to formalize the following fundamental theorem (Chazal et al., 2009a; Chazal et al., 2016a).

Theorem 7 (Stability of persistence). *Let $\mathbb{V}$ and $\mathbb{W}$ be two q-tame persistence modules. If $\mathbb{V}$ and $\mathbb{W}$ are δ-interleaved for some $\delta \geq 0$, then*

$$d_b(dgm(\mathbb{V}), dgm(\mathbb{W})) \leq \delta.$$

Although purely algebraic and rather abstract, this result is an efficient tool to easily establish concrete stability results in TDA. For example, we can easily recover the first persistence stability result that appeared in the literature (Cohen-Steiner et al., 2005).

Theorem 8. *Let $f, g: M \to \mathbb{R}$ be two real-valued functions defined on a topological space M that are q-tame, that is, such that the sublevel set filtrations of f and g induce q-tame modules at the homology level. Then for any integer k,*

$$d_b(dgm_k(f), dgm_k(g)) \leq \|f - g\|_\infty = \sup_{x \in M} |f(x) - g(x)|$$

where $\mathrm{dgm}_k(f)$ (resp. $\mathrm{dgm}_k(g)$) is the persistence diagram of the persistence module $(H_k(f^{-1}(-\infty, r)) | r \in \mathbb{R})$ (resp. $(H_k(g^{-1}(-\infty, r)) | r \in \mathbb{R})$) where the linear maps are the one induced by the canonical inclusion maps between sublevel sets.

Proof. Denoting $\delta = \|f - g\|_\infty$, we have that for any $r \in \mathbb{R}$, $f^{-1}(-\infty, r) \subseteq g^{-1}(-\infty, r + \delta)$ and $g^{-1}(-\infty, r) \subseteq f^{-1}(-\infty, r + \delta)$. This interleaving between the sublevel sets of f induces a δ-interleaving between the persistence modules at the homology level, and the result follows from the direct application of Theorem 7.

Theorem 7 also implies a stability result for the persistence diagrams of filtrations built on the top of data.

Theorem 9. *Let $\mathbb{X}$ and $\mathbb{Y}$ be two compact metric spaces and let $Filt(\mathbb{X})$ and $Filt(\mathbb{Y})$ be the Vietoris–Rips of Čech filtrations built on the top of $\mathbb{X}$ and $\mathbb{Y}$. Then*

$$d_b(dgm(Filt(\mathbb{X})), dgm(Filt(\mathbb{Y}))) \leq 2 d_{GH}(\mathbb{X}, \mathbb{Y})$$

where $dgm(Filt(\mathbb{X}))$ and $dgm(Filt(\mathbb{Y}))$ denote the persistence diagram of the filtrations $Filt(\mathbb{X})$ and $Filt(\mathbb{X})$.

As we already noticed in Example 3 of **Section 5.2**, the persistence diagrams can be interpreted as multiscale topological features of $\mathbb{X}$ and $\mathbb{Y}$. In addition, Theorem 9 tells us that these features are robust with respect to perturbations of the data in the Gromov–Hausdorff metric. They can be used as discriminative features for classification or other tasks (see, for example, Chazal et al. (2009b) for an application to nonrigid 3D shape classification).

We now give similar results for the alternative persistence homology representations introduced before. From the definition of the persistence landscape, we immediately observe that $\lambda(k, \cdot)$ is one-Lipschitz, and thus, stability properties similar to those for persistence diagrams are satisfied for the landscapes.

Proposition 1 (stability of persistence landscapes; Bubenik (2015)). Let dgm and dgm' be two persistence diagrams (without their essential parts). For any $t \in \mathbb{R}$ and any $k \in \mathbb{N}$, we have the following:

(i) $\lambda(k, t) \geq \lambda(k + 1, t) \geq 0$.
(ii) $|\lambda(k, t) - \lambda'(k, t)| \leq d_b(dgm, dgm'))$.

A large class of linear representations is continuous with respect to the Wasserstein metric W_s in the space of persistence diagrams and with respect to the Banach norm of the linear representation of persistence. Generally speaking, it is not always possible to upper bound the modulus of continuity of the linear representation operator. However, in the case where s = 1, it is even possible to show a stability result if the weight function takes small values for points close to the diagonal (see Divol and Lacombe (2020), Hofer et al. (2019b)).

Stability Versus Discriminative Capacity of Persistence Representations

The results of the study by Divol and Lacombe (2020) showed that continuity and stability are only possible with weigh functions taking small values for points close to the diagonal. However, in general, there is no specific reason to consider that points close to the diagonal are less important than others, given a learning task. In a machine learning perspective, it is also relevant to design linear representation with general weigh functions, although it would be more difficult to prove the consistency of the corresponding methods without at least the continuity of the representation. Stability is thus important but maybe too strong a requirement for many problems in data sciences. Designing linear representation that is sensitive to specific parts of persistence diagrams rather than globally stable may reveal a good strategy in practice.

6 STATISTICAL ASPECTS OF PERSISTENT HOMOLOGY

Persistence homology by itself does not take into account the random nature of data and the intrinsic variability of the topological quantity they infer. We now present a statistical approach to persistent homology, in the sense that data are considered to be generated from an unknown distribution. We start with several consistency results for persistent homology inference.

6.1 Consistency Results for Persistent Homology

Assume that we observe n points $(X_1, \ldots, X_n)$ in a metric space (M, ρ) drawn i. i. d. from an unknown probability measure μ whose support is a compact set denoted $\mathbb{X}_\mu$. The Gromov–Hausdorff distance allows us to compare $\mathbb{X}_\mu$ with compact metric spaces not necessarily embedded in M. In the following, an estimator $\hat{\mathbb{X}}$ of $\mathbb{X}_\mu$ is a function of $X_1 \ldots, X_n$ that takes values in the set of compact metric spaces.

Let $Filt(\mathbb{X}_\mu)$ and $Filt(\hat{\mathbb{X}})$ be two filtrations defined on $\mathbb{X}_\mu$ and $\hat{\mathbb{X}}$. Starting from Theorem 9; a natural strategy for estimating the persistent homology of $Filt(\mathbb{X}_\mu)$ consists in estimating the support $\mathbb{X}_\mu$. Note that in some cases, the space M can be unknown and the observations $X_1 \ldots, X_n$ are then only known through their pairwise distances $\rho(X_i, X_j)$, i, j = 1, ..., n. The use of the Gromov–Hausdorff distance allows us to consider this set of observations as an abstract metric space of cardinality n,

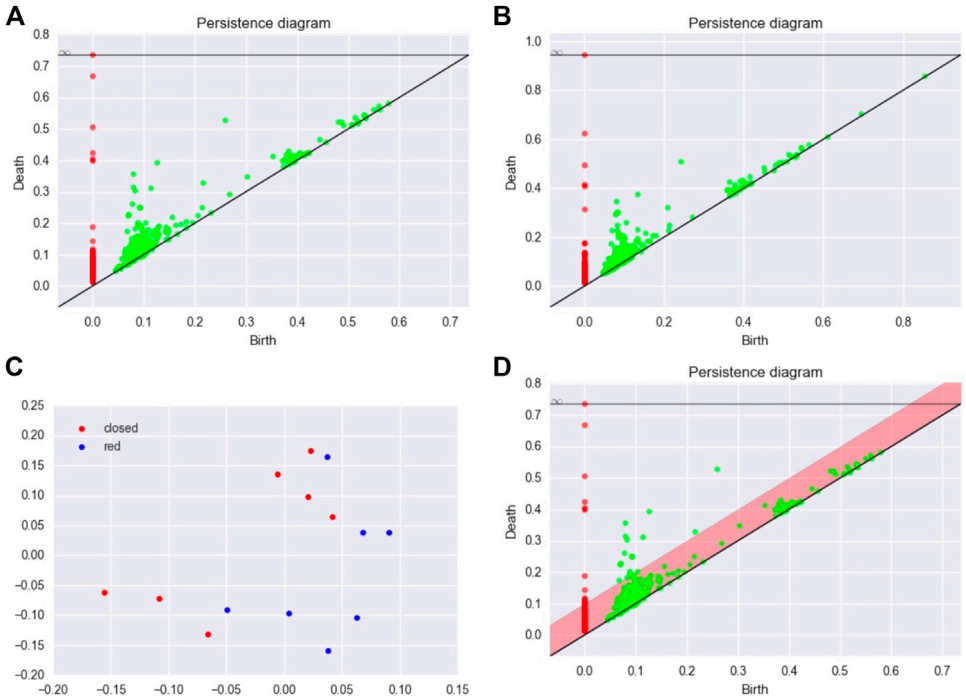


FIGURE 13 | (A,B) Two persistence diagrams for two configurations of MBP. **(C)** MDS configuration for the matrix of bottleneck distances. **(D)** Persistence diagram and confidence region for the persistence diagram of an MBP.

independently of the way it is embedded in M. This general framework includes the more standard approach consisting in estimating the support with respect to the Hausdorff distance by restraining the values of $\hat{\mathbb{X}}$ to the compact sets included in M.

The finite set $\mathbb{X}_n := \{X_1, \ldots, X_n\}$ is a natural estimator of the support $\mathbb{X}_\mu$. In several contexts discussed in the following, $\mathbb{X}_n$ shows optimal rates of convergence to $\mathbb{X}_\mu$ with respect to the Hausdorff distance. For some constants a, b > 0, we say that μ satisfies the (a, b)-standard assumption if for any $x \in \mathbb{X}_\mu$ and any r > 0,

$$\mu(B(x,r)) \geq \min\left(ar^b, 1\right). \tag{6}$$

This assumption has been widely used in the literature of set estimation under the Hausdorff distance (Cuevas and Rodríguez-Casal, 2004; Singh et al., 2009). Under this assumption, it can be easily derived that the rate of convergence of $dgm(Filt(\mathbb{X}_n))$ to $dgm(Filt(\mathbb{X}_\mu))$ for the bottleneck metric is upper bounded by $O(\frac{\log n}{n})^{1/b}$. More precisely, this rate upper bounds the minimax rate of convergence over the set of probability measures on the metric space (M, ρ) satisfying the (a, b)-standard assumption on M.

Theorem 10. *Chazal et al. (2014) For some positive constants a and b, let*

$$\mathcal{P} := \{\mu \text{ on } M \mid \mathbb{X}_\mu \text{ is compact and } \forall x \in \mathbb{X}_\mu, \forall r > 0, \mu(B(x,r)) \geq \min\left(1, ar^b\right)\}.$$

Then, it holds

$$\sup_{\mu \in \mathcal{P}} \mathbb{E}\left[\mathrm{d_b}\left(dgm\left(Filt\left(\mathbb{X}_\mu\right)\right), dgm\left(Filt\left(\mathbb{X}_n\right)\right)\right)\right] \leq C\left(\frac{\log n}{n}\right)^{1/b}$$

where the constant C only depends on a and b.

Under additional technical assumptions, the corresponding lower bound can be shown (up to a logarithmic term) (see Chazal et al. (2014)). By applying stability results, similar consistency results can be easily derived under alternative generative models as soon as a consistent estimator of the support under the Hausdorff metric is known. For instance, from the results of the study by Genovese et al. (2012) about Hausdorff support estimation under additive noise, it can be deduced that the minimax convergence rates for the persistence diagram estimation are faster than $(\log n)^{-1/2}$. Moreover, as soon as a stability result is available for some given representation of persistence, similar consistency results can be directly derived from the consistency for persistence diagrams.

Estimation of the Persistent Homology of Functions

Theorem 7 opens the door to the estimation of the persistent homology of functions defined on $\mathbb{R}^d$, on a submanifold of $\mathbb{R}^d$ or, more generally, on a metric space. The persistent homology of regression functions has also been studied by Bubenik et al. (2010). The alternative approach of Bobrowski et al. (2014), which was based on the inclusion map between nested pairs of estimated level sets, can be applied with kernel density and regression kernel estimators to estimate persistence homology of density functions and regression

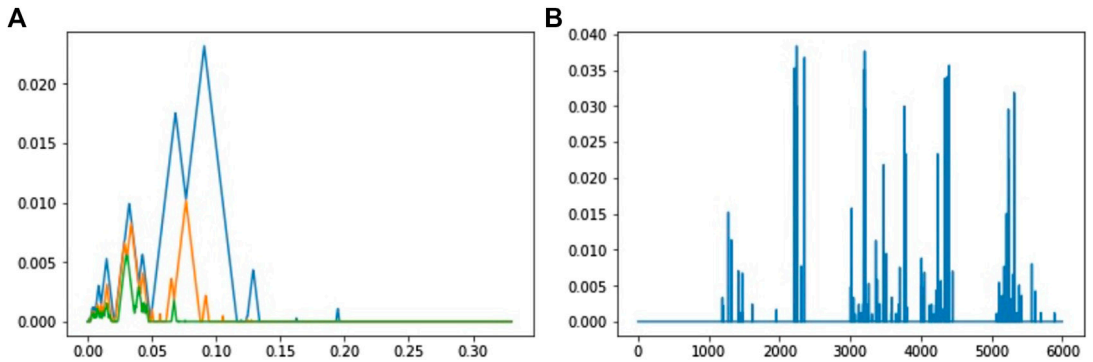


FIGURE 14 | (A) First three landscapes for zero-homology of the alpha shape filtration defined for a time series of acceleration of Walker A. **(B)** Variable importances of the landscape coefficients for the classification of walkers. The first 3,000 coefficients correspond to the three landscapes of dimension 0 and the last 3,000 coefficients to the three landscapes of dimension 1. There are 1,000 coefficients per landscape. Note that the first landscape of dimension 0 is always the same using the Rips complex (a trivial landscape), and consequently, the corresponding coefficients have a zero-importance value.

functions. Another direction of research on this topic concerns various versions of robust TDA. One solution is to study the persistent homology of the upper-level sets of density estimators (Fasy et al., 2014b). A different approach, more closely related to the distance function, but robust to noise, consists in studying the persistent homology of the sublevel sets of the distance to measure defined in **Section 4.4** (Chazal et al., 2017).

6.2 Statistic of Persistent Homology Computed on a Point Cloud

For many applications, in particular when the support of the point cloud is not drawn on or close to a geometric shape, persistence diagrams can be quite complex to analyze. In particular, many topological features are closed to the diagonal. Since they correspond to topological structures that die very soon after they appear in the filtration, these points are generally considered as noise (see **Figure 13** for an illustration). Confidence regions of persistence diagrams are rigorous answers to the problem of distinguishing between the signal and the noise in these representations.

The stability results given in **Section 5.7** motivate the use of the bottleneck distance to define confidence regions. However, alternative distances in the spirit of Wasserstein distances can be proposed too. When estimating a persistence diagram dgm with an estimator $\widehat{dgm}$, we typically look for some value η_α such that

$$P\left(\mathrm{d_b}\left(\widehat{dgm}, dgm\right) \geq \eta_\alpha\right) \leq \alpha,$$

for $\alpha \in (0, 1)$. Let B_α be the closed ball of radius α for the bottleneck distance, centered at $\widehat{dgm}$ in the space of persistence diagrams. Following Fasy et al. (2014b), we can visualize the signatures of the points belonging to this ball in various ways. One first option is to center a box of a side length of 2α at each point of the persistence diagram $\widehat{dgm}$. An alternative solution is to visualize the confidence set by adding a band at (vertical) distance $\eta_\alpha/2$ from the diagonal (the bottleneck distance being defined for the ℓ_∞ norm) (see **Figure 13** for an illustration). The points outside the band are then considered as significant topological features (see Fasy et al. (2014b) for more details).

Several methods have been proposed in the study by Fasy et al. (2014b) to estimate η_α in different frameworks. These methods mainly rely on stability results for persistence diagrams; confidence sets for diagrams can be derived from confidence sets in the sample space.

Subsampling Approach

This method is based on a confidence region for the support K of the distribution of the sample in the Hausdorff distance. Let $\tilde{\mathbb{X}}_b$ be a subsample of size b drawn from the sample $\tilde{\mathbb{X}}_n$, where $b = o(n/\log n)$. Let $q_b(1-\alpha)$ be the quantile of the distribution of $Haus(\tilde{\mathbb{X}}_b, \mathbb{X}_n)$. Take $\hat{\eta}_\alpha := 2\hat{q}_b(1-\alpha)$, where $\hat{q}_b$ is an estimation $q_b(1-\alpha)$ using a standard Monte Carlo procedure. Under a (a, b) standard assumption and for an n large enough, Fasy et al. (2014b) showed that

$$P\left(\mathrm{d_b}\left(dgm\left(Filt\left(K\right)\right), dgm\left(Filt\left(\mathbb{X}_n\right)\right)\right) > \hat{\eta}_\alpha\right)$$

$$\leq P\left(Haus\left(K, \mathbb{X}_n\right) > \hat{\eta}_\alpha\right) \leq \alpha + O\left(\frac{b}{n}\right)^{1/4}.$$

Bottleneck Bootstrap

The stability results often lead to conservative confidence sets. An alternative strategy is the bottleneck bootstrap introduced in the study by Chazal et al. (2016b). We consider the general setting where a persistence diagram $\widehat{dgm}$ is defined from the observation $(X_1, \ldots, X_n)$ in a metric space. This persistence diagram corresponds to the estimation of an underlying persistence diagram dgm, which can be related, for instance, to the support of the measure, or to the sublevel sets of a function related to this distribution (for instance, a density function when the X_i's are in $\mathbb{R}^d$). Let $(X_1^*, \ldots, X_n^*)$ be a sample from the empirical measure defined from the observations $(X_1, \ldots, X_n)$. Let also $\widehat{dgm}^*$ be the persistence diagram derived from this sample. We can then take for η_α the quantity $\hat{\eta}_\alpha$ defined by

$$P\Big(\mathrm{d_b}\Big(\widehat{dgm}^{*}, \widehat{dgm}\Big) > \hat{\eta}_\alpha \mid X_1, \ldots, X_n\Big) = \alpha. \tag{7}$$

Note that $\hat{\eta}_\alpha$ can be easily estimated using Monte Carlo procedures. It has been shown in the study by Chazal et al. (2016b) that the bottleneck bootstrap is valid when computing the sublevel sets of a density estimator.

Bootstrapping Persistent Betti Numbers

As already mentioned, confidence regions based on stability properties of persistence may lead to very conservative confidence regions. Based on the concepts of stabilizing statistics Penrose and Yukich (2001), asymptotic normality for persistent Betti numbers has been shown recently by Krebs and Polonik (2019) and Roycraft et al. (2020) under very mild conditions on the filtration and the distribution of the sample cloud. In addition, bootstrap procedures are also shown to be valid in this framework. More precisely, a smoothed bootstrap procedure together with a convenient rescaling of the point cloud seems to be a promising approach for boostrapping TDA features from point cloud data.

6.3 Statistic for a Family of Persistent Diagrams or Other Representations

Up to now in this section, we were only considering statistics based on one single observed persistence diagram. We now consider a new framework where several persistence diagrams (or other representations) are available, and we are interested in providing the central tendency, confidence regions, and hypothesis tests for topological descriptors built on this family.

6.3.1 Central Tendency for Persistent Homology

Mean and Expectations of Distributions of Diagrams

The space of persistence diagrams being a general metric space but not a Hilbert space, the definition of a mean persistence diagram is not obvious and unique. One first natural approach to defining a central tendency in this context is to consider Fréchet means of distributions of diagrams. Their existence has been proven in the study by Mileyko et al. (2011), and they have also been characterized in the study by Turner et al. (2014a). However, they may not be unique, and they turn out to be difficult to compute in practice. To partly overcome these problems, different approaches have been recently proposed based on numerical optimal transport Lacombe et al. (2018) or linear representations and kernel-based methods Divol and Chazal (2020).

Topological Signatures From Subsamples

Central tendency properties of persistent homology can also be used to compute topological signatures for very large data sets, as an alternative approach to overcome the prohibitive cost of persistence computations. Given a large point cloud, the idea is to extract many subsamples, to compute the persistence landscape for each subsample, and then to combine the information.

For any positive integer m, let $X = \{x_1, \ldots, x_m\}$ be a sample of m points drawn from a measure μ in a metric space M and which support is denoted by $\mathbb{X}_\mu$. We assume that the diameter of $\mathbb{X}_\mu$ is finite and upper bounded by $\frac{T}{2}$, where T is the same constant as in the definition of persistence landscapes in **Section 5.4**. For ease of exposition, we focus on the case k = 1 and the set $\lambda(t) = \lambda(1, t)$. However, the results we present in this section hold for k > 1. The corresponding persistence landscape (associated with the persistence diagram of the Čech or Rips–Vietoris filtration) is λ_X and we denote by Ψ_μ^m the measure induced by $\mu^{\otimes m}$ on the space of persistence landscapes. Note that the persistence landscape λ_X can be seen as a single draw from the measure Ψ_μ^m. The point-wise expectations of the (random) persistence landscape under this measure is defined by $\mathbb{E}_{\Psi_\mu^m}[\lambda_X(t)], t \in [0, T]$. The average landscape $\mathbb{E}_{\Psi_\mu^m}[\lambda_X]$ has a natural empirical counterpart, which can be used as its unbiased estimator. Let $S_1^m, \ldots, S_\ell^m$ be ℓ independent samples of size m from $\mu^{\otimes m}$. We define the empirical average landscape as

$$\overline{\lambda_\ell^m}(t) = \frac{1}{b}\sum_{i=1}^{b} \lambda_{S_i^m}(t), \quad \text{for all } t \in [0, T], \tag{8}$$

and propose to use $\overline{\lambda_\ell^m}$ to estimate $\lambda_{\mathbb{X}_\mu}$. Note that computing the persistent homology of $\mathbb{X}_n$ is O(exp(n)), whereas computing the average landscape is O(b exp(m)).

Another motivation for this subsampling approach is that it can also be applied when μ is a discrete measure with the support $\mathbb{X}_N = \{x_1, \ldots, x_N\}$ lying in a metric space M. This framework can be very common in practice, when a continuous (but unknown) measure is approximated by a discrete uniform measure μ_N on $\mathbb{X}_N$.

The average landscape $\mathbb{E}_{\Psi_\mu^m}[\lambda_X]$ is an interesting quantity on its own, since it carries some stable topological information about the underlying measure μ, from which the data are generated.

Theorem 11. *[Chazal et al. (2015a)] Let $X \sim \mu^{\otimes m}$ and $Y \sim \nu^{\otimes m}$, where μ and ν are two probability measures on M. For any $p \geq 1$, we have*

$$\left\| \mathbb{E}_{\Psi_\mu^m}[\lambda_X] - \mathbb{E}_{\Psi_\nu^m}[\lambda_Y] \right\|_\infty \leq 2\, m^{\frac{1}{p}} W_p(\mu, \nu),$$

where W_p is the pth Wasserstein distance on M.

The result of Theorem 11 is useful for two reasons. First, it tells us that for a fixed m, the expected "topological behavior" of a set of m points carries some stable information about the underlying measure from which the data are generated. Second, it provides a lower bound for the Wasserstein distance between two measures, based on the topological signature of samples of m points.

6.3.2 Asymptotic Normality

As in the previous section, we consider several persistence diagrams (or other representations). The next step after giving central tendency descriptors of persistence homology is to provide asymptotic normality results for these quantities together with bootstrap procedures to derive confidence regions. It is of course easier to show such results for functional representations of persistence. In the studies by Chazal et al. (2015b), Chazal et al. (2015c), following this strategy, confidence bands for landscapes are proposed from

the observation of landscapes $\lambda_1, \ldots, \lambda_N$ drawn i. i. d. from a random distribution in the space of landscapes. The asymptotic validity and the uniform convergence of the multiplier bootstrap is shown in this framework. Note that similar results can also be proposed for many representations of persistence, in particular by showing that the corresponding functional spaces are Donsker spaces.

6.4 Other Statistical Approaches to Topological Data Analysis

Statistical approaches for TDA are seeing an increasing interest and many others have been proposed in recent years or are still subject to active research activities, as illustrated in the following non-exhaustive list of examples.

Hypothesis Testing

Several methods have been proposed for hypothesis testing procedures for persistent homology, mostly based on permutation strategies and for two-sample testing. Robinson and Turner (2017) focused on pairwise distances of persistence diagrams, whereas Berry et al. (2020) studied more general functional summaries. Hypothesis tests based on kernel approaches have been proposed in the study by Kusano (2019). A two-stage hypothesis test of filtering and testing for persistent images was also presented in the study by Moon and Lazar (2020).

Persistence Homology Transform

The representations introduced before are all transformations derived from the persistence diagram computed from a fixed filtration built over a data set. The persistence homology transform introduced in the studies by Curry et al. (2018), Turner et al. (2014b) to study shapes in $\mathbb{R}^d$ takes a different path by looking at the persistence homology of the sublevel set filtration induced by the projection of the considered shape in each direction in $\mathbb{R}^d$. It comes with several interesting properties; in particular, the persistence homology transform is a sufficient statistic for distributions defined on the set of geometric and finite simplicial complexes embedded in $\mathbb{R}^d$.

Bayesian Statistics for Topological Data Analysis

A Bayesian approach to persistence diagram inference has been proposed in the study by Maroulas et al. (2020) by viewing a persistence diagram as a sample from a point process. This Bayesian method computes the point process posterior intensity based on a Gaussian mixture intensity for the prior.

6.5 Persistent Homology and Machine Learning

Using TDA and, more specifically, persistent homology for machine learning is a subject that attracts a lot of information and generated an intense research activity. Although the recent progress in this area goes far beyond the scope of this article, we briefly introduce the main research directions with a few references to help the newcomer to the field to get started.

Topological Data Analysis for Exploratory Data Analysis and Descriptive Statistics

In some domains, TDA can be fruitfully used as a tool for exploratory analysis and visualization. For example, the Mapper algorithm provides a powerful approach to exploring and visualizing the global topological structure of complex data sets. In some cases, persistence diagrams obtained from data can be directly interpreted and exploited for better understanding of the phenomena from which the data have been generated. This is, for example, the case in the study of force fields in granular media (Kramar et al., 2013) or of atomic structures in glass (Nakamura et al., 2015) in material science, in the study of the evolution of convection patterns in fluid dynamics (Kramár et al., 2016), and in machining monitoring (Khasawneh and Munch, 2016) or in the analysis of nanoporous structures in chemistry (Lee et al., 2017) where topological features can be rather clearly related to specific geometric structures and patterns in the considered data.

Persistent Homology for Feature Engineering

There are many other cases where persistence features cannot be easily or directly interpreted but present valuable information for further processing. However, the highly nonlinear nature of diagrams prevents them from being immediately used as standard features in machine learning algorithms.

Persistence landscapes and linear representations of persistence diagrams offer a first option to convert persistence diagrams into elements of a vector space that can be directly used as features in classical machine learning pipelines. This approach has been used, for example, for protein binding (Kovacev-Nikolic et al., 2016), object recognition (Li et al., 2014), or time series analysis. In the same vein, the construction of kernels for persistence diagrams that preserve their stability properties has recently attracted some attention. Most of them have been obtained by considering diagrams as discrete measures in $\mathbb{R}^2$. Convolving a symmetrized (with respect to the diagonal) version of persistence diagrams with a 2D Gaussian distribution, Reininghaus et al. (2015) introduced a multiscale kernel and applied it to shape classification and texture recognition problems. Considering the Wasserstein distance between projections of persistence diagrams on lines, Carriere et al. (2017) built another kernel and tested its performance on several benchmarks. Other kernels, still obtained by considering persistence diagrams as measures, have also been proposed in the study by Kusano et al. (2017).

Various other vector summaries of persistence diagrams have been proposed and then used as features for different problems. For example, basic summaries were considered in the study by Bonis et al. (2016) and combined with quantization and pooling methods to address nonrigid shape analysis problems; Betti curves extracted from persistence diagrams were used with one-dimensional convolutional neural networks (CNNs) to analyze time-dependent data and recognize human activities from inertial sensors in the studies by Dindin et al. (2020), Umeda (2017); persistence images were introduced in the study by Adams et al. (2017) and were considered to address some inverse problems using linear machine learning models in the study by Obayashi et al. (2018).

The kernels and vector summaries of persistence diagrams mentioned above are built independently of the considered data analysis or learning task. Moreover, it appears that in many cases, the relevant topological information is not carried by the whole persistence diagram but is concentrated in some localized regions that may not be obvious to identify. This usually makes the choice of a relevant kernel or vector summary very difficult for the user. To overcome this issue, various authors have proposed learning approaches that allow us to learn the relevant topological features for a given task. In this direction, Hofer et al. (2017) proposed a deep learning approach to learn the parameters of persistence image representations of persistence diagrams, while Kim et al. (2020) introduced a neural network layer for persistence landscapes. In the study by Carrière et al. (2020a), the authors introduced a general neural network layer for persistence diagrams that can be either used to learn an appropriate vectorization or directly integrated in a deep neural network architecture. Other methods, inspired from *k*-means, propose unsupervised methods to vectorize persistence diagrams (Royer et al., 2021; Zieliński et al., 2010), some of them coming with theoretical guarantees (Chazal et al., 2020).

Persistent Homology for Machine Learning Architecture Optimization and Model Selection

More recently, TDA has seen new developments in machine learning where persistent homology is no longer used for feature engineering but as a tool to design, improve, or select models (see Carlsson and Gabrielsson (2020), Chen et al. (2019), Gabrielsson and Carlsson (2019), Hofer et al. (2019a), Moor et al. (2020), Ramamurthy et al. (2019), Rieck et al. (2019)). Many of these tools rely on the introduction of loss or regularization functions depending on persistent homology features, raising the problem of their optimization. Building on the powerful tools provided by software libraries such as PyTorch or TensorFlow, practical methods allowing us to encode and optimize a large family of persistence-based functions have been proposed and experimented on (Poulenard et al., 2018; Gabrielsson et al., 2020). A general framework for persistence-based function optimization based on stochastic subgradient descent algorithms with convergence guarantees has been recently proposed and implemented in an easy-to-use software tool (Carriere et al., 2020b). With a different perspective, another theoretical framework to study the differentiable structure of functions of persistence diagrams has been proposed in the study by Leygonie et al. (2021).

7 TOPOLOGICAL DATA ANALYSIS FOR DATA SCIENCES WITH THE GUDHI LIBRARY

In this section, we illustrate TDA methods using the Python library GUDHI[11] (Maria et al., 2014) together with popular libraries such as NumPy (Walt et al., 2011), scikit-learn (Pedregosa et al., 2011), and pandas (McKinney, 2010). This section aims at demonstrating that the topological signatures of TDA can be easily computed and exploited using GUDHI. More illustrations with Python notebooks can be found in the tutorial GitHub[12] of GUDHI.

7.1 Bootstrap and Comparison of Protein Binding Configurations

This example is borrowed from Kovacev-Nikolic et al. (2016). In this article, persistent homology is used to analyze protein binding, and more precisely, it compares closed and open forms of the maltose-binding protein (MBP), a large biomolecule consisting of 370 amino acid residues. The analysis is not based on geometric distances in $\mathbb{R}^3$ but on a metric of dynamical distances defined by

$$D_{ij} = 1 - |C_{ij}|,$$

where C is the correlation matrices between residues. The data can be downloaded at this link[13].

```
import numpy as np
import gudhi as gd
import pandas as pd
import seaborn as sns

corr_protein=pd.read_csv("mypath/1anf.corr_1.
txt", header=None, delim_whitespace=True)
dist_protein_1 = 1- np.abs(corr_protein_1.values)
rips_complex_1 = gd.RipsComplex(distance_
matrix=dist_protein_1, max_edge_length=1.1)
simplex_tree_1 = rips_complex_1.create_simplex_
tree(max_dimension=2)
diag_1 = simplex_tree_1.persistence()
gd.plot_persistence_diagram(diag_1)
```

For comparing persistence diagrams, we use the bottleneck distance. The block of statements given below computes persistence intervals and computes the bottleneck distance for zero-homology and one-homology as follows:

```
interv0_1 = simplex_tree_1.persistence_
intervals_in_dimension(0)
interv0_2 = simplex_tree_2.persistence_
intervals_in_dimension(0)
bot0 = gd.bottleneck_distance(interv0_
1,interv0_2)

interv1_1 = simplex_tree_1.persistence_
intervals_in_dimension(1)
interv1_2 = simplex_tree_2.persistence_
intervals_in_dimension(1)
bot1 = gd.bottleneck_distance(interv1_1,
interv1_2)
```

[11]http://gudhi.gforge.inria.fr/python/latest/

[12]https://github.com/GUDHI/TDA-tutorial

[13]https://www.researchgate.net/publication/301543862_corr

In this way, we can compute the matrix of bottleneck distances between the fourteen MBPs. Finally, we apply a multidimensional scaling method to find a configuration in $\mathbb{R}^2$ which almost matches with the bottleneck distances (see **Figure 13C**). We use the scikit-learn library for the MDS as follows:

```
import matplotlib.pyplot as plt
from sklearn import manifold

mds = manifold.MDS(n_components=2,
dissimilarity="precomputed")
config = mds.fit(M).embedding_

plt.scatter(config [0:7,0], config [0:7, 1],
color='red', label="closed")
plt.scatter(config [7:1,0], config [7:1, 1],
color='blue', label="red")
plt.legend(loc=1)
```

We now define a confidence band for a diagram using the bottleneck bootstrap approach. We resample over the lines (and columns) of the matrix of distances, and we compute the bottleneck distance between the original persistence diagram and the bootstrapped persistence diagram. We repeat the procedure many times, and finally, we estimate the quantile 95% of this collection of bottleneck distances. We take the value of the quantile to define a confidence band on the original diagram (see **Figure 13D**). However, such a procedure should be considered with caution because as far as we know, the validity of the bottleneck bootstrap has not been proven in this framework.

7.2 Classification for Sensor Data

In this experiment, the 3D acceleration of 3 walkers (A, B, and C) has been recorded using the sensor of a smartphone[14]. Persistence homology is not sensitive to the choice of axes, and so no preprocessing is necessary to align the 3 time series according to the same axis. From these three time series, we have picked, at random, sequences of 8 s in the complete time series, that is, 200 consecutive points of acceleration in $\mathbb{R}^3$. For each walker, we extract 100 time series in this way. The next block of statements computes the persistence for the alpha complex filtration for data_A_sample, one of the 100 time series of acceleration of Walker A.

```
alpha_complex_sample = gd.AlphaComplex
(points = data_A_sample)
simplex_tree_sample = alpha_complex_sample.
create_simplex_tree(max_alpha_square=0.3)
diag_Alpha = simplex_tree_sample.persistence()
```

From diag_Alpha, we can then easily compute and plot the persistence landscapes (see **Figure 14A**). For all 300 time series, we compute the persistence landscapes for dimensions 0 and 1, and we compute the first three landscapes for the 2 dimensions. Moreover, each persistence landscape is discretized on 1,000 points. Each time series is thus described by 6,000 topological variables. To predict the walker from these features, we use a random forest (Breiman, 2001), which is known to be efficient in such a high-dimensional setting. We split the data into train and test samples at random several times. We finally obtain an averaged classification error of around 0.95. We can also visualize the most important variables in the random forest (see **Figure 14B**).

[14]The dataset can be downloaded at the link http://bertrand.michel.perso.math.cnrs.fr/Enseignements/TDA/data_acc

8 DISCUSSION

In this introductory article, we propose an overview of the most standard methods in the field of topological data analysis. We also provide a presentation of the mathematical foundations of TDA, on the topological, algebraic, geometric, and statistical aspects. The robustness of TDA methods (coordinate invariance and deformation invariance) and the compressed representation of data they offer make their use very interesting for data analysis, machine learning, and explainable AI. Many applications have been proposed in this direction during the last few years. Finally, TDA constitutes an additional possible approach in the data scientist toolbox.

Of course, TDA is suited to address all kinds of problems. Practitioners may face several potential issues when applying TDA methods. On the algorithmic aspects, computing persistence homology can be time and resource consuming. Even if there is still room for improvement, recent computational advances have enabled TDA to be an effective method for data science, thanks to libraries like GUDHI, for example. Moreover, combing TDA using quantization methods, graph simplification, or dimension reduction methods may reduce the computational cost of the TDA algorithms. Another potential problem we can face with TDA is that returning to the data point to interpret the topological signatures can be tricky because these signatures correspond to classes of equivalence of cycles. This can be a problem when there is a need to identify which part of the point cloud "has created" a given topological signature. TDA is in fact more suited to solving data science problems dealing with a family of point clouds, each data point being described by its persistent homology. Finally, the topological and geometric information that can be extracted from the data is not always efficient for solving a given problem in the data sciences alone. Combining topological signatures with other types of descriptors is generally a relevant approach.

Today, TDA is an active field of research, at the crossroads of many scientific fields. In particular, there is currently an intense effort to effectively combine machine learning, statistics, and TDA. In this perspective, we believe that there is still a need for statistical results which demonstrate and quantify the interest of these data science approaches based on TDA.

AUTHOR CONTRIBUTIONS

All authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

FUNDING

This work was partly supported by the French ANR Chair in Artificial Intelligence TopAI.

ACKNOWLEDGMENTS

We thank Kovacev-Nikolic et al. (2016) for making their data available.

REFERENCES

Aamari, E., Kim, J., Chazal, F., Michel, B., Rinaldo, A., Wasserman, L., et al. (2019). Estimating the Reach of a Manifold. *Electron. J. Stat.* 13 (1), 1359–1399. doi:10.1214/19-ejs1551

Adams, H., Emerson, T., Kirby, M., Neville, R., Peterson, C., Shipman, P., et al. (2017). Persistence Images: a Stable Vector Representation of Persistent Homology. *J. Machine Learn. Res.* 18 (8), 1–35.

Anai, H., Chazal, F., Glisse, M., Ike, Y., Inakoshi, H., Tinarrage, R., et al. (2020). "Dtm-based Filtrations," in *Topological Data Analysis* (Springer), 33–66. doi:10.1007/978-3-030-43408-3_2

Balakrishna, S., Rinaldo, A., Sheehy, D., Singh, A., and Wasserman, L. A. (2012). Minimax Rates for Homology Inference. *J. Machine Learn. Res. - Proc. Track* 22, 64–72.

Berry, E., Chen, Y.-C., Cisewski-Kehe, J., and Fasy, B. T. (2020). Functional Summaries of Persistence Diagrams. *J. Appl. Comput. Topol.* 4 (2), 211–262. doi:10.1007/s41468-020-00048-w

Biau, G., Chazal, F., Cohen-Steiner, D., Devroye, L., and Rodriguez, C. (2011). A Weighted K-Nearest Neighbor Density Estimate for Geometric Inference. *Electron. J. Stat.* 5, 204–237. doi:10.1214/11-ejs606

Biscio, C. A., and Møller, J. (2019). The Accumulated Persistence Function, a New Useful Functional Summary Statistic for Topological Data Analysis, with a View to Brain Artery Trees and Spatial point Process Applications. *J. Comput. Graphical Stat.* 28, 671–681. doi:10.1080/10618600.2019.1573686

Bobrowski, O., Mukherjee, S., and Taylor, J. (2014). *Topological Consistency via Kernel Estimation*. arXiv preprint arXiv:1407.5272.

Boissonnat, J.-D., Chazal, F., and Yvinec, M. (2018). *Geometric and Topological Inference*, Vol. 57. Cambridge University Press.

Bonis, T., Ovsjanikov, M., Oudot, S., and Chazal, F. (2016). "Persistence-based Pooling for Shape Pose Recognition," in ProceedingsComputational Topology in Image Context - 6th International Workshop, CTIC 2016. Marseille, France, June 15-17, 2016. 19–29. doi:10.1007/978-3-319-39441-1_3

Brécheteau, C. (2019). A Statistical Test of Isomorphism between Metric-Measure Spaces Using the Distance-To-A-Measure Signature. *Electron. J. Stat.* 13 (1), 795–849. doi:10.1214/19-ejs1539

Brécheteau, C., and Levrard, C. (2020). A K-Points-Based Distance for Robust Geometric Inference. *Bernoulli* 26 (4), 3017–3050. doi:10.3150/20-bej1214

Breiman, L. (2001). Random Forests. *Machine Learn.* 45 (1), 5–32. doi:10.1023/a:1010933404324

Brown, A., Bobrowski, O., Munch, E., and Wang, B. (2020). Probabilistic Convergence and Stability of Random Mapper Graphs. *J. Appl. Comput. Topol.* 5, 99–140. doi:10.1007/s41468-020-00063-x

Brüel-Gabrielsson, R., Nelson, B. J., Dwaraknath, A., Skraba, P., Guibas, L. J., and Carlsson, G. (2019). *A Topology Layer for Machine Learning*. arXiv preprint arXiv:1905.12200.

Bubenik, P., Carlsson, G., Kim, P. T., and Luo, Z.-M. (2010). Statistical Topology via morse Theory Persistence and Nonparametric Estimation. *Algebraic Methods Stat. Probab.* 516, 75–92. doi:10.1090/conm/516/10167

Bubenik, P. (2015). Statistical Topological Data Analysis Using Persistence Landscapes. *J. Machine Learn. Res.* 16, 77–102.

Buchet, M., Chazal, F., Dey, T. K., Fan, F., Oudot, S. Y., and Wang, Y. (2015a). "Topological Analysis of Scalar fields with Outliers," in *Proc. Sympos. On Computational Geometry*.

Buchet, M., Chazal, F., Oudot, S., and Sheehy, D. R. (2015b). "Efficient and Robust Persistent Homology for Measures," in Proceedings of the 26th ACM-SIAM symposium on Discrete algorithms. SIAM. doi:10.1137/1.9781611973730.13

Cadre, B. (2006). Kernel Estimation of Density Level Sets. *J. Multivar. Anal.* 97 (4), 999–1023. doi:10.1016/j.jmva.2005.05.004

Carlsson, G., and Gabrielsson, R. B. (2020). "Topological Approaches to Deep Learning," in *Topological Data Analysis* (Springer), 119–146. doi:10.1007/978-3-030-43408-3_5

Carlsson, G. (2009). Topology and Data. *Bull. Amer. Math. Soc.* 46 (2), 255–308. doi:10.1090/s0273-0979-09-01249-x

Carriere, M., Chazal, F., Glisse, M., Ike, Y., and Kannan, H. (2020). *A Note on Stochastic Subgradient Descent for Persistence-Based Functionals: Convergence and Practical Aspects*. arXiv preprint arXiv:2010.08356.

Carrière, M., Chazal, F., Ike, Y., Lacombe, T., Royer, M., and Umeda, Y. (2020). "Perslay: a Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures," in International Conference on Artificial Intelligence and Statistics (PMLR), 2786–2796.

Carrière, M., and Michel, B. (2019). *Approximation of Reeb Spaces with Mappers and Applications to Stochastic Filters*. arXiv preprint arXiv:1912.10742.

Carriere, M., Michel, B., and Oudot, S. (2018). Statistical Analysis and Parameter Selection for Mapper. *J. Machine Learn. Res.* 19 (12).

Carriere, M., and Oudot, S. (2017). "Sliced Wasserstein Kernel for Persistence Diagrams," in *To Appear in ICML-17*.

Carrière, M., and Oudot, S. (2015). *Structure and Stability of the 1-dimensional Mapper*. arXiv preprint arXiv:1511.05823.

Carrière, M., and Rabadán, R. (2020). "Topological Data Analysis of Single-Cell Hi-C Contact Maps," in *Topological Data Analysis* (Springer), 147–162. doi:10.1007/978-3-030-43408-3_6

Chazal, F., Chen, D., Guibas, L., Jiang, X., and Sommer, C. (2011a). Data-driven Trajectory Smoothing. Proc. ACM SIGSPATIAL GIS. doi:10.1145/2093973.2094007

Chazal, F., Cohen-Steiner, D., Glisse, M., Guibas, L., and Oudot, S. (2009a). Proximity of Persistence Modules and Their Diagrams. *SCG*, 237–246. doi:10.1145/1542362.1542407

Chazal, F., Cohen-Steiner, D., Guibas, L. J., Mémoli, F., and Oudot, S. Y. (2009b). Gromov-hausdorff Stable Signatures for Shapes Using Persistence. *Comput. Graphics Forum (proc. SGP 2009)* 28, 1393–1403. doi:10.1111/j.1467-8659.2009.01516.x

Chazal, F., Cohen-Steiner, D., and Lieutier, A. (2009d). A Sampling Theory for Compact Sets in Euclidean Space. *Discrete Comput. Geom.* 41 (3), 461–479. doi:10.1007/s00454-009-9144-8

Chazal, F., Cohen-Steiner, D., and Lieutier, A. (2009c). Normal Cone Approximation and Offset Shape Isotopy. *Comp. Geom. Theor. Appl.* 42 (6-7), 566–581. doi:10.1016/j.comgeo.2008.12.002

Chazal, F., Cohen-Steiner, D., Lieutier, A., and Thibert, B. (2008). Stability of Curvature Measures. *Comput. Graphics Forum (proc. SGP 2009)*, 1485–1496.

Chazal, F., Cohen-Steiner, D., and Mérigot, Q. (2010). Boundary Measures for Geometric Inference. *Found. Comput. Math.* 10, 221–240. doi:10.1007/s10208-009-9056-2

Chazal, F., Cohen-Steiner, D., and Mérigot, Q. (2011b). Geometric Inference for Probability Measures. *Found. Comput. Math.* 11 (6), 733–751. doi:10.1007/s10208-011-9098-0

Chazal, F., de Silva, V., Glisse, M., and Oudot, S. (2016a). The Structure and Stability of Persistence Modules. *SpringerBriefs in Mathematics*. Springer. doi:10.1007/978-3-319-42545-0

Chazal, F., Fasy, B. T., Lecci, F., Michel, B., Rinaldo, A., and Wasserman, L. (2014a). "Robust Topological Inference: Distance to a Measure and Kernel Distance," in *To Appear in JMLR.* .

Chazal, F., Fasy, B. T., Lecci, F., Michel, B., Rinaldo, A., and Wasserman, L. (2015a). "Subsampling Methods for Persistent Homology," in To appear in Proceedings of the 32 st International Conference on Machine Learning (ICML-15).

Chazal, F., Fasy, B. T., Lecci, F., Rinaldo, A., Singh, A., and Wasserman, L. (2013a). *On the Bootstrap for Persistence Diagrams and Landscapes*. arXiv preprint arXiv:1311.0376.

Chazal, F., Fasy, B. T., Lecci, F., Rinaldo, A., and Wasserman, L. (2015b). Stochastic Convergence of Persistence Landscapes and Silhouettes. *J. Comput. Geom.* 6 (2), 140–161.

Chazal, F., Glisse, M., Labruère, C., and Michel, B. (2014b). *Convergence Rates for Persistence Diagram Estimation in Topological Data Analysis*. Proceedings of Machine Learning Research.

Chazal, F., Guibas, L. J., Oudot, S. Y., and Skraba, P. (2013b). Persistence-based Clustering in Riemannian Manifolds. *J. ACM (Jacm)* 60 (6), 41. doi:10.1145/2535927

Chazal, F. (2017). "High-dimensional Topological Data Analysis," in *Handbook of Discrete and Computational Geometry*. 3rd Ed- To appear. chapter 27. (CRC Press).

Chazal, F., Huang, R., and Sun, J. (2015c). Gromov-Hausdorff Approximation of Filamentary Structures Using Reeb-type Graphs. *Discrete Comput. Geom.* 53 (3), 621–649. doi:10.1007/s00454-015-9674-1

Chazal, F., Levrard, C., and Royer, M. (2020). *Optimal Quantization of the Mean Measure and Application to Clustering of Measures*. arXiv preprint arXiv: 2002.01216.

Chazal, F., and Lieutier, A. (2008). Smooth Manifold Reconstruction from Noisy and Non-uniform Approximation with Guarantees. *Comput. Geom.* 40 (2), 156–170. doi:10.1016/j.comgeo.2007.07.001

Chazal, F., Massart, P., and Michel, B. (2016b). Rates of Convergence for Robust Geometric Inference. *Electron. J. Statist.* 10, 2243–2286. doi:10.1214/16-ejs1161

Chazal, F., and Oudot, S. Y. (2008). "Towards Persistence-Based Reconstruction in Euclidean Spaces," in Proceedings of the twenty-fourth annual symposium on Computational geometry (New York, NY, USA: SCG '08ACM), 232–241. doi:10.1145/1377676.1377719

Chen, C., Ni, X., Bai, Q., and Wang, Y. (2019). "A Topological Regularizer for Classifiers via Persistent Homology," in The 22nd International Conference on Artificial Intelligence and Statistics, 2573–2582.

Chen, Y.-C., Genovese, C. R., and Wasserman, L. (2015). *Density Level Sets: Asymptotics, Inference, and Visualization*. arXiv preprint arXiv:1504.05438.

Chen, Y.-C., Genovese, C. R., and Wasserman, L. (2017). Density Level Sets: Asymptotics, Inference, and Visualization. *J. Am. Stat. Assoc.* 112 (520), 1684–1696. doi:10.1080/01621459.2016.1228536

Cohen-Steiner, D., Edelsbrunner, H., Harer, J., and Mileyko, Y. (2010). Lipschitz Functions Have L P -Stable Persistence. *Found. Comput. Math.* 10 (2), 127–139. doi:10.1007/s10208-010-9060-6

Cohen-Steiner, D., Edelsbrunner, H., and Harer, J. (2005). *Stability of Persistence Diagrams*. SCG, 263–271. doi:10.1145/1064092.1064133

Cuevas, A., and Rodríguez-Casal, A. (2004). On Boundary Estimation. *Adv. Appl. Probab.* 36 (2), 340–354. doi:10.1239/aap/1086957575

Curry, J., Mukherjee, S., and Turner, K. (2018). *How many Directions Determine a Shape and Other Sufficiency Results for Two Topological Transforms*. arXiv preprint arXiv:1805.09782.

De Silva, V., and Carlsson, G. (2004). "Topological Estimation Using Witness Complexes," in Proceedings of the First Eurographics Conference on Point-Based Graphics (Switzerland, Switzerland: Aire-la-VilleEurographics Association), 157–166. SPBG'04.

De Silva, V., and Ghrist, R. (2007). Homological Sensor Networks. *Notices Am. Math. Soc.* 54 (1).

Devroye, L., and Wise, G. L. (1980). Detection of Abnormal Behavior via Nonparametric Estimation of the Support. *SIAM J. Appl. Math.* 38 (3), 480–488. doi:10.1137/0138038

Dey, T. K., Mémoli, F., and Wang, Y. (2016). "Multiscale Mapper: Topological Summarization via Codomain Covers," in Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (Society for Industrial and Applied Mathematics), 997–1013. doi:10.1137/1.9781611974331.ch71

Dey, T. K., Mémoli, F., and Wang, Y. (2017). Topological Analysis of Nerves, Reeb Spaces, Mappers, and Multiscale Mappers. Proc. Sympos. Comput . Geom.(SoCG). .

Dindin, M., Umeda, Y., and Chazal, F. (2020). "Topological Data Analysis for Arrhythmia Detection through Modular Neural Networks," in Canadian Conference on Artificial Intelligence (Springer), 177–188. doi:10.1007/978-3-030-47358-7_17

Divol, V., and Chazal, F. (2020). The Density of Expected Persistence Diagrams and its Kernel Based Estimation. *J. Comput. Geom.* 10 (2), 127–153.

Divol, V., and Lacombe, T. (2020). Understanding the Topology and the Geometry of the Persistence Diagram Space via Optimal Partial Transport. *J. Appl. Comput. Topol.* 5 (1), 1–53. doi:10.1007/s41468-020-00061-z

Edelsbrunner, H., Letscher, D., and Zomorodian, A. (2002). Topological Persistence and Simplification. *Discrete Comput. Geom.* 28, 511–533. doi:10.1007/s00454-002-2885-2

Fasy, B. T., Kim, J., Lecci, F., and Maria, C. (2014a). *Introduction to the R Package Tda*. arXiv preprint arXiv:1411.1830.

Fasy, B. T., Lecci, F., Rinaldo, A., Wasserman, L., Balakrishnan, S., and Singh, A. (2014b). Confidence Sets for Persistence Diagrams. *Ann. Stat.* 42 (6), 2301–2339. doi:10.1214/14-aos1252

Federer, H. (1959). Curvature Measures. *Trans. Amer. Math. Soc.* 93, 418. doi:10.1090/s0002-9947-1959-0110078-1

Frosini, P. (1992). "Measuring Shapes by Size Functions," in *Intelligent Robots and Computer Vision X: Algorithms and Techniques* (International Society for Optics and Photonics), Vol. 1607, 122–133.

Gabrielsson, R. B., and Carlsson, G. (2019). "Exposition and Interpretation of the Topology of Neural Networks," in 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA) (IEEE), 1069–1076. doi:10.1109/icmla.2019.00180

Genovese, C. R., Perone-Pacifico, M., Verdinelli, I., and Wasserman, L. (2012). Manifold Estimation and Singular Deconvolution under Hausdorff Loss. *Ann. Statist.* 40, 941–963. doi:10.1214/12-aos994

Ghrist, R. (2017). *Homological Algebra and Data*. preprint.

Grove, K. (1993). Critical point Theory for Distance Functions. *Proc. Symposia Pure Math.* 54. doi:10.1090/pspum/054.3/1216630

Guibas, L., Morozov, D., and Mérigot, Q. (2013). Witnessed K-Distance. *Discrete Comput. Geom.* 49, 22–45. doi:10.1007/s00454-012-9465-x

Hatcher, A. (2001). *Algebraic Topology*. Cambridge Univ. Press.

Hensel, F., Moor, M., and Rieck, B. (2021). A Survey of Topological Machine Learning Methods. *Front. Artif. Intell.* 4, 52. doi:10.3389/frai.2021.681108

Hofer, C. D., Kwitt, R., and Niethammer, M. (2019b). Learning Representations of Persistence Barcodes. *J. Machine Learn. Res.* 20 (126), 1–45.

Hofer, C., Kwitt, R., Niethammer, M., and Dixit, M. (2019a). "Connectivity-optimized Representation Learning via Persistent Homology,"in Proceedings of the 36th International Conference on Machine Learning Vol. 97. Proceedings of Machine Learning Research (PMLR), 2751–2760.

Hofer, C., Kwitt, R., Niethammer, M., and Uhl, A. (2017). *Deep Learning with Topological Signatures*. arXiv preprint arXiv:1707.04041.

Khasawneh, F. A., and Munch, E. (2016). Chatter Detection in Turning Using Persistent Homology. *Mech. Syst. Signal Process.* 70-71, 527–541. doi:10.1016/j.ymssp.2015.09.046

Kim, K., Kim, J., Zaheer, M., Kim, J., Chazal, F., and Wasserman, L. (2020). "Pllay: Efficient Topological Layer Based on Persistence Landscapes," in 34th Conference on Neural Information Processing Systems (NeurIPS)

Kovacev-Nikolic, V., Bubenik, P., Nikolić, D., and Heo, G. (2016). Using Persistent Homology and Dynamical Distances to Analyze Protein Binding. *Stat. Appl. Genet. Mol. Biol.* 15 (1), 19–38. doi:10.1515/sagmb-2015-0057

Kramar, M., Goullet, A., Kondic, L., and Mischaikow, K. (2013). Persistence of Force Networks in Compressed Granular media. *Phys. Rev. E Stat. Nonlin Soft Matter Phys.* 87 (4), 042207. doi:10.1103/PhysRevE.87.042207

Kramár, M., Levanger, R., Tithof, J., Suri, B., Xu, M., Paul, M., et al. (2016). Analysis of Kolmogorov Flow and Rayleigh-Bénard Convection Using Persistent Homology. *Physica D: Nonlinear Phenomena* 334, 82–98. doi:10.1016/j.physd.2016.02.003

Krebs, J. T., and Polonik, W. (2019). *On the Asymptotic Normality of Persistent Betti Numbers*. arXiv preprint arXiv:1903.03280.

Kusano, G., Fukumizu, K., and Hiraoka, Y. (2017). *Kernel Method for Persistence Diagrams via Kernel Embedding and Weight Factor*. arXiv preprint arXiv:1706.03472.

Kusano, G., Hiraoka, Y., and Fukumizu, K. (2016). "Persistence Weighted Gaussian Kernel for Topological Data Analysis," in International Conference on Machine Learning. 2004–2013.

Kusano, G. (2019). On the Expectation of a Persistence Diagram by the Persistence Weighted Kernel. *Jpn. J. Indust. Appl. Math.* 36 (3), 861–892. doi:10.1007/s13160-019-00374-2

Lacombe, T., Cuturi, M., and Oudot, S. (2018). *Large Scale Computation of Means and Clusters for Persistence Diagrams Using Optimal Transport*. NeurIPS.

Lee, Y., Barthel, S. D., Dłotko, P., Moosavi, S. M., Hess, K., and Smit, B. (2017). Quantifying Similarity of Pore-Geometry in Nanoporous Materials. *Nat. Commun.* 8, 15396. doi:10.1038/ncomms15396

Leygonie, J., Oudot, S., and Tillmann, U. (2019). *A Framework for Differential Calculus on Persistence Barcodes*. arXiv preprint arXiv:1910.00960.

Li, C., Ovsjanikov, M., and Chazal, F. (2014). "Persistence-based Structural Recognition," in ,2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2003–2010. doi:10.1109/cvpr.2014.257

Li, M. Z., Ryerson, M. S., and Balakrishnan, H. (2019). Topological Data Analysis for Aviation Applications. *Transportation Res. E: Logistics Transportation Rev.* 128, 149–174. doi:10.1016/j.tre.2019.05.017

Lum, P. Y., Singh, G., Lehman, A., Ishkanov, T., Vejdemo-Johansson, M., Alagappan, M., et al. (2013). Extracting Insights from the Shape of Complex Data Using Topology. *Sci. Rep.* 3, 1236. doi:10.1038/srep01236

Maria, C., Boissonnat, J.-D., Glisse, M., and Yvinec, M. (2014). "The Gudhi Library: Simplicial Complexes and Persistent Homology," in *International Congress on Mathematical Software* (Springer), 167–174. doi:10.1007/978-3-662-44199-2_28

Maroulas, V., Nasrin, F., and Oballe, C. (2020). A Bayesian Framework for Persistent Homology. *SIAM J. Math. Data Sci.* 2 (1), 48–74. doi:10.1137/19m1268719

McKinney, W. (2010). "Data Structures for Statistical Computing in python," in Proceedings of the 9th Python in Science Conference (TX: SciPy Austin), Vol. 445, 51–56. doi:10.25080/majora-92bf1922-00a

Mileyko, Y., Mukherjee, S., and Harer, J. (2011). Probability Measures on the Space of Persistence Diagrams. *Inverse Probl.* 27 (12), 124007. doi:10.1088/0266-5611/27/12/124007

Moon, C., and Lazar, N. A. (2020). *Hypothesis Testing for Shapes Using Vectorized Persistence Diagrams*. arXiv preprint arXiv:2006.05466.

Moor, M., Horn, M., Rieck, B., and Borgwardt, K. (2020). "Topological Autoencoders," in International Conference on Machine Learning (PMLR), 7045–7054.

Nakamura, T., Hiraoka, Y., Hirata, A., Escolar, E. G., and Nishiura, Y. (2015). Persistent Homology and many-body Atomic Structure for Medium-Range Order in the Glass. *Nanotechnology* 26 (30), 304001. doi:10.1088/0957-4484/26/30/304001

Niyogi, P., Smale, S., and Weinberger, S. (2011). A Topological View of Unsupervised Learning from Noisy Data. *SIAM J. Comput.* 40 (3), 646–663. doi:10.1137/090762932

Niyogi, P., Smale, S., and Weinberger, S. (2008). Finding the Homology of Submanifolds with High Confidence from Random Samples. *Discrete Comput. Geom.* 39 (1-3), 419–441. doi:10.1007/s00454-008-9053-2

Obayashi, I., and Hiraoka, Y. (2017). *Persistence Diagrams with Linear Machine Learning Models*. arXiv preprint arXiv:1706.10082.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-Learn: Machine Learning in Python. *J. Machine Learn. Res.* 12 (Oct), 2825–2830.

Penrose, M. D., and Yukich, J. E. (2001). Central Limit Theorems for Some Graphs in Computational Geometry. *Ann. Appl. Probab.* 11, 1005–1041. doi:10.1214/aoap/1015345393

Petrunin, A. (2007). "Applied Manifold Geometry," in *Surveys in Differential Geometry* (Somerville, MA: Int. Press), Vol. XI, 137–483. doi:10.1142/9789812770721_0003

Phillips, J. M., Wang, B., and Zheng, Y. (2014). *Geometric Inference on Kernel Density Estimates*. arXiv preprint 1307.7760.

Pike, J. A., Khan, A. O., Pallini, C., Thomas, S. G., Mund, M., Ries, J., et al. (2020). Topological Data Analysis Quantifies Biological Nano-Structure from Single Molecule Localization Microscopy. *Bioinformatics* 36 (5), 1614–1621. doi:10.1093/bioinformatics/btz788

Polonik, W. (1995). Measuring Mass Concentrations and Estimating Density Contour Clusters-An Excess Mass Approach. *Ann. Stat.* 23, 855–881. doi:10.1214/aos/1176324626

Poulenard, A., Skraba, P., and Ovsjanikov, M. (2018). Topological Function Optimization for Continuous Shape Matching. *Comput. Graphics Forum* 37, 13–25. doi:10.1111/cgf.13487

Qaiser, T., Tsang, Y.-W., Taniyama, D., Sakamoto, N., Nakane, K., Epstein, D., et al. (2019). Fast and Accurate Tumor Segmentation of Histology Images Using Persistent Homology and Deep Convolutional Features. *Med. image Anal.* 55, 1–14. doi:10.1016/j.media.2019.03.014

Ramamurthy, K. N., Varshney, K., and Mody, K. (2019). "Topological Data Analysis of Decision Boundaries with Application to Model Selection," in International Conference on Machine Learning (PMLR), 5351–5360.

Reininghaus, J., Huber, S., Bauer, U., and Kwitt, R. (2015). "A Stable Multi-Scale Kernel for Topological Machine Learning," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4741–4748. doi:10.1109/cvpr.2015.7299106

Rieck, B. A., Togninalli, M., Bock, C., Moor, M., Horn, M., Gumbsch, T., and Borgwardt, K. (2019). "Neural Persistence: A Complexity Measure for Deep Neural Networks Using Algebraic Topology," in International Conference on Learning Representations (ICLR 2019) (OpenReview)

Rieck, B., Yates, T., Bock, C., Borgwardt, K., Wolf, G., Turk-Browne, N., et al. (2020). Uncovering the Topology of Time-Varying Fmri Data Using Cubical Persistence. *Adv. Neural Inf. Process. Syst.* 33.

Robins, V. (1999). Towards Computing Homology from Finite Approximations. *Topology Proc.* 24, 503–532.

Robinson, A., and Turner, K. (2017). Hypothesis Testing for Topological Data Analysis. *J. Appl. Comput. Topol.* 1 (2), 241–261. doi:10.1007/s41468-017-0008-7

Roycraft, B., Krebs, J., and Polonik, W. (2020). *Bootstrapping Persistent Betti Numbers and Other Stabilizing Statistics*. arXiv preprint arXiv:2005.01417

Royer, M., Chazal, F., Levrard, C., Ike, Y., and Umeda, Y. (2021). "Atol: Measure Vectorisation for Automatic Topologically-Oriented Learning," in International Conference on Artificial Intelligence and Statistics (PMLR)

Barannikov, S. (1994). "The Framed morse Complex and its Invariants," in *Adv. Soviet Math.*, Vol. 21. Providence, RI: Amer. Math. Soc., 93–115. doi:10.1090/advsov/021/03

Seversky, L. M., Davis, S., and Berger, M. (2016). "On Time-Series Topological Data Analysis: New Data and Opportunities," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 59–67. doi:10.1109/cvprw.2016.131

Singh, A., Scott, C., and Nowak, R. (2009). Adaptive Hausdorff Estimation of Density Level Sets. *Ann. Statist.* 37 (5B), 2760–2782. doi:10.1214/08-aos661

Singh, G., Mémoli, F., and Carlsson, G. E. (2007). 8. Tensor Decomposition. SPBG, 91–100. Citeseer. doi:10.1137/1.9780898718867.ch8

Sizemore, A. E., Phillips-Cremins, J. E., Ghrist, R., and Bassett, D. S. (2019). The Importance of the Whole: Topological Data Analysis for the Network Neuroscientist. *Netw. Neurosci.* 3 (3), 656–673. doi:10.1162/netn_a_00073

Skraba, P., Ovsjanikov, M., Chazal, F., and Guibas, L. (2010). "Persistence-based Segmentation of Deformable Shapes," in Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 (IEEE Computer Society Conference on), 45–52. doi:10.1109/cvprw.2010.5543285

Smith, A. D., Dłotko, P., and Zavala, V. M. (2021). Topological Data Analysis: Concepts, Computation, and Applications in Chemical Engineering. *Comput. Chem. Eng.* 146, 107202. doi:10.1016/j.compchemeng.2020.107202

Tsybakov, A. B. (1997). On Nonparametric Estimation of Density Level Sets. *Ann. Stat.* 25 (3), 948–969. doi:10.1214/aos/1069362732

Turner, K., Mileyko, Y., Mukherjee, S., and Harer, J. (2014a). Fréchet Means for Distributions of Persistence Diagrams. *Discrete Comput. Geom.* 52 (1), 44–70. doi:10.1007/s00454-014-9604-7

Turner, K., Mukherjee, S., and Boyer, D. M. (2014b). Persistent Homology Transform for Modeling Shapes and Surfaces. *Inf. Inference* 3 (4), 310–344. doi:10.1093/imaiai/iau011

Umeda, Y. (2017). Time Series Classification via Topological Data Analysis. *Trans. Jpn. Soc. Artif. Intell.* 32 (3), D–G72_1. doi:10.1527/tjsai.d-g72

van der Walt, S., Colbert, S. C., and Varoquaux, G. (2011). The Numpy Array: a Structure for Efficient Numerical Computation. *Comput. Sci. Eng.* 13 (2), 22–30. doi:10.1109/mcse.2011.37

Villani, C. (2003). *Topics in Optimal Transportation*. American Mathematical Society.

Wasserman, L. (2018). Topological Data Analysis. *Annu. Rev. Stat. Appl.* 5, 501–532. doi:10.1146/annurev-statistics-031017-100045

Yao, Y., Sun, J., Huang, X., Bowman, G. R., Singh, G., Lesnick, M., et al. (2009). Topological Methods for Exploring Low-Density States in Biomolecular Folding Pathways. *J. Chem. Phys.* 130 (14), 144115. doi:10.1063/1.3103496

Zieliński, B., Lipiński, M., Juda, M., Zeppelzauer, M., and Dłotko, P. (2010). "Persistence Bag-Of-Words for Topological Data Analysis," in Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19).

Zomorodian, A., and Carlsson, G. (2005). Computing Persistent Homology. *Discrete Comput. Geom.* 33 (2), 249–274. doi:10.1007/s00454-004-1146-y

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers in Artificial Intelligence

ORIGINAL RESEARCH
published: 11 October 2021
doi: 10.3389/frai.2021.681117

Minimal Cycle Representatives in Persistent Homology Using Linear Programming: An Empirical Study With User's Guide

Lu Li[1], *Connor Thompson*[2], *Gregory Henselman-Petrusek*[3], *Chad Giusti*[4] *and Lori Ziegelmeier*[1]*

[1]Mathematics, Statistics, and Computer Science Department, Macalester College, Saint Paul, MN, United States, [2]Department of Mathematics, Purdue University, West Lafayette, IN, United States, [3]Mathematical Institute, University of Oxford, Oxford, United Kingdom, [4]Department of Mathematical Sciences, University of Delaware, Newark, DE, United States

OPEN ACCESS

Edited by:
Kathryn Hess,
École Polytechnique Fédérale de Lausanne, Switzerland

Reviewed by:
Gard Spreemann,
Telenor Research, Norway
Henri Riihimäki,
University of Aberdeen, United Kingdom

***Correspondence:**
Lori Ziegelmeier
lziegel1@macalester.edu

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Artificial Intelligence

Received: 15 March 2021
Accepted: 14 May 2021
Published: 11 October 2021

Citation:
Li L, Thompson C, Henselman-Petrusek G, Giusti C and Ziegelmeier L (2021) Minimal Cycle Representatives in Persistent Homology Using Linear Programming: An Empirical Study With User's Guide. Front. Artif. Intell. 4:681117. doi: 10.3389/frai.2021.681117

Cycle representatives of persistent homology classes can be used to provide descriptions of topological features in data. However, the non-uniqueness of these representatives creates ambiguity and can lead to many different interpretations of the same set of classes. One approach to solving this problem is to optimize the choice of representative against some measure that is meaningful in the context of the data. In this work, we provide a study of the effectiveness and computational cost of several ℓ_1 minimization optimization procedures for constructing homological cycle bases for persistent homology with rational coefficients in dimension one, including uniform-weighted and length-weighted edge-loss algorithms as well as uniform-weighted and area-weighted triangle-loss algorithms. We conduct these optimizations via standard linear programming methods, applying general-purpose solvers to optimize over column bases of simplicial boundary matrices. Our key findings are: 1) optimization is effective in reducing the size of cycle representatives, though the extent of the reduction varies according to the dimension and distribution of the underlying data, 2) the computational cost of optimizing a basis of cycle representatives exceeds the cost of computing such a basis, in most data sets we consider, 3) the choice of linear solvers matters a lot to the computation time of optimizing cycles, 4) the computation time of solving an integer program is not significantly longer than the computation time of solving a linear program for most of the cycle representatives, using the Gurobi linear solver, 5) strikingly, whether requiring integer solutions or not, we almost always obtain a solution with the same cost and almost all solutions found have entries in {-1, 0, 1} and therefore, are also solutions to a restricted ℓ_0 optimization problem, and 6) we obtain qualitatively different results for generators in Erdős-Rényi random clique complexes than in real-world and synthetic point cloud data.

Keywords: topological data analysis, computational persistent homology, minimal cycle representatives, generators, linear programming, l_1 and l_0 minimization

1 INTRODUCTION

Topological data analysis (TDA) uncovers mesoscale structure in data by quantifying its shape using methods from algebraic topology. Topological features have proven effective when characterizing complex data, as they are qualitative, independent of choice of coordinates, and robust to some choices of metrics and moderate quantities of noise (Carlsson, 2009; Ghrist, 2014). As such, topological features extracted from data have recently drawn attention from researchers in various fields including, for example, neuroscience (Bendich et al., 2016; Giusti et al., 2016; Sizemore et al., 2019), computer graphics (Singh et al., 2007; Brüel-Gabrielsson et al., 2020), robotics (Vasudevan et al., 2011; Bhattacharya et al., 2015), and computational biology (Bhaskar et al., 2019; Ulmer et al., 2019; McGuirl et al., 2020) [including the study of protein structure (Xia and Wei, 2014; Kovacev-Nikolic et al., 2016; Xia et al., 2018)].

The primary tool in TDA is persistent homology (PH) (Ghrist, 2008), which describes how topological features of data, colloquially referred to as "holes," evolve as one varies a real-valued parameter. Each hole comes with a geometric notion of dimension which describes the shape that encloses the hole: connected components in dimension zero, loops in dimension one, shells in dimension two, and so on. From a parameterized topological space $X = (X_t)_{t\in S\subset\mathbb{R}_{\geq 0}}$, for each dimension n, PH produces a collection $\mathrm{Barcode}_n(X)$ of lifetime intervals $\mathcal{L}$ which encode for each topological feature the parameter values of its birth, when it first appears, and death, when it no longer remains.

A basic problem in the practical application of PH is interpretability: given an interval $\mathcal{L} \in \mathrm{Barcode}_n(X)$, how do we understand it in terms of the underlying data? A reasonable approach would be to find an element of the homology class, also known as a cycle representative, that witnesses structure in the data that has meaning to the investigator. In the context of geometric data, this takes the form of an "inverse problem," constructing geometric structures corresponding to each persistent interval in the original input data. For example, a representative for an interval $\mathcal{L} \in \mathrm{Barcode}_1(X)$ consists of a closed curve or linear combination of closed curves which enclose a set of holes across the family of spaces $(X_t)_{t\in\mathcal{L}\subset S}$. Cycle representatives are used in Emmett et al. (2015) to annotate particular loops as chromatin interactions, and Wu et al. (2017) uses cycle representatives to study and locate and reconstruct fine muscle columns in cardiac trabeculae restoration.

An important challenge, however, is that cycle representatives are not uniquely defined. For example, in the left-hand image in **Figure 1** adapted from Carlsson (2009), two curves enclose the same topological feature and thus, represent the same persistent homology class. We often want to find a cycle that captures not only the existence but also information about the location and shape of the hole that the homology class has detected. This often means optimizing an application-dependent property using the underlying data, e.g. finding a minimal length or bounding area/volume using an appropriate metric. The algorithmic problem of selecting such optimal representatives is currently an active area of research (Chen and Freedman, 2010a; Dey et al., 2011; Wu et al., 2017; Obayashi, 2018; Dey et al., 2019).

There are diverse notions of optimality we may wish to consider in a given context, and which may have significant impact on the effectiveness or suitability of optimization, including.

- weight assignment to chains (uniform vs. length or area weighted),
- choice of loss function (ℓ_0 vs. ℓ_1),
- formulation of the optimization problem (cycle size vs. bounded area or volume), and
- restrictions on allowable coefficients (rational, integral, or {0, 1, -1}).

Each has a unique set of advantages and disadvantages. For example, optimization using the ℓ_0 norm with {0, 1, -1} coefficients is thought to yield the most interpretable results, but ℓ_0 optimization is NP-hard, in general (Chen and Freedman, 2010b). The problem of finding ℓ_1 optimal cycles with rational coefficients, can be formulated as a more tractable linear programming problem. While some literature exists to inform this choice (Dey et al., 2011; Escolar and Hiraoka, 2016; Obayashi, 2018), questions of basic importance remain, including:

Q1 How do the computational costs of the various optimization techniques compare? How much do these costs depend on the choice of a particular linear solver?
Q2 What are the statistical properties of optimal cycle representatives? For example, how often does the support of a representative form a single loop in the underlying graph? And, how much do optimized cycles coming out of an optimization pipeline differ from the representative that went in?
Q3 To what extent does choice of technique matter? For example, how often does the length of a length-weighted optimal cycle match the length of a uniform-weighted optimal cycle? And, how often are ℓ_1 optimal representatives ℓ_0 optimal?

Given the conceptual and computational complexity of these problems [see Chen and Freedman (2010b)], the authors expect that formal answers are unlikely to be available in the near future. However, even where theoretical results are available, strong empirical trends may suggest different or even contrary principles to the practitioner. For example, while the persistence calculation is known to have matrix multiplication time complexity (Milosavljević et al., 2011), in practice the computation runs almost always in linear time. Therefore, the authors believe that a careful empirical exploration of questions one to three will be of substantial value.

In this paper, we undertake such an exploration in the context of one-dimensional persistent homology over the field of rationals, $\mathbb{Q}$. We focus on linear programming (LP) and mixed-integer programming (MIP) approaches due to their ease of use, flexibility, and adaptability. In doing so, we present a new treatment of parameter-dependence (vis-a-vis selection of simplex-wise refinements) relevant to common

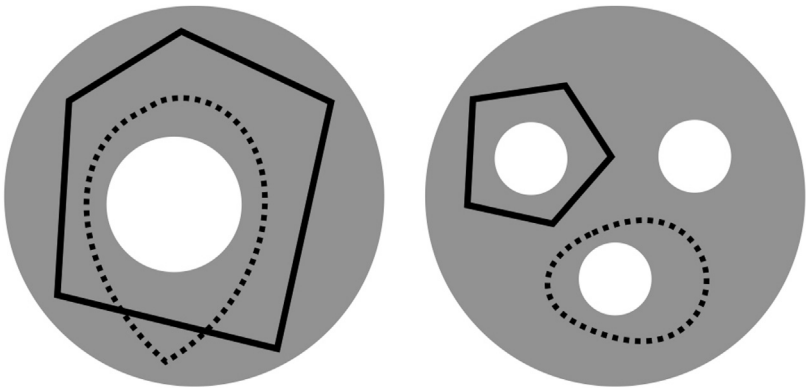


FIGURE 1 | Two disks (gray) — which we regard as 2-dimensional simplicial complexes, though the explicit decomposition into simplices is not shown—with different numbers of holes (white) and cycle representatives (black solid or dotted) adapted from (Carlsson, 2009). The disk on the left has a single 2-dimensional "hole" ($\beta_1 = 1$), and the two loops around it are cycle representatives for the same homology class. Similarly, the disk on the right has three "holes" ($\beta_1 = 3$) and the two loops shown are cycle representatives for different homology classes.

cases of rational cycle representative optimization (Escolar and Hiraoka, 2016; Obayashi, 2018), such as finding optimal cycle bases for the persistent homology of the Vietoris-Rips complex of a point cloud. We restrict our attention to one-dimensional homology to limit the number of reported statistics and data visualizations presented, although the methods discussed could be applied to any homological dimension.

The paper is organized as follows. **Section 2** provides an overview of some key concepts in TDA to inform a reader new to algebraic topology and establish notation. Then, we provide a survey of previous work on finding optimal persistent cycle representatives in **Section 3**, and formulate the methods used in this paper to find different notions of minimal cycle representatives via LP and MIP in **Section 4**. **Section 5** describes our experiments, including overviews of the data and the hardware and software we use for our analysis. In **Section 6**, we discuss the results of our experiments. We conclude and describe possible future work in **Section 7**.

2 BACKGROUND: TOPOLOGICAL DATA ANALYSIS AND PERSISTENT HOMOLOGY

In this section, we introduce key terms in algebraic and computational topology to provide minimal background and establish notation. For a more thorough introduction see, for example, Hatcher et al. (2002), Ghrist (2008), Edelsbrunner and Harer (2008), Carlsson (2009), Edelsbrunner and Harer (2010), and Topaz et al. (2015).

Given a discrete set of sample data, we approximate the topological space underlying the data by constructing a *simplicial complex*. This construction expresses the structure as a union of vertices, edges, triangles, tetrahedrons, and higher dimensional analogues (Carlsson, 2009).

2.1 Simplicial Complexes

A simplicial complex is a collection K of non-empty subsets of a finite set V. The elements of V are called vertices of K, and the elements of K are called simplices. A simplicial complex has the following properties: 1) $\{v\}$ in K for all $v \in V$, and 2) $\tau \subset \sigma$ and $\sigma \in K$ guarantees that $\tau \in K$.

Additionally, we say that a simplex has dimension n or is an n-simplex if it has cardinality $n + 1$. We use $\mathbf{S}_n(K)$ to denote the collection of n-simplices contained in K.

While there are a variety of approaches to create a simplicial complex from data, our examples use a standard construction for approximation of point clouds. Given a metric space X with metric d and real number $\varepsilon \geq 0$, the Vietoris-Rips complex for X, denoted by $\mathbf{VR}_\varepsilon(X)$, is defined as

$$\mathbf{VR}_\varepsilon(X) = \{\sigma \in \mathbf{S}_n(K) \big| d(x,y) \leq \varepsilon \text{ for all } x, y \in \sigma\}.$$

That is, given a set of discrete points X and a metric d, we build a **VR** complex at scale ε by forming an n-simplex if and only if $n+1$ points in X are pairwise within ε distance of each other.

2.2 Chains and Chain Complexes

Given a simplicial complex K and an abelian group G, the group of n-chains in K with coefficients in G is defined as

$$\mathbf{C}_n(K;G) := G^{\mathbf{S}_n(K)}.$$

Formally, we regard $G^{\mathbf{S}_n(K)}$ as a group of functions $\mathbf{S}_n(K) \to G$ under element-wise addition. Alternatively, we may view $\mathbf{C}_n(K;G)$ as a group of formal G-linear combinations of n-simplices, i.e., $\{\sum_\sigma x_\sigma \sigma | x_\sigma \in G \text{ and } \sigma \in \mathbf{S}_n(K)\}$.

Remark 2.1. We will focus on the cases where G is $\mathbb{Q}$ (the field of rationals), $\mathbb{Z}$ (the group of integers), or $\mathbb{F}_2$ (the 2-element field). Since we are most interested in the case $G = \mathbb{Q}$, we adopt the shorthand $\mathbf{C}_n(K) = \mathbf{C}_n(K;\mathbb{Q})$.

An element $\mathbf{x} = (x_\sigma)_{\sigma \in \mathbf{S}_n(K)} \in G^{\mathbf{S}_n(K)}$ is called an n-chain of K. As in this example, see we will generally use a bold-face symbol for the tuple $\mathbf{x}$ and corresponding light-face symbols for entries x_σ. The support of an n-chain is the set of simplices on which x_σ is nonzero:

$$\mathrm{supp}(\mathbf{x}) := \{\sigma \in \mathbf{S}_n(K) \,|\, x_\sigma \neq 0\}.$$

The ℓ_0 norm[1] and ℓ_1 norm[2] of $\mathbf{x}$ are defined as

$$\|\mathbf{x}\|_0 := \left|\mathrm{supp}(\mathbf{x})\right| \qquad \|\mathbf{x}\|_1 := \sum_{\sigma \in \mathbf{S}_n(K)} |x_\sigma|.$$

Remark 2.2 (Indexing conventions for chains and simplices). As chains play a central role in our discussion, it will be useful to establish some special conventions to describe them. These conventions depend on the availability of certain linear orders, either on the set of vertices or the set of simplices.

Case 1: Vertex set V has a linear order $\leq$. Every vertex set V discussed in this text will be assigned a (possibly arbitrary) linear order. Without risk of ambiguity, we may therefore write

$$(v_0, \ldots, v_n)$$

for the n-chain that places a coefficient of 1 on $\sigma = \{v_0 \leq \ldots \leq v_n\}$ and 0 on all other simplices.

Case 2: Simplex set $\mathbf{S}_n(K)$ has a linear order $\leq$. We will sometimes define a linear order on $\mathbf{S}_n(K)$. This determines a unique bijection $\sigma^{(n)} : \{1, \ldots, |\mathbf{S}_n(K)|\} \to \mathbf{S}_n(K)$ such that $\sigma_i^{(n)} \leq \sigma_j^{(n)}$ iff $i \leq j$. This bijection determines an isomorphism.

$$\phi : \mathbf{C}_n(K; G) = G^{\mathbf{S}_n(K)} \to G^{|\mathbf{S}_n(K)|}.$$

such that $\phi(\mathbf{x})_i = x_{\sigma_i}$ for all i. Provided a linear order $\leq$, we will use $\mathbf{x}$ to denote both $\mathbf{x}$ and $\phi(\mathbf{x})$ and rely on context to clarify the intended meaning.

For each $n \geq 1$, the boundary map $\partial_n : \mathbf{C}_n(K) \to \mathbf{C}_{n-1}(K)$ is the linear transformation defined on a basis vector $(v_0, v_1, \ldots, v_n)$ by

$$\partial_n (v_0, v_1, \ldots, v_n) = \sum_{i=0}^{n} (-1)^i (v_0, \ldots, \widehat{v_i}, \ldots, v_n),$$

where $\widehat{v_i}$ omits v_i from the vector. This map extends linearly from the basis of n-simplices to any n-chain in $\mathbf{C}_n(K)$. By an abuse of notation, we also denote the matrix representation of this boundary map, known as the boundary matrix, as ∂_n. The boundary matrix is parametrized by the n-simplices $S_n(K)$ along the columns and $n-1$ simplices $S_{n-1}(K)$ along the rows.

The collection $(\mathbf{C}_n(K))_{n \geq 0}$ along with the boundary maps $(\partial_n)_{n \geq 0}$ form a chain complex

$$\ldots \mathbf{C}_{n+1}(K) \xrightarrow{\partial_{n+1}} \mathbf{C}_n(K) \xrightarrow{\partial_n} \mathbf{C}_{n-1}(K) \xrightarrow{\partial_{n-1}} \ldots \xrightarrow{\partial_3} \mathbf{C}_2(K) \xrightarrow{\partial_2} \mathbf{C}_1(K) \xrightarrow{\partial_1} \mathbf{C}_0(K) \xrightarrow{\partial_0} 0.$$

Remark 2.3 (Indexing conventions for boundary matrices). In general, boundary matrix ∂_n is regarded as an element of $G^{\mathbf{S}_{n-1}(K) \times \mathbf{S}_n(K)}$, that is, as an array with columns labeled by n-simplices and rows labeled by $n-1$ simplices. However, given linear orders on $\mathbf{S}_{n-1}(K)$ and $\mathbf{S}_n(K)$, we may naturally regard ∂_n as an element of $G^{|\mathbf{S}_{n-1}(K)| \times |\mathbf{S}_n(K)|}$, see **Remark 2.2**.

2.3 Cycles, Boundaries

The boundary of an n-chain $\mathbf{x}$ is $\partial_n(\mathbf{x})$. An n-cycle is an n-chain with zero boundary. The set of all n-cycles forms a subspace $\mathbf{Z}_n(K) := \mathbf{ker}(\partial_n)$ of $\mathbf{C}_n(K)$. An n-boundary is an n-chain that is the boundary of $(n+1)$ chains. The set of all n-boundaries forms a subspace $\mathbf{B}_n(K) := \mathbf{im}(\partial_{n+1})$ of $\mathbf{C}_n(K)$. We refer to $\mathbf{Z}_n$ and $\mathbf{B}_n$ as the space of cycles and space of boundaries, respectively.

It can be shown that $\partial_n \circ \partial_{n+1}(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathbf{C}_{n+1}(K)$; colloquially, "a boundary has no boundary." Equivalently, $\partial_n \circ \partial_{n+1}$ is the zero map. Since the boundary map takes a boundary to 0, an n-boundary must also be an n-cycle. Therefore, $\mathbf{B}_n(K) \subseteq \mathbf{Z}_n(K)$.

2.4 Homology, Cycle Representatives

The nth homology group of K is defined as the quotient.

$$\mathbf{H}_n(K) := \mathbf{Z}_n(K)/\mathbf{B}_n(K).$$

Concretely, elements of $\mathbf{H}_n(K)$ are cosets of the form $[\mathbf{z}] = \{\mathbf{z}' \in \mathbf{Z}_n(K) \,|\, \mathbf{z}' - \mathbf{z} \in \mathbf{B}_n(K)\}$.[3] An element $h \in \mathbf{H}_n(K)$ is called an n-dimensional homology class. We say that a cycle $\mathbf{z} \in \mathbf{Z}_n(K)$ represents h, or that $\mathbf{z}$ is a cycle representative of h if $h = [\mathbf{z}]$. We say that $\mathbf{z}$ and $\mathbf{z}'$ are homologous if $[\mathbf{z}] = [\mathbf{z}']$.

Example: Consider the example in **Figure 2A**, which illustrates two homologous 1-cycles and the example in **Figure 2B**, which illustrates two non-homologous cycles.

Remark 2.4. The term homological generator has been used differently by various authors: to refer to an arbitrary nontrivial homology class, an element in a (finite) representation of $\mathbf{H}_n(K)$, as a set of cycles which generate the homology group, or (particularly in literature surrounding optimal cycle representatives) interchangeably with cycle representative. We favor the term cycle representative, to avoid ambiguity.

2.5 Betti Numbers, Cycle Bases

A (dimension-n) homological cycle basis for $\mathbf{H}_n(K)$ is a set of cycles $\mathcal{B} = \{\mathbf{z}^1, \ldots, \mathbf{z}^m\}$ such that $[\mathbf{z}^i] \neq [\mathbf{z}^j]$ when $i \neq j$, and $\{[\mathbf{z}^1], \ldots, [\mathbf{z}^m]\}$ is a basis for $\mathbf{H}_n(K)$. Modulo boundaries, every n-cycle can be expressed as a unique linear combination in $\mathcal{B}$.

Homological cycle bases have several useful interpretations. It is common, for example, to think of a 1-cycle as a type of "loop," generalizing the intuitive notion of a loop as a simple closed curve to include more intricate structures, and to regard the operation of adding boundaries as a generalized form of "loop-deformation." Framed in this light, a homological cycle basis $\mathcal{B}$ for $\mathbf{H}_1(K)$ can be regarded as a basis for the space of

[1]The ℓ_0 "norm" is not a real norm as it does not satisfy the homogeneous requirement of a norm. For example, scaling a vector $\mathbf{x}$ by a constant factor does not change its ℓ_0 "norm."

[2]See **Remark 2.1** These choices of groups have a natural notion of absolute value.

[3]More generally, we denote the groups of cycles and boundaries with coefficients in G as $\mathbf{Z}_n(K; G)$ and $\mathbf{B}_n(K; G)$. The (*dimension-n*) *homology of K with coefficients in G* is $\mathbf{H}_n(K; G) = \mathbf{Z}_n(K; G)/\mathbf{B}_n(K; G)$.

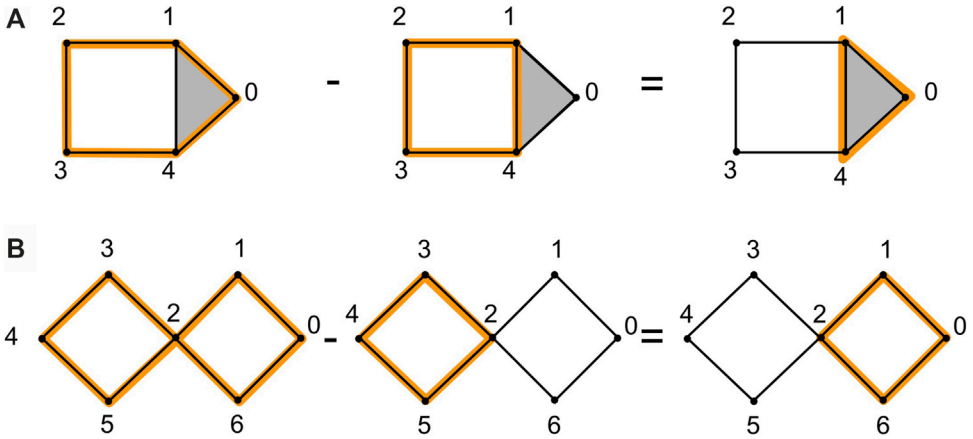


FIGURE 2 | We show an example of homologous cycles in **(A)**, adapted from (Topaz et al., 2015). The 1-cycle $(0,1)+(1,2)+(2,3)+(3,4)-(0,4)$ and the 1-cycle $(1,2)+(2,3)+(3,4)-(4,1)$ are homologous because their difference is the boundary of $(0,1,4)$. Subfigure **(B)** shows an example of non-homologous cycles. The 1-cycle $(\sum_{i=0}^{4}(i,i+1))-(5,2)+(2,6)-(0,6)$ and the 1-cycle $(2,3)+(3,4)+(4,5)-(2,5)$ are not homologous because their difference is a cycle $(0,1)+(1,2)+(2,6)-(0,6)$ which is not a linear combination of boundaries of 2-simplices.

loops-up-to-deformation in K. Higher dimensional analogs of loops involve closed "shells" made up of n-simplices.

Another interpretation construes each nontrivial homology class $[\mathbf{z}] \neq 0$ as a hole in K. Such holes are "witnessed" by loops or shells that are not homologous to the zero cycle. Viewed in this light, $\mathbf{H}_n(K)$ can naturally be regarded as the space of $(n+1)$ dimensional holes in K. The rank of the nth homology group

$$\beta_n(K) := \dim(\mathbf{H}_n(K)) = \dim(\mathbf{Z}_n(K)) - \dim(\mathbf{B}_n(K)),$$

therefore quantifies the "number of gray independent holes" in K. We call β_n the nth Betti number of K.

Example: Consider the gray disks in **Figure 1** [similar to Carlsson (2009)] with different numbers of holes and cycle representatives.

2.6 Filtrations of Simplicial Complexes

A filtration on a simplicial complex K is a nested sequence of simplicial complexes $K_\bullet = (K_{\varepsilon_i})_{i\in\{1,\ldots,T\}}$ such that

$$K_{\varepsilon_1} \subseteq K_{\varepsilon_2} \subseteq \cdots \subseteq K_{\varepsilon_T} = K,$$

where $\varepsilon_1 < \cdots < \varepsilon_T$ are real numbers. A filtered simplicial complex is a simplicial complex equipped with a filtration $K_\bullet$.

Example Let X be a metric space with metric d, and let $\varepsilon_1 < \cdots < \varepsilon_T$ be an increasing sequence of non-negative real numbers. Then the sequence $K_\bullet = (K_{\varepsilon_i})_{i\in\{1,\ldots,T\}}$ defined by $K_{\varepsilon_i} = \mathbf{VR}_{\varepsilon_i}(X)$ is a filtration on K.

The data of a filtered complex is naturally captured by the birth function on simplices, defined

$$\text{Birth} : K \to \mathbb{R},\ \sigma \mapsto \min\{\varepsilon_i : \sigma \in K_{\varepsilon_i}\}.$$

We regard the pair (K, Birth) as a simpilicial complex whose simplices are weighted by the birth function. For convenience, we will implicitly identify the sequence $K_\bullet$ with this weighted complex. Thus, for example, when we say that $\sigma \in K$ has birth parameter t, we mean that $\sigma \in K$ and $\text{Birth}(\sigma) = t$.

Definition 2.5. A filtration $K_\bullet$ is simplex-wise if one can arrange the simplices of K into a sequence $(\sigma_1, \ldots, \sigma_{|K|})$ such that $K_{\varepsilon_i} = \{\sigma_1, \ldots, \sigma_i\}$ for all i. A simplex-wise refinement of $K_\bullet$ is a simplex-wise filtration $K'_\bullet$ such that each space in $K_\bullet$ can be expressed in form $\{\sigma_1, \ldots, \sigma_j\}$ for some j.

As an immediate corollary, given a simplex-wise refinement of $K_\bullet$, we may naturally interpret each boundary matrix ∂_n as an element of $G^{|\mathbf{S}_{n-1}(K)|\times|\mathbf{S}_n(K)|}$, see **Remark 2.3** Under this interpretation, columns (respectively, rows) with larger indices correspond to simplices with later birth times; that is, birth time increases as one moves left-to-right and top-to-bottom.

2.7 Filtrations of Chain Complexes

If we regard $\mathbf{C}_n(K_{\varepsilon_i}; G)$ as a family of formal linear combinations in $\mathbf{S}_n(K_{\varepsilon_i})$, then it is natural to consider $\mathbf{C}_n(K_{\varepsilon_i}; G)$ as a subgroup of $\mathbf{C}_n(K_{\varepsilon_j}; G)$ for all $i < j$. In particular, we have an inclusion map

$$\iota : \mathbf{C}_n(K_{\varepsilon_i}; G) \to \mathbf{C}_n(K_{\varepsilon_j}; G),$$
$$\textstyle\sum_{\sigma\in\mathbf{S}_n(K_{\varepsilon_i})} x_\sigma \sigma \mapsto \sum_{\sigma\in\mathbf{S}_n(K_{\varepsilon_i})} x_\sigma \sigma + \sum_{\tau\notin\mathbf{S}_n(K_{\varepsilon_i})} 0\cdot\tau.$$

Given a simplex-wise refinement $K'_\bullet$, one can naturally regard $\mathbf{c}$ as an element $(c_1, c_2, \ldots)$ of $G^{|\mathbf{S}_n(K_{\varepsilon_i})|}$. From this perspective, ι has a particularly simple interpretation, namely "padding" by zeros:

$$\iota(\mathbf{c}) = (\underbrace{c_1, c_2, \ldots}_{\mathbf{c}}, 0, \ldots, 0).$$

Similar observations hold when one replaces $\mathbf{C}_n$ with either $\mathbf{Z}_n$, the space of cycles, or $\mathbf{B}_n$, the space of boundaries.

2.8 Persistent Homology, Birth, Death

The notion of birth for simplices has a natural extension to chains, as well as a variant called death. Formally, the birth and death parameters of $\mathbf{c} \in \mathbf{C}_n(K)$ are

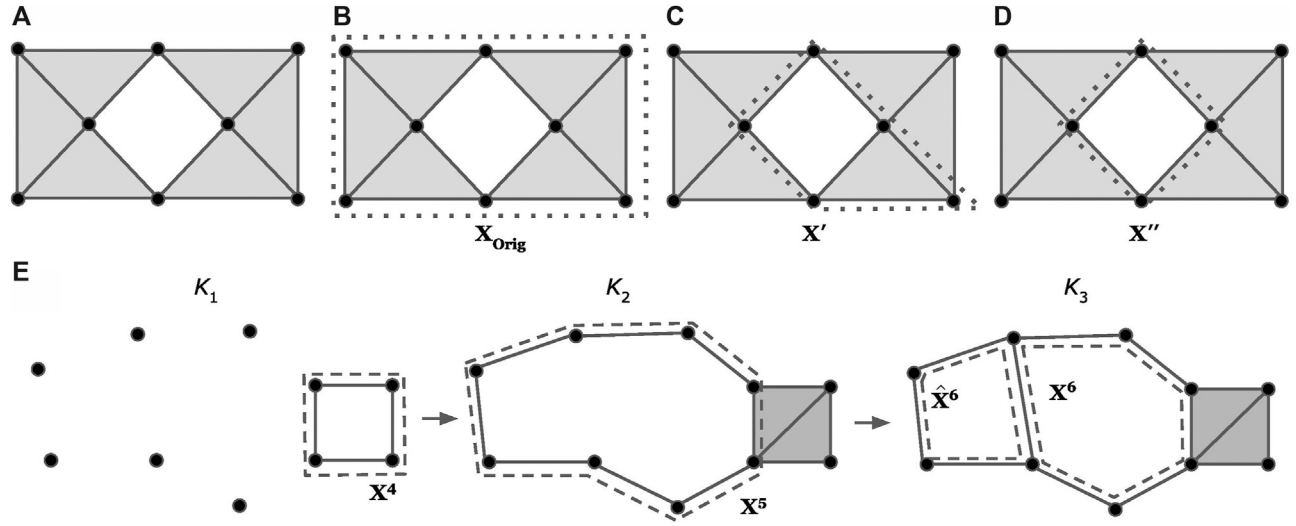


FIGURE 3 | Examples of optimizing a cycle representative (using the notion of minimizing edges) within the same homology class **(A-D)** and using a basis of cycle representatives **(E)**, modified examples adapted from (Escolar and Hiraoka, 2016; Obayashi, 2018). The dotted lines represent a cycle representative for the enclosed "hole." Intuitively, we consider $\mathbf{x}''$ in **(D)** as the optimal cycle representative since it consists of the smallest number of edges. Subfigure **(E)** shows a case where we optimize a cycle representative using a basis of cycle representatives. In **(E)**, $\{\mathbf{x}^4, \mathbf{x}^5, \mathbf{x}^6\}$ is the original basis of cycle representatives. We can substitute $\mathbf{x}^6$ with $\hat{\mathbf{x}}^6$, which we can obtain by adding $\mathbf{x}^5$ to $\mathbf{x}^6$, and thus obtain $\{\mathbf{x}^4, \mathbf{x}^5, \hat{\mathbf{x}}^6\}$ as the new basis of cycle representatives.

$$\text{Birth}(\mathbf{c}) = \min\{\varepsilon_i : \mathbf{c} \in \mathbf{C}_n(K_{\varepsilon_i})\}$$

$$\text{Death}(\mathbf{c}) = \begin{cases} \min\{\varepsilon_i : \mathbf{c} \in \mathbf{B}(K_{\varepsilon_i})\} & \mathbf{c} \in \mathbf{B}(K) \\ \infty & \text{else.} \end{cases}$$

In the special case where $\mathbf{c}$ is a cycle, Birth($\mathbf{c}$) is the first parameter value where [$\mathbf{c}$] represents a homology class, and Death($\mathbf{c}$) is the first parameter value where [$\mathbf{c}$] represents the zero homology class. Thus, the half-open lifespan interval

$$\mathcal{L}(\mathbf{c}) = [\text{Birth}(\mathbf{c}), \text{Death}(\mathbf{c})),$$

is the range of parameters over which $\mathbf{c}$ represents a well-defined, nonzero homology class.

A (dimension-n) persistent homology cycle basis is a subset $\mathcal{B} \subseteq \mathbf{Z}_n(K)$ with the following two properties:

1. Each $\mathbf{z} \in \mathcal{B}$ has a nonempty lifespan interval.
2. For each $i \in \{1, \ldots, T\}$, the set

$$\mathcal{B}_{\varepsilon_i} := \{\mathbf{z} \in \mathcal{B} : \varepsilon_i \in \mathcal{L}(\mathbf{z})\},$$

is a homological cycle basis for $\mathbf{H}_n(K_{\varepsilon_i})$.

Every filtration of simplicial complexes $(K_{\varepsilon_i})_{i \in \{1,\ldots,T\}}$ admits a persistent homological cycle basis $\mathcal{B}$ (Zomorodian and Carlsson, 2005). Moreover, it can be shown that the multiset of lifespan intervals (one for each basis vector), called the dimension-n barcode of $K_\bullet$,

$$\text{Barcode}_n = \{\mathcal{L}(\mathbf{z}) : \mathbf{z} \in \mathcal{B}\},$$

is invariant over all possible choices of persistent homological cycle bases $\mathcal{B}$ (Zomorodian and Carlsson, 2005).

Example: Consider the sequence of simplicial complexes (K_1, K_2, K_3) shown in **Figure 3E**. The set $\mathcal{B} = \{\mathbf{x}^4, \mathbf{x}^5, \mathbf{x}^6\}$ is a (dimension-1) persistent homological cycle basis of the filtration. The associated dimension-1 barcode is $\text{Barcode}_1 = \{[1, 2), [2, \infty), [3, \infty)\}$ where $[2, \infty)$ and $[3, \infty)$ are the lifespans of $\mathbf{x}^5$ and $\mathbf{x}^6$, respectively.

Barcodes are among the foremost tools in topological data analysis (Ghrist, 2008; Edelsbrunner and Harer, 2008), and they contain a great deal of information about a filtration. For example, it follows immediately from the definition of persistent homological cycle bases that $\beta_n(K_{\varepsilon_i}) = |\mathcal{B}_{\varepsilon_i}|$ for all n and i. Consequently,

$$\beta_n(K_{\varepsilon_i}) = |\{J \in \text{Barcode}_n : \varepsilon_i \in J\}|.$$

2.9 Computing PH Cycle Representatives

Barcodes and persistent homology bases may be computed via the so-called $R = DV$ decomposition (Cohen-Steiner et al., 2006) of the boundary matrices ∂_n. Details are discussed in the **Supplementary Material**.

3 RELATED WORK ON MINIMIZING CYCLE REPRESENTATIVES

One important problem in TDA is interpreting homological features. In general, a lifetime interval $\mathcal{L}$ corresponding to a feature may be represented by many different cycle representatives. As discussed in Chen et al. (2008), localizing homology classes can be characterized as finding a representative cycle with the most concise geometric measure. As an illustrative example from Escolar and Hiraoka (2016), **Figure 3A** shows a simplicial complex K with $\mathbf{H}_1(K)$ isomorphic to $\mathbb{Q}$ or equivalently, $\beta_1 = 1$; it contains one hole. **Figures 3B–D** display three cycle representatives, $\mathbf{x}^{Orig}$, $\mathbf{x}'$, and $\mathbf{x}''$, each of

which represents the same homology class (heuristically, they encircle the same hole). We intuitively prefer $\mathbf{x}''$ as a representative, since it involves the fewest edges and "hugs" the hole most tightly. Given a simplicial complex K and a nontrivial cycle $\mathbf{x}^{Orig}$ on it, we are interested in finding a cycle representative that is optimal with respect to some geometric criterion. In this section, we discuss previous studies on optimal cycle representatives.

Minimal cycle representatives have proven useful in many applications. Hiraoka et al. (2016) use TDA to geometrically analyze amorphous solids. Their analysis using minimal cycle representatives explicitly captures hierarchical structures of the shapes of cavities and rings. Wu et al. (2017) discuss an application of optimal cycles in Cardiac Trabeculae Restoration, which aims to reconstruct trabeculae, very complex muscle structures that are hard to detect by traditional image segmentation methods. They propose to use topological priors and cycle representatives to help segment the trabeculae. However, the original cycle representative can be complicated and noisy, causing the reconstructed surface to be messy. Optimizing the cycle representatives makes the cycle more smooth and thus, leads to more accurate segmentation results. Emmett et al. (2015) use PH to analyze chromatin interaction data to study chromatin conformation. They use loops to represent different types of chromatin interactions. To annotate particular loops as interactions, they need to first localize a cycle. Thus, they propose an algorithm to locate a minimal cycle representative for a given PH class using a breadth-first search, which finds the shortest path that contains the edge that enters the filtration at the birth time of the cycle and is homologically independent from the minimal cycles of all PH classes born before the current cycle.

There are several approaches used to define an optimal cycle representative. Dey et al. (2011) propose an algorithm to find an optimal homologous 1-cycle for a given homology class via linear programming. That is, they consider a single homology class $[\mathbf{x}]$ and search for a homologous cycle representative that minimizes some geometric measure within that class, for instance, the number of 1-simplices within the representative. Escolar and Hiraoka (2016) extend this approach to find an optimal cycle by using cycles outside of a single homology class to "factor out" redundant information. In this approach, an optimal cycle representative is no longer guaranteed to be homologous to the original representative, but the collection of cycle representatives have each been independently optimized and the collection still forms a homology basis. Further, Escolar and Hiraoka (2016) extends this approach to achieve a filtered cycle basis, although we note that it is not guaranteed to be a persistent homology basis. The two approaches in Dey et al. (2011) and Escolar and Hiraoka (2016) aim to minimize the number of 1-simplices in a cycle representative. Obayashi (2018) proposes an alternative algorithm for finding volume-optimal cycles in persistent homology, which minimize the number of 2-simplices which the cycle representative bounds, also using linear programming. These methods serve as the foundation for our present paper and are discussed in more detail in the rest of this section.

In addition to linear programming, many researchers have contributed to the problem of computing optimal cycles: Wu et al. (2017) propose an algorithm for finding shortest persistent 1-cycles. They first construct a graph based on the given simplicial complex and then compute annotation for the given complex. The annotation assigns all edges different vectors and can be used to verify if a cycle belongs to the desired group of cycles. They then find the shortest path between two vertices of the edge born at the birth time of the original cycle representative using a new A^* heuristic search strategy. Their algorithm is a polynomial time algorithm but in the worst case, the time complexity is exponential to the number of topological features. Dey et al. (2010) propose a polynomial-time algorithm that computes a set of loops from a VR complex of the given data whose lengths approximate those of a shortest basis of the one dimensional homology group $\mathbf{H}_1$. In Dey et al. (2019), show that finding optimal (minimal) persistent 1-cycles is NP-hard and then propose a polynomial time algorithm to find an alternative set of meaningful cycle representatives. This alternative set of representatives is not always optimal but still meaningful because each persistent 1-cycle is a sum of shortest cycles born at different indices. They find shortest cycles using Dijkstra's algorithm by considering the 1-skeleton as a graph. This list is by no means exhaustive, and does not touch on the wide variety of related approaches, e.g. Chen and Freedman (2010b), which attempts to fit cycle representatives within a ball of minimum radius.

In the next subsection, we briefly introduce some basic notions of linear programming, and then in the subsequent three subsections, we survey the optimization problems on which the present work is based.

3.1 Background: Linear Programming

Linear programming seeks to find a set of decision variables $\mathbf{x} = (x_1, \ldots, x_\eta)^T$ which optimize a linear cost (or objective) function $\mathbf{c}^T\mathbf{x}$ subject to a set of linear (in)equality constraints $\mathbf{a}_1^T\mathbf{x} = b_1, \ldots, \mathbf{a}_\mu^T\mathbf{x} = b_\mu$. Any linear optimization problem can be written as a Linear Program (LP) in standard form

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0, \end{array} \tag{1}$$

where A is the $\mu \times \eta$ matrix with coefficients of the constraints as rows and $\mathbf{b} = (b_1, \ldots, b_\mu)^T$. Linear programming is well-studied and discussed in many texts (Bertsimas and Tsitsiklis, 1997; Vanderbei, 2014; Boyd and Vandenberghe, 2004).

The optimal solution $\mathbf{x}^\star$ satisfies the constraints while optimizing the objective function, yielding the optimal cost $\mathbf{c}^T\mathbf{x}^\star$. The feasible set of solutions in a linear optimization problem is a polyhedron defined by the linear constraints. In general, the optimal solution of a (non-degenerate) LP will occur at a vertex of the polyhedron and can be solved with the standard simplex algorithm, which traverses through the edges of the polytope to vertices in a cost reducing manner, or interior point methods, which traverse along the inside of the polytope to reach an optimal vertex. In the worst-case, the complexity of

the simplex method is exponential, yet it often runs remarkably fast, while interior point methods are polynomial time algorithms.

Standard LPs search for real-valued optimal solutions, but in some instances, a restriction of the decision variables, such as requiring integral solutions, may be necessitated. The mixed integer programming (MIP) problem is written

$$\begin{array}{ll} \text{minimize} & \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{y} \\ \text{subject to} & A\mathbf{x} + B\mathbf{y} = \mathbf{b} \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0} \\ & \mathbf{x} \text{ integer}, \end{array} \tag{2}$$

for matrices A, B and vectors $\mathbf{b}, \mathbf{c}, \mathbf{d}$. A standard LP has fewer constraints, and thus, will have optimal cost less than or equal to that of the analogous MIP. MIPs are much more challenging to solve than LPs, as they are discrete as opposed to convex optimization problems, and no efficient general algorithm is known (Bertsimas and Tsitsiklis, 1997). However, LP relaxations, (exponential-time) exact, (polynomial-time) approximation, and heuristic algorithms can be used to obtain solutions to MIPs.

In this paper, we determine optimal cycle representatives with both LP and MIP formulations.

3.2 Minimal Cycle Representatives of a Homology Class

Given a homology class $h = [\mathbf{x}^{Orig}] \in \mathbf{H}_n(K; G)$ and a function $\text{loss}\colon \mathbf{Z}_n(K; G) \to \mathbb{R}$, how does one find a cycle representative of h on which loss attains minimum? This problem is equivalent to solving the following program defined in Dey et al. (2011):

$$\begin{array}{ll} \text{minimize} & \text{loss}(\mathbf{x}) \\ \text{subject to} & \mathbf{x} = \mathbf{x}^{Orig} + \partial_{n+1}\mathbf{w} \\ & \mathbf{w} \in \mathbf{C}_{n+1}(K; G). \end{array} \tag{3}$$

This formulation considers all cycle representatives homologous to $\mathbf{x}^{\text{Orig}}$, i.e. that differ by a boundary, and selects the optimal representative $\mathbf{x}$ which minimizes loss. The program in **Eq. 3** is correct because the coset h can be expressed in the form

$$h = \mathbf{x}^{\text{Orig}} + \mathbf{B}_n(K; G) = \{\mathbf{x}^{Orig} + \partial_{n+1}\mathbf{w} | \mathbf{w} \in \mathbf{C}_{n+1}(K; G)\}.$$

In practice, a cycle representative $\mathbf{x}^{Orig}$ is almost always provided together with the initial problem data (which consists of K, G, loss, and h), so the central challenge lies with solving the program in **Eq. 3**.

Several variants of the program in **Eq. 3** have been studied, especially where $\text{loss}(\mathbf{x}) = \|\mathbf{x}\|_0$ or $\text{loss}(\mathbf{x}) = \|\mathbf{x}\|_1$. For a survey of results when $G = \mathbb{F}_2$, see Chen and Freedman (2010b). For a discussion of results when $G = \mathbb{Z}$, see Dey et al. (2011). Broadly speaking, minimizing against ℓ_0 tends to be hard, even when K has attractive properties such as embeddability in a low-dimensional Euclidean space (Borradaile et al., 2020). Minimizing against ℓ_1 is hard when $G = \mathbb{F}_2$ (since, in this case, $\ell_1 = \ell_0$), but tractable via linear programming when $G \in \{\mathbb{Q}, \mathbb{R}\}$.

An interesting variant of the minimal cycle representative problem is the minimal persistent cycle representative problem. This problem was described in Chen et al. (2008) and may be formulated as follows: given an interval $[a, b) \in \text{Barcode}_n(K_\bullet)$, solve

$$\begin{array}{ll} \text{minimize} & \text{loss}(\mathbf{x}) \\ \text{subject to} & \text{Birth}(\mathbf{x}) = a \\ & \text{Death}(\mathbf{x}) = b \\ & \mathbf{x} \in \mathbf{Z}_n(K_a; G), \end{array} \tag{4}$$

for $\mathbf{x}$. An advanced treatment of this problem can be found in (Chen et al., 2008) for special case where 1) $G = \mathbb{F}_2$, 2) loss is a weighted sum of incident edges, and 3) the birth function assigns distinct values to any two simplices of the same dimension, and 4) $n = 1$.

3.3 Minimal Homological Cycle Bases

The program in **Eq. 3** has a natural extension when G is a field. This extension focuses not on the smallest representative of a single homology class, but the smallest homological cycle basis. It may be formally expressed as follows:

$$\begin{array}{ll} \text{minimize} & \sum_{\mathbf{x} \in \mathcal{B}} \text{loss}(\mathbf{x}) \\ \text{subject to} & \mathcal{B} \in \text{HCB}_n(K; G), \end{array} \tag{5}$$

where $\text{HCB}_n(K, G)$ is the family of dimension-n homological cycle bases of $\mathbf{H}_n(K; G)$. Thus, the program is finding a complete generating set $\mathcal{B}$ for all of the homological cycles of dimension n where each element has been minimized in some sense.

It is natural to wonder whether a solution to the program in **Eq. 5** could be obtained by first calculating an arbitrary (possibly non-minimal) homological cycle basis $\mathcal{B} = \{\mathbf{x}^1, \ldots, \mathbf{x}^m\}$ and then selecting an optimal cycle representative $\mathbf{z}^i$ from each homology class $[\mathbf{x}^i]$. Unfortunately, the resulting basis need not be optimal. To see why, consider the simplicial complex K_3 shown in **Figure 3E**, taking G to be $\mathbb{Q}$ and loss to be the ℓ_0 norm. Complex $K_\bullet$ has several different homological cycle bases in degree 1, including $\mathcal{B}_0 := \{\hat{\mathbf{x}}^6, \mathbf{x}^6\}$, $\mathcal{B}_1 := \{\mathbf{x}^5, \mathbf{x}^6\}$, and $\mathcal{B}_2 := \{\mathbf{x}^5, \hat{\mathbf{x}}^6 + \mathbf{x}^4\}$. However, only $\mathcal{B}_0$ is ℓ_0 minimal. Moreover, each of the cycle representatives $\mathbf{x}^5, \mathbf{x}^6, \hat{\mathbf{x}}^6$ is already minimal within its homology class, so element-wise minimization will not transform $\mathcal{B}_1$ or $\mathcal{B}_2$ into optimal bases, as might have been hoped.

As with the minimal cycle representative problem, the minimal homological cycle basis problem has been well-studied in the special case where loss is the ℓ_0 norm and $G = \mathbb{F}_2$. In this case, the program in **Eq. 5** is NP-hard to approximate for $n > 1$, but $O(n^3)$ when $n = 1$ (Dey et al., 2018). Several interesting variants and special cases have been developed in the $n = 1$ case, as well Erickson and Whittlesey (2009), Dey et al. (2010), and Chen and Freedman (2010). We are not currently aware of a systematic treatment for the case $G \in \{\mathbb{Q}, \mathbb{R}\}$.

A natural variant of the minimal homological cycle basis program in **Eq. 5** is the minimal persistent homological cycle basis problem

$$\begin{array}{ll} \text{minimize} & \sum_{\mathbf{x} \in \mathcal{B}} \text{loss}(\mathbf{x}) \\ \text{subject to} & \mathcal{B} \in \text{PrsHCB}_n(K_\bullet; G), \end{array} \tag{6}$$

where $\text{PrsHCB}_n(K_\bullet; G)$ is the set of persistent homological cycle bases. This is a stricter condition than the program in **Eq. 5** in that not only does it require that the elements of $\mathcal{B}$ form a generating

set of all cycles of dimension n, but the barcode associated to $\mathcal{B}$ must match $\text{Barcode}_n(K_\bullet)$. That is, the multisets of birth/death pairs must be identical.

The program in **Eq. 6** is much more recent than the program in **Eq. 5**, and consequently appears less in the literature. In the special case where every bar in the multiset $\text{Barcode}_n(K_\bullet)$ has multiplicity 1 (i.e. there are no duplicate bars), the program in **Eq. 6** can be solved by making one call to the minimal persistent cycle representative program in **Eq. 4** for each bar. In particular, the method of Chen et al. (2008) may be applied to obtain a minimal persistent basis when the correct hypotheses are satisfied: $G = \mathbb{F}_2$, loss is a weighted sum of incident simplices, there are distinct birth times for all simplices of the same dimension, and $n = 1$. In general, however, bars of multiplicity two are possible, and in this case repeated application of the program in **Eq. 4** will be insufficient.

3.4 Minimal Filtered Cycle Space Bases

A close cousin of the minimal homological cycle basis the program in **Eq. 5** is the minimal filtered cycle basis problem, which may be formulated as follows

$$\begin{array}{ll}\text{minimize} & \sum_{\mathbf{x}\in\mathcal{C}}\text{loss}(\mathbf{x}) \\ \text{subject to} & \mathcal{C}\in\text{FCB}(K_\bullet;G),\end{array} \tag{7}$$

where $\text{FCB}(K_\bullet)$ is the family of all bases $\mathcal{C}$ of $\mathbf{Z}_n(K_{\varepsilon_T})$ such that $\mathcal{C}$ contains a basis for each subspace $\mathbf{Z}_n(K_{\varepsilon_i})$, for $i \in \{1,\dots,T\}$.

Escolar and Hiraoka (2016) provide a polynomial time solution via linear programming when.

1. loss is the ℓ_1 norm,
2. $G = \mathbb{Q}$, and
3. $K_\bullet$ is a simplex-wise filtration [without loss of generality, $K_\bullet = (K_1,\dots,K_T)$].

Their key observation is that $\mathcal{C}$ is an optimal solution to the program in **Eq. 6** if and only if $\mathcal{C}$ can be expressed as a collection $\{\mathbf{z}^j : j \in J\}$ where

1. the set $J = \{j : \mathbf{Z}_n(K_{j-1}) \subsetneq \mathbf{Z}_n(K_j)\}$ that indexes the cycles is the list of filtrations at which a novel n-cycle appears, and.
2. for each $j \in J$, the cycle $\mathbf{z}^j$ first appears in K_j and is a minimizer for the loss function among all such cycles, i.e. $\mathbf{z}^j \in \text{argmin}_{\mathbf{z}\in\mathbf{Z}_n(K_j)\setminus\mathbf{Z}_n(K_{j-1})}\text{loss}(\mathbf{z})$.

The authors formulate this problem as

$$\begin{array}{ll}\text{minimize} & \|\mathbf{x}\|_1 \\ \text{subject to} & \mathbf{x} = \mathbf{x}^{Orig} + \sum_{r\in R} w_r g^r + \sum_{s\in S} v_s f^s \\ & \mathbf{w}\in\mathbb{Q}^R \\ & \mathbf{v}\in\mathbb{Q}^S,\end{array} \tag{8}$$

where $\mathbf{x}^{Orig} \in \mathbf{Z}_N(K_j)\setminus\mathbf{Z}_N(K_{j-1})$ is a novel cycle representative at filtration j; $\{g^r : r \in R\}$ is a basis for $\mathbf{B}_n(K_{j-1})$[4]; and $\{g^r : r \in R\} \cup \{f^s : s \in S\}$ is an extension of the given basis for $\mathbf{B}_n(K_{j-1})$ to a basis for $\mathbf{Z}_n(K_{j-1})$. That is, $\mathbf{x}^{Orig}$ is a cycle that has just appeared in the filtration. To optimize it, we are allowed to consider linear combinations of both boundaries, $\{g^r\}$, and cycles, $\{f^s\}$, born before $\mathbf{x}^{Orig}$. The cycle $\mathbf{x}$ obtained in this way cannot have a birth time before that of $\mathbf{x}^{Orig}$, but may have a different death time if $[\sum_{s\in S} v_s f^s]$ dies later than $[\mathbf{x}^{Orig}]$.

The algorithm developed in Escolar and Hiraoka (2016) is cleverly constructed to extract $\mathbf{x}^{Orig}$, $\{g^r : r \in R\}$, and $\{f^s : s \in S\}$ from matrices which are generated in the normal course of a barcode calculation.

Remark 3.1. It is important to distinguish between PrsHCB and FCB, hence between the optimization the programs in **Eqs 6**, **7**. As Escolar and Hiraoka (2016) point out, given $\mathcal{B} \in$ PrsHCB and $\mathcal{C} \in$ FCB, one can always find an injective function $\phi : \mathcal{B} \to \mathcal{C}$ such that $\text{Birth}(\mathbf{z}) = \text{Birth}(\phi(\mathbf{z}))$ for all $\mathbf{z}$. However, this does not imply that $\phi(\mathcal{B}) \in$ PrsHCB, as the deaths of each cycle may not coincide. Indeed, the question of whether a persistent homological cycle basis can be extracted from $\mathcal{C}$ by any means is an open question, so far as we are aware. We provide an example in **Figure 4** where the cycle basis obtained by optimizing each cycle using the program in **Eq. 7** is not a persistent homology cycle basis $\mathcal{B}$.

Though **Remark 3.1** is a bit disappointing for those interested in persistent homology, the machinery developed to study the program in **Eq. 7** is nevertheless interesting, and we will discuss an adaptation.

3.5 Volume-Optimal Cycles: Minimizing Over Bounding Chains

Schweinhart (2015) and Obayashi (2018) consider a different notion of minimization: volume[5] optimality. This approach focuses on the "size" of a bounding chain; it is specifically designed for cycle representatives in a persistent homological cycle basis.

Obayashi (2018) formalizes the approach as follows. First, assume a simplex-wise filtration $K_\bullet$; without loss of generality, $K_\bullet = (K_1,\dots,K_T)$, and we may enumerate the simplices of K_T such that $K_i = \{\sigma_1,\dots,\sigma_i\}$ for all i. Since each simplex has a unique birth time, each interval in $\text{Barcode}_n(K_\bullet) = \{[b_1,d_1),\dots,[b_N,d_N)\}$ has a unique left endpoint. Fix $[b_i,d_i) \in \text{Barcode}_n(K_\bullet)$ such that $d_i < \infty$ (in the case $d_i = \infty$, volume is undefined). It can be shown that σ_{b_i} is an n-simplex and σ_{d_i} is an $(n+1)$ simplex. We use $\tau_k = \sigma_k$ below when the dimension of σ_k is equal to $n+1$.

A persistent volume $\mathbf{v}$ for $[b_i,d_i)$ is an $(n+1)$ chain $\mathbf{v} \in \mathbf{C}_{n+1}(K_{d_i})$ such that[6]

$$\mathbf{v} = \tau_{d_i} + \sum_{\tau_k\in\mathcal{F}_{n+1}} \alpha_k \tau_k \tag{9}$$

$$(\partial_{n+1}\mathbf{v})_\sigma = 0 \quad \forall\sigma \in \mathcal{F}_n \tag{10}$$

$$(\partial_{n+1}\mathbf{v})_{\sigma_{b_i}} \neq 0, \tag{11}$$

[4]Because of the assumption that $K_\bullet$ is a simplex-wise filtration, if there is a new n-cycle in K_j then there cannot also be a new $(n+1)$ simplex, so this is also a basis for $\mathbf{B}_n(K_j)$.

[5]This notion of volume differs from that of Chen and Freedman (2010b). The latter refers to volume as the ℓ_0 norm of a chain, while the former (which we discuss in this section) refers to the ℓ_0 norm of a *bounding* chain.

[6]If we regard $\partial_{n+1}\mathbf{v}$ as a function $\mathbf{S}_n(K_{d_i}) \to \mathbb{Q}$, then $(\partial_{n+1}\mathbf{v})_\sigma$ is the value taken by $\partial_{n+1}\mathbf{v}$ on simplex σ. Alternatively, if we regard $\partial_{n+1}\mathbf{v}$ as a linear combination of n-simplices, then $(\partial_{n+1}\mathbf{v})_\sigma$ is the coefficient placed by $\partial_{n+1}\mathbf{v}$ on σ.

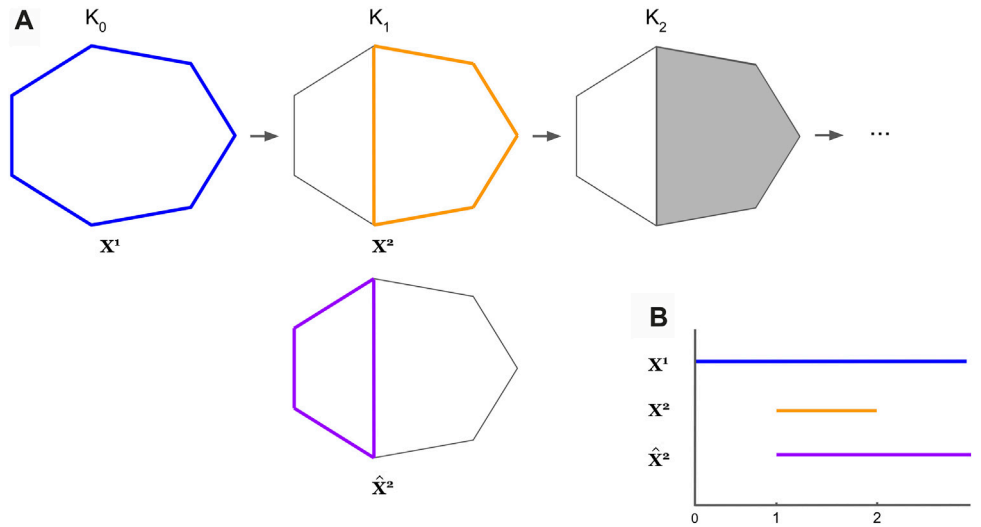


FIGURE 4 | An example where the optimal cycles obtained from **Eq. 8** do not form a persistent homological cycle basis. The thickened colored cycles in Subfigure **(A)** represent a cycle representative for the hole it encloses, and the bar with the corresponding color in Subfigure **(B)** records the lifespan of the cycle. In Subfigure **(A)**, we see $\mathcal{L}(\mathbf{x}^1) = [0, \infty)$, $\mathcal{L}(\mathbf{x}^2) = [1, 2)$ Then, $\{\mathbf{x}^1, \mathbf{x}^2\}$ forms a basis for the persistent homological cycles. The cycle representative $\tilde{\mathbf{x}}^2$ is an optimal cycle representative obtained by solving **Eq. 7** for the filtered simplicial complex K_2. However, $\mathcal{L}(\tilde{\mathbf{x}}^2) = [1, \infty)$, and thus $\{\mathbf{x}^1, \tilde{\mathbf{x}}^2\}$ is no longer a persistent homological cycle basis.

where $\mathcal{F}_n = \{\sigma_k \in \mathbf{S}_n(K) : b_i < k < d_i\}$ denotes the n-simplices alive in the window between the birth and death time of the interval under consideration.

We interpret these equations as follows: Given a persistence interval $[b_i, d_i)$, condition **Eq. 9** implies that $\mathbf{v}$ only contains $n+1$ simplices born between b_i and d_i and must contain the $n+1$ simplex born at d_i. Condition (**Eq. 10**) ensures that the boundary of $\mathbf{v}$ contains no n-simplex born after b_i, and condition (**Eq. 11**) ensures that the boundary of $\mathbf{v}$ contains the n-simplex born at b_i. This guarantees that $\partial_{n+1}\mathbf{v}$ exists at step b_i, does not exist before step b_i, and dies at step d_i.

Theorem 3.2. (Obayashi, 2018). Suppose that $[b_i, d_i) \in \text{Barcode}_n(K_\bullet)$ and $d_i < \infty$.

1. Interval $[b_i, d_i)$ has a persistent volume.
2. If $\mathbf{v}$ is a persistent volume for $[b_i, d_i)$ then $\mathcal{L}(\partial_{n+1}\mathbf{v}) = [b_i, d_i)$.
3. Suppose that $\mathcal{B}$ is an n-dimensional persistent homological cycle basis for $K_\bullet$, that $\mathbf{x}^{Orig} \in \mathcal{B}$ is the basis vector corresponding to $[b_i, d_i)$, and that $\mathbf{v}$ is a persistent volume for $[b_i, d_i)$. Then, $(\mathcal{B} \backslash \{\mathbf{x}^{Orig}\}) \cup \{\partial_{n+1}\mathbf{v}\}$ is also a persistent homological cycle basis.

By **Theorem 3.2**, for any barcode composed of finite intervals, one can construct a persistent homological cycle basis from nothing but (boundaries of) persistent volumes! Were we to build such a basis, it would be natural to ask for volumes that are optimal with respect to some loss function; that is, we might like to solve

$$\begin{array}{ll} \text{minimize} & \text{loss}(\mathbf{v}) \\ \text{subject to} & (9), (10), (11) \\ & \mathbf{v} \in \mathbf{C}_{n+1}(K_{d_i}), \end{array} \tag{12}$$

for each barcode interval $[b_i, d_i)$. A solution $\mathbf{v}$ to the program in **Eq. 12** is called an optimal volume; its boundary, $\mathbf{x} = \partial_{n+1}\mathbf{v}$ is called a volume-optimal cycle.

It is interesting to contrast ℓ_0 minimal cycle representatives for an interval[7] $[b_i, d_i)$ with ℓ_0 volume-optimal cycle for the same interval. Consider, for example, **Figure 5**. For the persistence interval $[b_i, d_i)$, the cycle with minimal number of edges is $(a,b) + (b,c) + (c,d) + (d,a)$. However, the volume-optimal cycle would be found as follows: considering K_{d_i}, we must find the fewest 2-simplices whose boundary captures the persistence interval. In this case, we would have an optimal volume $(a,b,e) + (b,c,e) + (a,d,e)$ and volume-optimal cycle $(a,b) + (b,c) + (c,e) + (e,d) + (d,a)$.

3.6 ℓ_0 vs. ℓ_1 Optimization

As mentioned above, it is common to choose $\text{loss}(\mathbf{x}) = \|\mathbf{x}\|_0$ or $\text{loss}(\mathbf{x}) = \|\mathbf{x}\|_1$.[8] A linear program (LP) with ℓ_1 objective function is polynomial time solvable. However, an objective function with the ℓ_0 norm restricted to {0, 1, -1} coefficients is often preferred as the output of such a problem is highly interpretable: a cycle representative with minimal number of edges or enclosing the minimal number of triangles. Yet, ℓ_0

[7]Technically, this notion is not well-defined; to be formal, we should fix a persistent homology cycle basis $\mathcal{B}$, fix a cycle representative $\mathbf{z} \in \mathcal{B}$ with lifespan interval $[b_i, d_i)$, and ask for an ℓ_0 cycle representative in the same homology class, $[\mathbf{z}] \in \mathbf{H}_n(K_{b_i})$, as per the program in **Eq. (3)**. However, in simple cases the intended meaning is clear.

[8]Other choices of loss function, e.g., the ℓ_p norm, are common throughout mathematical optimization. While we focus on ℓ_0 and ℓ_1 due to their tendency to produce sparse solutions, other choices may be better or worse suited, depending on the intended application. For example, since ℓ_2 loss imposes lighter penalties on small errors and heavier penalties on large ones (as compared to ℓ_1), it is especially sensitive to outliers; this makes it useful for tasks such as function estimation. On the other hand, by imposing relatively heavy penalties on small errors, ℓ_1 loss encourages sparsity (Tahbaz-Salehi and Jadbabaie, 2008; Tahbaz-Salehi and Jadbabaie, 2010).

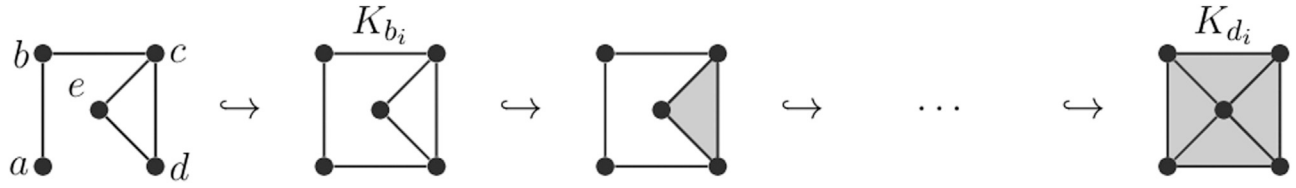


FIGURE 5 | A situation in which a volume-optimal cycle is different from the uniform minimal cycle. Consider the filtered simplicial complex pictured. For the persistence interval $[b_i, d_i)$, the cycle with minimal 0-norm (fewest number of edges) is $(a,b) + (b,c) + (c,d) + (d,a)$. However, the volume-optimal cycle would be found as follows: considering K_{d_i}, we must find the fewest 2-simplices whose boundary captures the persistence interval. In this case, we would have an optimal volume $(a,b,e) + (b,c,e) + (a,d,e)$ and volume-optimal cycle $(a,b) + (b,c) + (c,e) + (e,d) + (d,a)$.

optimization is known to be NP-hard (Tahbaz-Salehi and Jadbabaie, 2010).

The ℓ_1 norm promotes sparsity and often gives a good approximation of ℓ_0 optimization (Tahbaz-Salehi and Jadbabaie, 2008; Tahbaz-Salehi and Jadbabaie, 2010), but the solution may not be exact. Yet, if all of the coefficients of the solution x are restricted to 0 or ± 1 in the optimization problem, then the ℓ_0 and ℓ_1 norms are identical. A looser restriction, as proposed in Escolar and Hiraoka (2016), would be to solve an optimization with ℓ_1 objective function with integer constraints on the solution.

Requiring the solution to be integral also allows us to understand the optimal solution more intuitively than having fractional coefficients. Such an optimization problem is called a mixed integer program (MIP), which is known to be slower than linear programming and is NP-hard (Obayashi, 2018). Many variants of integer programming special to optimal homologous cycles, in particular, have been shown to be hard as well (Borradaile et al., 2020). In **Section 4**, we discuss the optimization problems we implement, where each is solved both as an LP with an ℓ_1 norm in the objective function and an MIP by adding the constraint that $\mathbf{x}$ is integral.

Dey et al. (2011) gives the totally unimodularity sufficient condition which guarantees that an LP and MIP give the same optimal solution. A matrix is totally unimodular if the determinant of each square submatrix is -1, 0, or 1. Dey et al. (2011) give conditions for when the ∂_{n+1} matrix is totally unimodular. If the totally unimodularity condition is not satisfied, then an LP may not give the desired result. As totally unimodularity is not guaranteed for all boundary matrices (Henselman and Dłotko, 2014), we cannot rely on this condition.

3.7 Software Implementations

Edge-minimal cycles: Software implementing the edge-loss method introduced in Escolar and Hiraoka (2016) can be found at Escolar and Hiraoka (2021). This is a C++ library specialized for 3-dimensional point clouds.

Triangle-loss optimal cycles: The volume optimization technique introduced in Obayashi (2018) is available through the software platform HomCloud, available at Obayashi et al. (2021). The code can be accessed by unarchiving the HomCloud package (for example, https://homcloud.dev/download/homcloud-3.1.0.tar.gz) and picking the file homcloud-x.y.z/homcloud/optvol.py.

4 PROGRAMS AND SOLUTION METHODS

The present work focuses on linear programming (LP) and mixed integer programming (MIP) optimization of 1-dimensional persistent homology cycle representatives with $\mathbb{Q}$ – coefficients. While the methods discussed below can be applied to any homological dimension, we limit the scope of the present work to dimension one. As described in **Section 3**, we follow two general approaches: those that measure loss as a function of n-simplices, and those that measure loss as a function of $n+1$ simplices. Motivated by the $n = 1$ case, we refer to the former as edge-loss methods and the latter as triangle-loss methods. For our empirical analysis, four variations (corresponding to two binary parameters) are chosen from each approach, yielding a total of eight distinct optimization problems.

Concerning implementation, we find that triangle-loss methods [namely, Obayashi (2018)] can be applied essentially as discussed in that paper. The greatest challenge to implementing this approach is the assumption of an underlying simplex-wise filtration. This necessitates parameter choices and preprocessing steps not included in the optimization itself; we discuss how to execute these steps below.

Implementation of edge-loss methods is slightly more complex. For binary coefficients ($G = \mathbb{F}_2$) a variety of combinatorial techniques have been implemented in dimension 1 (Chen et al., 2008; Zhang and Wu, 2019). Escolar and Hiraoka (2016) provide an approach for $\mathbb{Q}$ – coefficients, but in general this may not yield a persistent homology cycle basis, see **Remark 3.1**. In addition to the triangle-loss method mentioned in **Section 3.5**, Obayashi (2018) introduces a modified form of this edge-loss method which *does* guarantee a persistent homology basis, but assumes a simplex-wise filtration. We show that this approach can be modified to remove the simplex-wise filtered constraint.

Neither of the approaches presented here is guaranteed to solve the minimal persistent homology cycle basis problem, the program in **Eq. 6**. In the case of triangle-loss methods, this is due to the (arbitrary) choice of a total order on simplices. In the case of edge-loss methods, it is due to the choice of an initial persistent homology cycle basis.

In the remainder of this section, we present the eight programs studied, including any modifications from existing work.

4.1 Structural Parameters

Each program addressed in our empirical study may be expressed in the following form

$$\begin{aligned} \text{minimize} \quad & \|W\mathbf{x}\|_1 = \sum_{\sigma} W[\sigma,\sigma]\left(\mathrm{x}_\sigma^+ + \mathrm{x}_\sigma^-\right) \\ \text{subject to} \quad & \mathbf{x} = \mathrm{x}^+\text{-}\mathrm{x}^- \\ & \mathbf{x}^+, \mathbf{x}^- \geq 0 \\ & \mathbf{x} \in \mathcal{X}, \end{aligned} \tag{13}$$

where $\mathcal{X}$ is a space of feasible solutions and W is a diagonal matrix with nonnegative entries. These programs vary along 3 parameters:

1. *Chain dimension of* $\boldsymbol{x}$. If $\mathcal{X}$ is a family of 1-chains, then we say that the program in **Eq. 13** is an edge-loss program. If $\mathcal{X}$ is a family of 2-chains, we say that the program in **Eq. 13** is a triangle-loss program.
2. *Integrality*. The program is integral if each $\boldsymbol{x} \in \mathcal{X}$ has integer coefficients; otherwise we call the problem non-integral.
3. *Weighting*. For each loss type (edge vs. triangle) we consider two possible values for W: identity and non-identity. In the identity case, all edges (or triangles) are weighted equally; we call this a uniform-weighted problem. In the non-identity case we weigh each entry according to some measurement of "size" of the underlying simplex (length, in the case of edges, and area, in the case of triangles).[9] There is precedent for such weighting schemes in existing literature (Chen et al., 2008; Dey et al., 2011).

Edge-loss and triangle-loss programs will be denoted Edge and Tri, respectively. Integrality will be indicated by a superscript I (integer) or NI (non-integer). Uniform weighting will be denoted by a subscript *Unif* (uniform); non-uniform weighting will be indicated by subscript *Len* (for edge-loss programs) or *Area* (for triangle-loss programs). Thus, for example, Edge^I_{Len} denotes a length-weighted edge-loss program with integer constraints.

4.2 Edge-Loss Methods

Our approach to edge-loss minimization, based on work by Escolar and Hiraoka (2016), is summarized in **Algorithm 1**. As in Escolar and Hiraoka (2016), we obtain $\mathbf{x}$ by taking a linear combination of $\mathbf{x}^{Orig}$ with not only boundaries but cycles as well; consequently $\mathbf{x}$ need not be homologous to $\mathbf{x}^{Orig}$.

Our pipeline differs from Escolar and Hiraoka (2016) in three respects. First, we perform all optimizations after the persistence calculation has run. On the one hand, this means that our persistence calculations fail to benefit from the memory advantages offered by optimized cycles; on the other hand, separating the calculations allows one to "mix and match" one's favorite persistence solver with one's favorite linear solver, and we anticipate that this will be increasingly important as new, more efficient solvers of each kind are developed. Second, we introduce additional constraints which guarantee that $\mathcal{B}^* \in \mathrm{PrsHCB}$ [and, moreover, $\mathcal{L}(\mathbf{x}) = \mathcal{L}(\mathbf{x}^{Orig})$ for each $\mathbf{x}^{Orig} \in \mathcal{B}$]. Third, we remove the hypothesis

Algorithm 1 | Edge-loss persistent cycle minimization

1: Compute a persistent homology basis $\mathcal{B}$ for homology in dimension 1, with coefficients in $\mathbb{Q}$, using the standard matrix decomposition procedure described in the **Supplementary Material**. Arrange the elements of $\mathcal{B}$ into an ordered sequence $Z^0 = (\mathbf{z}^{0,1}, \ldots, \mathbf{z}^{0,m})$.
2: **for** $j = 0, \ldots, m-1$ **do**
3: Solve the program in **Eq. 14** to optimize the $j+1$th element of Z^j. Let $\mathbf{x}$ denote the solution to this problem, and define Z^{j+1} by replacing the $j+1$th element of Z^j with $\mathbf{x}$. Concretely, $\mathbf{z}^{j+1,j+1} = \mathbf{x}$, and $\mathbf{z}^{j+1,k} = \mathbf{z}^{j,k}$ for $k \neq j$.
4: **end for**
5: Return $\mathcal{B}^* := \{\mathbf{z}^{m,1}, \ldots, \mathbf{z}^{m,m}\}$, the set of elements in Z^m.

of a simplex-wise filtration; this requires some technical modifications, whose motivation is explained in the **Supplementary Material**. The crux of this modification lies with the for loop, which replaces cycles that have been optimized in the cycle basis for later cycle optimization.

The program in **Eq. 14** optimizes the jth element of an ordered sequence of cycle representatives $Z = (\mathbf{z}^1, \ldots, \mathbf{z}^m)$. In particular, it seeks to minimize $\mathbf{x}^{Orig} := \mathbf{z}^j$. To define this program, we first construct a matrix A such that $A[:, i] = \mathbf{z}^i$ for $i = 1, \ldots, m$. We then define three index sets, $\mathcal{P}$, $\mathcal{Q}$, $\mathcal{R}$ such that

$$\begin{aligned} \mathcal{P} &= \left\{i : \mathrm{Birth}\left(\mathbf{z}^i\right) \leq \mathrm{Birth}\left(\mathbf{x}^{Orig}\right),\ \mathrm{Death}\left(\mathbf{z}^i\right) \leq \mathrm{Death}\left(\mathbf{x}^{Orig}\right),\ i \neq j\right\} \\ \mathcal{Q} &= \left\{\tau \in \mathbf{S}_{n+1}(K) : \mathrm{Birth}(\tau) \leq \mathrm{Birth}\left(\mathbf{x}^{Orig}\right)\right\} \\ \mathcal{R} &= \left\{\sigma \in \mathbf{S}_n(K) : \mathrm{Birth}(\sigma) \leq \mathrm{Birth}\left(\mathbf{x}^{Orig}\right)\right\}, \end{aligned}$$

That is, $\mathcal{P}$ indexes the set of cycles $\mathbf{z}^i$ such that $\mathbf{z}^i$ is born (respectively, dies) by the time that $\mathbf{z}^j$ is born (respectively, dies), excluding the original cycle $\mathbf{z}^j$ itself. Set $\mathcal{Q}$ is the family of triangles born by $\mathrm{Birth}(\mathbf{x}^{Orig})$, and set $\mathcal{R}$ is the family of edges born by $\mathrm{Birth}(\mathbf{x}^{Orig})$.

With these definitions in place, we now formalize the general edge-loss problem as the program in **Eq. 14**, where $\partial_{n+1}[\mathcal{R}, \mathcal{Q}]$ denotes the submatrix of ∂_{n+1} indexed by triangles born by $\mathrm{Birth}(\mathbf{x}^{Orig})$ (along columns) and edges indexed by edges born by $\mathrm{Birth}(\mathbf{x}^{Orig})$. Likewise $A[\mathcal{R}, \mathcal{P}]$ is the column submatrix of A corresponding to cycles that are born before the birth time of $\mathbf{x}^{Orig}$ (and which die before the death time of $\mathbf{x}^{Orig}$), excluding $\mathbf{x}^{Orig}$ itself.

$$\begin{aligned} \text{minimize} \quad & \|W\mathbf{x}\|_1 = \sum_{\sigma \in R} \mathrm{W}[\sigma,\sigma]\left(x_\sigma^+ + x_\sigma^-\right) \\ \text{subject to} \quad & \left(\mathbf{x}^+\text{-}\mathbf{x}^-\right) = \mathbf{x}^{Orig}[\mathcal{R}] + \partial_{n+1}[\mathcal{R}, \mathcal{Q}]\mathbf{q} + A[\mathcal{R}, \mathcal{P}]\mathbf{p} \\ & \mathbf{p} \in \mathbb{Q}^{\mathcal{P}} \\ & \mathbf{q} \in \mathbb{Q}^{\mathcal{Q}} \\ & \mathbf{x} \in G^{\mathcal{R}} \\ & \mathbf{x}^+, \mathbf{x}^- \geq 0. \end{aligned} \tag{14}$$

Recall from **Section 4.1** that this program varies along two parameters (integrality and weighting). In integral programs $G = \mathbb{Z}$, whereas in nonintegral programs $G = \mathbb{Q}$. The weight matrix W is always diagonal, but in uniform-weighted programs $W[\sigma, \sigma] = 1$ for all $\sigma = R$, whereas in length-weighted programs $W[\sigma, \sigma]$ is the length of edge σ. The program in **Eq. 14** thus results in four variants:

$\mathrm{Edge}^{NI}_{Unif}$: Nonintegral edge-loss with uniform weights.
Edge^{I}_{Unif}: Integral edge-loss with uniform weights.

[9]These notions make sense due to our use of coefficient field $\mathbb{Q}$. The distance used to form a simplicial complex can be used to define length. We restrict our attention of area to points in Euclidean space.

Edge^{NI}_{Len}: Nonintegral edge-loss with edges weighted by length.
Edge^{I}_{Len}: Integral edge-loss with edges weighted by length.

The program in **Eq. 14** may have many more variables than needed, because ∂_{n+1} is often highly singular. Indeed, in applications, ∂_{n+1} can have hundreds or thousands of times as many columns as rows!

A simple means to reduce the size of the program in **Eq. 14**, therefore, is to replace $\mathcal{Q}$ with a subset $\widehat{\mathcal{Q}} \subseteq \mathcal{Q}$ such that $\partial_{n+1}[\mathcal{R}, \widehat{\mathcal{Q}}]$ is a column basis for $\partial_{n+1}[\mathcal{R}, \mathcal{Q}]$. Replacing $\mathcal{Q}$ with $\widehat{\mathcal{Q}}$ will not change the space of feasible values for $\mathbf{x}$ in the program in **Eq. 14**, but it can cut the number of decision variables significantly. In particular, one may take $\widehat{\mathcal{Q}} := \{\sigma : R[:, \sigma] \neq 0\}$ in the $R = \partial_{n+1} V$ decomposition of ∂_{n+1} described in the **Supplementary Material**. We also show correctness of this choice of $\widehat{\mathcal{Q}}$ there.

4.3 Triangle-Loss Methods

Our approach to triangle-loss optimization is essentially that of Obayashi (2018), plus a preprocessing step that converts more general problem data into the simplex-wise filtration format assumed in Obayashi (2018). There are several noteworthy methods for time and memory performance enhancement developed in Obayashi (2018), which we do not implement (e.g., using restricted neighborhoods $\mathcal{F}^{(r)}_{q}$ to reduce problem size), but which may substantially improve runtime and memory performance.

The original method makes the critical assumption that $K_\bullet$ is a simplex-wise filtration, more precisely, that there exists a linear order $\sigma_1 \leq \cdots \leq \sigma_{|K|}$ such that $K_i = \{\sigma_1, \ldots, \sigma_i\}$. This hypothesis allows one to map each finite-length interval $[i, j) \in \text{Barcode}_n(K_\bullet)$ to a unique pair of simplices (σ_i, σ_j), called a *birth/death pair*, where $\sigma_i \in \mathbf{S}_n(K)$ and $\sigma_j \in \mathbf{S}_{n+1}(K)$. This mapping makes it possible to formulate the program in **Eq. 12**. Unlike the general edge-loss the program in **Eq. 13**, one can formulate the program in **Eq. 12** without ever needing to choose an initial (non-optimal) cycle. Thus, for simplex-wise filtrations, the method of Obayashi (2018) has the substantial advantage of being "parameter free."

However, in many applied settings the filtration $K_\bullet$ is not simplex-wise. Indeed, even accessing information about the filtration can be difficult in modern workflows. Such is the case, for example, for the filtered Vietroris-Rips (VR) construction. In many VR applications, the user presents raw data in the form of a point cloud or distance matrix to a "black box" solver; the solver returns the barcode without ever exposing information about the filtered complex to the user. Thus, the problem of mapping intervals back to pairs of simplices has practical challenges in common applied settings.

To accommodate this more general form of problem data, we employ **Algorithm 2**. This procedure works by (implicitly) defining a simplex-wise refinement $K'_\bullet$ of $K_\bullet$, applying the method of Obayashi (2018) to this refinement, then extracting a persistent homology cycle basis for the subspace of finite intervals from the resulting data. More details, including recovery of a complete persistent homology cycle basis with infinite intervals,[10] and a proof of correctness can be found in the **Supplementary Material**.

Algorithm 2 | Triangle-loss persistent cycle minimization.

1: Place a filtration-preserving linear order $\leq^{(l)}$ on $\mathbf{S}_l(K)$ for each l.
2: Compute an $R = \partial_{n+1} V$ decomposition as described in (Cohen-Steiner et al., 2006) and the **Supplementary Material**. We then obtain a set Γ of birth/death pairs (σ, τ).
3: For each $(\sigma, \tau) \in \Gamma$ such that $\text{Birth}(\sigma) < \text{Birth}(\tau)$, put

$$\mathcal{F}_n := \{\sigma' \in \mathbf{S}_n(K) : \text{Birth}(\sigma') \leq \text{Birth}(\tau),\ \sigma \leq^{(n)} \sigma'\}$$
$$\mathcal{F}_{n+1} := \{\tau' \in \mathbf{S}_{n+1}(K) : \text{Birth}(\sigma) \leq \text{Birth}(\tau'),\ \tau' \leq^{(n+1)} \tau\},$$

and $\widehat{\mathcal{F}}_{n+1} := \mathcal{F}_{n+1} \cup \{\tau\}$. Compute a solution to the corresponding program in **Eq. 15**, and denote this solution by $\mathbf{v}^{\sigma,\tau}$
4: Put $\widehat{\mathcal{D}} := \{\partial_{n+1}(\mathbf{v}^{\sigma,\tau}) : (\sigma, \tau) \in \Gamma \text{ and } \text{Birth}(\sigma) < \text{Birth}(\tau)\}$ and let $\widehat{\mathcal{D}}' := \{\mathbf{z} \in \mathcal{M} : \text{Death}(\mathbf{z}) = \infty\}$, where $\mathcal{M}$ is a persistent homology cycle basis calculated by the standard $R = DV$ method.
5: Return $\mathcal{D} := \widehat{\mathcal{D}} \cup \widehat{\mathcal{D}}'$.

A key component of **Algorithm 2** is the program in **Eq. 15**, which we refer to as the triangle-loss program.

$$\begin{aligned} \text{minimize} \quad & \|W\mathbf{v}\|_1 = \sum_{i=1}^{n} W[\gamma, \gamma]\left(\mathbf{v}^+_\gamma + \mathbf{v}^-_\gamma\right) \\ \text{subject to} \quad & \partial_{n+1}[\sigma, \widehat{\mathcal{F}}_{n+1}]\mathbf{v} \neq 0 \\ & \partial_{n+1}[\mathcal{F}_n, \widehat{\mathcal{F}}_{n+1}]\mathbf{v} = 0 \\ & \mathbf{v}_\tau = 1 \\ & \mathbf{v}^+, \mathbf{v}^- \geq 0 \\ & \mathbf{v}^+, \mathbf{v}^- \in G^{\widehat{\mathcal{F}}_{n+1}}. \end{aligned} \tag{15}$$

This terminology is motivated by the special case $n = 1$, which is our focus for empirical studies. As with the general edge-loss program in **Eq. 15** varies along two parameters (integrality and weighting). In integral programs $G = \mathbb{Z}$, whereas in nonintegral programs $G = \mathbb{Q}$. The weight matrix W is always diagonal, but in uniform-weighted programs $W[\gamma, \gamma] = 1$ for all γ, whereas in *area*-weighted programs $W[\gamma, \gamma]$ is the area of triangle γ.[11] The program in **Eq. 15** thus results in four variants:

Tri^{NI}_{Unif}: Nonintegral triangle-loss with uniform weights.
Tri^{I}_{Unif}: Integral triangle-loss with uniform weights.
Tri^{NI}_{Area}: Nonintegral triangle-loss with edges weighted by area.
Tri^{I}_{Area}: Integral triangle-loss with edges weighted by area.

Remark 4.1. Algorithm 2 offers an effective means to apply the methods of Obayashi (2018) to some of the most common data sets in TDA. However, this is done at the cost of parameter-dependence; in particular, outputs depend on the choice of linear orders $\leq^{(l)}$. A brief discussion on how the choice of a total order $\leq$ in **Algorithm 2** may impact the difficulty of the linear programs one must solve is

[10] Recall volume is undefined for infinite intervals.
[11] We compute the area of a 2-simplex using Heron's Formula. We calculate area only for VR complexes whose vertices are points in Euclidean space, though more general metrics could also be considered.

discussed in the **Supplementary Material**. In particular, we explain why the total order implicitly chosen in **Algorithm 2** is reasonable, from a computational/performance standpoint.

4.4 Acceleration Techniques

We consider acceleration techniques to reduce the computational costs of the programs in **Eqs 14**, **15**.

4.4.1 Edge-Loss Methods

The technique used for edge-loss problems aims to reduce the number of decision variables in the program in **Eq. 14**. It does so by replacing a (large) set of decision variables indexed by $\mathcal{Q}$ with a much smaller set, $\widehat{\mathcal{Q}}$. See **Section 4.2** for details.

4.4.2 Triangle-Loss Methods

When ∂_n is large, the memory and computation time needed to construct the constraint matrix $\partial_{n+1}[\mathcal{F}_n, \widehat{\mathcal{F}}_{n+1}]$ can be nontrivial. In applications that require an optimal representative for every interval in the barcode, these costs can be incurred for hundreds or even thousands of programs. We consider two ways to generate the constraint matrices $\partial_{n+1}[\mathcal{F}_n, \widehat{\mathcal{F}}_{n+1}]$ for each of the intervals in a barcode: build $\partial_{n+1}[\mathcal{F}_n, \widehat{\mathcal{F}}_{n+1}]$ from scratch for each program, or build the complete boundary matrix ∂_{n+1} in advance; rather than recompute block submatrices for each program, we pass a slice of the complete matrix stored in memory.

The difference between these two techniques can be seen as a speed/memory tradeoff. As we will see in **Section 6.2**, the first approach is generally faster to optimize the entire basis of homology cycle representatives, but when the data set is large, the full boundary matrix ∂_{n+1} may be too large to store in memory.

5 EXPERIMENTS

In order to address the questions raised in **Section 1**, we conduct an empirical study of minimal homological cycle representatives in dimension one—as defined by the optimization problems detailed in **Section 4** — on a collection of point clouds, which includes both real world data sets and point samples drawn from four common probability distributions of varying dimension.

5.1 Real-World Data Sets

We consider 11 real world data sets from Otter et al. (2017), a widely used reference for benchmark statistics concerning persistent homology computations. There are 13 data sets considered by Otter et al. (2017), however, one of them (gray-scale image) is not available, and one of them is a randomly generated data set similar to our own synthetic data. We summarize information about the dimension, number of points, persistence computation time of each point cloud in **Table 1**. Below we provide brief descriptions of each data set, but we refer the interested reader to Otter et al. (2017) for further details.[12]

1. Vicsek biological aggregation model. The Vicsek model is a dynamical system describing the motion of particles. It was first introduced in Vicsek et al. (1995) and was analyzed using PH in Topaz et al. (2015). We consider a snapshot in time of a single realization of the model with each point specified by its (x, y) position and heading. To compute distances, the positions and headings are scaled to be between 0 and 1, and then distance is calculated on the unit cube with periodic boundary conditions. The distance between a and b is computed as $\min\{d(a,q) : q\text{-}b \in \{0,1,\text{-}1\}^3\}$. We denote this data by **Vicsek**.
2. Fractal networks. These networks are self-similar and are used to explore the connection patterns of the cerebral cortex (Sporns, 2006). The distances between nodes in this data set are defined uniformly at random by Otter et al. (2017). In another data set, the authors of Otter et al. (2017) define distances between nodes by using linear weight-degree correlations. We consider both data sets and found the results to be similar. Therefore, we opt to use the one with distances defined uniformly at random. We denote this data set by **Fract R**.
3. C.elegans neuronal network. This is an undirected network in which each node is a neuron, and edges represent synapses. It was studied using PH in Petri et al. (2013). Each nonzero edge weight is converted to a distance equal to its inverse by Otter et al. (2017). We denote this data by ***C.elegans***.
4. Genomic sequences of the HIV virus. This data set is constructed by taking 1,088 different genomic sequences of dimension 673. The aligned sequences were studied using PH in Chan et al. (2013) with sequences retrieved from (Los Alamos National Laboratory, 2021). Distances are defined using the Hamming distance, which is equal to the number of entries that are different between two genomic sequences. We denote this data by **HIV**.
5. Genomic sequences of H3N2. This data set contains 1,173 genomic sequences of H3N2 influenza in dimension 2,722. Distances are defined using the Hamming distance. We denote this data set as **H3N2**.
6. Human genome. This is a network representing a sample of the human genome studied using PH in Petri et al. (2013), which was created using data retrieved from Davis and Hu (2011). Distances are measured using Euclidean distance. We denote this data set by **Genome**.
7. U.S. Congress roll-call voting networks. In the two networks below, each node represents a legislator, and the edge weight is a number in $[0, 1]$ representing the similarity of the two legislators' past voting decisions. Distance between two nodes i, j are defined to be $1\text{-}w_{i,j}$.
   1. **House**. This is a weighted network of the House of Representatives from the 104th United States Congress.
   2. **Senate**. This is a weighted network of the Senate from the 104th United States Congress.
8. Network of network scientists. This data set represents the largest connected component of a collaboration network of network scientists (Newman, 2006). The edge weights indicate the number of joint papers between two authors. Distances are defined as the inverse of edge weight. We denote this data set by **Network**.

[12]We use the distance matrices found on the associated github page (Otter et al., 2017b), except in two cases. For the **Vicsek** data, we use a distance to account for the intended periodic boundary conditions of the model, and for the **genome** data, we use Euclidean distance as the distance matrix in Otter et al. (2017b) resulted in an integer overflow error.

9. Klein. The Klein bottle is a non-orientable surface with one side. This data set was created in Otter et al. (2017a) by linearly sampling 400 points from the Klein bottle using its "figure 8" immersion in $\mathbb{R}^3$. This data set originally contains (Borradaile et al., 2020) duplicate points, which we remove. Distances are measured using the Euclidean distance. We denote this data set by **Klein**.
10. Stanford Dragon graphic. This data set contains 1,000 points sampled uniformly at random by Otter et al. (2017) from 3-dimensional scans of the dragon (Stanford University Computer Graphics Laboratory, 1999). Distances are measured using the Euclidean distance. We denote this data set **Drag**.

5.2 Randomly Generated Point Clouds

We also generate a large corpus of synthetic point clouds, each containing 100 points in $\mathbb{R}^q$ with $q = 2, \ldots, 10$, drawn from normal, exponential, gamma, and logistic distributions. We produce 10 realizations for each distribution and dimension combination, for a total of 360 randomly generated point clouds. We use Euclidean distance to measure similarity between points and the Vietoris- Rips filtered simplicial complex to compute persistent homology.

5.3 Erdős-Rényi Random Complexes

To investigate which properties of homological cycle representatives could arise as the result of the underlying geometry of the point clouds, we also consider a common non-geometric model for random complexes: Erdős-Rényi random clique complexes. Here, we construct 100 symmetric dissimilarity matrices of size 100 × 100 by drawing entries i. i.d. from the uniform distribution on $[0, 1]$ for each pair of distinct points. As these dissimilarities are fully independent, they are in particular not subject to geometric constraints like the triangle inequality. A natural filtration is placed on these dissimilarity matrices by forming filtered simplicial complex $K_\bullet = (K_{\varepsilon_i})_{i\in\{1,\ldots,T\}}$ where $0 = \varepsilon_1 < \cdots < \varepsilon_T = 1$ to compute persistent homology.

5.4 Computations

For each of the data sets, we perform **Algorithms 1**, **2** (using Vietoris-Rips complexes with $\mathbb{Q}$ – coefficients) to find optimal bases $\mathcal{B}^*, \mathcal{D} \in$ PrsHCB. For comparison to the edge-loss problem in **Algorithm 1**, we also apply the program in **Eq. 8** to each representative in the persistent homology cycle basis to find a basis $\mathcal{C} \in$ FCB.

5.5 Hardware and Software

We test our programs on an iMac (Retina 5K, 27-inch, 2019) with a 3.6 GHz Intel Core i9 processor and 40 GB 2667 MHz DDR4 memory.

Software for our experiments is implemented in the programming language Julia; source code is available at Li and Thompson (2021). This code specifically implements **Algorithms 1**, **2** and the program in **Eq. 8**.[13]

Since our interest lies not only with the outputs of these algorithms but with the structure of the linear programs themselves Li and Thompson (2021), implements a standalone workflow that exposes the objects built internally within each pipeline. This library is simple by design, and does not implement the performance-enhancing techniques developed in Escolar and Hiraoka (2016) and Obayashi (2018). Users wishing to work with optimal cycle representatives for applications may consider these approaches discussed in **Section 3.7**.

To implement **Algorithms 1**, **2** in homological dimension one, the test library (Li and Thompson, 2021) provides three key functions: *A novel solver for persistence with* $\mathbb{Q}$ *– coefficients*. To compute cycle representatives for persistent homology with $\mathbb{Q}$ – coefficients, we implement a new persistent homology solver adapted from Eirene (Henselman-Petrusek, 2016). The adapted version uses native Eirene code as a subroutine to reduce the number of columns in the top dimensional boundary matrix in a way that is guaranteed not to alter the outcome of the persistence computation (Henselman and Ghrist, 2016).

Formatting of Inputs to Linear Programs. Having computed barcodes and persistent homology cycle representatives, library (Li and Thompson, 2021) provides built-in functionality to format the linear the program in **Eqs 14**, **15** for input to a linear solver. This "connecting" step is executed in pure Julia.

Wrappers for Linear Solvers. We use the Gurobi linear solver (Gurobi Optimization, 2020) and the GLPK solver (GNU Project, 2012). Both solvers can optimize both LPs and MIPs. Experiments indicate that Gurobi executes much faster than GLPK on this class of problems, and thus, we use it in the bulk of our computations. Both solvers are free for academic users.

6 RESULTS AND DISCUSSION

In this section, we investigate each of the questions raised in **Section 1** with the following analyses.

6.1 Computation Time Comparisons

We summarize results for Programs Edge^{NI}_{Unif}, Edge^{I}_{Unif}, Edge^{NI}_{Len}, Edge^{I}_{Len}, Tri^{NI}_{Unif}, and Tri^{I}_{Unif} in **Table 1** for data described in **Section 5.1** and **Table 2** for data described in **Section 5.2** and **Section 5.3**. Further, we summarize results for Programs Tri^{NI}_{Area} and Tri^{I}_{Area} in **Table 2** for data described in **Section 5.2**.[14] We use $T_{\textbf{persistence}}$ to denote the time taken to compute all original cycle representatives and their lifespans $\mathcal{L}$. We use $T^*_\bullet$ to denote the computation time for optimizing all generators found by the persistence algorithm, where the subscript denotes the cost function e.g. *E-Unif* or *T-Unif*, and the superscript denotes the nonintegral ^{NI} or integral ^{I} constraint. The $T^*_\bullet$ computations include the time required to construct the inputs to the solver for the edge-loss methods, and exclude the time required to construct the inputs to the triangle-loss methods, whose computation time is separately recorded in

[13]The program in **Eq. 8** is implemented analogously.

[14]We compute the area of a 2-simplex using Heron's Formula for data whose distances are measured using the Euclidean distance. For data with non-Euclidean distances, we find that there are triangles that do not obey the triangle inequality, thus, we only compute area-weighted triangle-loss cycles for data described in **Section 5.2**. As such, Tri^{NI}_{Area}, Tri^{NI}_{Area} do not appear in **Table 1** and the Erős-Rényi column of **Table 2**.

TABLE 1 | Summary of the experimental results of the data sets from Otter et al. (2017) as described in **Section 5.1**. The rows include the ambient dimension, number of points, the number of cycle representatives in $\mathbf{H}_1$, and the time (measured in seconds) it took to compute persistent homology for each data set. We also include the computation time taken to optimize the set of cycle representatives under six different optimization problems, and computation time of two different implementation choices for the triangle-loss optimal cycles: building the full ∂_2 boundary matrix once and extracting the part needed, or constructing part of the ∂_2 boundary matrix for each cycle representative. In this table, T stands for computation time measured in seconds with subscripts indicating the type of the optimal cycle and superscripts indicating whether the program was solved using linear programming (NI) or integer programming (I). The time taken to construct the input to the optimization problem is included in the optimization time for edge-loss minimal cycle representatives, but is excluded and separately listed in the last two rows for the triangle-loss minimal cycle representatives. For triangle-loss cycles, we were able to compute 115 out of the 117 cycle representatives for the **Genome** data set and 52 out of 57 cycle representatives for the **H3N2** data set due to memory constraints. The numbers in the parenthesis represent the other optimization statistics corresponding to the triangle-loss optimal cycles we were actually able to compute. The last two rows compare two ways of building the input $\partial_2[:, \widehat{\mathcal{F}}_2]$ matrix to the triangle-loss optimal cycle program. The penultimate row records the time of building the entire ∂_2 matrix once and then extracting columns born in the interval $[b_i, d_i]$ for each representative. The last row records the total time to iteratively build the part of the boundary matrix $\partial_2[:, \widehat{\mathcal{F}}_2]$ for each cycle representative.

	Klein	Vicsek	C.*elegans*	HIV	Genome	Fractal R	Network	House	Senate	Drag	H3N2
Ambient dimension	3	3	202	673	688	259	300	261	60	3	1,173
# Points	400	300	297	1,088	1,397	512	379	445	103	1,000	2,722
# Representatives	257	149	107	174	117 (115)	438	7	126	12	311	28 (26)
$T_{\text{persistence}}$	100.97	129.39	5.14	728.51	967.61	143.07	12.18	9.62	0.10	1,053.53	71,081.77
Edge-loss persistent homological cycle representatives (Eq. 14)											
$T^{I}_{E\text{-}Len}$	16.01	8.20	19.64	466.85	656.05	150.46	0.17	63.93	0.31	45.14	4,732.59
$T^{NI}_{E\text{-}Len}$	11.28	6.61	16.07	403.63	491.69	86.95	0.13	48.65	0.22	34.73	4,540.55
$T^{I}_{E\text{-}Unif}$	14.59	9.09	19.22	473.82	689.51	119.94	0.23	63.34	0.33	45.51	4,714.90
$T^{NI}_{E\text{-}Unif}$	11.38	5.55	15.63	404.95	492.66	83.40	0.12	48.88	0.22	33.88	4,547.37
Edge-loss filtered homological cycle represnetatives (Eq. 8)											
$T^{I}_{E\text{-}Len}$	16.93	8.64	20.41	468.22	1,144.17	155.08	0.17	62.20	0.30	67.77	2,999.24
$T^{NI}_{E\text{-}Len}$	10.29	5.51	16.15	403.74	973.15	88.66	0.13	48.24	0.22	50.25	2,829.12
$T^{I}_{E\text{-}Unif}$	15.14	8.32	19.76	476.84	1,191.44	142.4	0.24	61.82	0.31	68.63	2,937.16
$T^{NI}_{E\text{-}Unif}$	11.07	5.63	16.23	406.97	981.72	87.59	0.12	48.11	0.22	54.05	2,833.06
Triangle-loss persistent homological cycle representatives (Eq. 15)											
$T^{I}_{T\text{-}Unif}$	316.33	24.52	657.53	25,402.56	16,379.86	20,440.33	2.91	234.05	0.29	384.91	39,140.67
$T^{NI}_{T\text{-}Unif}$	154.36	19.18	540.06	23,260.12	14,535.42	18,279.82	2.47	206.63	0.18	277.93	36,401.50
T Build all	2.16	0.32	4.88	268.57	—	138.46	0.06	6.23	0.03	5.94	—
Total T build part	9.18	3.51	28.47	1688.10	415.79	917.42	0.28	45.02	0.05	106.64	1,236.80

order to compare two ways of constructing the input matrix, as discussed in **Section 4.4**. In each table, rows 1–3 provide information about the data by specifying ambient dimension, number of points, and number of cycle representatives. Row 4, labeled as $T_{\textbf{persistence}}$, gives the total time to compute persistent homology for the data, measured in seconds. Rows 5–12 (**Table 1**) and rows 5–14 (**Table 2**) give the total time to optimize all cycle representatives that are feasible to compute using each optimization technique. In the last two rows of each table, we provide the time of constructing the input to the triangle-loss methods using two different approaches described in **Section 4.4**. The penultimate row records the time of building the entire ∂_2 matrix once and then extracting $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$ for each representative. The last row records the total time to iteratively build the part of the boundary matrix $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$ for each cycle representative. In **Table 2**, the computation times displayed average all random samples from each dimension for each distribution.

The two numbers in parenthesis in the third row of **Table 1** indicate the actual number of representatives we were able to optimize using the triangle-loss methods (all edge-loss representatives were optimized). For the **Genome** and **H3N2** data sets, we are not able to compute all triangle-loss cycle representatives due to the large number of 2-simplices born between the birth and death interval of some cycles. For instance, for a particular cycle representative in the **Genome** data set, there were 10,522,991 2-simplices born in this cycle's lifespan. Also, given the large number of 2-simplices in the simplicial complex, we are not able to build the full ∂_2 matrix due to memory constraints, denoted by - in the penultimate row of **Table 1**.

Below we describe some insights on computation time drawn from the two tables.

6.1.1 Persistence and Optimization $T_{\text{persistence}}$ vs. $T^*_\bullet$

We observe that $T^*_\bullet$ [15] > $T_{\textbf{persistence}}$ e.g. for 5 out of the 11 real-world data sets described in **Section 5.1** when using the four edge-loss methods. The same inequality holds in seven out of the 11 data sets when using the two uniform-weighted triangle-loss methods. For all of the synthetic data described in **Sections 5.2** and **5.3**, we have $T^*_\bullet > T_{\textbf{persistense}}$ when using all eight optimization programs. Therefore, the computational cost of optimizing a basis of cycle representatives generally exceeds the cost of computing such a basis.

This somewhat surprising result highlights the computational complexity of the algorithms used both to compute persistence and to optimize generators. A common feature of both the persistence computation and linear optimization is that empirical performance typically outstrips asymptotic complexity by a wide margin; the persistence computation, for example, has cubic complexity in the size of the complex, but usually runs in linear time. Thus, worst-case

[15]Including the time of constructing the input to the optimization programs.

TABLE 2 | Summary of the experimental results for the synthetic, randomly generated data sets described in **Section 5.2** and **Section 5.3**. For each distribution, we sample 10 data sets each containing 100 points in ambient dimensions from 2–10. The computation time in this table averages the 10 random samples for each dimension and distribution combination. The number of cycle representatives is totaled over the 90 samples for each distribution. The rows of this table are analogous to those of **Table 1**, excluding the penultimate row of that table, as the time comparison is only done for the large real-world data sets.

	Normal	Gamma	Logistic	Exponential	Erdős-Rényi
Ambient dimension	2–10	2–10	2–10	2–10	NA
# Points	100	100	100	100	100
Total # representatives	4,815	3,706	4,456	3,788	34,214
Average $T_{\text{persistence}}$ (seconds)	2.80	2.12	2.01	2.63	2.20
Edge-loss persistent homological cycle representatives (Eq. 14)					
Average total $T^{I}_{E\text{-}Len}$	5.52	6.01	5.65	5.91	5.99
Average total $T^{NI}_{E\text{-}Len}$	4.37	4.55	4.32	4.47	4.99
Average total $T^{I}_{E\text{-}Unif}$	5.31	5.97	5.45	5.90	6.16
Average total $T^{NI}_{E\text{-}Unif}$	4.08	4.58	4.23	4.51	4.87
Edge-loss filtered homological cycle representatives (Eq. 8)					
Average total $T^{I}_{E\text{-}Len}$	5.32	6.46	6.27	6.88	7.44
Average total $T^{NI}_{E\text{-}Len}$	4.07	5.05	4.78	5.11	4.69
Average total $T^{I}_{E\text{-}Unif}$	5.23	6.46	6.25	6.66	6.25
Average total $T^{NI}_{E\text{-}Unif}$	4.17	4.94	4.61	5.29	4.64
Triangle-loss persistent homological cycle representatives (Eq. 15)					
Average total $T^{I}_{T\text{-}Unif}$	6.56	9.91	7.06	9.68	4.64
Average total $T^{NI}_{T\text{-}Unif}$	5.24	7.99	5.79	7.75	4.49
Average total $T^{I}_{T\text{-}\text{Area}}$	6.59	10.20	7.30	9.99	—
Average total $T^{I}_{T\text{-}\text{Area}}$	5.19	7.89	5.80	7.57	—
Average total T build all	1.40	1.71	1.56	1.07	1.24
Average total T build part	3.51	1.54	1.61	1.56	0.85

complexity paints an incomplete picture. Moreover, naive "back of the envelope" calculations are often hindered by lack of information. For example, the persistence computation (which essentially reduces to Gaussian elimination) typically processes each of the m columns of a boundary matrix ∂_n in sequence. The polytope of feasible solutions for an associated linear program (edge-loss or triangle-loss) may have many fewer or many more vertices than m, depending on the program; moreover, even if the number of vertices is very high, the number of *visited* vertices (e.g., by the simplex algorithm) can be much lower. Without knowing these numbers *a priori*, run times can be quite challenging to estimate. Empirical studies, such as the present one, give a picture of how these algorithms perform in practice.

6.1.2 Integral and Nonintegral Programs (T^I vs. T^{NI})

In **Tables 1** and **2**, we observe that ($T^I > T^{NI}$), i.e., the total computation time of optimizing a basis of cycle representatives using an integer program exceeds the computation time using a non-integer constrained program. Yet, T^I and T^{NI} are on the same order of magnitude, for both edge-loss methods and triangle-loss method.

Let $r_{E\text{-}Unif} = \frac{t^{I}_{E\text{-}Unif}}{t^{NI}_{E\text{-}Unif}}$, where $t^{*}_{\bullet}$ represents the computation time for optimizing a single cycle representative. We define $r_{E\text{-}Len}$ and $r_{T\text{-}Unif}$ similarly. We compute each for every cycle representative for data described in **Tables 1**, **2**. Let $\bar{r}_{\bullet}$ denote the average of $r_{\bullet}$ and $\sigma_{r_{\bullet}}$ denote the standard deviation of $r_{\bullet}$. We have $\bar{r}_{E\text{-}Unif} = 1.49, \sigma_{r_{E\text{-}Unif}} = 1.34$, $\bar{r}_{E\text{-}Len} = 1.55, \sigma_{r_{E\text{-}Len}} = 1.38$, $\bar{r}_{T\text{-}Unif} = 1.28, \sigma_{r_{T\text{-}Unif}} = 1.14$. **Figures 6A,C,E** plots $r_{\bullet}$ using scatter plots and **Figures 6B,D,F** displays the same data using box plots. The vertical axis represents the ratio between the MIP time and LP time of optimizing a cycle representative. The horizontal axis in the scatter plots represents the computation time to solve the LP. The red line in each subfigure represents the horizontal line $y = 1$. As we can see from the box plots, the ratio between the computation time of MIP and LP for most of the cycle representatives (>50%) is around 1 and less than 2. Although there are cases where the computation time of solving an MIP is 108.70 times the computation time of solving an LP, such cases happen only for cycle representatives with a very short LP computation time.

6.1.3 Triangle-Loss Vs. Edge-Loss Programs $T_{T\text{-}\bullet}$ vs. $T_{E\text{-}\bullet}$

We observe that the edge-loss optimal cycles are more efficient to compute than the triangle-loss cycles for more than 60.11% of the cycle representatives[16]. This aligns with our intuition because for representatives with a longer persistence, the number of columns in the boundary matrix $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$ grows faster than that of $\partial_1[:, \mathcal{Q}]$. Consequently, the edge-loss programs are feasible for all cycle representatives we experiment with, whereas the triangle-loss technique fails for six representatives due to the large problem size (with greater than twenty million triangles born between the life span of those cycle representatives).

6.1.4 Different Linear Solvers

The choice of linear solver can significantly impact the computational cost of the optimization problems. We perform experiments on length/uniform-minimal cycle representatives using the GLPK (GNU Project, 2012; Gurobi Optimization,

[16]Obayashi (2018) proposes a few techniques for accelerating the triangle-loss methods which we did not implement.

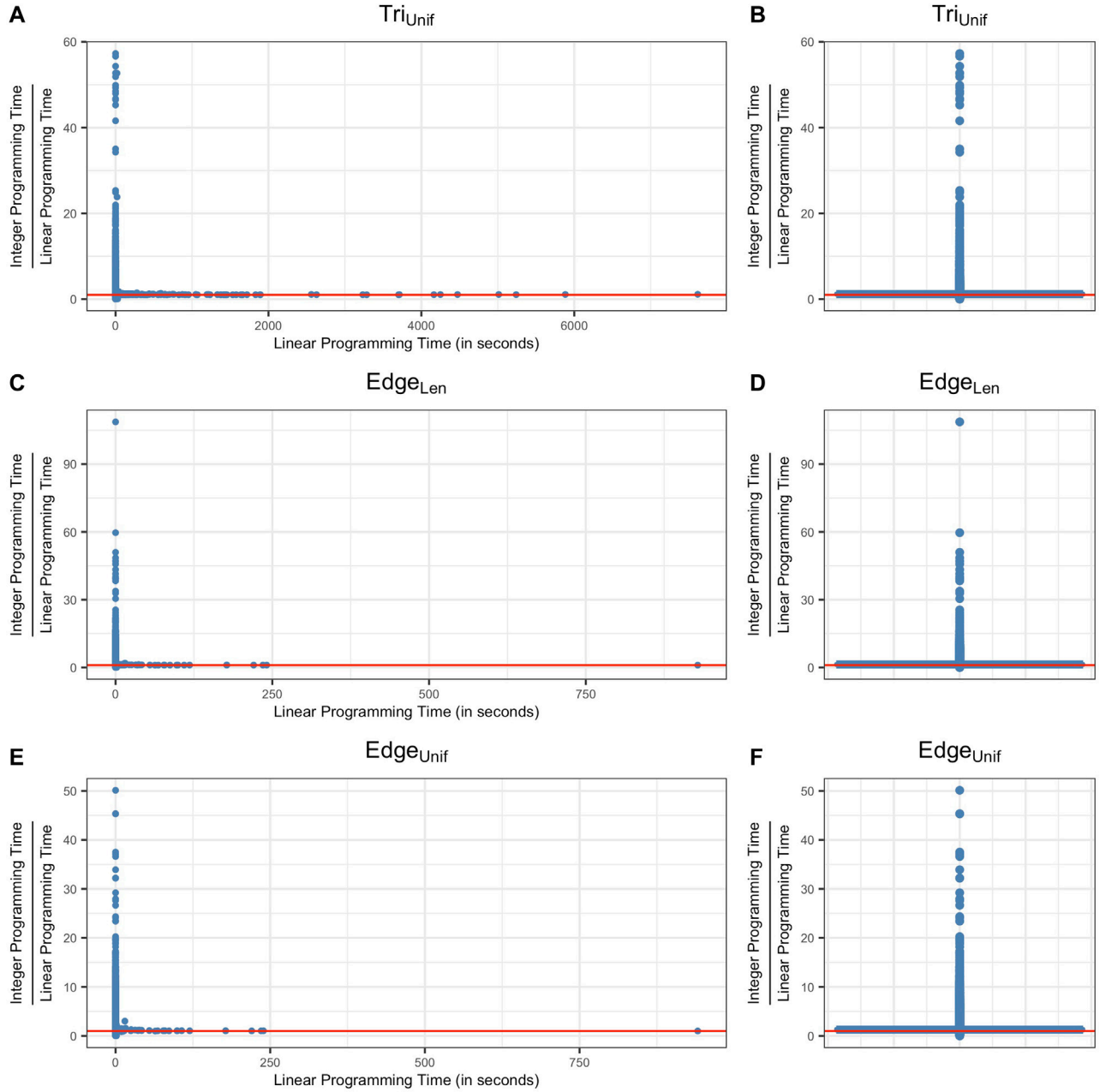


FIGURE 6 | Ratio between the computation time of solving an integer programming problem (Programs Tri^{I}_{Unif}, Edge^{I}_{Len}, Edge^{NI}_{Unif}) and the computation time of solving a linear programming problem (Programs Tri^{NI}_{Unif}, Edge^{NI}_{Len}, Edge^{NI}_{Unif}) for all of the cycle representatives from data sets described in **Sections 5.1**, **5.2** and **5.3**. Subfigures **(A) (C) (E)** plot the data using scatter plots and subfigures **(B) (D) (F)** show the same data using box plots. The vertical axis represents the ratio between the integer programming time and linear programming time of optimizing a cycle representative and the horizontal axis represents the computation time to solve the linear program. The red line in each subfigure represents the horizontal line $y = 1$, where the time for each optimization is equivalent. As we can see from the box plots, the ratio between the computation time of integer programming and linear programming for most of the cycle representatives (>50%) center around 1.

2020) linear solvers on 90 data sets drawn from the normal distribution with dimensions from 2 to 10 with a total of 4,815 cycle representatives. The median of the computation time ratio between using the GLPK solver and Gurobi solver is 2.22 for Program Edge^{NI}_{Unif}, 1.68 for Program Edge^{I}_{Unif}, 2.28 for Program Edge^{NI}_{Len}, and 1.73 for Program Edge^{I}_{Len}, and the computation time using the GLPK solver can be 30 times larger than the computation time using the Gurobi solver for some cycles, see figure in the **Supplementary Material**. Therefore, we use the Gurobi solver in all other analyses in this paper.

TABLE 3 | Computation time of three differently sized input boundary matrices to edge-loss and triangle-loss methods. The superscripts denote whether the program requires an integral solution or not, and the subscripts indicate the type of optimal cycle. All time is measured in seconds. We perform experiments on a small-sized data set (**Senate**) that consists of 103 points in dimension 60 and a medium-sized data set (**House**) that contains 445 points in dimension 261. For edge-loss methods, we consider three implementations to solve these optimization problems: using the full boundary matrix ∂_2, using the basis columns and all rows $\partial_2[:, \widehat{\mathcal{Q}}]$, and using the basis columns and deleting rows corresponding to edges born after the birth time of the cycle $\partial_2[\mathcal{R}, \widehat{\mathcal{Q}}]$. For triangle-loss methods, we consider three approaches to solve these optimization problems: zeroing out the columns in the boundary matrix outside of $[b_i, d_i]$ denoted as $\partial_{2_{zero}}$, deleting columns outside of this range $\partial_2[:, \widehat{\mathcal{F}}_2]$, and deleting both columns outside of $[b_i, d_i]$ and rows born after d_i denoted $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$. The **House** data set was too large to implement the first method.

		Edge-loss Optimal Cycles (Eq. 14)		
	T	∂_2	$\partial_2[:, \widehat{\mathcal{Q}}]$	$\partial_2[\mathcal{R}, \widehat{\mathcal{Q}}]$
Small Data Set (Senate)	$T^{NI}_{E\text{-}Unif}$	1.06	1.03	0.41
	$T^{I}_{E\text{-}Unif}$	1.25	1.23	0.60
	$T^{NI}_{E\text{-}Len}$	1.05	1.05	0.41
	$T^{I}_{E\text{-}Len}$	1.23	1.19	0.65
Medium Data Set (House)	$T^{NI}_{E\text{-}Unif}$	184.70	122.72	47.10
	$T^{I}_{E\text{-}Unif}$	188.88	147.27	64.64
	$T^{NI}_{E\text{-}Len}$	184.41	121.80	46.02
	$T^{I}_{E\text{-}Len}$	193.01	146.46	63.87
		Triangle-loss Optimal Cycles (Eq. 15)		
	T	$\partial_{2_{zero}}$	$\partial_2[:, \widehat{\mathcal{F}}_2]$	$\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$
Small Data Set (Senate)	$T^{NI}_{T\text{-}Unif}$	21.37	0.64	0.18
	$T^{I}_{T\text{-}Unif}$	24.51	0.86	0.29
Medium Data Set (House)	$T^{NI}_{T\text{-}Unif}$	—	297.34	203.63
	$T^{I}_{T\text{-}Unif}$	—	321.31	234.05

6.2 Performance of Acceleration Techniques

6.2.1 Edge-Loss Optimal Cycles

As discussed in **Section 4.4**, we accelerate edge-loss problems by replacing $\partial_2[:, \mathcal{Q}]$ with the column basis submatrix of $\partial_2[:, \widehat{\mathcal{Q}}]$. We further reduce the size of $\partial_2[:, \widehat{\mathcal{Q}}]$ by only including the rows corresponding to 1-simplices born before the birth time of the cycle, denoted as $\partial_2[\mathcal{R}, \widehat{\mathcal{Q}}]$. We perform experiments on a small-sized data set (**Senate**) that consists of 103 points in dimension 60 and a medium-sized data set (**House**) that contains 445 points in dimension 261. In **Table 3**, we report the computation time of solving the optimization problems in Programs Edge^{NI}_{Unif}, Edge^{I}_{Unif}, Edge^{NI}_{Len}, and Edge^{I}_{Len} using these three techniques of varying the size of the input boundary matrix. The results align with intuition that the optimizations are faster with fewer input variables, and thus, the third implementation is the most efficient among the three.

6.2.2 Triangle-Loss Optimal Cycles

As discussed in **Section 4.4**, there are also multiple approaches to creating the input to the triangle-loss problems. To recap, we restrict the boundary matrix ∂_2 to $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$ for a particular cycle representative $\mathbf{x}^i$. We can do so in various ways: 1) zeroing out the columns of ∂_2 not in $\widehat{\mathcal{F}}_2$ but maintaining the original size of the boundary matrix, 2a) building the entire boundary matrix ∂_2 once and then deleting the columns not in $\widehat{\mathcal{F}}_2$ for each representative, 2b) building the columns in $\widehat{\mathcal{F}}_2$ iteratively for each representative, and 3a/b) in conjunction with 2a) or 2b) respectively, reducing the rows of the boundary matrix of ∂_2 to only include the rows born before the death time of the cycle $\mathcal{F}_1$.

In **Table 3**, we summarize the computation time of solving Programs Tri^{NI}_{Unif} and Tri^{I}_{Unif} to find triangle-loss optimal cycles with three different sized boundary matrices as input: 1) zeroing out, 2b) deleting partial columns, and 3b) deleting partial rows and columns. Note that 2a) and 2b) both result in the same boundary matrix $\partial_2[:, \widehat{\mathcal{F}}_2]$. We again use the **Senate** and **House** data sets for analysis. We see that deleting partial rows and columns is the most efficient among the three implementations, which again matches intuition that reducing the number of variables accelerates the optimization problem.

We also ran experiments on the real-world data sets to compare the timing of building $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$ via methods 3a) and 3b) and summarize the results in the last two rows of **Table 1**. We find that approach 3a), where we build the entire matrix ∂_2 and then delete columns for each cycle representative, is in general faster than approach 3b), where the boundary matrix $\partial_2[\mathcal{F}_1, \widehat{\mathcal{F}}_2]$ is iteratively built for each representative. However, this latter approach can be more useful for large data sets, whose full boundary matrix ∂_2 might be too large to construct. For example, building the full boundary matrix for the **Genome** data set caused Julia to crash due to the large number of 2-simplices (453, 424, 290 triangles for the **Genome** data set and 3, 357, 641, 440 triangles for the **H3N2** data set). Whereas, by implenting 3b) where we rebuild a part of the boundary matrix for each representative, we were able to optimize 115 out of the 117 cycle representatives for the **Genome** data set and 52 of 57 cycle representatives for the **H3N2** data set.

6.3 Coefficients of Optimal Cycle Representatives in Data Sets From Section 5.1 and Section 5.2

As discussed in **Section 3.6**, the problem of solving an ℓ_0 optimization is desirable for its interpretability but doing so is NP-hard (Tahbaz-Salehi and Jadbabaie, 2010). Often, ℓ_0 optimization is approximated by an ℓ_1 optimization problem, which is solvable in polynomial time. If the coefficients of a solution of the ℓ_1 problem are in {-1, 0, 1}, then it is in fact an ℓ_0 solution to the restricted optimization problem where we require solutions to have entries in {-1, 0, 1} (Escolar and Hiraoka, 2016; Obayashi, 2018).

We find that 99.50% of the original, unoptimized cycle representatives obtained from data sets described in **Section 5.1** and 99.91% of the unoptimized cycle representatives obtained from data sets described in **Section 5.2** have coefficients in {-1, 0, 1}. All unoptimized cycle representatives turned out to have integral entries.

We then systematically check each solution of the eight programs Edge^{NI}_{Unif}, Edge^{I}_{Unif}, Edge^{NI}_{Len}, Edge^{I}_{Len}, Tri^{NI}_{Unif}, Tri^{I}_{Unif},

and Tri^{NI}_{Area}, Tri^{I}_{Area} across all data sets and all optimal cycle representatives from data discussed in **Sections 5.1** and **5.2**,[17] found by **Algorithms 1**, **2** and the program in **Eq. 8** to see if the coefficients are integral or in {-1, 0, 1}. We analyze the 18, 163 optimal cycle representatives and find the following consistent results.

All optimal solutions to the program in **Eq. 8** (edge-loss minimization of filtered cycle bases) and all but one of the solutions returned by **Algorithm 1** (edge-loss minimization of persistent cycle bases) had coefficients in {-1, 0, 1}; see the table in the **Supplementary Material** for details. The exceptional representative $\mathbf{x}^{NI}_{E\text{-}Unif}$ occurred in the **C.elegans** data set, with coefficients in {-0.5, 0, 0.5}. It corresponds to one of only a few cases where two intervals with equal birth and death time occur within the same data set; see **Section 6.6**. An interesting consequence of these fractional coefficients is that here, unlike all other cycle representatives from data discussed in **Section 5.1** and **Section 5.2**, the ℓ_0 norm and ℓ_1 norm differ. This accounts for the sole point that lies below the $y = 1$ line in the first column of row (B) in **Figure 8**.

On the one hand, this exceptional behavior could bear some connection to **Algorithm 1**. Recall that **Algorithm 1** operates by removing a sequence of cycles from a cycle basis, replacing each cycle with a new, optimized cycle on each iteration (that is, we swap the $j + 1^{th}$ element of Z^j with an optimized cycle $\mathbf{x}$ in order to produce Z^{j+1}). Replacing optimized cycles in the basis is key, since without replacement it would be possible in theory to get a set of optimized cycles that no longer form a basis. We verified that if we modify **Algorithm 1** to skip the replacement step, we achieve {-1, 0, 1} solutions for the exceptional **C.elegans** cycle representative (for the other repeated intervals we obtain the same optimal cycle with and without the replacement). On the other hand, we find that even with replacement the GLPK solver obtains a solution with coefficients in {-1, 0, 1}. Thus, every one of the cases considered produced {-1, 0, 1} coefficients for at least one of the two solvers, and the appearance of fractional coefficients may be naturally tied to the specific implementation of the solver used.

When solving the integral triangle-loss problem by **Algorithm 2**, we obtain two solutions whose boundaries $\mathrm{x} = \partial_2 \mathbf{v}$ have coefficients in {-1,0,1,2} for two different cycle representatives from the logistic distribution data set. However, the corresponding solutions $\mathbf{v}$ of these cycle representatives do have coefficients in {-1,0,1}.

The surprising predominance of solutions in {-1, 0, 1} suggests that in most cases, the modeler can reap both the computational advantage of ℓ_1 solutions and the theoretical and interpretability advantages of ℓ_0 solutions[18] by solving an ℓ_1 optimization problem. Further, we find that the optimum cost is the same whether we require an integer solution or not for more than 99.97% of solutions to Program Edge_{Len}, 100% of solutions to Edge_{Unif}, and 100% of solutions to Tri_{Unif}. Thus, the modeler can drop the integral constraint to save computation time while still being able to achieve an integral solution in most cases.

6.4 Comparing Optimal Cycle Representatives Against Different Loss Functions

We compare the optimal cycle representatives against different loss functions to study the extent to which the solutions produced by each technique vary. We consider two loss functions on an H_1 cycle representative $\mathbf{x} \in \mathbf{Z}_1(K)$:

$$L_{E\text{-}Len}(\mathbf{x}) = \sum_{\sigma \in \text{supp}(\mathbf{x})} \text{Length}(\sigma),$$

where $\text{Length}(\sigma)$ is the distance—as designated by the metric d used to define the VR complex—between the two vertices of a 1-simplex σ, and

$$L_{E\text{-}Unif}(\mathbf{x}) = \|\mathbf{x}\|_0 = \left|\text{supp}(\mathbf{x})\right|,$$

the number of 1-simplices (edges) in a representative.

We also consider two loss functions on 2-chains $\mathbf{v} \in \mathbf{C}_2(K)$, namely area-weighted loss:

$$L_{T\text{-}Area}(\mathbf{v}) = \sum_{\tau \in \text{supp}(\mathbf{v})} \text{Area}(\tau),$$

where $\text{Area}(\tau)$ is the area of a 2-simplex as computed by Heron's Formula, and uniform-weighted loss

$$L_{T\text{-}Unif}(\mathbf{v}) = \|\mathbf{v}\|_0 = \left|\text{supp}(\mathbf{v})\right|.$$

Remark 6.1. These weighted ℓ_0 loss functions differ from the objective functions used in the optimization problems presented in **Section 4**, which measure weighted ℓ_1 norm. However, weighted ℓ_0 norm and weighted ℓ_1 norm agree on solutions with {-1, 0, 1} coefficients, and (as reported in **Section 6.3**) nearly all cycle representatives for **Sections 5.1** and **5.2** data satisfy this condition, both pre- and post-optimization.In the special case where $\text{supp}(\mathbf{x})$ determines a simple closed polygonal curve c with vertices $(p^1, q^1), \ldots, (p^n, q^n) \in \mathbb{R}^2$, we also use the Surveyor's Area Formula (Braden, 1986) to quantify area of $\mathbf{X}$ as

$$L_{Sur\text{-}Area}(c) = \frac{1}{2}\left|\sum_{i=1}^{n} p^i q^{i+1} - p^{i+1} q^i\right|,$$

where, by convention, $p^{i+1} = p^1$ and $q^{i+1} = q^1$. We evaluate this function only when (i) the ambient point cloud of the VR complex is a subset of $\mathbb{R}^2$, b) $\text{supp}(\mathbf{x})$ forms a graph-theoretic cycle when regarded as a subset of edges in the combinatorial graph formed by 1-skeleton of K, and 3) no pair of distinct closed line segments intersect one another.

In the case when we compute the loss function of a corresponding optimal solution, we use the notation for the cost $C^*_\bullet := L_\bullet(\mathbf{x}^*_\bullet)$ to an edge-loss problem that finds optimal solution $\mathbf{x}^*_\bullet$, and $C^*_\bullet := L_\bullet(\mathbf{v}^*_\bullet)$ to a triangle-loss problem that finds optimal solution $\mathbf{v}^*_\bullet$. For instance, $C^{NI}_{E\text{-}Unif} = L_{E\text{-}Unif}(\mathbf{x}^{NI}_{E\text{-}Unif})$. We will also compute the loss functions of optimal solutions from differing optimizations. For instance, $L_{E\text{-}Len}(\mathbf{x}^{NI}_{E\text{-}Unif})$, and in that case, we do not use the $C^*_\bullet$ notation.

[17]We discuss the coefficients of the Erdős-Rényi complexes of **Section 5.3** in **Section 6.7**.

[18]Recall that, in the current discussion, ℓ_0 optimality refers to the *restricted* integer problem where coefficients are constrained to lie in {-1, 0, 1}. The unrestricted problem (about which we have nothing to say) may have quite different properties.

Figure 7 shows an example of various optimal cycle representatives obtained from Programs Edge^{NI}_{Unif}, Edge^{NI}_{Len}, Tri^{NI}_{Unif}, and Tri^{NI}_{Area} on an example point cloud drawn from the normal distribution in $\mathbb{R}^2$. In this example, solutions obtained from **Algorithm 1** and the program in **Eq. 8** are the same. Each subfigure is labeled by program in the upper left corner. The values of different loss functions evaluated on each optimal representative appear in the upper right corner. We do not compute $L_{T\text{-}Unif}$ or $L_{T\text{-}Area}$ of the optimal edge-loss minimal cycle representatives, as no bounding 2-chain for this 1-cycle is specified in the optimization.[19] We observe that various notions of optimality lead to differing cycle representatives, yet each solution to an optimization problem minimizes the loss function it is intended to optimize. This will not always be the case, as we will see momentarily.

Figure 8 reports ratios on the losses $L_{E\text{-}Unif}$, $L_{E\text{-}Len}$, and $L_{Sur\text{-}Area}$[20] for the eight PrsHCB optimization problems detailed in **Section 4** as well as the four edge-loss FCB problems from the program in **Eq. 8**, evaluated on the data from **Sections 5.1** and **5.2**. These ratios suggest that the uniform-weighted and length-weighted edge-loss cycles do minimize what they set out to minimize, namely, the number of edges and the total length, respectively. We also observe that intuitively the less-constrained solutions to the FCB program in **Eq. 8** can have a lower cost than the more-constrained solutions to the PrsHCB program in **Eq. 14**.

We also see that the edge-loss-minimal cycles have similar loss in terms of length and number of edges ($L_{E\text{-}Len}$ and $L_{E\text{-}Unif}$) whereas the triangle-loss-minimal cycles can have larger losses ($L_{E\text{-}Len}(\mathbf{x}_{T\text{-}Unif})$ and $L_{E\text{-}Unif}(\mathbf{x}_{T\text{-}Unif})$). We find that 63.28% of the $L_{E\text{-}Unif}$ minimal cycle representatives are also $L_{E\text{-}Len}$ minimal while 99.66% of the $L_{E\text{-}Len}$ minimal cycle representatives are also $L_{E\text{-}Unif}$ minimal across all cycle representatives from all data sets for PrsHCB cycles. Similarly, we find that 61.31% of the $L_{E\text{-}Unif}$ minimal cycle representatives are also $L_{E\text{-}Len}$ minimal while 99.32% of the $L_{E\text{-}Len}$ minimal cycle representatives are also $L_{E\text{-}Unif}$ minimal across all cycle representatives from all data sets for FCB cycles. This suggests that modelers can often use the length-weighted minimal cycle to substitute the uniform-weighted minimal cycle. However, the triangle-loss cycles can potentially provide very different results.

Counterintuitively, the $L_{T\text{-}Area}$ optimal cycle representative might not be the representative that encloses the smallest surveyor's area. As shown in **Figure 8**, we observe that 15.55% of $\mathbf{x}^{NI}_{E\text{-}Unif}$, 13.14% of $\mathbf{x}^{I}_{E\text{-}Unif}$, 23.59% of $\mathbf{x}^{NI}_{E\text{-}Len}$, and 23.59% of $\mathbf{x}^{I}_{E\text{-}Len}$ for the cycles from PrsHCB using the program in **Eq. 14** have an area smaller than that of the triangle-loss area-weighted optimal cycle $\mathbf{x}^{NI}_{T\text{-}Area}$. Similarly, 15.55% of $\mathbf{x}^{NI}_{E\text{-}Unif}$, 12.87% of $\mathbf{x}^{I}_{E\text{-}Unif}$, 24.53% of $\mathbf{x}^{NI}_{E\text{-}Len}$, and 24.53% of $\mathbf{x}^{I}_{E\text{-}Len}$ for the cycles from FCB using the program in **Eq. 8** have an area smaller than that of the triangle-loss area-weighted optimal cycle $\mathbf{x}^{NI}_{T\text{-}Area}$. Lastly, 3.22% of $\mathbf{x}^{I}_{T\text{-}Unif}$, 2.81% of $\mathbf{x}^{NI}_{T\text{-}Unif}$, and 2.95% of $\mathbf{x}^{I}_{T\text{-}Area}$ for the cycles found using the program in **Eq. 15** have an area smaller than that of the triangle-loss area-weighted optimal cycle $\mathbf{x}^{NI}_{T\text{-}Area}$.

In **Figure 9**, we provide an example illustrating why the triangle-loss area-weighted optimal cycle, solving Programs Tri^{NI}_{Area}, or Tri^{I}_{Area}, might not be the cycle that encloses the smallest surveyor's area. Another reason why the area-weighted triangle-loss cycles could have a larger enclosed area is that in the optimization problems, the loss function is the sum of the triangles the cycle bounds, not the real enclosed area. Therefore, the area-weighted triangle-loss cycle will have the optimal area-weighted optimal cost, but not necessarily the smallest enclosed area.

6.5 Comparative Performance and Precision of LP Solvers

Our experiments demonstrate that the choice of linear solver may impact speed, frequency of obtaining integer solutions, and frequency of obtaining ℓ_0 optimal solutions. While these particular results are subject to change due to regular updates to each platform, they illustrate the degree to which these factors can vary.

As discussed in **Section 6.1**, the GLPK solver performs much slower than the Gurobi solver in an initial set of experiments. The GLPK solver also finds non-integral solutions when solving a linear programming problem in Programs Edge^{NI}_{Unif}, and Edge^{NI}_{Len} more often than the Gurobi solver. On the same set of experiments as in **Section 6.1**, when finding the FCB using the program in **Eq. 8**, 9.74% of the edge-loss length-weighted minimal cycle representatives have non-integral entries, and 8.32% of the edge-loss uniform-weighted minimal cycle representatives have non-integral entries when using the GLPK solver, whereas when using the Gurobi solver, 0.12% of the length-weighted minimal cycle representatives have non-integral entries, and 0.04% of the uniform-weighted minimal cycle representatives have non-integral entries. For the length-weighted minimal cycle representatives, the non-integral solutions differ from an ℓ_0 optimal solution by a margin of machine error with both solvers. However, for the uniform-weighted minimal cycle representatives, the GLPK solver has 1.83% of its non-integral solutions differing from an ℓ_0 optimal solution by a margin not of machine epsilon, and the Gurobi solver has 0.02% of its non-integral solutions differing from an ℓ_0 optimal solution by a margin greater than machine epsilon. For the GLPK solver, when solving Program Edge^{NI}_{Unif}, instead of finding an integral solution, it occasionally finds a solution with fractional entries that sum to 1. For example, instead of assigning an edge a coefficient of 1, it sometimes assigns two edges each with a coefficient of 0.5. In that way, the solution is still ℓ_1 optimal, but no longer ℓ_0 optimal. Thus, the choice of linear solver may affect the optimization results.

6.6 Statistical Properties of Optimal Cycle Representatives With Regard to Various Other Quantities of Interest

6.6.1 Support of a Representative Forming a Single Loop in the Underlying Graph

The support of the original cycle, $\text{supp}(\mathbf{x}^{Orig}) \subseteq \mathbf{S}_1(K)$, need not be a cycle in the graph-theoretic sense. Concretely, this means that the nullity, p, of column submatrix $\partial_1[:, \mathbf{x}^{Orig}]$ may be strictly greater than 1. We refer to p informally as the "number of loops" in $\mathbf{x}^{Orig}$.

We are interested in exploring how often the support of an original cycle representative forms a single loop in

[19]We formulated an Obayashi-style linear program similar to Program in **Eq. 15** to compute the volume of edge-loss optimal cycles but in many cases it had no feasible solution.

[20]Recall, we only compute L_{Area} on the 2-dimensional distribution data.

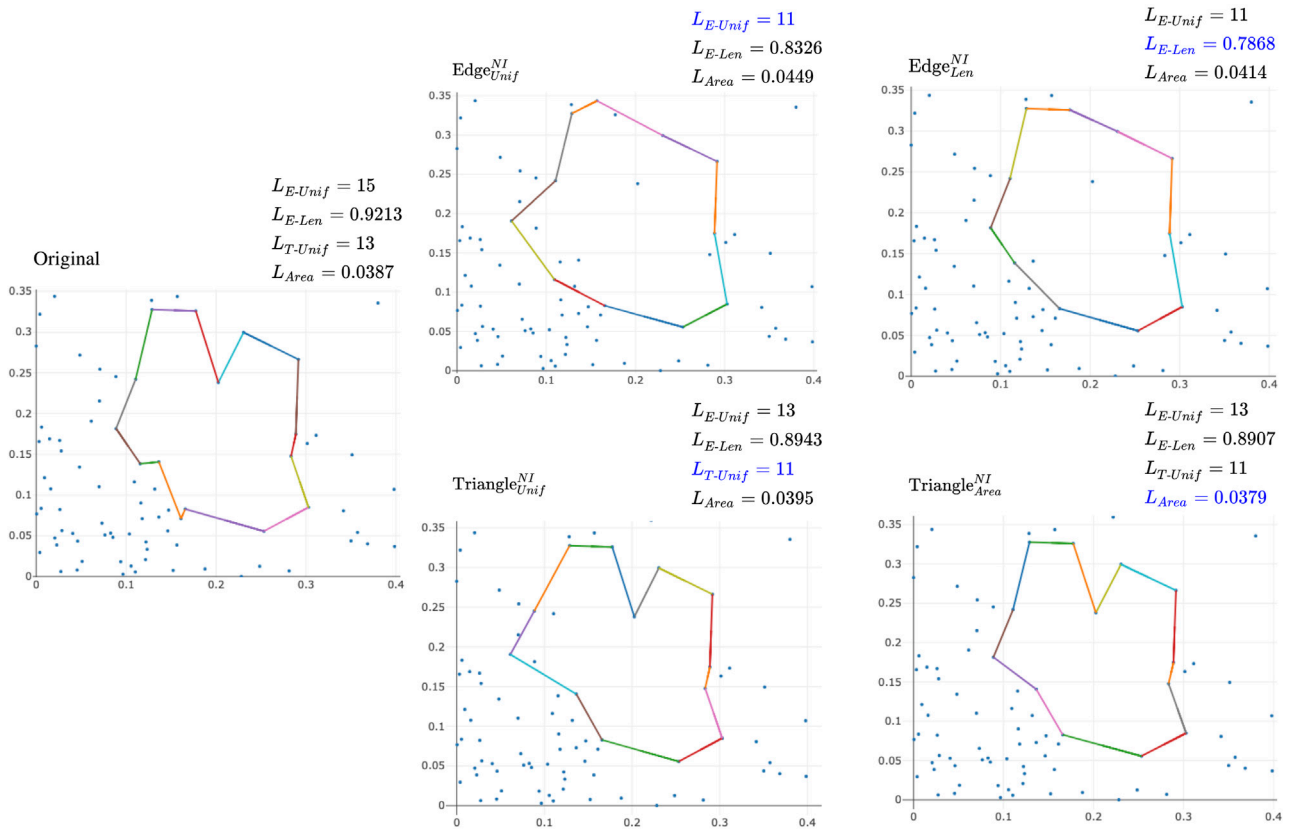


FIGURE 7 | Examples of different optimal cycles and cost against different loss functions using a point cloud of 100 points with ambient dimension two randomly drawn from a normal distribution. The upper left corner of each subfigure labels the optimization algorithm used to optimize the original cycle representative. The upper right corner of each subfigure records the different measures of the size of the optimal representative. Blue text represents the measure an algorithm sets out to optimize.

the underlying graph. We analyze each of the 360 synthetic data sets of various dimensions and distributions discussed in **Section 5.2** as well as the 100 Erdős-Rényi random complexes discussed in **Section 5.3** and display the results in **Figure 10**. We find that the majority of the original cycle representatives have one loop. After optimizing these cycle representatives with the edge-loss methods, we verify that all FCB and PrsHCB optimal cycles only have one loop, whereas 0.13% of the triangle-loss cycles have two loops. However, we observe that 91.93% of the optimal cycle representatives for Erdős-Rényi complexes have 1 loop, 5.81% have two loops, and 2.14% have more than two loops, with 15 as the maximum number of loops.

As shown in **Figure 11** the reduction in size of the original cycle, formalized as $\frac{C^*_\bullet}{L_\bullet(\mathbf{x}^{Orig})}$, correlates closely with the reduction in the number of loops by the optimization.

6.6.2 Overall Effectiveness of Optimization ($L_\bullet(\mathbf{x}^*_\bullet)$ vs. $L_\bullet(\mathbf{x}^{Orig})$)

We compare the optimal representatives against the original cycle representatives[21] with respect to edge-loss functions $L_{E\text{-}Unif}$ and $L_{E\text{-}Len}$. As shown in **Figure 12**, we find that the optimizations are in general effective in reducing the size of the cycle representative, especially for representatives with larger size. On each of the subfigures, the horizontal axis is the size of the original cycle representative and the vertical axis is the ratio between the loss of each optimal representative and the loss of the original representative:

$$\frac{C^*_\bullet}{L_\bullet(\mathbf{x}^{Orig})}.$$

The average ratio $\frac{C^{NI}_{E\text{-}Unif}}{L_{E\text{-}Unif}(\mathbf{x}^{Orig})}$ is 83.17%, aggregated over cycle representatives obtained from data described in **Section 5.1** and 90.35% aggregated over cycle representatives obtained from data described in **Section 5.2** for cycles obtained from the program in **Eq. 14**. The average ratio $\frac{C^{NI}_{E\text{-}Len}}{L_{E\text{-}Len}(\mathbf{x}^{Orig})}$ is 87.02% over cycle representatives obtained from data described in **Section 5.1** and 90.41% over cycle representatives obtained from data describedra **Section 5.2** for cycles obtained from the program in **Eq. 14**. The average ratio $\frac{C^{NI}_{T\text{-}Unif}}{L_{T\text{-}Unif}(\mathbf{x}^{Orig})}$ is 88.34% over cycle representatives obtained from data described in **Section 5.1** and 95.54% over cycle representatives obtained from data described in **Section 5.2** for cycles obtained from the program in **Eq. 15**.

[21]The remainder of this subsection excludes the Erdős-Rényi cycles.

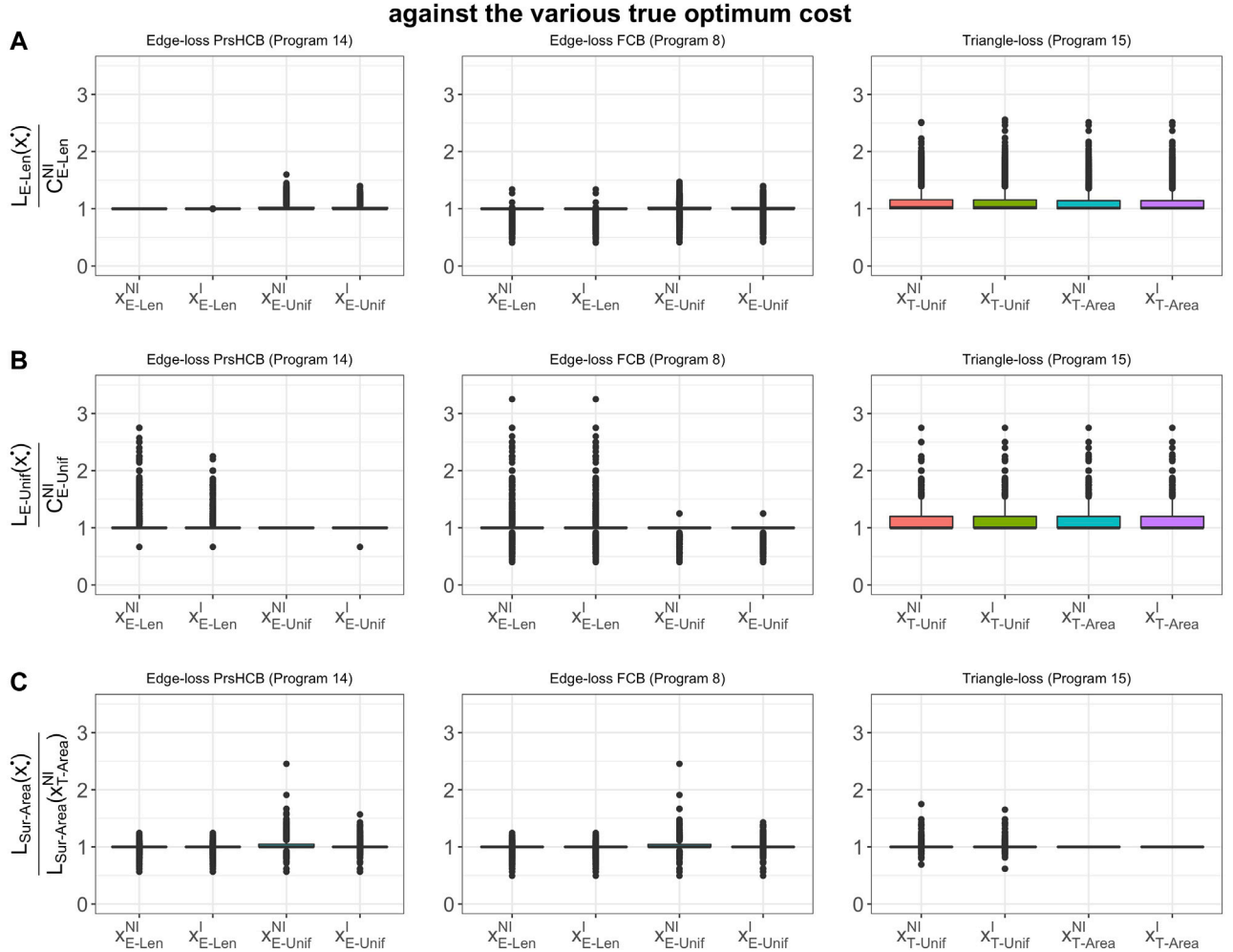


FIGURE 8 | Box plots of the ratios between **(A)** $L_{E\text{-}Len}(\mathbf{x}^*_\bullet)$ and $C^{NI}_{E\text{-}Len}$ **(B)** $L_{E\text{-}Unif}(\mathbf{x}^*_\bullet)$ and $C^{NI}_{E\text{-}Unif}$, and **(C)** $L_{Sur\text{-}Area}(\mathbf{x}^*_\bullet)$ and $L_{Sur\text{-}Area}(x^{NI}_{T\text{-}Area})$. Within each row, the denominator is fixed across all three columns, and corresponds to the PrsHCB cycles which are solutions to Programs Edge^{NI}_{Unif} row **(A)**, Edge^{NI}_{Len} row **(B)**, or Tri^{NI}_{Area} row **(C)**. The horizontal axis of each subplot is the type of optimal representative. The cost of the optimal solutions to the programs in **Eqs 8**, **14**, and Program Tri^{NI}_{Area} was equal regardless of the presence of an integer constraint in nearly all examples (as discussed in **Section 6.3**), resulting in two columns in each row having ratio 1. The data used in **(A)** and **(B)** aggregate over all cycle representatives from data described in **Sections 5.1** and **5.2**. The data used in **(C)** aggregate the 746 cycle representatives from 40 point clouds with ambient dimension of two from data described in **Section 5.2**. We observe that some edge-loss and uniform-weighted-triangle-loss optimal cycles have a surveyor's area strictly smaller than the denominator in row **(C)**; refer to **Figure 8** and **Section 6.4** to see why this may happen. It is possible for $L_{E\text{-}Unif}(\mathbf{x}^*_\bullet)$ to be strictly smaller than $C^{NI}_{E\text{-}Unif}$ because the cycle $\mathbf{x}^{NI}_{E\text{-}Unif}$ is calculated to be optimal relative to ℓ_1 loss, not $L_{E\text{-}Unif}$, which is a measure of ℓ_0 loss. We observe this behavior in the first plot on the second row, discussed in detail in **Section 6.3**.

6.6.3 Comparing Solutions to Integral Programs and Non-Integral Programs ($\mathbf{x}^{NI}_\bullet$ vs. $\mathbf{x}^{I}_\bullet$)

Among all cycle representatives found by solving the program in **Eq. 14**, 66.38% of them have $\mathbf{x}^{NI}_{E\text{-}Unif} = \mathbf{x}^{I}_{E\text{-}Unif}$, and 92.52% of them have $\mathbf{x}^{NI}_{E\text{-}Len} = \mathbf{x}^{I}_{E\text{-}Len}$. We find $\mathbf{x}^{NI}_{T\text{-}Unif} = \mathbf{x}^{I}_{T\text{-}Unif}$ for 80.44% of the cycle representatives and $\mathbf{x}^{NI}_{T\text{-}Area} = \mathbf{x}^{I}_{T\text{-}Area}$ for 100% of the cycle representatives when using the triangle-loss program in **Eq. 15**. Thus, the presence or absence of integer constraints rarely impacts the result of an area- or length-weighted program, but often impacts solutions of a uniform-weighted program. We saw in **Section 6.3** that essentially all solutions had coefficients in $\{-1, 0, 1\}$ regardless of integer or non-integer constraints. As such, we conjecture that the higher rate of different solutions in the uniform-weighted problems could result from a larger number of distinct optimal solutions and be a feature of particular choice of solution selected by the linear solvers, rather than the non-existence of a particular integer solution.

6.6.4 Cycle Representative Size for Different Distributions and Dimensions

Figure 13 provides a summary of the size and number of cycle representatives found for each distribution data set described in

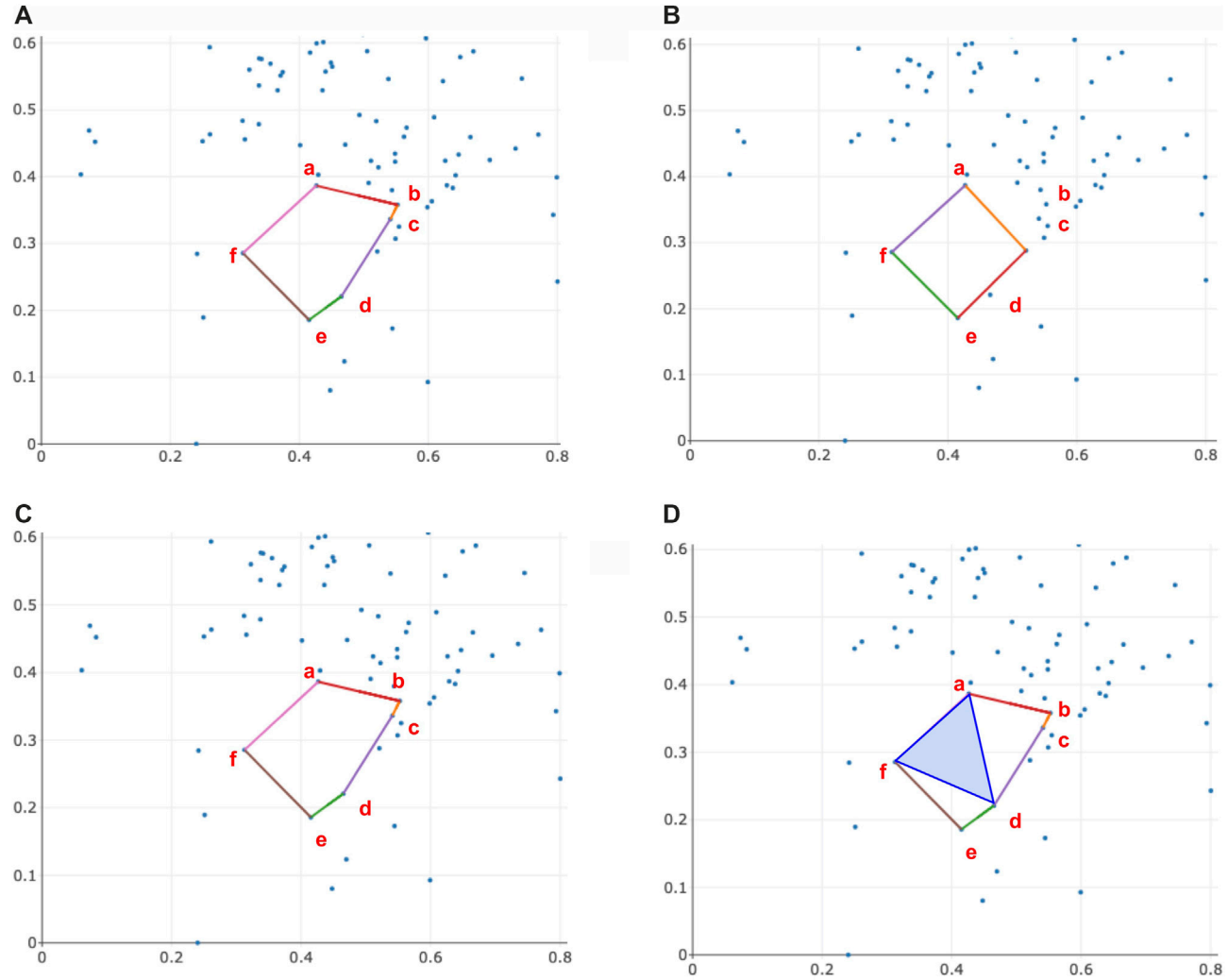


FIGURE 9 | An example illustrating when the area enclosed by the triangle-loss area-weighted optimal cycle, solution to Program Tri^{NI}_{Area}, can be larger than the area enclosed by the edge-loss length-weighted minimal cycle, solution to Program Edge^{NI}_{Len} **(A)** is the original cycle of a representative point cloud in $\mathbb{R}^2$ drawn from the normal distribution **(B)** is the length-weighted edge-loss optimal cycle **(C)** is the area-weighted triangle-loss optimal cycle, in this example, it is the same cycle as the original cycle **(D)** is the area-weighted minimal cycle where the blue shaded area marks the triangle born at the death time of the cycle. Constraint **Eq. 9** specifies that the area-weighted optimal cycle must contain the 2-simplex born at the death time of the cycle. Therefore, this cycle must contain (a, d, f) because it was born at the death time. The length-weighted minimal cycle does not have this constraint, and as such, can result in a smaller area.

Section 5.2. We observe that there tend to be more and larger (with respect to ℓ_0 norm) representatives in higher dimensions.

6.6.5 Duplicate Intervals in the Barcode

Of all data sets analyzed, only **Klein** and **C.elegans** have barcodes in which two or more intervals had equal birth and death times (that is, bars with multiplicity ≥ 2). Among the 107 total intervals of the **C.elegans** data set, there are 75 unique intervals, 10 intervals with multiplicity two, and one interval each with multiplicity three, four, and five. The duplicate bars in the **C.elegans** data set are noteworthy for having produced the sole example of an optimized cycle representative $\mathbf{x}^{NI}_{E\text{-}Unif}$ with coefficients outside $\{-1, 0, 1\}$ (in particular, it had coefficients in $\{-0.5, 0.5\}$).

Among the 257 total intervals of the **Klein** data set, there are 179 unique intervals, 1 interval that is repeated twice, and two intervals that are repeated 38 times. For the **Klein** data set, if we replace the distance matrix provided by Otter et al. (2017) with the Euclidean distance matrix calculated using Julia (the maximum difference between the two matrices is on the scale of 10^{-5}), we obtain only one interval that is repeated twice. This indicates that duplicate intervals are rare in practice, at least in dimension 1.

6.6.6 Edge-Loss Cycle Representatives FCB vs. PrsHCB

We find that for 84.52% of Edge^{NI}_{Unif}, 90.84% of Edge^{I}_{Unif}, 93.49% of Edge^{NI}_{Len}, and 93.49% of Edge^{I}_{Len}, the FCB edge-loss cycle representatives found by the program in **Eq. 8** and the PrsHCB edge-loss cycles from the program in **Eq. 14** are the same, i.e. the ℓ_1 norm of their difference is 0. As mentioned in **Remark 3.1**, the FCB cycles may not have the same death time as $\mathbf{x}^{Orig}$. For the real-world data sets, 6.72% of the (Edge^{NI}_{Len}) and (Edge^{I}_{Len}), 7.65% of the (Edge^{NI}_{Unif}) and

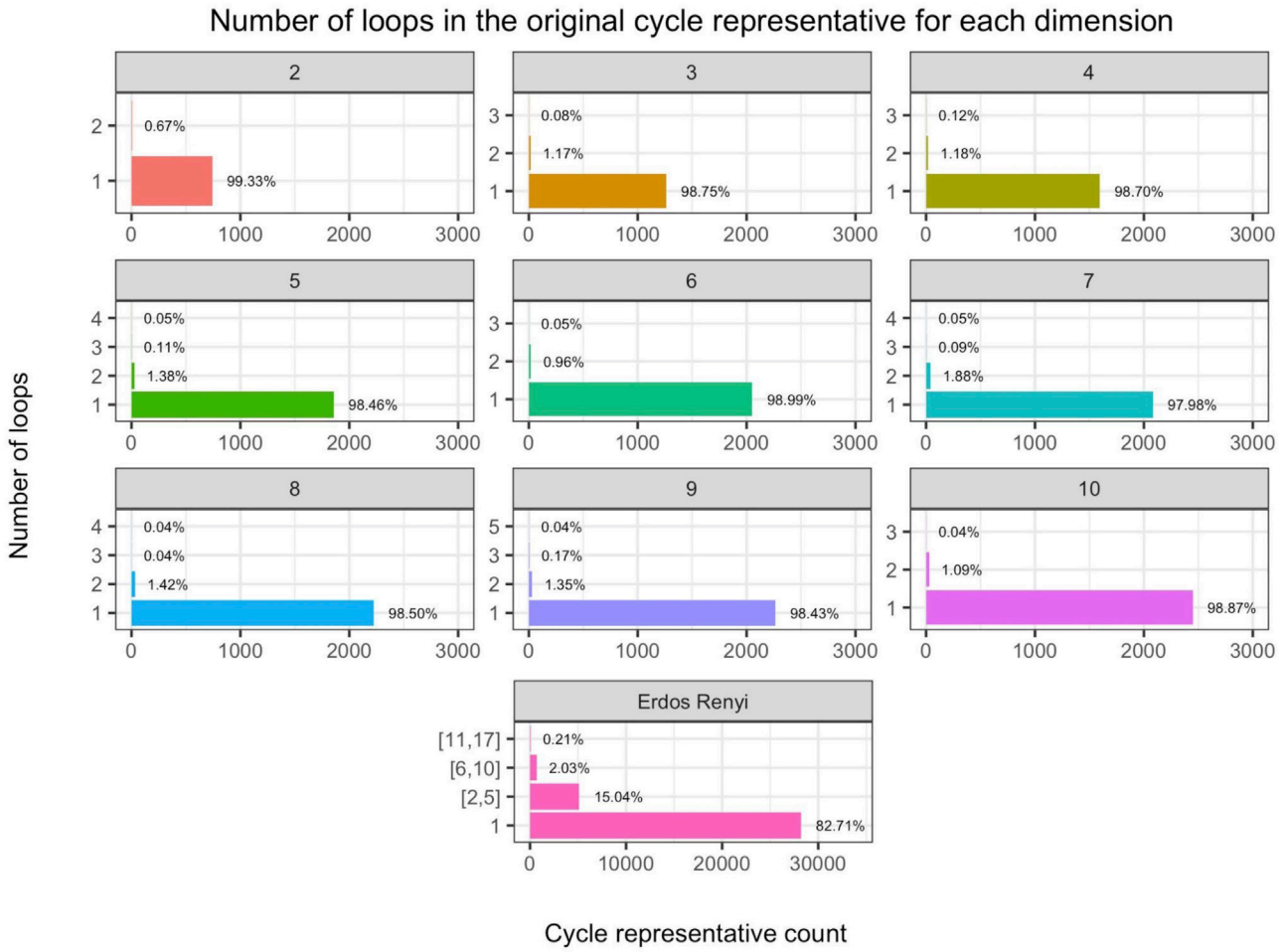


FIGURE 10 | (Rows 1–3) Number of loops in the original cycle representative aggregated by dimension (labeled by subfigure title) in the 360 randomly generated distribution data sets discussed in **Section 5.2** and (Row 4) same for the Erdős-Rényi random complexes discussed in **Section 5.3**, where we bin cycle representatives that have two to five loops, 6–10, loops, or more than 10 loops. The horizontal axis is the number of cycle representatives and the vertical axis is the number of loops in the original representative. We observe that for the distribution data, an original cycle representative can have up to 5 loops in higher dimensions, and in general, it is uncommon to find an original representative with multiple loops. However, we observe that 17.47% of the cycle representatives for Erdős-Rényi complexes have more than 1 loop, with a maximum number of 17 loops in a cycle representative.

4.48% of (Edge^{I}_{Unif}) have lifetimes different than $\mathbf{x}^{Orig}$. For the randomly generated distribution data sets, 7.11% of the (Edge^{NI}_{Len}) and (Edge^{NI}_{Len}), 8.06% of the $(\text{Edge}^{NI}_{Unif})$ and 4.25% of (Edge^{I}_{Len}) have lifetimes different than $\mathbf{x}^{Orig}$. All cycle representatives with lifetimes different than $\mathbf{x}^{Orig}$ have death time beyond that of $\mathbf{x}^{Orig}$.

6.7 Optimal Cycle Representatives for Erdős-Rényi Random Clique Complexes

We observe qualitatively different behavior in cycle representatives from the Erdős-Rényi random clique complexes. Among the 34, 214 cycle representatives from the 100 dissimilarity matrices found by solving the programs in **Eqs 14**, **15**, we find that 91.04% of the original cycle representatives have entries in $\{-1, 0, 1\}$ and 99.75% of the original cycle representatives have integral entries. We have 3.89% of the length-weighted edge-loss representatives, 4.49% of the uniform-weighted edge-loss representatives, and 3.52% of the uniform-weighted triangle-loss representatives with entries not in $\{-1, 0, 1\}$. We find 2.66% of the length-weighted edge-loss representatives, 3.57% of the uniform-weighted edge-loss representatives, and 1.58% of the uniform-weighted triangle-loss representatives with non-integral entries when not requiring integral solutions.

We find $\frac{L_{E-Unif}(\mathbf{x}^{NI}_{E-Unif})}{L_{E-Unif}(\mathbf{x}^{I}_{E-Unif})} > 1$ for 1.07% of the cycle representatives and $\frac{L_{E-Len}(\mathbf{x}^{NI}_{E-Len})}{L_{E-Len}(\mathbf{x}^{I}_{E-Len})} > 1$ for 1.09% of the representatives. All such representatives have entries outside of $\{-1, 0, 1\}$ and involve some fractional entries. An average of 96.75% of the nonzero entries in the reduced boundary matrices are in $\{-1, 1\}$, 2.15% in $\{-2, 2\}$, and 0.27% with an absolute value greater than or equal to 3.

Because of the non-integrality of some original cycle representatives found by the persistence algorithm, we fail to find an integral solution for 0.27% of the edge-loss representatives and 0.11% of the triangle-loss representatives.

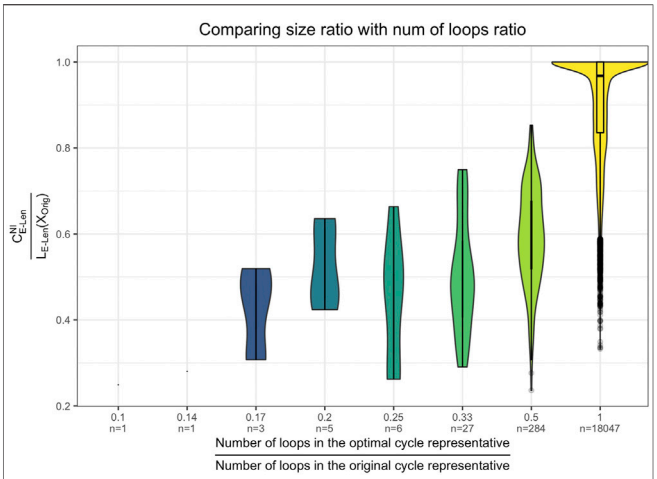


FIGURE 11 | Violin plot of the effectiveness of the optimization as a function of the number of loops in the original cycle representative. Results are aggregated over the data sets from **Section 5.1** and **Section 5.2**. The *x*-axis shows the size reduction in terms of number of loops, and the *y*-axis shows the size reduction in terms of the length of the cycle. We see that in general, the reduction in size of the original cycle mostly comes from the reduction in the number of loops by the optimization.

A partial explanation for this behavior can be found in the work of (Costa et al., 2015). Here, the authors show that a connected two-dimensional simplicial complex for which every subcomplex has fewer than three times as many edges as vertices must have the homotopy type of a wedge of circles, 2-spheres, and real projective planes. With high probability, certain ranges of thresholds for the i.i.d. dissimilarity matrices on which the Erdős-Rényi random complexes are built produces random complexes with approximately such density patterns at each vertex. Thus, some of the persistent cycles are highly likely to correspond to projective planes. Because of their non-orientability, the corresponding minimal generators could be expected to have entries outside of the range $\{-1, 0, 1\}$.

7 CONCLUSION

In this work, we provide a theoretical, computational, and empirical user's guide to optimizing cycle representatives against four criteria of optimality: total length, number of edges, internal volume, and area-weighted internal volume. Utilizing this framework, we undertook a study on statistical properties of minimal cycle representatives for $\mathbf{H}_1$ homology found via linear programming. In doing so, we made the following four main contributions.

1. We developed a publicly available code library (Li and Thompson, 2021) to compute persistent homology with rational coefficients, building on the software package Eirene (Henselman and Ghrist, 2016) and implemented and extended algorithms from (Escolar and Hiraoka, 2016; Obayashi, 2018) for computing minimal cycle representatives. The library employs standard linear solvers (GLPK and Gurobi) and implements various acceleration techniques described in **Section 4.4** to make the computations more efficient.
2. We formulated specific recommendations concerning procedural factors that lie beyond the scope of the optimization problems per se (for example, the process used to generate inputs to a solver) but which bear directly on the overall cost of computation, and of which practitioners should be aware.
3. We used this library to compute optimal cycle representatives for a variety of real-world data sets and randomly generated point clouds. Somewhat surprisingly, these experiments demonstrate that computationally advantageous properties are typical for persistent cycle representatives in data. Indeed, we find that we are able to compute uniform/length-weighted optimal cycles for all data sets we considered, and that we are able to compute triangle-loss optimal cycles for all but six cycle representatives, which fail due to the large number of triangles (more than 20 million) used in the optimization problem. Computation time information is summarized in **Table 1** and **Table 2**.
   Consequently, heuristic techniques may provide efficient means to extract solutions to cycle representative optimization problems across a broad range of contexts. For example, we find that edge-loss optimal cycles are faster to compute than triangle-loss optimal cycles for cycle representatives with a longer persistence interval, whereas for cycles with shorter persistence intervals, the triangle-loss cycle can be less computationally expensive to compute.
4. We provided statistics on various minimal cycle representatives found in these data, such as their effectiveness in reducing the size of the original cycle representative found by the persistence algorithm and their effectiveness evaluated against different loss functions. In doing so, we identified consistent trends across samples that address the questions raised in **Section 1**.
   a. Optimal cycle representatives are often significant improvements in terms of a given loss function over the initial cycle representatives provided by persistent homology computations (typically, by a factor of 0.3–1.0). Interestingly, we find that area-weighted triangle-loss optimal cycle representatives can enclose a greater area than length- or uniform-weighted optimal cycle representatives.
   b. We find that length-weighted edge-loss optimal cycles are also optimal with respect to a uniform-weighted edge-loss function upwards of 99% of the time in the data we studied. This suggests that one can often find a solution that is both length-weighted minimal and uniform-weighted minimal by solving only the length-weighted minimal optimization problem. However, the triangle-loss optimal cycles can have a relatively higher length-weighted edge-loss or uniform-weighted edge-loss than the length/uniform-weighted minimal cycles. Thus, computing triangle-loss optimal cycles might provide distinct information and insights.
   c. Strikingly, all but one ℓ_1 optimal representatives were also ℓ_0 optimal (that is, ℓ_0 optimal among cycles taking $\{0, 1, -1\}$ coefficients; ℓ_0 optimality among cycles taking $\mathbb{Z}$ coefficients was not tested) in the real-world and synthetic point cloud data. Thus, it appears that solutions to the NP-hard problem of finding ℓ_0 optimal cycle representatives can often be solved using linear programming in real data. In the Erdős-Rényi random complexes, qualitatively different behavior was found; this may relate to the fact that spaces in this random

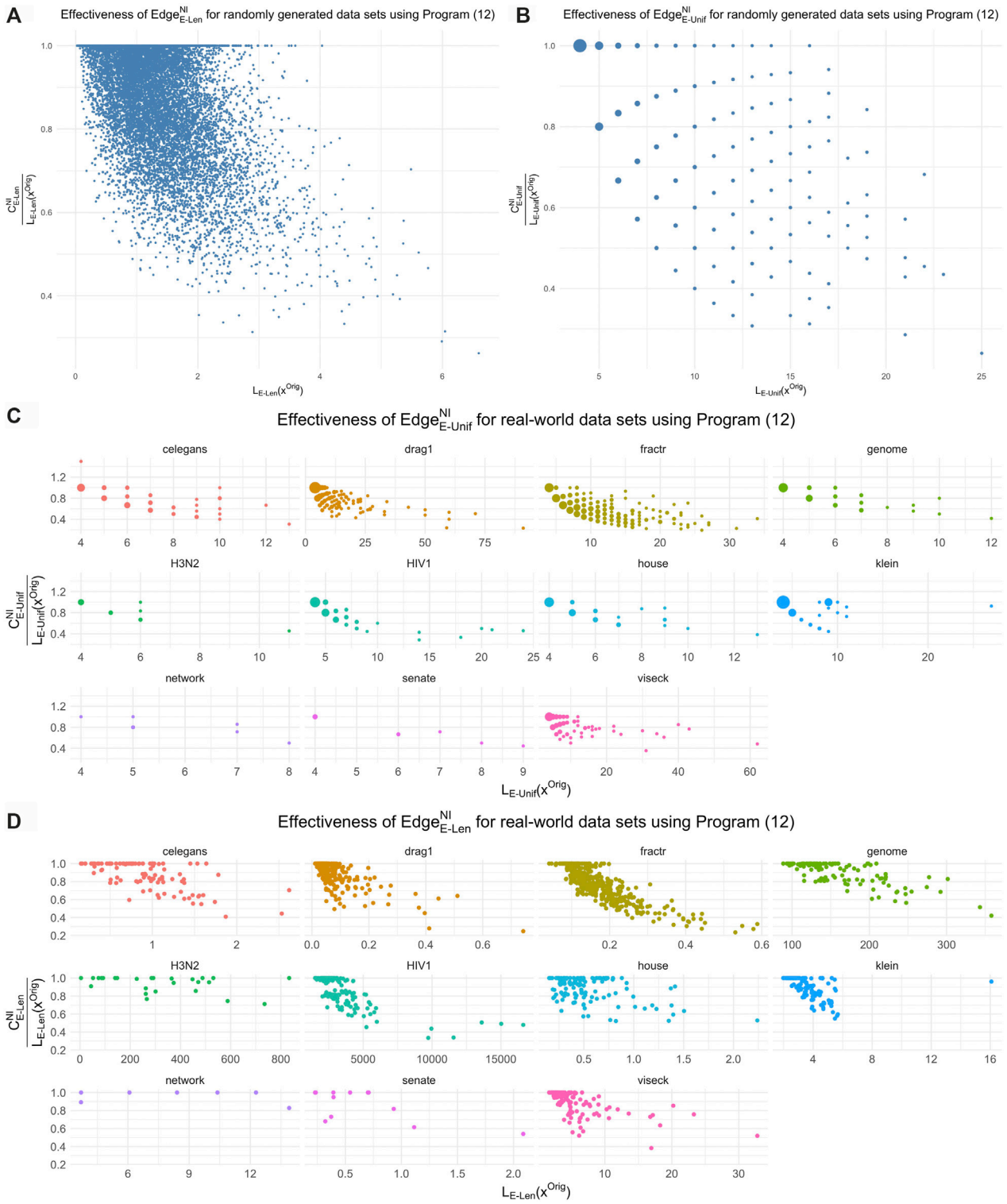


FIGURE 12 | The effectiveness of length-weighted and uniform-weighted optimization for the data sets in **Sections 5.1** and **5.2** in reducing the size of the original cycle representative found by the persistence algorithm. In each subfigure, the horizontal axis is the size of the original representative and the vertical axis is the ratio between the size of the optimal representative and the size of the original representative. The uniform-weighted graphs appear more sparse because reductions in the cost $L_{T\text{-}Unif}(\mathbf{x}^{Orig})$ can only come in multiples of the reciprocal of the original length. The node size in the uniform-weighted graphs corresponds to the number of overlapping points.

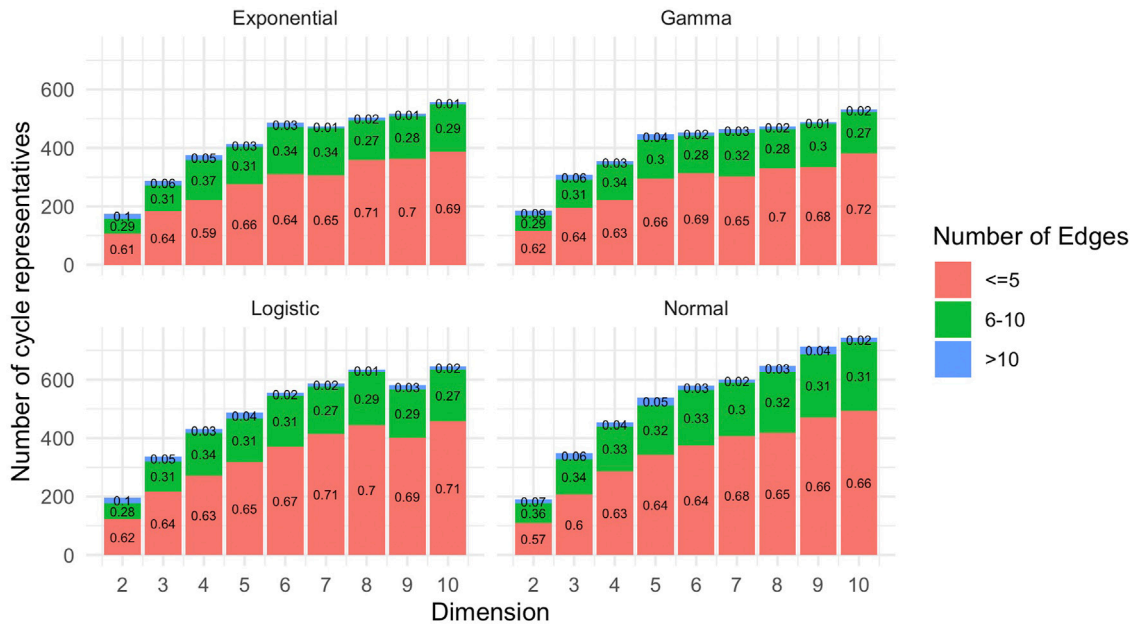


FIGURE 13 | The number of original cycle representatives and the number of edges within each original representative for data described in **Section 5.2**. These plots aggregate all cycle representatives for each dimension of a particular distribution. The horizontal axis for each subplot is the dimension of the data set, and the vertical axis is the number of cycle representatives found in each dimension. In general, we see there are more cycle representatives in higher dimensional data sets. Each bar is partitioned by the number of edges of the representative. We observe that as dimension increases, there tend to be more cycle representatives with more edges.

family contain non-orientable subcomplexes with high probability.

Several questions lie beyond the scope of this text and merit future investigation. For example, while the methods discussed in **Section 4** apply equally to homology in any dimension, we have focused our empirical investigation exclusively in dimension one; it would be useful and interesting to compare these results with homology in higher dimensions. It would likewise be interesting to compare with different weighting strategies on simplices, and loss functions other than ℓ_0 and ℓ_1, e.g. ℓ_2. Future work may also consider whether the modified approach to the edge-loss minimization program in **Eq. 14** could be incorporated into persistence solvers themselves, as pioneered in (Escolar and Hiraoka, 2016). Unlike the programs formulated in this earlier work, the program in **Eq. 14** requires information about the death times of cycles in addition to their births; typically this information is not available until after the persistence computation has already finished, so new innovations would probably be needed to make progress in this direction.

DATA AVAILABILITY STATEMENT

The datasets presented in this study can be found in online repositories. The names of the repository/repositories and accession number(s) can be found below: https://github.com/TDAMinimalGeneratorResearch/minimal-generator.

AUTHOR CONTRIBUTIONS

GH-P wrote the Eirene code. LL and CT wrote the rest of the code and performed all experiments. LL created all figures and tables. GH-P and LZ designed, directed, and supervised the project. GH-P developed the theory in the **Supplementary Material**. All authors contributed to the analysis of the results and to the writing of the manuscript.

FUNDING

This material is based upon work supported by the National Science Foundation under grants no. DMS-1854683, DMS-1854703, and DMS-1854748. GH-P is a member of the Centre for Topological Data Analysis, supported by the EPSRC grant New Approaches to Data Science: Application Driven Topological Data Analysis EP/R018472/1.

ACKNOWLEDGMENTS

The authors are grateful to David Turner for helpful advice on selection of linear solvers. They also thank and acknowledge the initial work done by Robert Angarone and Sophia Wiedemann on this study.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at: https://www.frontiersin.org/articles/10.3389/frai.2021.681117/full#supplementary-material

REFERENCES

Bendich, P., Marron, J. S., Miller, E., Pieloch, A., and Skwerer, S. (2016). Persistent Homology Analysis of Brain Artery Trees. *Ann. Appl. Stat.* 10, 198–218. doi:10.1214/15-AOAS886

Bertsimas, D., and Tsitsiklis, J. (1997). *Introduction to Linear Optimization*. (Belmont, MA: Athena Scientific).

Bhaskar, D., Manhart, A., Milzman, J., Nardini, J. T., Storey, K. M., Topaz, C. M., et al. (2019). Analyzing Collective Motion with Machine Learning and Topology. *Chaos: Interdiscip. J. Nonlinear Sci.* 29, 123125. doi:10.1063/1.5125493

Bhattacharya, S., Ghrist, R., and Kumar, V. (2015). Persistent Homology for Path Planning in Uncertain Environments. *IEEE Trans. Robotics* 31, 578–590. doi:10.1109/TRO.2015.2412051

Borradaile, G., Maxwell, W., and Nayyeri, A. (2020). "Minimum Bounded Chains and Minimum Homologous Chains in Embedded Simplicial Complexes," in *36th International Symposium on Computational Geometry (SoCG 2020)*. *Leibniz Int. Proc. Informatics* 21, 21:1–21:15.

Boyd, S., and Vandenberghe, L. (2004). *Convex Optimization*. Belmont, MA: Cambridge University Press.

Braden, B. (1986). The Surveyor's Area Formula. *Coll. Math. J.* 17, 326–337. doi:10.1080/07468342.1986.11972974

Brüel-Gabrielsson, R., Ganapathi-Subramanian, V., Skraba, P., and Guibas, L. J. (2020). Topology-aware Surface Reconstruction for point Clouds. *Comput. Graphics Forum* 39, 197–207. doi:10.1111/cgf.14079

Carlsson, G. (2009). Topology and Data. *Bull. Am. Math. Soc.* 46, 255–308.

Chan, J. M., Carlsson, G., and Rabadan, R. (2013). Topology of Viral Evolution. *Proc. Natl. Acad. Sci.* 110, 18566–18571. doi:10.1073/pnas.1313480110

Chen, C., and Freedman, D. (2010a). Measuring and Computing Natural Generators for Homology Groups. *Comput. Geometry* 43, 169–181. doi:10.1016/j.comgeo.2009.06.004

Chen, C., and Freedman, D. (2010b). "Hardness Results for Homology Localization," in *SODA '10: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*. USA: Society for Industrial and Applied Mathematics).

Chen, C., and Freedman, D. (2008). "Quantifying Homology Classes. Albers S," in *25th International Symposium On Theoretical Aspects Of Computer Science*. (Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum Fuer Informatik). Editor P. Weil, 1, 169–180. doi:10.4230/LIPIcs.STACS.2008.1343

Cohen-Steiner, D., Edelsbrunner, H., and Morozov, D. (2006). "Vines and Vineyards by Updating Persistence in Linear Time," in Proceedings of the Twenty-Second Annual Symposium on Computational Geometry, 119–126.

Costa, A., Farber, M., and Horak, D. (2015). Fundamental Groups of Clique Complexes of Random Graphs. *Trans. Lond. Math. Soc.* 2, 1–32. doi:10.1112/tlms/tlv001

Davis, T. A., and Hu, Y. (2011). The university of florida Sparse Matrix Collection. *ACM Trans. Math. Softw.* 38, 1–25. doi:10.1145/2049662.2049663

Dey, T. K., Sun, J., and Wang, Y. (2010). "Approximating Loops in a Shortest Homology Basis from point Data," in *SoCG '10: Proceedings of the Twenty-Sixth Annual Symposium on Computational Geometry*. (New York, NY, USA: Association for Computing Machinery), 166–175. doi:10.1145/1810959.1810989

Dey, T. K., Hirani, A. N., and Krishnamoorthy, B. (2011). Optimal Homologous Cycles, Total Unimodularity, and Linear Programming. *SIAM J. Comput.* 40, 1026–1044. doi:10.1137/100800245

Dey, T. K., Li, T., and Wang, Y. (2018). "Efficient Algorithms for Computing a Minimal Homology Basis," in *Latin American Symposium on Theoretical Informatics*. Springer, 376–398.

Dey, T. K., Hou, T., and Mandal, S. (2019). "Persistent 1-cycles: Definition, Computation, and its Application," in *International Workshop on Computational Topology in Image Context*. Málaga, Spain: Springer, 123–136.

Edelsbrunner, H., and Harer, J. (2008). Persistent Homology–A Survey. *Discrete Comput. Geometry - DCG* 453, 257–282. doi:10.1090/conm/453/08802

Edelsbrunner, H., and Harer, J. (2010). *Computational Topology: An Introduction*. Providence, RI: Applied Mathematics (American Mathematical Society).

Emmett, K., Schweinhart, B., and Rabadan, R. (2015). *Multiscale Topology of Chromatin Folding*. arXiv preprint arXiv:1511.01426.

Erickson, J., and Whittlesey, K. (2009). "Greedy optimal homotopy and homology generators. *SODA* 5, 1038–1046.

Escolar, E. G., and Hiraoka, Y. (2016). "Optimal Cycles for Persistent Homology via Linear Programming," in *Optimization in the Real World*. Tokyo: Springer, 79–96.

Escolar, E., and Hiraoka, Y. (2021). *OptiPersLP - optimal cycles in persistence via linear programming*. Available at: https://bitbucket.org/remere/optiperslp/src/master/.

Ghrist, R. (2008). Barcodes: the Persistent Topology of Data. *Bull. Am. Math. Soc.* 45, 61–75. doi:10.1090/s0273-0979-07-01191-3

Ghrist, R. (2014). *Elementary Applied Topology*. (Seattle: CreateSpace Independent Publishing Platform).

Giusti, C., Ghrist, R., and Bassett, D. S. (2016). Two's Company, Three (Or More) Is a Simplex. *J. Comput. Neurosci.* 41, 1–14.

GNU Project (2012). *GLPK (GNU Linear Programming Kit)*. Available at:.

Gurobi Optimization, L. (2020). *Gurobi Optimizer Reference Manual*.

Hatcher, A., and Press, C. U. of Mathematics CUD (2002). Algebraic Topology. *Algebraic Topology*. Cambridge, UK: Cambridge University Press.

Henselman, G., and Dłotko, P. (2014). "Combinatorial Invariants of Multidimensional Topological Network Data," In IEEE Global Conference on Signal and Information Processing (GlobalSIP). IEEE, 828–832.

Henselman, G., and Ghrist, R. (2016). *Matroid Filtrations and Computational Persistent Homology*. Available at: http://adsabs.harvard.edu/abs/2016arXiv160600199H (Accessed May, 2021).

Henselman-Petrusek, G. (2016). Eirene (Julia Library for Computational Persistent Homology). Available at: https://github.com/Eetion/Eirene.jl (Accessed May, 2021). Commit 4f57d6a0e4c030202a07a60bc1bb1ed1544bf679.

Hiraoka, Y., Nakamura, T., Hirata, A., Escolar, E. G., Matsue, K., and Nishiura, Y. (2016). Hierarchical Structures of Amorphous Solids Characterized by Persistent Homology. *Proc. Natl. Acad. Sci.* 113, 7035–7040. doi:10.1073/pnas.1520877113

Kovacev-Nikolic, V., Bubenik, P., Nikolić, D., and Heo, G. (2016). Using Persistent Homology and Dynamical Distances to Analyze Protein Binding. *Stat. App. Genet. Mol Biol.* 15, 19–38. doi:10.1515/sagmb-2015-0057

Li, L., and Thompson, C. (2021). *Source Code,* Minimal Cycle Representatives In Persistent Homology Using Linear Programming: An Empirical Study With User's Guide. Available at: https://github.com/TDAMinimalGeneratorResearch/minimal-generator (Accessed May, 2021).

Los Alamos National Laboratory (2021). *HIV Database*. Available at: http://www.hiv.lanl.gov/content/index.

McGuirl, M. R., Volkening, A., and Sandstede, B. (2020). Topological Data Analysis of Zebrafish Patterns. *Proc. Natl. Acad. Sci.* 117, 5113–5124. doi:10.1073/pnas.1917763117

Milosavljević, N., Morozov, D., and Skraba, P. (2011). "Zigzag Persistent Homology in Matrix Multiplication Time," in *SoCG '11: Proceedings of the Twenty-Seventh Annual Symposium on Computational Geometry*. New York, NY, USA: Association for Computing Machinery), 216–225. doi:10.1145/1998196.1998229

Newman, M. E. (2006). Finding Community Structure in Networks Using the Eigenvectors of Matrices. *Phys. Rev. E* 74, 036104. doi:10.1103/physreve.74.036104

Obayashi, I., Wada, T., Tunhua, T., Miyanaga, J., and Hiraoka, Y. (2021). *HomCloud: Data Analysis Software Based on Persistent Homology)*. Available at: https://homcloud.dev/.

Obayashi, I. (2018). Volume-optimal Cycle: Tightest Representative Cycle of a Generator in Persistent Homology. *SIAM J. Appl. Algebra Geometry* 2, 508–534.

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., and Harrington, H. A. (2017a). A Roadmap for the Computation of Persistent Homology. *EPJ Data Sci.* 6, 17. doi:10.1140/epjds/s13688-017-0109-5

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., and Harrington, H. A. (2017b). A Roadmap for the Computation of Persistent Homology. Available at: https://github.com/n-otter/PH-roadmap

Petri, G., Scolamiero, M., Donato, I., and Vaccarino, F. (2013). Topological Strata of Weighted Complex Networks. *PloS one* 8, e66506. doi:10.1371/journal.pone.0066506

Schweinhart, B. (2015). *Statistical Topology of Embedded Graphs*. Ph.D. thesis. Princeton, NJ: Princeton University.

Singh, G., Mémoli, F., and Carlsson, G. E. (2007). Topological Methods for the Analysis of High Dimensional Data Sets and 3d Object Recognition. *SPBG* 91, 100. doi:10.2312/SPBG/SPBG07/091-100

Sizemore, A. E., Phillips-Cremins, J. E., Ghrist, R., and Bassett, D. S. (2019). The Importance of the Whole: Topological Data Analysis for the Network Neuroscientist. *Netw. Neurosci.* 3, 656–673. doi:10.1162/netn_a_00073

Sporns, O. (2006). Small-world Connectivity, Motif Composition, and Complexity of Fractal Neuronal Connections. *Bio. Syst.* 85, 55–64. doi:10.1016/j.biosystems.2006.02.008

Stanford University Computer Graphics Laboratory (1999). *The Stanford 3D Scanning Repository*. (Accessed May, 2021).

Tahbaz-Salehi, A., and Jadbabaie, A. (20082008). Distributed Coverage Verification in Sensor Networks without Location Information. *IEEE Conf. Decis. Control.*, 4170–4176. doi:10.1109/CDC.2008.4738751

Tahbaz-Salehi, A., and Jadbabaie, A. (2010). Distributed Coverage Verification in Sensor Networks without Location Information. *IEEE Trans. Automatic Control.* 55, 1837–1849. doi:10.1109/TAC.2010.2047541

Topaz, C. M., Ziegelmeier, L., and Halverson, T. (2015). Topological Data Analysis of Biological Aggregation Models. *PloS one* 10, e0126383. doi:10.1371/journal.pone.0126383

Ulmer, M., Ziegelmeier, L., and Topaz, C. M. (2019). A Topological Approach to Selecting Models of Biological Experiments. *PLOS ONE* 14, 1–18. doi:10.1371/journal.pone.0213679

Vanderbei, R. J. (2014). "Linear Programming Foundations and Extensions," in *International Series in Operations Research Management Science*. 4th ed. edn.v.2014 New York: Springer.

Vasudevan, R., Ames, A. D., and Bajcsy, R. (2011). Human Based Cost from Persistent Homology for Bipedal Walking," in *18th IFAC World Congress*. *IFAC Proc. Volumes* 44, 3292–3297. doi:10.3182/20110828-6-IT-1002.03807

Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., and Shochet, O. (1995). Novel Type of Phase Transition in a System of Self-Driven Particles. *Phys. Rev. Lett.* 75, 1226–1229. doi:10.1103/PhysRevLett.75.1226

Wu, P., Chen, C., Wang, Y., Zhang, S., Yuan, C., Qian, Z., et al. (2017). "Optimal Topological Cycles and Their Application in Cardiac Trabeculae Restoration," in *Proceedings of the Twenty- IJCAI-19*, International Joint Conferences on 1136 Artificial Intelligence Organization, 80–92.

Xia, K., and Wei, G. W. (2014). Persistent Homology Analysis of Protein Structure, Flexibility, and Folding. *Int. J. Numer. Methods Biomed. Eng.* 30, 814–844. doi:10.1002/cnm.2655

Xia, K., Li, Z., and Mu, L. (2018). Multiscale Persistent Functions for Biomolecular Structure Characterization. *Bull. Math. Biol.* 80, 1–31. doi:10.1007/s11538-017-0362-6

Zhang, X., Wu, P., Yuan, C., Wang, Y., Metaxas, D., and Chen, C. (2019). Heuristic Search for Homology Localization Problem and its Application in Cardiac Trabeculae Reconstruction. *Proceedings of the Twenty-Eighth1135International Joint Conference on Artificial Intelligence, IJCAI-19*, 1312–1318. doi:10.24963/ijcai.2019/182

Zomorodian, A., and Carlsson, G. (2005). Computing Persistent Homology. *Discrete Comput. Geometry* 33, 249–274. doi:10.1007/s00454-004-1146-y

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

frontiers | Frontiers in Big Data

ORIGINAL RESEARCH
published: 26 April 2022
doi: 10.3389/fdata.2022.668356

Contagion Dynamics for Manifold Learning

*Barbara I. Mahler**

Mathematical Institute, University of Oxford, Oxford, United Kingdom

Contagion maps exploit activation times in threshold contagions to assign vectors in high-dimensional Euclidean space to the nodes of a network. A point cloud that is the image of a contagion map reflects both the structure underlying the network and the spreading behavior of the contagion on it. Intuitively, such a point cloud exhibits features of the network's underlying structure if the contagion spreads along that structure, an observation which suggests contagion maps as a viable manifold-learning technique. We test contagion maps and variants thereof as a manifold-learning tool on a number of different synthetic and real-world data sets, and we compare their performance to that of Isomap, one of the most well-known manifold-learning algorithms. We find that, under certain conditions, contagion maps are able to reliably detect underlying manifold structure in noisy data, while Isomap fails due to noise-induced error. This consolidates contagion maps as a technique for manifold learning. We also demonstrate that processing distance estimates between data points before performing methods to determine geometry, topology and dimensionality of a data set leads to clearer results for both Isomap and contagion maps.

Keywords: dimensionality reduction, manifold learning, topological data analysis, persistent homology, contagion

OPEN ACCESS

Edited by:
Umberto Lupo,
Swiss Federal Institute of Technology Lausanne, Switzerland

Reviewed by:
Dane Taylor,
University at Buffalo, United States
Bengier Ulgen Kilic,
University at Buffalo, United States, in collaboration with reviewer DT
Miroslav Kramar,
University of Oklahoma, United States

***Correspondence:**
Barbara I. Mahler
mahler@maths.ox.ac.uk

Specialty section:
This article was submitted to Machine Learning and Artificial Intelligence, a section of the journal Frontiers in Big Data

Received: *16 February 2021*
Accepted: *02 February 2022*
Published: *26 April 2022*

Citation:
Mahler BI (2022) Contagion Dynamics for Manifold Learning. Front. Big Data 5:668356. doi: 10.3389/fdata.2022.668356

1. INTRODUCTION

Manifold-learning techniques aim to identify low-dimensional manifold structure in high-dimensional data (Lee and Verleysen, 2007). High-dimensional point-cloud data may represent a large number of features on a collection of objects. Some of these features may be redundant or irrelevant, thus giving the data lower-dimensional intrinsic structure. Alternatively, high-dimensional point-cloud data with low-dimensional intrinsic structure may arise as a sample of points from a low-dimensional manifold that is embedded in a high-dimensional space.

Consider, for instance, data points that lie on a plane in three-dimensional space. Principal component analysis (PCA), a classical dimensionality-reduction technique (Sorzano et al., 2014), can find the directions along which the data has maximum variance as well as the relative importance of these directions. In the case of the plane embedded in three-dimensional space, PCA returns three vectors: two of positive weight spanning the plane and one vector of zero weight that is orthogonal to the plane. PCA can thus identify the plane underlying the ostensibly three-dimensional data. More generally, consider data points that are concentrated around a low-dimensional manifold (reflecting the underlying information) that is embedded in a high-dimensional space. PCA is a *linear* dimensionality-reduction method: If the manifold is non-linear, PCA is unable to detect the low-dimensional space underlying the data set. This is where manifold-learning techniques (as a type of *non-linear* dimensionality reduction) can be effective. The purpose of manifold learning is to uncover low-dimensional manifold structure of a data set in a high-dimensional feature space, even if the structure of the data is curved.

A common procedure for non-linear dimensionality reduction is the following:

1. Create a network on the data points, such as by defining an edge between any two nodes within some distance ϵ (producing the *ϵ-neighborhood graph*) or by connecting each point to its k closest neighbors (producing the *k-nearest-neighbor graph*).
2. Define some notion of distance between data points based on this network [e.g., shortest-path length for Isomap (Tenenbaum et al., 2000)]. The aim is to approximate the actual geodesic distance on the underlying manifold.
3. Map points to some space based on the pairwise distances. Possible ways of doing this include the following: (a) use a multidimensional scaling algorithm, which finds an embedding that preserves pairwise distances as well as possible; or (b) take distances to be coordinates in a space of dimension equal to the number of data points [the approach that contagion maps take as a manifold-learning technique (Taylor et al., 2015)].

Many well-established manifold-learning techniques perform poorly when faced with noisy data. Isomap, for instance, can be very sensitive to noise. Consider, for example, a noisy point sample on the Swiss roll (see e.g., **Figure 2A**). Noise can lead to two points on adjacent sheets lying close together. The k-nearest-neighbor graph might then have an edge that connects the corresponding nodes (i.e., a "short-circuit error"), although the points lie far apart in the intrinsic geometry. Consequently, Isomap falsely considers the two points to be close, and it thus fails as a manifold-learning technique in this case.

Contagion maps (Taylor et al., 2015) can circumvent Isomap's "short-circuit error" issue by exploiting the "social reinforcement" phenomenon that characterizes threshold contagions. When the threshold of a contagion is small enough to allow spreading *via* a single edge, the associated contagion map can be viewed as a variant of Isomap and is similarly sensitive to the type of noise described above. For larger thresholds, however, a single errant edge in the k-nearest-neighbor graph cannot carry a contagion, and, as a result, the contagion map does not view the two points as close and performs well as a manifold-learning technique. When a contagion on a network spreads as a wavefront exclusively *via* edges between nodes that are close together in the intrinsic geometry of the underlying domain, we call this *wavefront propagation* (WFP). When it spreads *via* edges that connect nodes that are far apart from each other in the underlying domain, thereby creating new contagion clusters in regions of the network that are far from the previously infected regions, we call this *appearance of new clusters* (ANC). If a contagion spreads predominantly *via* WFP, then, intuitively, the point cloud that is the image of the corresponding contagion map exhibits features of the network's underlying structure, and this has been confirmed for particular classes of networks in Taylor et al. (2015) and Mahler (2021). This recovery of underlying structure under certain spreading dynamics suggests contagion maps as a manifold-learning technique.

We use persistent homology, a method from topological data analysis (Edelsbrunner and Harer, 2008), as well as more established statistical techniques to perform manifold learning based on contagion dynamics, and we compare this approach to Isomap-type algorithms. One of the most common applications of persistent homology is the task of recovering a manifold from a random, potentially noisy sample of points (Carlsson, 2009). This application illuminates the natural overlap of topological data analysis with manifold learning. Both are designed to find shape regardless of exact geometry (including measures of curvature and length), and both aim to be robust to noise. Traditionally, they differ in their respective approaches and, as a result, in their ability to identify different types of structural features. Persistent homology can, for example, identify a sphere (by its topological features in dimensions 1 and 2), but not a Swiss roll (as it has no non-trivial topological features). A manifold-learning algorithm like Isomap, on the other hand, can "unroll" the Swiss roll (under favorable conditions), but cannot see that a sphere is a two-dimensional manifold: Isomap detects a sphere's lowest embedding dimension 3, but cannot see its intrinsic 2-dimensional structure. In this article, we combine the two approaches by processing our data *via* a manifold-learning-type procedure first and then computing persistent homology (along with some other measures) based on this processed data.

The manifold-learning-type procedure that we use is based either on activation times in a threshold contagion or on shortest-path distances between nodes in a network built on the given data. In both cases, we compare two different approaches: We either pass the distance estimates directly into a pipeline for analyzing dimensionality, topology, and geometry, or we first process them to create points in high-dimensional space, whose pairwise distances we then pass into the same pipeline. We not only compare the contagion-based approach to the Isomap-type one, but also examine the effect of this pre-processing step on our results. We find that the contagion-based approach proves successful in many cases where the Isomap-type approach breaks down and that the pre-processing step leads to clearer results in general.

2. MATERIALS AND METHODS

The fundamental hypothesis that our algorithms are built on is that our data come as samples from some underlying submanifold of $\mathbb{R}^n$, which we want to infer. To this end, we perform variants of a procedure whose basic steps are as follows.

First, if a data set is given in the form of a point cloud, we start by constructing a *neighborhood graph* (V, E) based on this point cloud. We do so by associating a set V of nodes to the data points (denoting by i the node associated to point p_i) and defining the edge set E to build either (1) a *k-nearest-neighbor graph* by connecting each point to its k closest neighbors, or (2) an *ϵ-neighborhood graph* by connecting any two points that are within ϵ from one another:

1. Given some $k \in \mathbb{N}$, we have $(i,j) \in E$ if and only if p_i is in the set of the k nearest neighbors of p_j or vice versa.
2. Given some $\epsilon \in \mathbb{R}_{>0}$, we have $(i,j) \in E$ if and only if $d_2(p_i, p_j) \leq \epsilon$.

In both types of neighborhood graph, one can either weight the edges by the corresponding pairwise distances or treat the edges as unweighted[1].

If we are given data in the form of a network, we use this given (weighted or unweighted) network instead.

Second, we calculate a notion of distance between points that is aimed to be an estimate for the actual geodesic distance on the underlying manifold. The idea is that only the pairwise Euclidean distances between neighboring pairs of points approximate the geodesic distances sufficiently, and that estimates for geodesic distances between non-neighboring pairs of points can be inferred from the distances between neighboring points by "tracing" through the neighborhood graph.

The precise pairwise distances between adjacent points in a neighborhood graph provide information that is relevant to estimating the geodesic distances between non-neighboring points. Unweighted neighborhood graphs forget this information and thus tend to lead to less accurate approximations to the geodesic distances. The loss of information from taking an unweighted graph can be greater for k-nearest-neighbor graphs than for ϵ-neighborhood graphs, because, in the latter case, the pairwise distances are within a range that is capped by ϵ.

We examine dimensionality, topology, and geometry as follows. We perform classical multidimensional scaling (MDS) (Torgerson, 1958) based either on the dissimilarity measure (i.e., activation times in the case of contagion-based algorithms, and shortest-path lengths in the case of Isomap-type algorithms) directly or on the Euclidean distances between points in high-dimensional space whose coordinates are the distance estimates. MDS aims to embed a point cloud in a given low-dimensional vector space in a way that preserves given distances between pairs of points as well as possible. It does so by minimizing a cost function called "strain": Given a matrix $D = \left(d_{ij}\right)_{i,j\in I}$ of pairwise distances, or "dissimilarities" (not necessarily satisfying the defining properties of a metric), and some Euclidean target space Y, MDS finds coordinate vectors $\left\{y_i \in Y\right\}_{i\in I}$ that minimize the cost function

$$E = \|\tau(D) - \tau(D_Y)\|_{L^2},$$

where $D_Y = \left(\|y_i - y_j\|_2\right)_{i,j\in I}$, $\|M\|_{L^2} = \sqrt{\sum_{i,j} M_{ij}^2}$, and $\tau(M) = -HSH/2$ with $S_{ij} = M_{ij}^2$ and $H_{ij} = \delta_{ij} - 1/|I|$ (Mardia et al., 1979).

MDS is a linear dimensionality-reduction technique when applied to Euclidean distances. By applying it to sensible approximations to the geodesic distances between data points, we hope to recover potentially curved structure. In other words, we hope to use MDS to achieve non-linear dimensionality reduction.

Given an embedding *via* MDS to Euclidean space of dimension p, we calculate its *residual variance* (Cox and Cox, 2010),

$$R_p = 1 - \left(\rho^{(p)}\right)^2,$$

where $\rho^{(p)}$ is the Pearson correlation coefficient (Pearson, 1895) between the given pairwise distances $\left\{d_{ij}\right\}_{i,j\in I}$ and the corresponding pairwise Euclidean distances $\left\{\|y_i - y_j\|_2\right\}_{i,j\in I}$ between points in the embedding.

We determine the *approximate embedding dimension* P of the data (according to the given pairwise distances) by finding the smallest dimension such that the residual variance of the embedding *via* MDS to that dimension is less than 5%, that is,

$$P = \min\{p \mid R_p < 0.05\}.^2$$

In addition to these dimensionality considerations *via* MDS, we analyze our data topologically by computing the persistent homology of the Vietoris–Rips filtration (Ghrist, 2008) based either on the dissimilarity measure directly or on the Euclidean distances between points in the associated high-dimensional point cloud. When a base-geometry is given, we also examine our data geometrically through a Pearson correlation coefficient between that base-geometry and the given dissimilarity measure. Note that such a known base-geometry to compare our processed data to is not usually given in manifold-learning applications. The measure is, however, useful when testing the algorithm on benchmark data.

When analyze the point cloud given by the columns of D, we are essentially applying our methods to a dissimilarity matrix that encodes the pairwise distances between the column vectors of D. We denote the operator that maps D to that matrix as

$$p_{\text{dist}} : \mathcal{M}_{m,n}(\mathbb{R}) \longrightarrow \mathcal{M}_{n,n}(\mathbb{R}),$$

with $\left(p_{\text{dist}}(D)\right)_{ij} = d_2(D_{*i}, D_{*j})$.

The methods for analyzing dimensionality, topology, and geometry described above were first used in tandem in Taylor et al. (2015) on contagion maps. That is, they were used after applying p_{dist} to a matrix holding the activation times in multiple realizations of a threshold contagion.

2.1. Contagion Maps

First, given point-cloud data, we obtain a neighborhood graph as described above. If the given data is a network, we work with this network directly instead. We denote the graph by (V, E), the number of nodes (i.e., the number of points in the case of point-cloud data) by $|V| = N$, and the graph's binary adjacency matrix by A. In order to get an estimate for the intrinsic distance between pairs of points, we then consider a threshold contagion on this network. We denote the state of node $i \in V$ at time t by $\eta_i(t)$, which takes the value 1 if it is *active* and the value 0 if it is *inactive*.

[1]Note that, as the "weights" in this graph are really distances, a small weight of an edge indicates a strong connection between its incident nodes.

[2]In practice, we put a cap of 100 on P, so if the approximate embedding dimension is 100 or larger, we record it as 100.

Given a set of seed nodes consisting of a node $j \in V$ together with its immediate neighbors

$$S^{(j)} = \{j\} \cup \{k \mid A_{jk} \neq 0\}$$

that are active at time $t = 0$, and a threshold T, we update node states synchronously in discrete time steps according to the following rule. If $\eta_i(t) = 1$, then $\eta_i(t+1) = 1$. If $\eta_i(t) = 0$, then

$$\eta_i(t+1) = 1 \quad \text{if and only if} \quad f_i > T,$$
$$\text{where} \quad f_i = \frac{1}{d}\sum_{k \in V} A_{ik}\eta_k(t), \quad \text{and} \quad d \ \text{is the node degree.}$$

Given a network (V, E) and a threshold T, a contagion seed yields a deterministic process, which we call a *realization* of the contagion model with T on (V, E). The activation time of node i in the realization seeded around node j is the smallest t such that $\eta_i(t) = 1$, and we denote it by $x_j^{(i)}$. If node i is never activated in the realization that is seeded around node $j \in V$, we set $x_j^{(i)} = 2N$ (i.e., larger than any actual activation time).

One can now work directly with this set of activation times, that is, treat the activation times as estimates for the geodesic distance between points on the underlying manifold, and use them to examine geometry, topology, and dimensionality of the data. Alternatively, one can first apply p_{dist}. That is, one can work with points whose coordinate vectors are given by the columns of the dissimilarity matrix $D_{\text{cont}} = \left(x_i^{(j)}\right)_{i,j \in V}$ that holds the activation times (or of a symmetrization of this dissimilarity matrix given by $D_{\text{cont}} + (D_{\text{cont}})^{\text{T}}$). Using terminology from Taylor et al. (2015), producing such a point cloud is equivalent to mapping the nodes *via* a *contagion map*. The *regular contagion map* associated to (V, E) and T is the function from the set V of nodes to $\mathbb{R}^N$ that is defined by

$$i \mapsto x^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_N^{(i)}]^T.$$

Similarly, the *symmetric contagion map* associated to (V, E) and T is the function from the set V of nodes to $\mathbb{R}^N$ that is defined by

$$i \mapsto [x_1^{(i)} + x_i^{(1)}, \ldots, x_N^{(i)} + x_i^{(N)}]^T.$$

In this article, we work with symmetric contagion maps exclusively.

We examine dimensionality, topology, and geometry based on the activation times directly, that is, based on the entries of the matrix $D_{\text{cont}} + (D_{\text{cont}})^{\text{T}}$, as well as after applying p_{dist} (i.e., we analyze the symmetric contagion map).

2.2. Isomap

The Isomap algorithm (Tenenbaum et al., 2000) is essentially a combination of a shortest-path algorithm with MDS. In a sense, Isomap works as a "non-linear version" of MDS which accommodates for potential curvature of data by incorporating a shortest-paths algorithm to estimate geodesic distances between data points.

The original Isomap algorithm proceeds as follows. First, given point-cloud data, Isomap starts by building a neighborhood graph (V, E) on the point cloud, as described at the beginning of section 2. Next, Isomap calculates the shortest-path lengths between pairs of nodes in this network using some shortest-path algorithm. We use the Floyd–Warshall algorithm (Floyd, 1962; Warshall, 1962) in this work. The set of shortest-path lengths can be recorded in a *dissimilarity matrix* $D_{\text{iso}} = \left(d_{\text{G}}(i,j)\right)_{i,j \in V}$[3], to which Isomap finally applies MDS to map the data points to a low-dimensional space.

As in the case of contagion maps, if data is given in the form of a network, we will work with this given network instead of a neighborhood graph. Moreover, in addition to the original Isomap algorithm, which simply projects points *via* MDS based on the set of shortest-path lengths (i.e., the entries in D_{iso}), we calculate the residual variances of these projections, and we also examine this set topologically (*via* the persistent homology of the Vietoris–Rips filtration based on these shortest-path lengths) and geometrically [*via* a Pearson correlation (Pearson, 1895) with some given base-geometry], when possible. Furthermore, we analyze the point cloud given by the columns (or, equivalently, rows) of D_{iso}, that is, we analyze the entries in $p_{\text{dist}}(D_{\text{iso}})$.

Note that a contagion map with threshold $T = 0$ is approximately equivalent to a version of Isomap that uses an unweighted neighborhood graph.

2.3. Workflow

Our workflow is composed of multiple stages, at each of which one can choose from a number of different options. This leads to exponentially many possible procedures that one can use to analyze a given data set. First, given point-cloud data, one needs to choose the type of neighborhood graph to build on this data set as well as the defining parameter k or ϵ. Next, one needs to pick a way of estimating geodesic distances based on this graph. We choose either shortest-path distances or activation times in a threshold contagion, that is, we follow either the Isomap algorithm or that of contagion maps. In the case of contagion maps, one also needs to choose a threshold parameter T. Given the set of estimates for the geodesic distances, i.e., the dissimilarity matrix that encodes the shortest-path (D_{iso}) distances or activation times (D_{cont}), one can apply further methods either to these estimates directly or to the pairwise distances between the points whose coordinate vectors are the columns (or rows, by symmetry) of this dissimilarity matrix. For either of these choices one can finally apply methods to determine dimensionality, topology, and geometry. **Figure 1** shows a schematic representation of our workflow. We apply different

[3]The length of a shortest path between nodes i and j is denoted by $d_{\text{G}}(i,j)$. Note that the function $d_{\text{G}} : V \times V \to [0, \infty)$ is a metric on the set of nodes or (equivalently) on the set of data points.

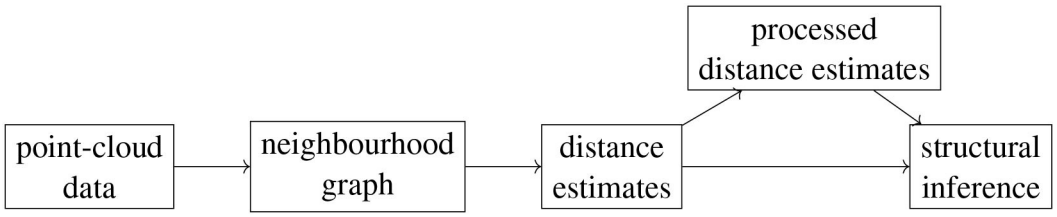


FIGURE 1 | Schematic representation of the workflow.

subsets of the full analysis to the different data sets that we study.

3. RESULTS

We apply Isomap, as well as contagion maps with several different thresholds, to three different data sets. First, we consider point samples from a Swiss roll, a classical benchmark data set for dimensionality reduction and one of the first data sets to which Isomap was applied (Tenenbaum et al., 2000). We then analyze a couple of representatives of the class of torus-based networks that was studied in Mahler (2021). The toroidal structure underlying these networks allows us to examine the topological aspect of our methods by looking to recover the torus' non-trivial topological features. Finally, we consider a data set that represents the conformation space of the cyclo-octane molecule (Martin et al., 2010). This data set is known to have non-trivial topological features and is an example of a naturally occurring data set.

3.1. Noisy Samples From a Swiss Roll

We first examine point samples from a Swiss roll that are obtained by taking regularly spaced points on the Swiss roll surface and then adding various levels of noise to these points. This way of generating data makes it possible to have direct control over the data, so one can explore how different algorithms react to slight variations of the data (in terms of e.g., density, noise level, or uniformity). We use this data set primarily to explore the effects of the parameter k or ϵ when building a neighborhood graph.

We start by taking points on a Swiss roll at a density of approximately 50 per unit area, regularly spaced with respect to the intrinsic geodesic distance on the Swiss roll (see **Figure 2A**). We then add *Gaussian noise* with a specified signal-to-noise ratio (S/N) to these regularly spaced points (see **Figure 2B**). That is, for each point $p = [p_1, p_2, p_3]$ in this regularly spaced point sample, we add independent, identically distributed noise drawn from a zero-mean normal distribution to each of its coordinates to obtain a perturbation p_{noisy} of the point:

$$p_{\text{noisy}} = [p_1 + n_1, p_2 + n_2, p_3 + n_3],$$

where $n_i \sim \mathcal{N}\left(0, \sigma^2\right)$ with $\sigma^2 = 10^{\frac{-S/N}{10}}$.

We test Isomap and contagion maps on this noisy point cloud to see how well each of them sees the underlying 2-dimensional space. Each algorithm starts by building a neighborhood graph on the points (see **Figures 2C,D**).

We have already touched on Isomap's sensitivity to "short-circuit errors" in the introduction. However, a careful choice of ϵ (or k) when constructing the neighborhood graph can mitigate such errors to some extent (Balasubramanian et al., 2002). The goal is to find an ϵ (or k) that is small enough to avoid short-circuit edges but not so small that the resulting graph "corrupts" the underlying space. One may choose to simply vary ϵ over a range and implement Isomap on all of the neighborhood graphs that thus arise. This approach is in the same vein as considering the full range of thresholds for the contagion map algorithm, and it works whenever there exists a range of ϵ (or k) for which the resulting neighborhood graphs correctly represent the underlying topology. However, for some data sets—particularly those that are sparse and incorporate a high level of noise—it is impossible to find a value for ϵ (or k) that strikes a balance between covering the underlying topology and not making "short-circuit errors". In other words, the range of ϵ (or k) that "corrupt" the underlying topology and the range of ϵ (or k) that make "short-circuit errors" overlap, leaving no values of ϵ (or k) that yield neighborhood graphs that correctly represent the underlying topology. For such data sets, Isomap is inadequate as a manifold-learning tool, but contagion maps may be effective.

See **Figures 3**, **4** for examples on the Swiss roll that illustrate the concepts in the above paragraph. In particular, **Figure 3** shows an example of a data set for which a careful choice of ϵ generates a neighborhood graph that both captures the underlying manifold and does not include "short-circuit" edges. By contrast, **Figure 4** shows an example of a data set for which no choice of ϵ produces a neighborhood graph that accurately represents the underlying manifold. **Figure 5** shows the residual variances of projecting this data set *via* MDS to dimensions 1 to 10 when based on Isomap and when based on the contagion map with $T = 0.2$ (both starting with a 0.18-neighborhood graph). For the contagion map, the residual variance plunges at dimension 2, thereby correctly identifying the intrinsic dimension of the data. The residual variances for Isomap, on the other hand, only decrease slighty and continuously across the increasing target dimensions. That is, Isomap fails to see the correct intrinsic structure of the data when starting with an ϵ-neighborhood graph with $\epsilon = 0.18$ (or any other value of ϵ), while contagion map — with a suitable choice of ϵ and T — correctly identifies the underlying structure. For contagion maps to work in this case, the value of ϵ had to be chosen large enough for the neighborhood

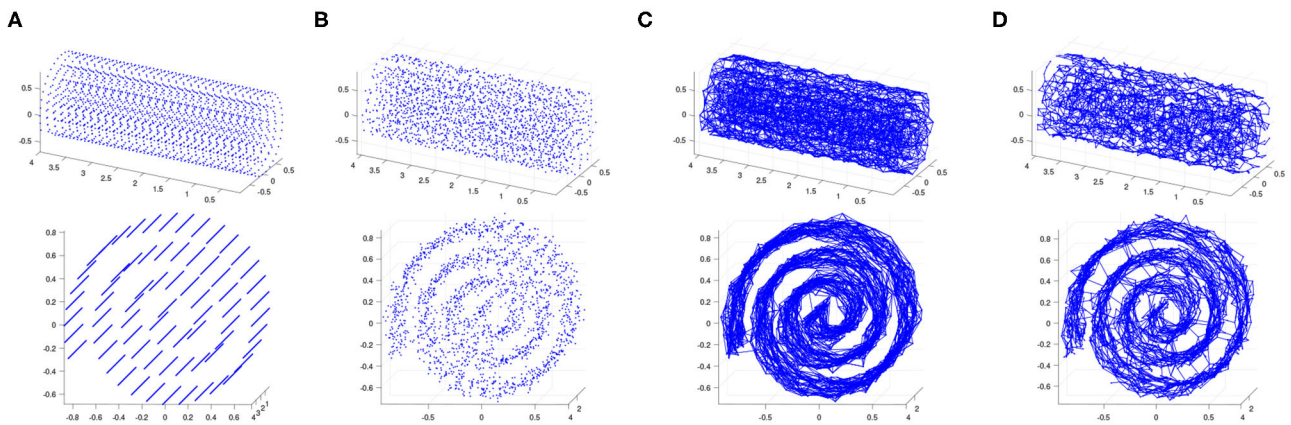


FIGURE 2 | (A) Regularly spaced points on a Swiss roll at density 50 per unit area. **(B)** Gaussian noise is added at a signal-to-noise ratio of 30. **(C)** The 5-nearest-neighbor graph on the noisy point cloud. **(D)** The 0.18-neighborhood graph on the noisy point cloud. We show two views of each plot.

graph to cover the underlying manifold, and the value of T had to be picked small enough to carry the contagion and large enough to be resistant to the unavoidable noisy inter-sheet edges in the ϵ-neighborhood graph.

We now consider 2,000 points of the Swiss roll data set[4] from Tenenbaum et al. (2000). This data set consists of 20,000 points in total.

Both Isomap and contagion maps perform adequately for the high signal-to-noise ration of $S/N = 20$ (i.e., low noise level) with all four examined versions of neighborhood graphs (see **Figure 6**).

For the lowest signal-to-noise ratio (i.e., greatest noise level) that we consider (namely $S/N = 5$) the 8-nearest-neighbor graph includes noisy inter-sheet edges. As a result, Isomap does not detect the intrinsic dimension 2 of the Swiss roll when using the 8-nearest neighbor graph, and neither does the contagion map with $T = 0$ or $T = 0.1$, thresholds for which the activation times in our threshold contagion are close to the shortest paths in the unweighted neighborhood graph (see **Figure 7**). With $T = 0.2$, however, the contagion map does correctly recover the intrinsic dimension 2, as this threshold is just large enough to be robust to the occurring noisy edges. For thresholds larger than $T = 0.2$, many of the realizations of our threshold contagion leave nodes in the neighborhood graph inactive (recorded as "infinite" activation times), and, as a result, the residual variances of the embeddings based on these activation times are large for all considered dimensions (1 to 10). This illustrates that, while contagion maps can be a powerful tool when dealing with such noisy edges, a suitable choice of threshold can be a delicate matter.

Similarly, for signal-to-noise ratio $S/N = 5$, Isomap fails when based on the 4.5-neighborhood graph or the 5-neighborhood graph, but the contagion map for a threshold of $T = 0.2$ successfully identifies the intrinsic dimension 2 for both examined values of the neighborhood parameter ϵ. Furthermore, when based on a 5-neighborhood graph, Isomap fails even for the higher $S/N = 10$, as the 5-neighborhood graph includes noisy inter-sheet edges even for this lower noise level (see **Figure 8**).

Note that our measures for topology and geometry are not useful for this data set, as the underlying manifold has no non-trivial topological features, and there is no base-geometry provided.

Our experiments on this classical manifold-learning data showcase examples where contagion maps have the power to detect low-dimensional structure, even when Isomap is unsuccessful. Crucial to contagion maps' success is the careful choice of suitable neighborhood graph parameter and contagion threshold.

3.2. Torus-Based Networks

We consider the torus-based model described in Mahler (2021) with $N = 2{,}500$ nodes and a degree of *geometric edges* of $d^{\mathrm{G}} = 8$. Networks in this model are similar to Kleinberg's small-world like network (Kleinberg, 2000). They consist of a periodic grid of nodes that are connected *via geometric edges* (i.e., edges between neighboring nodes) in a regular manner, and to which *non-geometric edges* are added according to a probability distribution. We add first 2 and then 4 non-geometric edges per node uniformly at random and apply versions of both Isomap and contagion maps (with thresholds $T = 0, 0.1, \ldots, 1$) to the resulting networks. Namely, we calculate the shortest-path lengths between pairs of nodes in these two unweighted networks, as well as the activation times in all realizations of our threshold contagion that are seeded at the direct neighborhoods of individual nodes. We thus obtain two dissimilarity matrices: one holding the shortest-path lengths (D_{iso}) and one holding the symmetrized activation times (D_{cont}). We analyze the information held in these two dissimilarity matrices geometrically, topologically, and in terms of dimensionality in two ways each. We first analyze the estimated pairwise geodesic distances held in each dissimilarity matrix directly, and we then consider the point clouds that results from taking columns (or, equivalently, rows) of each

[4]https://web.mit.edu/cocosci/isomap/datasets.html

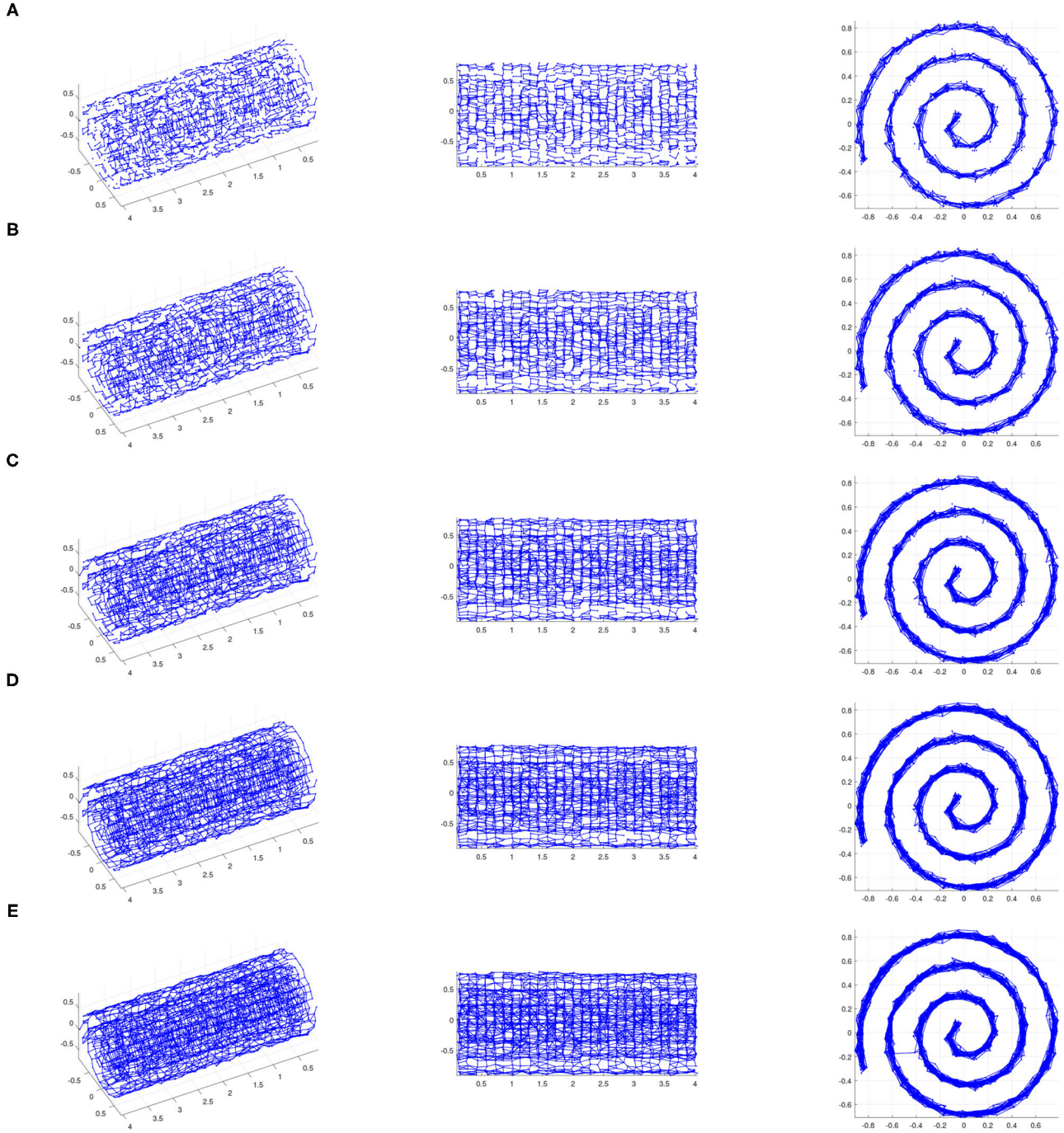


FIGURE 3 | The ϵ-neighborhood graphs on a noisy point sample from a Swiss roll with $S/N = 35$ for **(A)** $\epsilon = 0.14$, **(B)** $\epsilon = 0.15$, **(C)** $\epsilon = 0.16$, **(D)** $\epsilon = 0.17$, and **(E)** $\epsilon = 0.18$ (We show three views of each plot.). If ϵ is too small (e.g., $\epsilon = 0.14$), the ϵ-neighborhood graph does not adequately "cover" the underlying Swiss roll. If ϵ is too large (e.g., $\epsilon = 0.18$), the ϵ-neighborhood graph includes inter-sheet edges, and thus does not represent the underlying Swiss roll. However, there exists a range of ϵ for which the ϵ-neighborhood graph covers the underlying surface and does not include inter-sheet edges, thus providing an authentic representation of the underlying Swiss roll. In other words, the pairwise distances between points whose corresponding nodes are adjacent in the ϵ-neighborhood graph for such ϵ approximate the actual geodesic sufficiently, and approximate pairwise geodesic distances between other point pairs can be inferred through the Isomap algorithm. This is an example of a data set for which Isomap is suitable as a manifold-learning technique with a careful choice of ϵ.

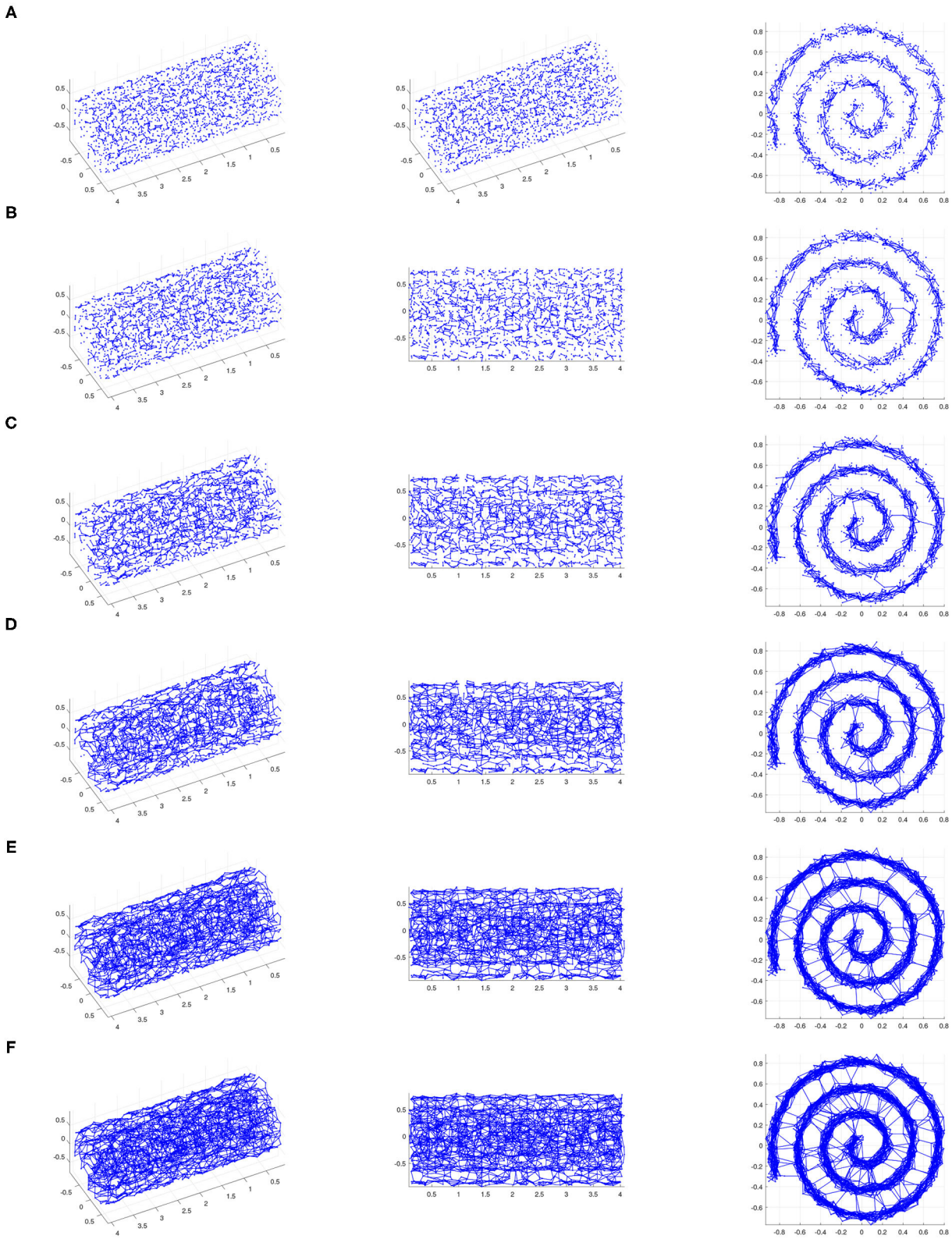


FIGURE 4 | The ϵ-neighborhood graphs on a noisy point sample from a Swiss roll with $S/N = 30$ for **(A)** $\epsilon = 0.11$, **(B)** $\epsilon = 0.12$, **(C)** $\epsilon = 0.14$, **(D)** $\epsilon = 0.16$, **(E)** $\epsilon = 0.18$, and **(F)** $\epsilon = 0.19$ (We show three views of each plot.). For small ϵ, the ϵ-neighborhood graph does not adequately "cover" the underlying Swiss roll. As ϵ increases, noisy inter-sheet edges appear (for e.g., $\epsilon = 0.12$) before ϵ is large enough for the neighborhood graph to adequately "cover" the underlying Swiss roll. This is an example of a data set for which Isomap cannot be used successfully as a manifold-learning technique with any choice of ϵ.

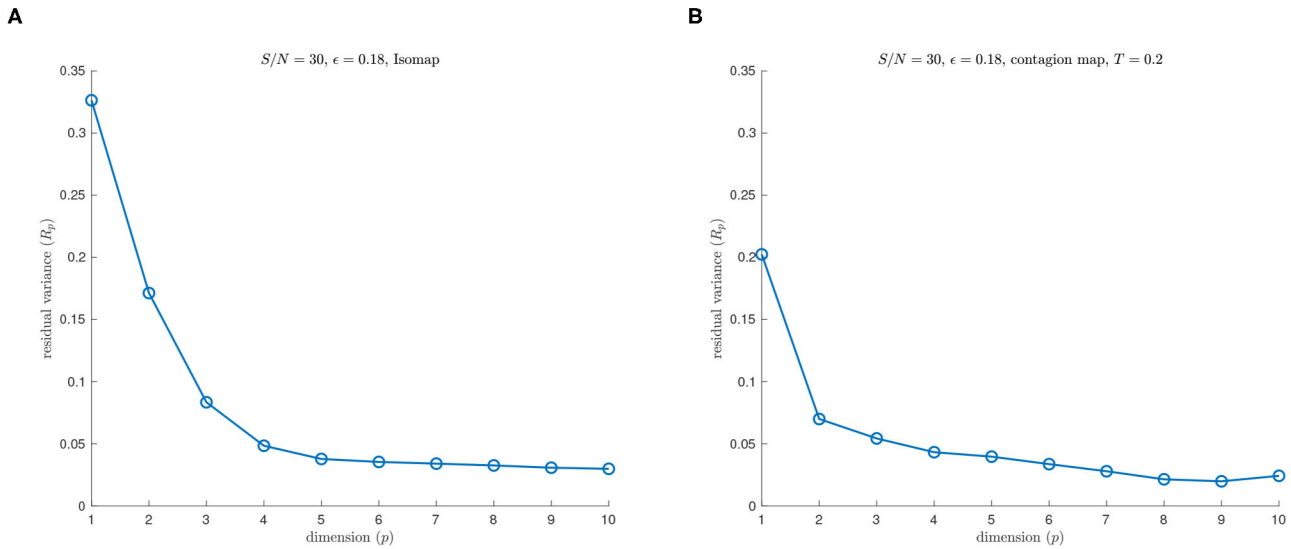


FIGURE 5 | (A) Isomap and **(B)** contagion map with $T = 0.2$ (both starting with a 0.18-neighborhood graph) on a noisy sample from a Swiss roll with $S/N = 30$.

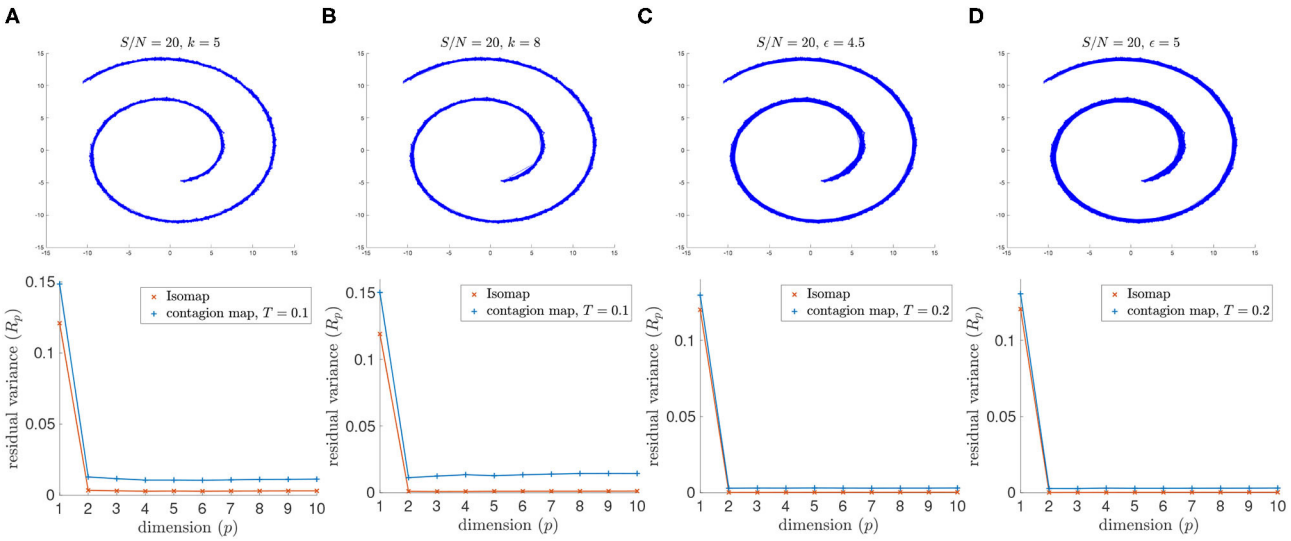


FIGURE 6 | (Top row) Neighborhood graphs on 2,000 points of the Swiss roll data set from Tenenbaum et al. (2000) with white Gaussian noise added with $S/N = 20$. (Bottom row) The residual variances of MDS projections to dimensions 1–10 resulting from the approximate pairwise geodesic distances between nodes based on shortest paths on the weighted graph (i.e., Isomap) and the activation times in a threshold contagion on the unweighted graph (i.e., contagion maps). The neighborhood graphs are **(A)** the 5-nearest-neighbor graph, **(B)** the 8-nearest-neighbor graph, **(C)** the 4.5-neighborhood graph, and **(D)** the 5-neighborhood graph.

dissimilarity matrix as the coordinate vectors of points in $\mathbb{R}^{2500}$.

In detail, we perform the following analyzes:

- In terms of dimensionality: We perform MDS based on the entries in each dissimilarity matrix to dimensions 1–10 and record the residual variance for each dimension. We also perform MDS on the point cloud that results from taking columns (or, equivalently, rows) of each dissimilarity matrix as the coordinate vectors of points in $\mathbb{R}^{2500}$. In both cases, we identify the approximate embedding dimension as the lowest dimension such that the residual variance when projecting down to that dimension *via* MDS is below 5%.
- Topologically: We build Vietoris–Rips filtrations based on the approximations to the pairwise geodesic distances (i.e., the entries in each dissimilarity matrix) and compute their persistent homologies. We also build Vietoris–Rips filtrations on the points cloud that results from taking columns (or, equivalently, rows) of each dissimilarity matrix as the coordinate vectors of points in $\mathbb{R}^{2500}$ and compute their persistent homologies.

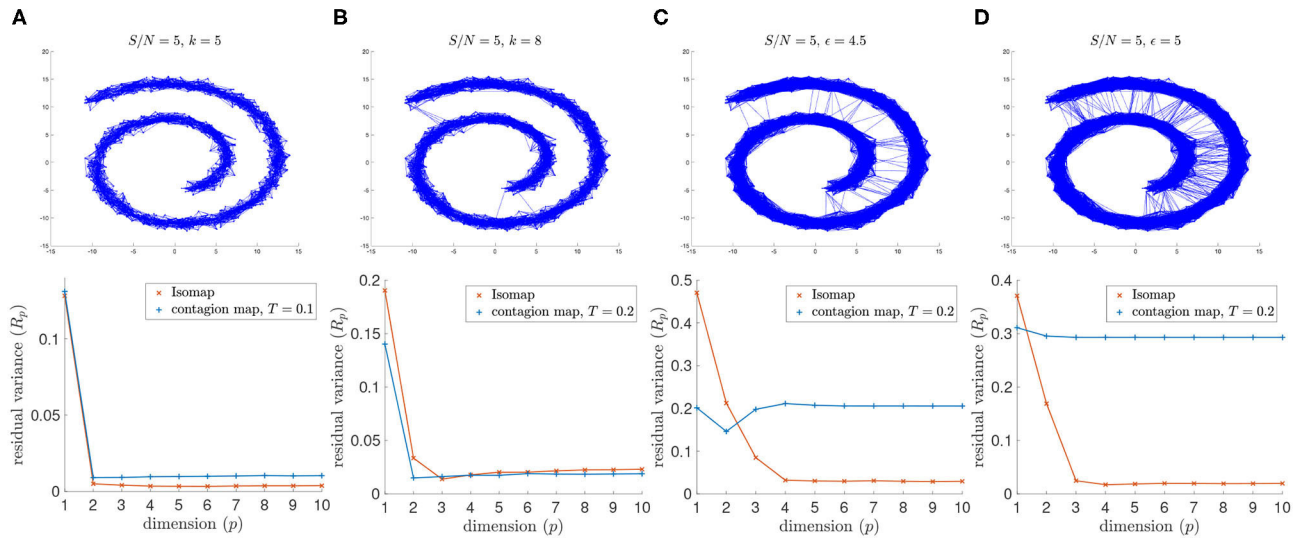


FIGURE 7 | (Top row) Neighborhood graphs on 2,000 points of the Swiss roll data set from Tenenbaum et al. (2000) with white Gaussian noise added with $S/N = 5$. (Bottom row) The residual variances of MDS projections to dimensions 1 to 10 resulting from the approximate pairwise geodesic distances between nodes based on shortest paths on the weighted graph (i.e., Isomap) and the activation times in a threshold contagion on the unweighted graph (i.e., contagion maps). The neighborhood graphs are **(A)** the 5-nearest neighbor graph, **(B)** the 8-nearest neighbor graph, **(C)** the 4.5-neighborhood graph, and **(D)** the 5-neighborhood graph.

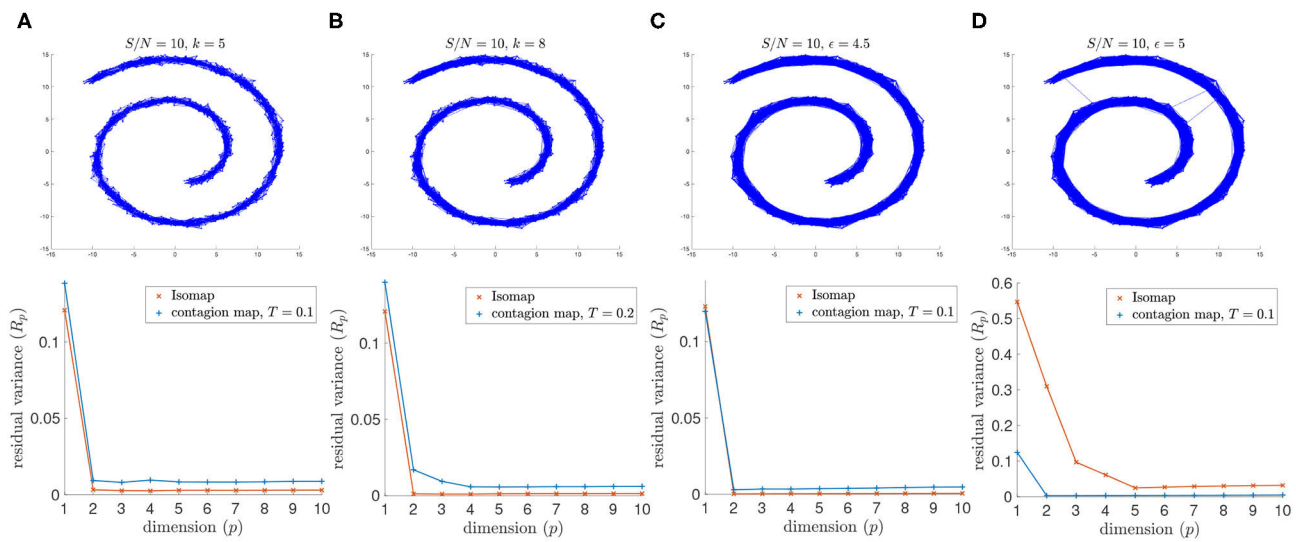


FIGURE 8 | (Top row) Neighborhood graphs on 2,000 points of the Swiss roll data set from Tenenbaum et al. (2000) with white Gaussian noise added with $S/N = 10$. (Bottom row) The residual variances of MDS projections to dimensions 1 to 10 resulting from the approximate pairwise geodesic distances between nodes based on shortest paths on the weighted graph (i.e., Isomap) and the activation times in a threshold contagion on the unweighted graph (i.e., contagion maps). The neighborhood graphs are **(A)** the 5-nearest neighbor graph, **(B)** the 8-nearest neighbor graph, **(C)** the 4.5-neighborhood graph, and **(D)** the 5-neighborhood graph.

- Geometrically: We calculate the Pearson correlation coefficient between the entries in each dissimilarity matrix and the corresponding pairwise distances between regularly spaced points on a torus:

$$\frac{1}{2\pi}\left(\cos\frac{2\pi x}{n}, \sin\frac{2\pi x}{n}, \cos\frac{2\pi y}{n}, \sin\frac{2\pi y}{n}\right) \in \mathbb{T} = \frac{1}{2\pi}\mathbb{S}^1 \times \frac{1}{2\pi}\mathbb{S}^1 \subset \mathbb{R}^4, \quad x, y \in \{0, 1, \ldots, n-1\}. \tag{1}$$

We also calculate the Pearson correlation coefficient between the pairwise distances between points in the point cloud that results from taking columns (or, equivalently, rows) of each dissimilarity matrix as the coordinate vectors of points in $\mathbb{R}^{2500}$ and the corresponding pairwise distances between the regularly spaced points on a torus specified in (1).

We find that Isomap is unable to infer the underlying torus structure from these networks. Contagion maps, however, detect

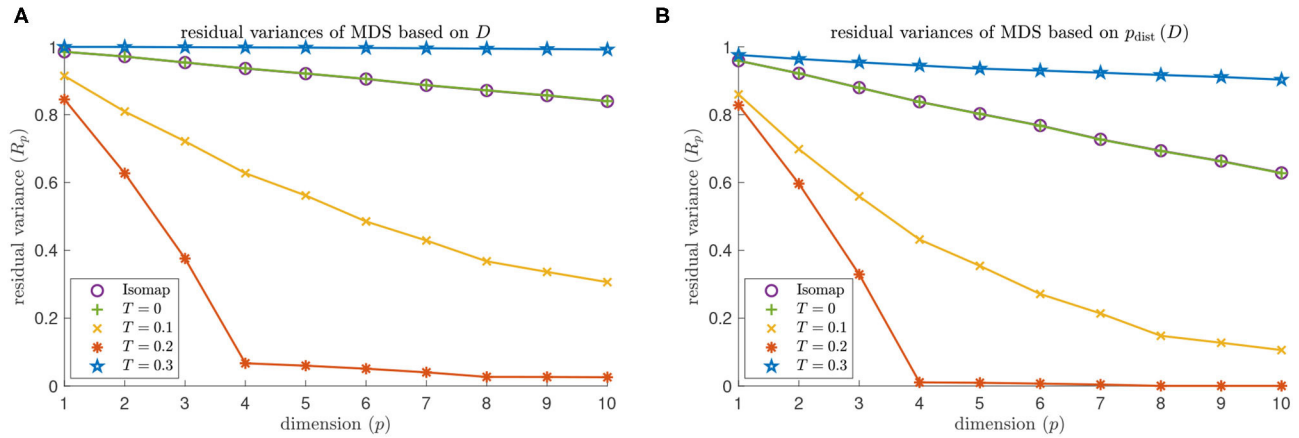


FIGURE 9 | Dimensionality results on our torus-based network with $d^{\text{NG}} = 2$. **(A)** Residual variances of MDS based on the estimated geodesic distances (i.e., the entries in D_{iso} and D_{cont}) according to Isomap, and according to contagion maps with thresholds $T = 0$, $T = 0.1$, $T = 0.2$, and $T = 0.3$. **(B)** Residual variances of MDS based on the point cloud (i.e., the rows of the D_{iso} and D_{cont}) according to Isomap, and according to contagion maps with thresholds $T = 0$, $T = 0.1$, $T = 0.2$, and $T = 0.3$.

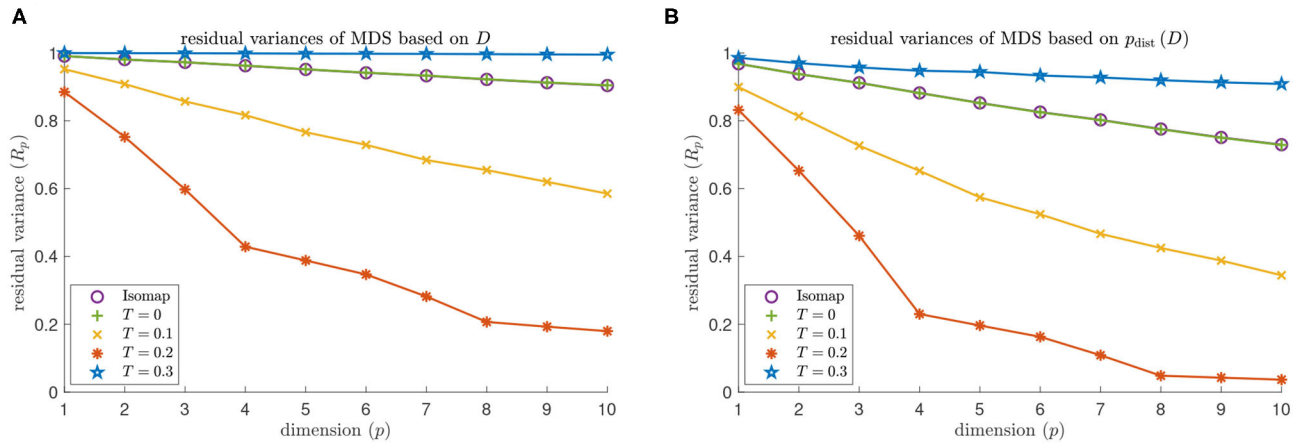


FIGURE 10 | Dimensionality results on our torus-based network with $d^{\text{NG}} = 4$. **(A)** Residual variances of MDS based on the estimated geodesic distances (i.e., the entries in D_{iso} and D_{cont}) according to Isomap, and according to contagion maps with thresholds $T = 0$, $T = 0.1$, $T = 0.2$, and $T = 0.3$. **(B)** Residual variances of MDS based on the point cloud (i.e., the rows of the D_{iso} and D_{cont}) according to Isomap, and according to contagion maps with thresholds $T = 0$, $T = 0.1$, $T = 0.2$, and $T = 0.3$.

the characteristics of the torus when using a threshold of $T \approx 0.2$. The results in this section illustrate the utility of contagion maps for spatial network data that incorporates noisy edges and its potential to outperform Isomap in such scenarios, but they also highlight that a careful choice of T is critical.

Dimensionality

MDS based on Isomap does not identify the embedding dimension 4 of a torus for either $d^{\text{NG}} = 2$ or $d^{\text{NG}} = 4$ (see the purple data points in **Figures 9**, **10**). The residual variance based on contagion maps does have a dip at dimension 4 for the threshold $T = 0.2$ (see the red data points in **Figures 9**, **10**) but does not when the threshold is $T = 0.1$ or $T = 0.3$. Note that for Isomap, as well as contagion maps with all considered thresholds, the residual variances are smaller for all considered dimensions when the analysis is done on the point cloud, making the results for contagion map with $T = 0.2$ look sharper.

We identify the approximate embedding dimensions for Isomap and contagion maps with thresholds $T = 0{,}0.1, \ldots, 1$ and show the results in **Figure 11**. Both versions of Isomap (the one performing MDS based on the entries in D_{iso} and the one performing MDS based on the distances between the rows of D_{iso}) vastly overestimate the embedding dimension for the torus-based network with $d^{\text{NG}} = 2$ and the one with $d^{\text{NG}} = 4$. When performing MDS based on the entries in D_{iso}, the approximate embedding dimension is at least 100 (which is the dimension at which we cap our computations). When

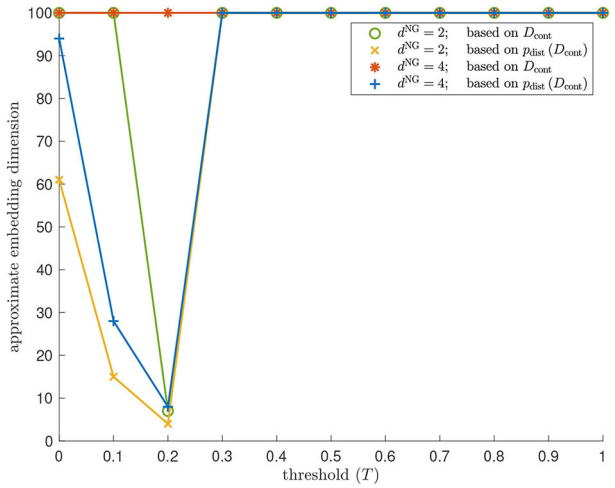


FIGURE 11 | Dimensionality results on our torus-based network with $d^{NG} = 2$ (colored in green and yellow) and $d^{NG} = 4$ (colored in red and blue). Approximate embedding dimension according to contagion maps with different values of threshold T and critical value 5% using the dissimilarity matrix D_{cont} (colored in green and red), and the point cloud whose coordinate vectors are the rows in D_{cont} (colored in yellow and blue). For $d^{NG} = 2$, the results for Isomap are practically identical to those for the contagion map with $T = 0$: The approximate embedding dimension is at least 100 (which is the dimension at which we cap our computations) when based on the entries in D_{iso}, and it is 61 when based on the point cloud whose coordinate vectors are given by the rows of D_{iso}. The approximate embedding dimension according to contagion maps reaches a minimum value for $T = 0.2$ both when working with D_{cont} and when working with $p_{dist}(D_{cont})$. This minimal value is 7 in the former case (see the data points colored in green), and is 4 in the latter case (see the data points colored in yellow). Similarly, for $d^{NG} = 4$, the results for Isomap are practically identical to those for the contagion map with $T = 0$: The approximate embedding dimension is at least 100 (which is the dimension at which we cap out computations) when based on the entries in D_{iso}; and it is 95 when based on the point cloud whose coordinate vectors are given by the rows of D_{iso}. The approximate embedding dimension according to contagion maps reaches a minimum value of 8 for $T = 0.2$ when working with $p_{dist}(D_{cont})$ (see the data points colored in blue).

performing MDS based on the distances between the rows of D_{iso}, the embedding dimensions are still very large: The embedding dimension is 61 for the network with $d^{NG} = 2$, and it is 95 for the network with $d^{NG} = 4$. Note that these results are practically identical to those the for the contagion map with $T = 0$, as we are working with the same unweighted graphs in both Isomap and contagion maps. For contagion maps with varying thresholds T, the approximate embedding dimension has a dip around $T = 0.2$, except when performing MDS based on the entries in D_{cont} for the network with $d^{NG} = 4$, in which case contagion maps return an embedding dimension of at least 100 for all thresholds (see the red data points in **Figure 12**). This suggests that, for thresholds close to 0.2, the contagion spreads predominantly *via* WFP along the underlying torus, making it possible to identify the underlying low-dimensional structure.

Topology

Figures 12, **13** show the barcodes corresponding to the persistent homology in dimension 1 of the Vietoris–Rips filtrations built according to the different versions of Isomap and contagion maps. The barcodes in dimension 1 of the Vietoris–Rips filtrations based on the estimated geodesic distances (i.e., the entries in D_{iso} and D_{cont}) do not seem to reveal any significant features for either Isomap or contagion maps (see **Figures 12A–E**, **13A–E**). The barcode in dimension 1 of the Vietoris–Rips filtration on the point cloud based on Isomap (i.e., given by the rows of D_{iso}) on the network with $d^{NG} = 2$ does feature two dominant bars (see **Figure 12F**), as do the barcodes corresponding to point clouds based on contagion maps for $T = 0$, $T = 0.1$, and $T = 0.2$ (i.e., given by the rows of D_{cont}) (see **Figures 12G–I**). Note, however, that in panels F–H, due to the order of the bars, the dominance appears slightly stronger than it actually is. For the network with $d^{NG} = 4$, the barcode in dimension 1 of the Vietoris–Rips filtration on the point cloud based on Isomap does not have any dominant bars, whereas the one based on the contagion map for $T = 0.2$ does (see **Figure 13I**).

Geometry

We examine the geometry of our Isomap and contagion map results by comparing the entries of D_{iso} and D_{cont}, as well as the point clouds given by the rows of these dissimilarity matrices, to the regularly spaced points on a torus specified in (1) *via* the Pearson correlation coefficient (see **Figure 14**). Isomap returns a low Pearson correlation in all cases, suggesting that the shortest-path distances are (as expected, given the large number of non-geometric edges) not good estimates for the distances along the torus that underlies these networks. Note that these results are practically identical to those for contagion map with $T = 0$, as we are working with the same unweighted graphs in both Isomap and contagion maps. For contagion maps with varying thresholds T, the Pearson correlation coefficient peaks around $T = 0.2$, suggesting that, for thresholds close to $T = 0.2$, the contagion spreads predominantly *via* WFP, making the activation times good estimates for the distance along the torus that underlies these networks.

These synthetic torus-based network data sets allowed us to explore all three structural measures (dimensionality, topology, and geometry), as the data's underlying structure is not only low-dimensional but has non-trivial topological features to recover, and we have a base-geometry to compare our contagion and shortest-path based distance estimates against.

Again, we emphasized the importance of finding a suitable value for the contagion threshold. In the case of these torus-based network data sets, a range of thresholds around $T = 0.2$ produce contagion maps that reveal the underlying toroidal structure. This optimal range for T could be predicted from the bifuracation analysis in Mahler (2021). In a true manifold-learning application, however, where we have no a priori knowledge about the data's underlying structure and how it is sampled from that underlying space, it is generally difficult to guess an appropriate value of T. A practical approach is to simply sweep over a range of incremental values of T and look for dips and peaks in the measures for topology, geometry, and dimensionality. Such dips and

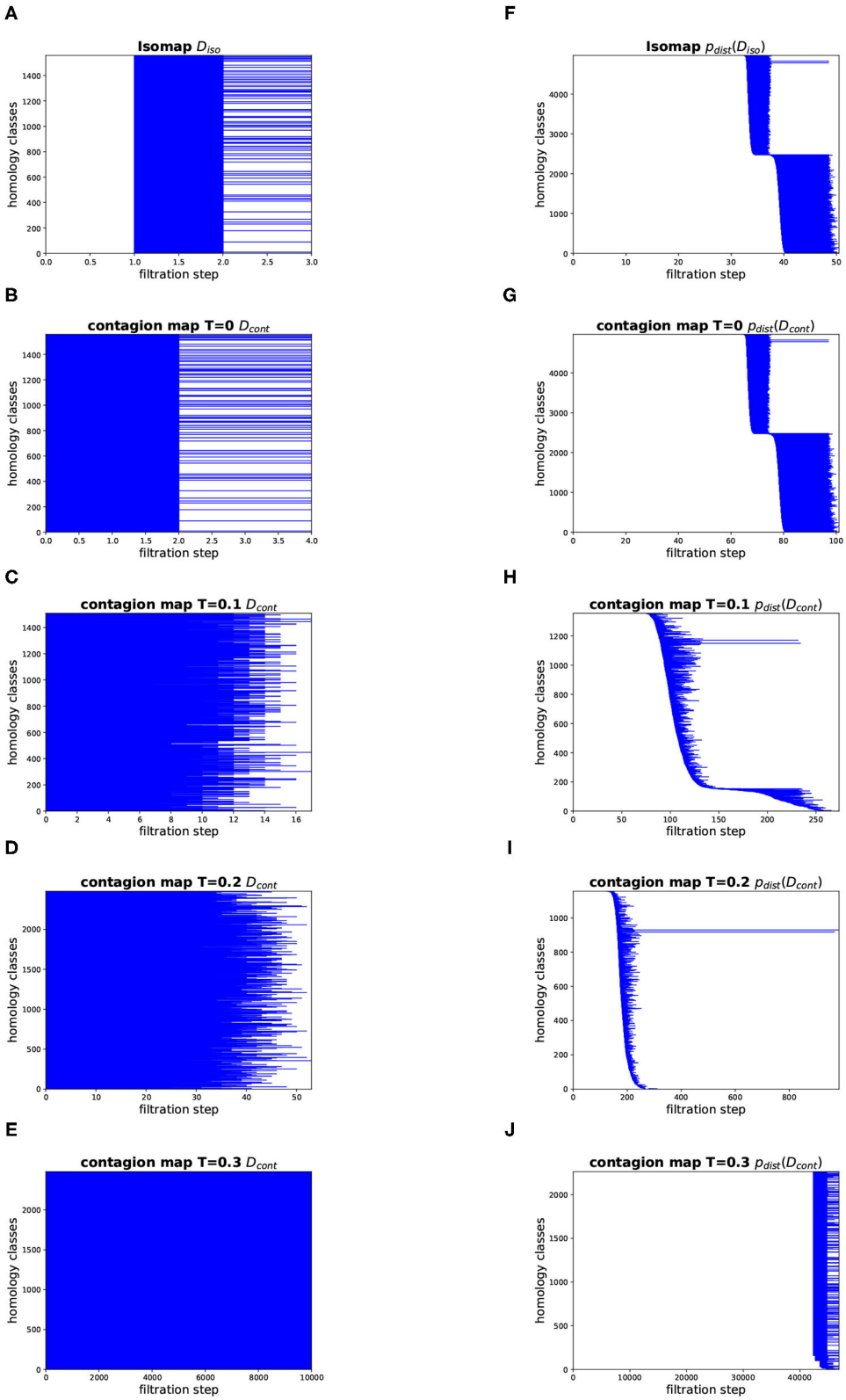


FIGURE 12 | Topology results on our torus-based network with $d^{NG} = 2$. (Left column) Dimension 1 barcodes of the Vietoris–Rips filtrations based on the estimated geodesic distances (i.e., the entries in D_{iso} and D_{cont}) according to **(A)** Isomap, and according to contagion maps with **(B)** $T = 0$, **(C)** $T = 0.1$, **(D)** $T = 0.2$, and **(E)** $T = 0.3$. (Right column) Barcodes of the Vietoris–Rips filtrations on the point clouds according to **(F)** Isomap, and according to contagion maps with **(G)** $T = 0$, **(H)** $T = 0.1$, **(I)** $T = 0.2$, and **(J)** $T = 0.3$.

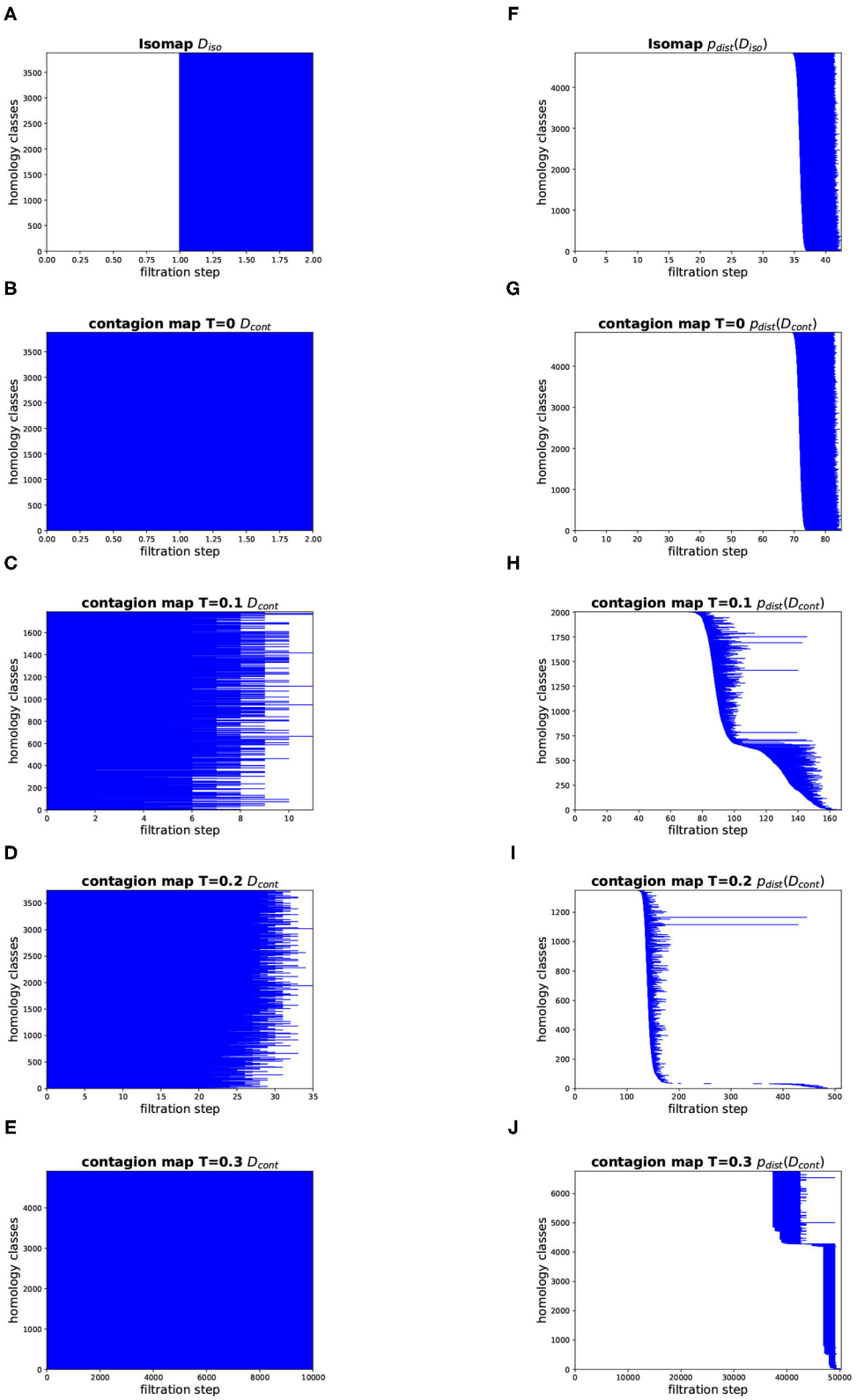


FIGURE 13 | Topology results on our torus-based network with $d^{NG} = 4$. (Left column) Dimension 1 barcodes of the Vietoris–Rips filtrations based on the estimated geodesic distances (i.e., the entries in D_{iso} and D_{cont}) according to **(A)** Isomap, and according to contagion maps with **(B)** $T = 0$, **(C)** $T = 0.1$, **(D)** $T = 0.2$, and **(E)** $T = 0.3$. (Right column) Barcodes of the Vietoris–Rips filtrations on the point clouds according to **(F)** Isomap, and according to contagion maps with **(G)** $T = 0$, **(H)** $T = 0.1$, **(I)** $T = 0.2$, and **(J)** $T = 0.3$.

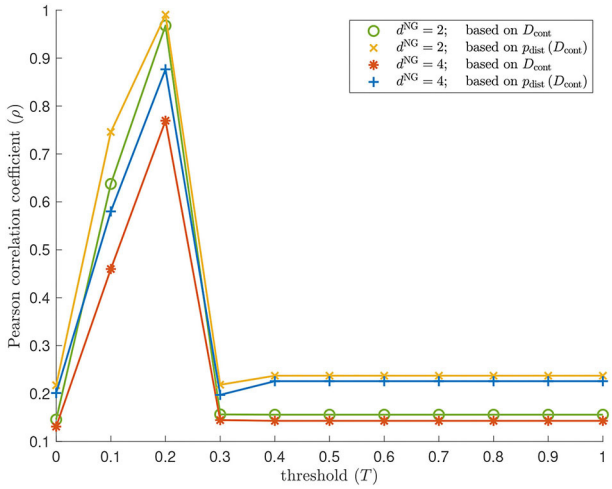


FIGURE 14 | Geometry results on our torus-based network with $d^{NG} = 2$ and $d^{NG} = 4$. Pearson correlation coefficient between the pairwise distances between the regularly spaced points on a torus specified in (1) and the following sets: the estimated geodesic distances (i.e., the entries in D_{cont}) according to contagion maps with thresholds $T = 0, 0.1, \ldots, 1$ on a torus-based network with $d^{NG} = 2$ (colored in green); the pairwise distances between points in $\mathbb{R}^{2500}$ whose coordinate vectors are the rows of D_{cont} according to contagion maps with thresholds $T = 0, 0.1, \ldots, 1$ on a torus-based network with $d^{NG} = 2$ (colored in yellow); the estimated geodesic distances (i.e., the entries in D_{cont}) according to contagion maps with thresholds $T = 0, 0.1, \ldots, 1$ on a torus-based network with $d^{NG} = 4$ (colored in red); the pairwise distances between points in $\mathbb{R}^{2500}$ whose coordinate vectors are the rows of D_{cont} according to contagion maps with thresholds $T = 0, 0.1, \ldots, 1$ on a torus-based network with $d^{NG} = 4$ (colored in blue). The results for Isomap are practically identical to those for the contagion map with $T = 0$: The Pearson correlation coefficient is 0.1458 (0.1311) when based on the entries in D_{iso} for a torus-based network with $d^{NG} = 2$ ($d^{NG} = 4$), and it is 0.2175 (0.2013) when based on the point cloud whose coordinate vectors are given by the rows of D_{iso} for a torus-based network with $d^{NG} = 2$ ($d^{NG} = 4$).

peaks correspond to values of T for which the contagion spreads predominantly as a wavefront, and therefore yields activation times that give good estimates for the actual intrinsic distances between data points and allow recovery of the underlying structure.

We also demonstrated that processing our distance estimates through p_{dist} before performing our analyzes to determine dimensionality, topology, and geometry improves these measures, giving clearer results. In some cases we saw that p_{dist} appears, in fact, to be a crucial step in the manifold-learning pipeline.

3.3. Conformation Space of the Cyclo-Octane Molecule

The cyclo-octane molecule $(CH_2)_8$ consists of a ring of eight carbon atoms, each bonded with two hydrogen atoms. A *conformation* of a molecule is a possible spatial arrangement of its atoms (modulo rotation and translation) (Moss, 1996). The conformation of a molecule can be specified by the coordinates of each of its atoms in three-dimensional space, giving a point in $\mathbb{R}^{3a}$, where a is the number of atoms in the molecule. (In this case, each coordinate of each atom in three-dimensional space is a feature and $\mathbb{R}^{3a}$ is the feature space). The set of such points for all conformations of a molecule is called its *conformation space*. Each conformation is accompanied by a state of potential energy of the molecule, and a conformation is more likely to occur the lower its associated potential energy. The cyclo-octane molecule has many conformations of comparable potential energy, and its conformation space has been studied in computational chemistry for over 50 years (Hendrickson, 1967; Pakes et al., 1981). Given the locations of the eight carbon atoms in a conformation of the cyclo-octane molecule, the locations of the hydrogen atoms are determined to minimize energy: The two covalent hydrogen atoms of each carbon atom are positioned to form a tetrahedral arrangement with the two neighboring carbon atoms that minimizes the potential energy of that subunit of the molecule. The conformation space of cyclo-octane thus lies in $\mathbb{R}^{3\times 8} = \mathbb{R}^{24}$. It is generally assumed that conformation spaces form low-dimensional manifolds, so identifying the structure of the conformation space of a molecule is essentially a manifold-learning problem. The conformation space of cyclo-octane has been shown to be the union of a sphere with a Klein bottle intersecting in two circles of singularities (Brown et al., 2008; Martin et al., 2010), forming a two-dimensional manifold with singularities.

Martin et al. (2010) analyzed a data set of 6,040 points in the conformation space of cyclo-octane, subsampled from a larger data set consisting of 1031644 cyclo-octane conformations. This data set is publicly available as part of the JAVAPLEX software package[5] (Tausz et al., 2014). To visualize this set of points, Martin et al. mapped the points from $\mathbb{R}^{24}$ to $\mathbb{R}^3$ *via* Isomap. We explore different versions of both Isomap and contagion maps on it.

Figure 15 shows the residual variances for projections *via* MDS onto dimensions 1 to 10 based on shortest-path distances, i.e., based on the entries in D_{iso}, (panel A) as well as for those based on activation times in contagions with thresholds $T = 0.1, 0.2$, and 0.4, i.e., based on the entries in D_{cont}, (panel B–D), and it shows the visualizations of the projection to 3D (panels E–H). We see that Isomap and contagion maps with low thresholds ($T = 0.1$ and 0.2) detect the embedding dimension of the underlying space, suggesting the absence of noisy edges in the 8-nearest-neighbor graph on this data set. Contagion maps with higher thresholds (e.g., 0.4) do not seem to reveal any meaningful structure. This is likely due to the contagion stabilizing before much of the graph has been activated, leading to many "infinite" activation times.

In **Figure 16**, we show barcodes corresponding to the persistent homology in dimension 1 of various Vietoris–Rips filtrations based on the cyclo-octane data set of 6040 points in $\mathbb{R}^{24}$. The barcode in **Figure 16A** corresponds to the Vietoris–Rips filtration built directly on the data points in their ambient space $\mathbb{R}^{24}$. This barcode has one dominant bar, suggesting that the Klein bottle and the sphere whose

[5]http://appliedtopology.github.io/javaplex/

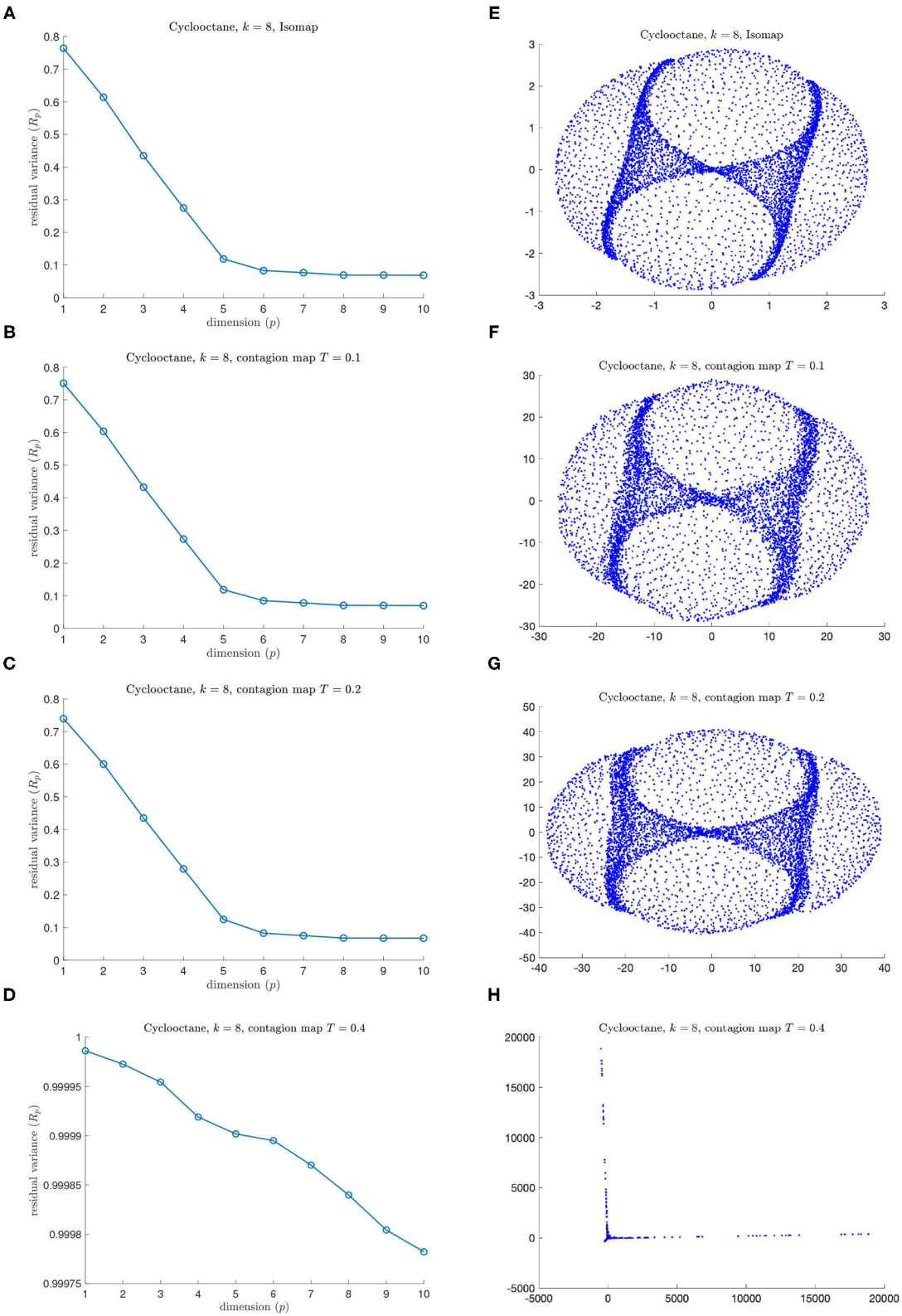


FIGURE 15 | Results for **(A,E)** Isomap and for **(B,C,D,F,G,H)** contagion maps for different values of T on the data set of 6040 points in the conformation space of cyclo-octane (using an 8-nearest neighbor neighborhood graph). **(A)** Residual variances for projections *via* MDS onto dimensions 1 to 10 in the original Isomap algorithm. **(E)** Visualization of Isomap in 3D **(B–D)**. Residual variances for projections onto dimensions 1 to 10 in the contagion-map algorithm for **(B)** $T = 0.1$, **(C)** $T = 0.2$, and **(D)** $T = 0.4$ **(F–H)**. Visualizations of projections to 3D for **(F)** $T = 0.1$, **(G)** $T = 0.2$, and **(H)** $T = 0.4$.

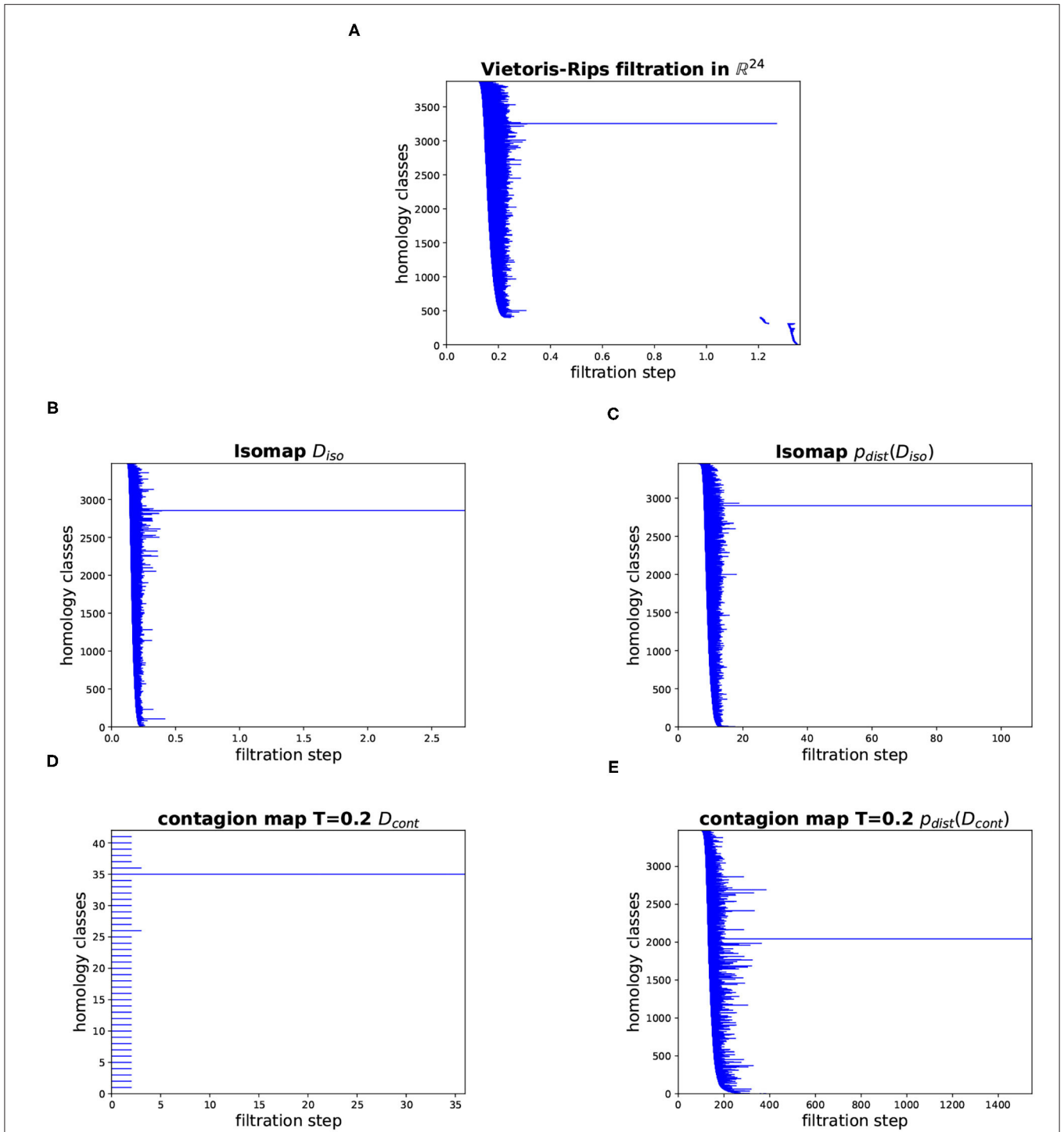


FIGURE 16 | Barcodes for the persistent homology in dimension 1 of various Vietoris–Rips filtrations based on the cyclo-octane data set. **(A)** Vietoris–Rips filtration directly on the data points in $\mathbb{R}^{24}$. **(B)** Vietoris–Rips filtration based on the shortest-path distances in the 8-nearest-neighbor graph (i.e., based on the entries in D_{iso}). **(C)** Vietoris–Rips filtration based on the points whose coordinate vectors are the rows of D_{iso} (corresponding to the 8-nearest-neighbor graph). **(D)** Vietoris–Rips filtration based on the activation times of the contagion with threshold $T = 0.2$ on the 8-nearest-neighbor graph (i.e., based on the entries in D_{cont}). **(E)** Vietoris–Rips filtration based on the points whose coordinate vectors are the rows of D_{cont} (corresponding to the contagion with threshold $T = 0.2$ on the 8-nearest-neighbor graph).

union is the conformation space of cyclo-octane intersect in a way that makes the 1-dimensional loop that is present in the homology of the Klein bottle over $\mathbb{Z}/2\mathbb{Z}$, but not over $\mathbb{Z}$, nullhomotopic. The other panels show barcodes corresponding to various Vietoris–Rips filtrations that, in a sense, mimic the Vietoris–Rips filtration built according to the intrinsic metric on the underlying manifold. Namely, they are Vietoris–Rips filtrations based on different versions of Isomap and contagion map, that is, based on estimates for the geodesic distance on the underlying manifold. **Figure 16B** shows the barcode corresponding to the shortest-path distances in the 8-nearest-neighbor graph (i.e., based on the entries in D_{iso}). **Figure 16C** shows the Vietoris–Rips filtration based on the points whose coordinate vectors are the columns of D_{iso}. **Figure 16D** shows the Vietoris–Rips filtration based on the activation times of the contagion with threshold $T = 0.2$ on the 8-nearest-neighbor graph (i.e., based on the entries in D_{cont}). **Figure 16E** shows the Vietoris–Rips filtration based on the points whose coordinate vectors are the columns of D_{cont}. All of these barcodes have one dominant bar, which is consistent with the homology of the underlying manifold. The barcode in **Figure 16D** has many bars with identical birth and death. This stems from the fact that activation times (in our contagion model) have integer values between 0 and $2N$ (where N is the number of node or, equivalently, data points), and so a Vietoris–Rips filtration based on these values has only few filtration steps at which simplices are added.

This sample of the conformation space of cyclo-octane is an example of a data set for which Isomap is successful, as are contagion maps with a sufficiently small contagion threshold T. Sweeping over a range of values of the threshold T includes the value zero, which corresponds to a contagion map that can be seen as equivalent to Isomap. We can therefore view contagion maps as an extension that includes a form of Isomap.

4. CONCLUSION

Isomap is a well-established manifold-learning tool and is useful for many data sets. It can successfully handle curvature of data in many cases, and the freedom of choosing the parameter k or ϵ when creating a neighborhood graph allows it to handle noise to some extent. However, when faced with particularly sparse and noisy data, Isomap is prone to so-called "short-circuit errors", which in some cases cannot be avoided regardless of the choice of k or ϵ. For such data, contagion maps can yield better reconstructions. For a suitable choice of the threshold parameter T, single noisy edges that occur in a neighborhood graph do not carry a contagion and thus do not distort the estimate of the geodesic distances *via* activation times significantly. In other words, with the right choice of T, contagion maps are able to "exploit social reinforcement to silence noise".

We have demonstrated this on a number of synthetic and real-world data sets, including samples from the Swiss roll, a classical benchmark data set for manifold learning. Some of the data sets we examined were point clouds, on which we built different neighborhood graphs in the first step of our algorithm. Others were already in the form of a network, on which we could directly consider threshold contagions and shortest paths.

We analyzed the activation times in multiple realizations of a threshold contagion directly in terms of dimensionality, topology, and geometry, and we also did the same analysis after performing the p_{dist}-operation, that is, after mapping data points to points in high-dimensional space based on these activation times. In doing so, we have added to what had been done with the original contagion map algorithm in Taylor et al. (2015) and Mahler (2021), which only examined point clouds in high-dimensional space. We studied these variants of the original contagion-map algorithm, and we did the analogous for Isomap, which, in its original form, only considered embedding dimension based on the shortest-path lengths directly. By comparing the variants of contagion maps and Isomap, we found not only that contagion maps perform better in many cases where Isomap breaks down due to noise-induced short circuit errors, but also that processing the distance estimates *via* the p_{dist}-operation before analyzing them leads to clearer results. Indeed, we saw in some instances that this operation seems to accentuate our results (see e.g., the red data points in **Figure 10A** vs. **Figure 10B**, or **Figure 11**). Even more remarkably, in our method for determining topological features of a data set, the p_{dist} operation does not only emphasize results but seems, in some cases, to be a necessary methodological step, bringing out features that are not detectable without this pre-processing step (see **Figures 12**, **13**). How exactly p_{dist} transforms a dissimilarity matrix and what effect this operation has on our various measures of geometry, topology, and dimensionality will be studied in future work.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article, further inquiries can be directed to the author.

AUTHOR CONTRIBUTIONS

The author confirms being the sole contributor of this work and has approved it for publication.

ACKNOWLEDGMENTS

The author thanks Vidit Nanda and Ulrike Tillmann for helpful discussions and useful comments on various versions of this article.

REFERENCES

Balasubramanian, M., Schwartz, E. L., Tenenbaum, J. B., de Silva, V., and Langford, J. C. (2002). The isomap algorithm and topological stability (technical comment). *Science* 295, 7–7. doi: 10.1126/science.295.5552.7a

Brown, W. M., Martin, S., Pollock, S. N., Coutsias, E. A., and Watson, J.-P. (2008). Algorithmic dimensionality reduction for molecular structure analysis. *J. Chem. Phys.* 129, 064118. doi: 10.1063/1.2968610

Carlsson, G. (2009). Topology and data. *Bull. Am. Math. Soc.* 46, 255–308. doi: 10.1090/S0273-0979-09-01249-X

Cox, T. F., and Cox, M. A. (2010). *Multidimensional Scaling*. Boca Raton, FL: CRC Press.

Edelsbrunner, H., and Harer, J. (2008). Persistent homology-a survey. *Contemp. Math.* 453, 257–282. doi: 10.1090/conm/453/08802

Floyd, R. W. (1962). Algorithm 97: shortest path. *Commun. ACM* 5, 345. doi: 10.1145/367766.368168

Ghrist, R. (2008). Barcodes: the persistent topology of data. *Bull. Am. Math. Soc.* 45, 61–75. doi: 10.1090/S0273-0979-07-01191-3

Hendrickson, J. B. (1967). Molecular geometry. v. evaluation of functions and conformations of medium rings. *J. Am. Chem. Soc.* 89, 7036–7043. doi: 10.1021/ja01002a036

Kleinberg, J. M. (2000). "The small-world phenomenon: an algorithmic perspective," in *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing* (New York, NY: ACM), 163–170. doi: 10.1145/335305.335325

Lee, J. A., and Verleysen, M. (2007). *Nonlinear Dimensionality Reduction*. Berlin: Springer Science & Business Media. doi: 10.1007/978-0-387-39351-3

Mahler, B. I. (2021). Analysis of contagion maps on a class of networks that are spatially embedded in a torus. *SIAM J. Appl. Math.* 81, 1416–1440. doi: 10.1137/18M1235910

Mardia, K. V., Kent, J. T., and Bibby, J. M. (1979). *Multivariate Analysis. Probability and Mathematical Statistics*. New York, NY: Academic Press Inc.

Martin, S., Thompson, A., Coutsias, E. A., and Watson, J.-P. (2010). Topology of cyclo-octane energy landscape. *J. Chem. Phys.* 132, 234115. doi: 10.1063/1.3445267

Moss, G. P. (1996). Basic terminology of stereochemistry (iupac recommendations 1996). *Pure Appl. Chem.* 68, 2193–2222. doi: 10.1351/pac199668122193

Pakes, P. W., Rounds, T. C., and Strauss, H. L. (1981). Conformations of cyclooctane and some related oxocanes. *J. Phys. Chem.* 85, 2469–2475. doi: 10.1021/j150617a013

Pearson, K. (1895). Notes on regression and inheritance in the case of two parents. *Proc. R. Soc. Lond.* 58, 240–242. doi: 10.1098/rspl.1895.0041

Sorzano, C. O. S., Vargas, J., and Montano, A. P. (2014). A survey of dimensionality reduction techniques. *arXiv[Preprint].arXiv: 1403.2877*.

Tausz, A., Vejdemo-Johansson, M., and Adams, H. (2014). *JavaPlex: A Research Software Package for Persistent (Co)Homology*. Available online at: http://appliedtopology.github.io/javaplex/ (accessed March 27, 2019).

Taylor, D., Klimm, F., Harrington, H. A., Kramár, M., Mischaikow, K., Porter, M. A., et al. (2015). Topological data analysis of contagion maps for examining spreading processes on networks. *Nat. Commun.* 6, 7723. doi: 10.1038/ncomms8723

Tenenbaum, J. B., de Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 2319–2323. doi: 10.1126/science.290.5500.2319

Torgerson, W. S. (1958). *Theory and Methods of Scaling*. New York, NY: John Wiley & Sons, Inc.

Warshall, S. (1962). A theorem on Boolean matrices. *J. ACM* 9, 11–12. doi: 10.1145/321105.321107

Conflict of Interest: The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Advantages of publishing in Frontiers

OPEN ACCESS
Articles are free to read for greatest visibility and readership

FAST PUBLICATION
Around 90 days from submission to decision

HIGH QUALITY PEER-REVIEW
Rigorous, collaborative, and constructive peer-review

TRANSPARENT PEER-REVIEW
Editors and reviewers acknowledged by name on published articles

Frontiers
Avenue du Tribunal-Fédéral 34
1005 Lausanne | Switzerland

Visit us: www.frontiersin.org
Contact us: frontiersin.org/about/contact

REPRODUCIBILITY OF RESEARCH
Support open data and methods to enhance research reproducibility

DIGITAL PUBLISHING
Articles designed for optimal readership across devices

FOLLOW US
@frontiersin

IMPACT METRICS
Advanced article metrics track visibility across digital media

EXTENSIVE PROMOTION
Marketing and promotion of impactful research

LOOP RESEARCH NETWORK
Our network increases your article's readership