Topology Preserving Data Reduction for Computing Persistent Homology

Nicholas O. Malott, Aaron M. Sens, and Philip A. Wilsey
Dept. of EECS, University of Cincinnati, Cincinnati, OH 45221, USA
Email: malottno@mail.uc.edu, sensam@mail.uc.edu, philip.wilsey@uc.edu

***Abstract*—An emerging method for data analysis is called Topological Data Analysis (TDA). TDA is based in the mathematical field of topology and examines the properties of spaces under continuous deformation. One of the key tools used for TDA is called *persistent homology* which considers the connectivity of points in a d-dimensional point cloud at different spatial resolutions to identify topological properties (holes, loops, and voids) in the space. Persistent homology then classifies the topological features by their persistence through the range of spatial connectivity. Unfortunately the memory and run-time complexity of computing persistent homology is exponential and current tools can only process a few thousand points in $\mathbb{R}^3$. Fortunately, the use of data reduction techniques enables persistent homology to be applied to much larger point clouds. Techniques to reduce the data range from random sampling of points to clustering the data and using the cluster centroids as the reduced data. While several data reduction approaches appear to preserve the large topological features present in the original point cloud, no systematic study comparing the efficacy of different data clustering techniques in preserving the persistent homology results has been performed. This paper explores the question of topology preserving data reductions and describes formally when and how topological features can be mischaracterized or lost by data reduction techniques. The paper also performs an experimental assessment of data reduction techniques and resilient effects on the persistent homology. In particular, data reduction by random selection is compared to cluster centroids extracted from different data clustering algorithms.**

***Index Terms*—topological data analysis; persistent homology; data reduction; sampling; data mining; unsupervised learning**

I. Introduction

The ubiquitous deployment of electronic and computer based data collection systems has created massive data sets that defy human analysis. As a result, a broad collection of mechanized data analysis/data mining techniques have emerged. One approach for analyzing data is based on the mathematical field of topology and is called *Topological Data Analysis (TDA)* [1]–[5]. Topology is a branch of mathematics that characterizes the properties of a space that are preserved under continuous deformation [6]. TDA leverages this aspect of topology to extract knowledge based on the shape and form of the data. There are two main techniques that apply TDA techniques for data analysis, namely: *Persistent Homology (PH)* and *mapper* [1]. TDA, and specifically PH, has been demonstrated as an effective data mining technique for a number of fields. For example, PH has been applied to analyze networks [7]–[9], brain artery trees [10], digital images [11]–[16], protein structures [17]–[19] gene sequences [20]–[22], and cardiovascular diseases [23], [24] to name a few.

Computational persistent homology explores the shape of the data as the data in the point cloud is interconnected at different spatial distances (often called ϵ distances). The computation records the ϵ distances when topological features (holes, loops, and voids) appear (called the *birth*) and disappear (called the *death*). The $< birth, death >$ pair defines the *persistence interval* of each topological feature and they can be displayed in a variety of different forms, including: barcodes, persistence diagrams, persistence landscapes, and persistence images [5]. In most cases, persistence intervals are paired with the dimension that they exist and so the persistence interval becomes $< dimension >, < birth >, < death >$.

Unfortunately, the computation of PH is exponential in both time and space [5]. This prevents the direct application of PH on big data. For example, the most efficient libraries for computing PH (currently Ripser [25], GUDHI [26], and Eirene [27]) can only process a few thousand data points in $\mathbb{R}^3$ (unless strict limits are otherwise placed on the PH processing parameters. This limitation has motivated studies to explore the application of data reduction techniques to permit the application of PH on larger data sets [28]–[31]. These techniques provide good approximations of the PH of the large features in the point cloud and achieve the PH computation at 3-4 orders of magnitude faster than using the entire point cloud (when the entire point cloud can be analyzed). The data reduction technique of Chazal *et al* [28] uses repeated trials of random selections of data from the original point cloud; the technique of de Silva and Carlsson [30] uses either random or Maxmin (which finds dispersed points across the point cloud) to select "landmark" points that have other nearby members in the point cloud for use; and Moitra *et al* [29], [31] use the centroids of k-means++ clusters. While experiments show that k-means++ centroids provide good results, more accurate results might be possible with other topologically preserving data reduction methods.

This paper compares different methods to reduce large point cloud data sets for computing PH. The study explores multiple scalings of data reduction using several different data reduction methods and evaluates how well each method preserves the topological features in the point cloud. The key objective is

Support for this work was provided in part by the National Science Foundation under grant IIS–1909096.

978-1-7281-6251-5/20/$31.00 ©2020 IEEE

to discover how different methods of data reduction sample the data such that the resulting sample $\mathcal{P}'$ is a topologically faithful representation of the original point cloud $\mathcal{P}$. By definition, the sample is not able to maintain all of the topological features of the original. However, many uses of TDA are interested only in the presence and shape of the larger (longer lived) topological features while considering the small (short lived) topological features insignificant. Thus, the specific goal of data reduction is to maintain the structure of the large topological features while removing the less concerning small topological features. This paper characterizes the different types of losses that can occur due to data reduction.

In addition, an experimental assessment that expands the study initiated by Moitra *et al* [29] with additional clustering algorithms and the random and Maxmin methods of [28], [30] is presented. The samples are composed of the cluster centroids from several different clustering algorithms, namely: k-means++ [32], Agglomerative single-linkage [33], and Agglomerative (Ward's) [34]. In addition comparison to random sampling (as performed in [28]) and the Maxmin algorithm developed to select landmark points by de Silva and Carlsson [30] are included in this study. The PH result from each sampling method is compared to the PH result from the full data set where the size of the data permits. While the theory shows many possible losses from sampling, the experimental results show that in practice the large topological features are well-preserved by many of the clustering and random sampling data reductions. However, as the data reductions increase in scale, the PH results are better preserved by only a few of the data clustering algorithms.

The remainder of this paper is organized as follows. Section II presents some of the background on PH and topology preservation. Section III briefly describes related work. Section IV outlines the general strategy for data reduction with cluster centroids. Section V provides the motivation for data reduction using different clustering algorithms. Section VI presents the experimental results on several different data sets. Finally, we conclude the paper with some remarks in Section VII.

II. Background

This section contains a brief overview of persistent homology and highlights some of the formal theories on the preservation of topological structure by various data clustering algorithms [35]. A detailed overview is available at [1], [5].

A. Persistent Homology

Homology is a means to characterize the features of a topological space. *Homology groups* at different dimensions d represent the topological features in that dimension. The Betti number at each dimension d is the rank of the d^{th} homology group, which corresponds to the number of holes in d [1].

Computing homology on a finite metric space is achieved by approximating the space with a representative complex formed from the points in that space. While there are several types of complexes (cubical, CW, or simplicial [5]) simplicial complexes are the most widely used and they will be used in this paper. Simplicial complexes are composed of simplicies; generalizations of a triangle to any number of dimensions. For example, a 0-simplex is a point, an edge is a 1-simplex, a triangle is a 2-simplex, tetrahedron a 3-simplex, and so on. However, examining a point cloud statically does not yield any meaningful topological information [5]. Instead, the point cloud must be observed at multiple scales via geometric filtrations in order to discern topological features. Features that exist over multiple scales are "persistent" and therefore meaningful from a topology standpoint. *Persistent Homology* (PH) is then the notion of homology applied to multiple geometric scales on a finite metric space.

One of the most common and computationally feasible complexes to construct filtrations is the *Vietoris-Rips* (VR) complex [36]. The filtration is a set of subcomplexes with each subcomplex based on a distance ϵ_i. The ϵ distances begin at 0 and increases until all points in the point cloud are connected (although in practice a maximum ϵ distance is often defined). As ϵ_i increases, topological features (holes, voids, and loops) appear and disappear in the space. Topologically speaking, the ϵ distance that a topological feature first appears in the filtration is called the *birth*. The *death* of a topological feature occurs at the ϵ distance where that topological feature no longer exists in the filtration. Each $< birth, death >$ tuple for a topological feature is called a *persistence interval*. Typically longer persistence intervals represent topologically meaningful features of the data and shorter ones represent noise [1]. However, in some cases the shorter persistence intervals are also of interest [10], [37].

B. Preserving Topology via Clustering Algorithms

A theoretical framework for the usage of clustering algorithms to preserve topology of a space is developed by Carlsson and Memoli [35]. Preliminary work by Niyogi *et al.* [38] shows that homological information of a manifold can be inferred from a random sampling of points distributed around it. In this case the manifold $\mathcal{M}$ is characterized as a low-dimensional underlying geometric space of the points in question. Assumptions are made that all probability distributions of points around the manifold are supported by it and any noise points are distributed via Gaussians. Given these strict assumptions Niyogi *et al.* [38] were able to show it is possible to infer higher order homological information from a sampling and reconstruction of connected components. In other words, low dimensional topological features can be used to infer higher dimensional features. In practice it is nearly impossible to satisfy the assumptions of Niyogi *et al*; especially for experimental data [39]. In order to obtain topological information from noisy data that does not lie around a manifold *persistence* must be incorporated with any homology analysis. Persistence allows one to determine which homology groups of a space are not created by noise. Persistence also identifies which features continue existing as the dimension of the homology groups increases. This solves the issue of not being able to rely on an underlying manifold for a data set. Carlsson extends the work of [38] by exploring

what occurs when a more robust sampling method beyond random sampling is used [39]. Carlsson and Memoli later found that clustering algorithms in particular enable persistent homology to be computed on reduced data sets with provable preservation of persistent features [35].

As outlined in [35], [39] a clustering algorithm $\mathbf{C}$ will take an input set of points and create a mapping which separates the points into a set of output points in partitions $\mathcal{P}$. Carlsson notes that clustering is a statistical method of sampling and mapping the connected components of a topological space to a partitioned space [39]. This mapping is unique because it acts as a function between two disparate mathematical areas — topology and set theory. This allows Carlsson to use category theory to characterize how clustering algorithms change topological spaces. A brief description of category theory is given below; see [40], [41] for a detailed background.

Category theory allows for any entity that fulfills certain conditions to be meaningfully compared to another entity that fulfills the same conditions. Generally these entities are mathematical in nature, however, they do not have to be. These entities are known as *categories*. For something to be referred to as a category, it must be a set of objects with morphisms (mappings/transformations) between pairs of objects and possess composition and identity properties. In this case, the original data set and the reduced data set are both categories and the clustering algorithm $\mathbf{C}$ is a *functor* between them. By definition, functors preserve the composition and identity properties of the categories they map from. This mapping is continuous. This usage of clustering algorithms as functors allowed Carlsson and Memoli [35] to develop theories about how they preserve topological structures of a space and persistence of topological features in the mapping they create.

III. Related Work

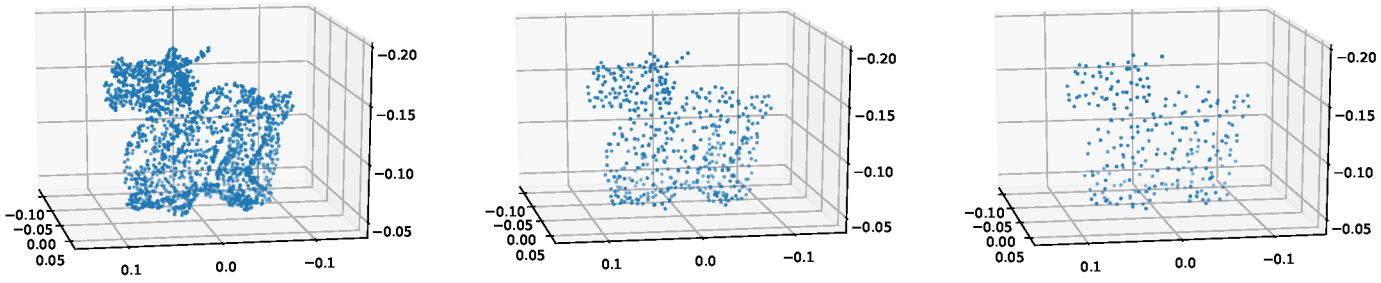
The computation of PH via the VR complex becomes intractable as the size of a small point cloud extends beyond a few thousand points in $\mathbb{R}^3$. A number of efforts have been made to simplify the computation wherever possible. The primary bottlenecks of computing PH are the size of the simplicial complex and the size and reduction of the boundary matrix [5]. Early work to alleviate these bottlenecks was achieved by modifications to the simplicial complex, primarily concerned with sparsifying the complex. In particular, Sheehy developed two theoretical methods to sparsify the Vietoris Rips complex using net trees [42]. Sheehy shows that the use of net trees to remove points and their incident simplices from the VR complex does not change the topology. However, this sparsification did not scale well to larger point clouds [43]. To attack this problem, Dey *et al.* [43] implements a method to approximate VR filtrations on much larger point clouds than Sheehy through batch collapse of simplices. Dey's usage of batch and cluster set distances as opposed to vertex distances when merging complexes resulted in increased scalability from the initial approach described in [44]. More recently, Brehm and Hardering [45] were able to implement a more scalable version of Sheehy's sparsification method in the Julia library `Sparips`. `Sparips` first builds a contraction tree (similar to a cover tree [46]) over the raw data before the construction of the boundary matrix. The boundary matrix is then sparsified using information from the contraction tree. `Sparips` provides a tighter bound on approximation of PH than [43] and obtains performance comparable to GUDHI. All of these approaches reduce the number of complexes constructed from the original point cloud in order to identify significant topological features on larger point clouds.

Data reduction through sampling has also been explored to expand the processing capabilities of PH libraries. Chazal *et al.* [28] utilize repeated random sampling of the original point cloud and compute PH from the average landscape of those samples. However, as noted by Sheehy [42], the combination of different persistent diagrams into a single diagram did not yield the same accuracy; strict assumptions had to be made on the data for the best results. Moitra *et al.* [29] used a similar approach that sampled the point cloud using k-means++. Because the k-means++ algorithm samples the data through multiple iterations to reduce the WCSS error, PH only needs to be calculated on one sample, as opposed to the multiple samples in Chazal *et al*. Moitra *et al.* shows that sampling the data using k-means++ preserves significant topological features and has similar results to Chazal *et al* [28]. In addition, k-means++ also allows for the use of *upscaling* of the reduced data to increase the accuracy of the persistent intervals computed from the sampled point cloud [31]. However, these studies did not explore the impact of data reduction at increasingly large reduction percentages or for the broader impact that various other data reduction methods would have on the results of computing PH on reduced data.

Finally, some studies have attempted to use random projection to enable the computation of PH on high dimensional data sets. Random projection allows higher dimensional data to be mapped to lower dimensions, while preserving distances between points with bounded error [47]. Sheehy uses this idea to prove that random projection preserves the persistent homology of the point cloud to a comparable bound [48]. From the theoretical results, Ramamurthy [49] conducted experiments on PH of randomly projected point clouds and showed that the persistence diagrams were similar for a variety of random projections.

IV. Topology Preserving Data Reduction

The focus of this paper is the transformation of a point cloud $\mathcal{P}$ that is too large for computing PH to another point cloud $\mathcal{P}'$ with fewer total points such that the PH can be computed on a representative point cloud. Ideally, the transformation should be such that $\mathcal{P}$ and $\mathcal{P}'$ maintain the structure of large topological structures, homeomorphic to some degree. An example of a suitable mapping is illustrated in Figure 1. The leftmost image has $2,000$ points and represents the original point cloud $\mathcal{P}$; the center and rightmost images represent two topologically preserving reductions of 500 and 250 points and are possible representations of the reduced point cloud $\mathcal{P}'$. The general approach is to partition the original point



Original point cloud of 2000 points | Reduced point cloud of 500 points | Reduced point cloud of 250 points

Fig. 1. Reduction of the Stanford Dragon triangulated mesh model with k-means++

cloud and use the geometric centers of the partitions as the set of points in $\mathcal{P}'$. In addition to preserving the topological structure, the ideal data reduction methods must be able to operate on large data sets. Finally, there are additional benefits if the data reduction is achieved by partitioning. In particular, the partitioning approach (achieved through data clustering or otherwise) is desired over random sampling methods [28], [30] as it permits for *upscaling* of the partitions on the periphery of a topological feature to recover a more accurate persistence interval for that feature [31].

As illustrated in Figure 1, the application of partitioning centroids to accelerate the computation of persistence changes the distances between points in the point cloud, but preserves the general shape of the point cloud. This change in homology can have impact on the $< birth, death >$ persistence intervals produced from the PH computation, and even the mischaracterization of some persistence intervals. More precisely, the approximation of the PH from a data reduced point cloud can introduce a bounded error on the lifespans of topological features [29] that is directly related to the maximum radius of the partition, r_{max}. In practice this error should be significantly less and experimental results show that the actual error is well below this limit [29].

However, there are limits to representing the point cloud through increasing data reductions; at some point, the topologically significant features of the original data will fail to emerge.[1] It is important to understand how and when these features might disappear. Furthermore, in some cases it is possible that a topological feature will be lost or shifted to a lower dimension (when the centroid data does not form sufficient simplices surrounding the feature in all dimensions that map to the convex hull of the feature in the full data set). While this is possible, it generally occurs at more extreme levels of data reduction. Preliminary experiments in this paper and elsewhere [29], [31] show that the identification of the larger topological features in a point cloud are well preserved by data reduction.

The remainder of this section expands on the technical approach of this method. In addition to providing details on the possible implementation methods for this approach, the challenges, errors, and perceived solutions are reviewed. In general a worst case analysis is presented and, in practice, many of the challenges and performance issues are not as significant as outlined in this section. For the remainder of this paper, the following symbols will be used:

- d, the dimension of the elements in the point cloud,
- $\mathcal{P} = \{\mathbb{R}^d\}$, the point cloud,
- $N = ||\mathcal{P}||$, the total number of points in the point cloud,
- M, the targeted upper limit of total points in $\mathbb{R}^d$ to be used in the PH computation, and
- $\epsilon = (\epsilon_0, \epsilon_1, \cdots \epsilon_q)$, the sequence of ϵ distances used for computing PH where $\epsilon_i < \epsilon_j$ for $i < j$. Further let $\epsilon_{min} = \epsilon_0$ and $\epsilon_{max} = \epsilon_q$.

A. Partitioning

This section examines the data reduction step as a generic partitioning of the points in the original point cloud. Although some partitioning methods may result in more complete solutions, in general most d-dimensional spatial partitioning approaches will work [39]. To enable the computation of PH on the reduced data set, the partitioning should define no more than M partitions. Each point in the original point cloud must be placed in one and only one partition. More precisely, let $\hat{\mathcal{P}} = \{p \mid p \subset \mathcal{P}\}$ be a partitioning of $\mathcal{P}$, then $\forall p, q \in \hat{\mathcal{P}} \mid p \neq q, p \cap q = \emptyset$ and $\cup_{p \in \hat{P}}\ p = \mathcal{P}$.

From this partitioning, the algorithm will then select a single representative point from each partition to define a new point cloud with fewer total points than the original point cloud. While any data point (actual or representative) from each partition can serve this purpose, this work will examine the specific use of the geometric center of each partition where the geometric center is defined as the mean of each dimension of all the points in that partition. More precisely, let $\mathcal{P}'$ be the set of geometric centers of the partitions $\hat{\mathcal{P}}$, then if $p_i \in \mathcal{P}'$ is the centroid for partition $\hat{p}_i \in \hat{P}$ then $p_i = \frac{\sum_{q \in \hat{p}_i} q}{||p_i||}$. In the remainder of this paper, the term *centroid* will be used to denote the geometric center of a partition.

The principle objective of this step is to build a point cloud $\mathcal{P}'$ such that the larger topological features present in $\mathcal{P}$ are also present in $\mathcal{P}'$. In the remainder of this paper, the following additional terms will be used:

- $\hat{\mathcal{P}}$, the partitions,
- $\mathcal{P}'$, the centroids,

[1] In the extreme case where, for example, the reduction to a single point, the entire structure is lost. However, in this study, we will study more practical limits and not further discuss these extreme cases.

- r_i, the distance from the partition centroid, $\mathcal{P'}_i \in \mathcal{P'}$, to the most distant point in that partition, and
- $r_{max} = max(r_i)$, the maximum r_i of all the partitions.

Using the topologically similar (but smaller) point could $\mathcal{P'}$ to approximate the PH of $\mathcal{P}$ will identify the large topological features (as defined by the bounds of Section IV-B). In particular, let $\mathcal{B}$ be the boundary of points in the complex defining a $d \geq 2$-dimensional topological feature and let $s_{\mathcal{B}} = max(distance(b_i, b_j))\ \forall b_i, b_j \in \mathcal{B}$. Then define the term "large topological feature" to be any feature with diameter $s_{\mathcal{B}} > 2r_{max}$. That is, a large topological feature has a diameter that is not contained within the largest partition of $\hat{\mathcal{P}}$. Depending on its location in the partitions, any topological feature with a diameter smaller than $2r_{max}$ may or may not be identified during this step. In particular, any topological feature that falls within the boundary of a partition will be lost; any topological feature that extends beyond the boundaries of the partitions are likely to be retained. The degree to which features are lost may be significantly impacted by the mechanism/algorithm used to define the partitions.

B. Computing PH on Partition Centroids

The partitioning step is used to create a mapping from the original, large input point cloud $\mathcal{P}$ to a topologically similar but smaller (in terms of total points) point cloud $\mathcal{P'}$. Ideally this mapping will be performed in a manner that preserves the larger topological features of the original point cloud. Formally $||\mathcal{P'}|| < M < ||\mathcal{P}||$ so that PH can be computed on $\mathcal{P'}$ in the allotted time and space. This step is performed as follows. The PH of the original point cloud is estimated by a computation of PH on the partition centroids $\mathcal{P'}$. When computing PH on $\mathcal{P'}$ instead of $\mathcal{P}$, the smaller topological features that lie within a radius of any partition will be hidden from this PH computation. For the large topological features in $\mathcal{P}$, approximating the PH using $\mathcal{P'}$ can result in the following deviations from the PH results of $\mathcal{P}$:

1) The persistence interval $< birth, death >$ may occur at different (but bounded) ϵ distances.
2) A false topological feature not actually in $\mathcal{P}$ may be identified.
3) A topological feature of $\mathcal{P}$ in dimension $n > 2$ may present itself in $\mathcal{P'}$ at a different dimension m.
4) A topological feature might be lost by the data reduction step.

The frequency and significance of these deviations is influenced by the partitioning methods used to define $\hat{\mathcal{P}}$ and consequently $\mathcal{P'}$. For the remainder of this section, a worst case characterizations of these deviations will be presented.

C. Error Bounds on Resulting Persistence Intervals

As developed by Moitra *et al* [29].

Theorem 1. *The shift in the $< birth, death >$ persistence interval values arising when computing PH from P' instead of P is bounded by $2r_{max}$.*

Fortunately, this error can be reduced by an upscaling step [31]. More precisely, the approximate $< birth, death >$ interval for any topological feature identified from the estimated PH can be refined by recomputing the PH using all of (and only) the points from the partitions containing the centroids that form the boundary of the feature. This process is called *upscaling*. If the upscaled point cloud contains too many points for computing PH, an iterative repartitioning and upscaling can be performed to refine the approximation of the feature boundary. Of course there is a limit; if the convex hull of points on the boundary exceeds M, then the improvements by upscaling may be further limited.

D. False Topological Features

False topological features can arise due to false voids that lie between the centroids of the reduced point cloud $\mathcal{P'}$. That is, the gaps between the centroids due to the removal of the partition points can be such that the centroids of $\mathcal{P'}$ define a complex around a false topological feature. Formally, these false features can occur only when:

Theorem 2 (False voids from centroid gaps)**.** *False voids can appear when $\epsilon_{min} < r_{max} < \epsilon_{max}$.*

Proof. Consider a 2-dimensional space of 4 square partitions with radius r uniformly filled with points in $\mathcal{P}$, where ϵ_{min} <r< ϵ_{max} and where the minimum pairwise distance between any two points in $\mathcal{P}$ is less than ϵ_{min}. Then $\mathcal{P'}$ would consist of the centroids of these squares. Computing PH on $\mathcal{P'}$ would result in the discovery of a topological feature not present in the original point cloud $\mathcal{P}$. ■

False topological features can be pruned by upscaling [31].

E. Dimension Shift of Topological Feature

Estimating the PH of $\mathcal{P}$ using $\mathcal{P'}$ can also cause a topological feature to shift dimensions. Shifts into higher dimensions occur when the points in $\mathcal{P'}$ stretch a feature with a void space in new dimension. Shifts into lower dimensions occur when a topological feature that has a convex hull in k-dimensions in $\mathcal{P}$ loses a cover in one (or more) of the dimensions in $\mathcal{P'}$ so that the convex hull only occurs in $j < k$-dimensions. That is:

Theorem 3 (Feature shift to higher dimensions)**.** *The PH computation in $\mathcal{P'}$ may shift an identified topological feature into a higher dimension.*

Proof. Consider a point cloud in $\mathbb{R}^3$ composed of N points that contains a single 2-dimensional circle of radius $r > 2\epsilon_{max}$ in the xy-plane at $z = j$. Consider a partitioning of the space such that one partition is a square with sides of length r and located immediately above the 2-dimensional circle and with all points outside of the square defining a partition. The data reduction step will introduce a void space in the z dimension above and including the circle. ■

Feature shifts to higher dimensions are not a significant issue. They will be pushed back into the proper dimension with the upscaling computation.

Theorem 4 (Feature shift to lower dimensions)**.** *The PH computation in $\mathcal{P}'$ may shift an identified topological feature into a lower dimension.*

Proof. Consider a 3-dimensional space with a sphere at the origin with radius r_s ($\epsilon_{min} < r_s < \epsilon_{max}$) and with a uniform distribution of points extending some finite distance beyond the surface of the sphere. Consider a partitioning such that (a) a collection of partitions that lie in the xy-plane at $z = 0$ with the maximum radius r_{max} for all of these partitions is such that $\epsilon_{min} < r_{max} < \epsilon_{max}$ the remaining space lies in two partitions: one covering all points in $\mathcal{P}$ at $(x, y, z > 0)$ and with a centroid at $(x, y, z > \epsilon_{max})$; and covering all points in $\mathcal{P}$ at $(x, y, z < 0)$ and with with a centroid at $(x, y, z < \epsilon_{max})$. Then the resulting centroid points would be such that the sphere would only appear to the PH algorithm as a 2-dimensional circle and no connections would be made to the centroids above and below the $z = 0$ axis as they would lie outside the range of the ϵ values examined by the PH algorithm. ■

Feature shifts into lower dimensions are more problematic. They cannot easily be restored to their proper dimension and the feature will be lost.

F. Lost Topological Features

In rare cases, a topological feature can be lost when the points defining the boundary for the convex hull in $\mathcal{P}$ surrounding the topological feature are insufficient in $\mathcal{P}'$ for a corresponding convex hull to be defined. This issue may motivate the use of multiple partitioning steps to estimate PH with strategic partitioning methods.

V. Experimental Study

The motivation of this work is to be able to compute PH on big data. Preliminary data [29], [31] suggests that spherical clustering methods such as k-means++ can present a suitable partitioning of the data. However, it is unclear if this is the best method for data reduction or if other algorithms or heuristics provide better topologically preserving abstractions.

This experimental study examines the use of other clustering methods for data reduction. In particular hierarchical clustering algorithms are of interest due to their theoretical guarantees of preserving persistent features [35]. More precisely, a comparative analysis of k-means++, single-linkage agglomerative clustering, ward-linkage agglomerative clustering, and Silva's Witness maxmin sampling [30] is performed.

Density-based cluster algorithms such as DBSCAN [50], HDBSCAN [51], and mean-shift were evaluated but performed poorly due to the lack of ability to set k, the number of centroids to generate. This led to classifications at different bandwidths and parameters to attempt to approximate the point clouds of suitable size. That said, several suitably sized point clouds were derived using these methods. However, ultimately the reduced point clouds $\mathcal{P}'$ extracted using these density-based clustering algorithms failed to preserve the topological features on par with other methods. As a result these algorithms are excluded from the results in this section.

The experimental study measures the accuracy of the persistence intervals computed with the various approaches against a reference computation of the persistence intervals on the original point cloud. Accuracy at this stage is determined by the *Heat Kernel Distance (HKD)* [52], [53] between the original persistence intervals computed from $\mathcal{P}$ and the persistence intervals computed from $\mathcal{P}'$. The HKD gives a stable heat-kernel metric for classification applications and is a way to obtain topological inferences about an object using Gaussian kernel density estimates. Additionally heat kernel distance is robust to noise and outliers in data, which makes it a valuable analysis metric for comparing persistence intervals.

VI. Experimental Results

Each algorithm under comparison was implemented as a preprocessing step to reduce the input point cloud. In the case of clustering algorithms such as k-means++, the cluster centroids were output; for hierarchical cases the dendrogram was cut at a specific threshold. After each reduced point cloud was obtained, the PH library Ripser [25] was used to compute PH on the original and reduced point clouds. Ripser is currently a state-of-the-art library for computing PH with Vietoris-Rips complexes, efficient in both speed and memory performance. However, similar results can be obtained with GUDHI [26] or Eirene [27]. All experiments were conducted using an Ryzen Threadripper 1950X with 128GB of RAM.

The HKD comparisons are computed on persistence intervals separated by dimension. While an aggregate comparison of the HKD to the complete set of persistence intervals (independent of dimension) was performed the results did not provide any significant insights and, due to space considerations, they are not presented here. Accuracy measured by persistence interval dimension gives a notion of the preservation of all persistence intervals in the original data set. However, many of the persistence intervals contributing to the HKD results in this instance can be attributed to shorter persistence intervals or noise that are by definition going to be missing in the reduced point cloud. Since the focus on this paper is to how well these reduction methods preserve the significant, larger topological features, a deeper analysis is necessary. As a result, several methods to filter the persistence intervals by length are explored and compared. That is, the persistence intervals from the original point cloud are examined and cutoff lengths were established using several filtering methods in an attempt to isolate the significant persistence intervals for comparison. For example, one filter finds, by dimension, the shortest persistence interval from the longest 10% found in the original point cloud. This length is then used to filter persistence intervals computed from the original and reduced data sets. The HKD for these filtered results are then computed and reported. This will provide insight on how well the reduced data sets preserve the significant, larger topological features. In cases where the persistent homology of the original data set could not be computed due to resource constraints, the maximum number

of points was reduced and compared to continued reduction with the same algorithm.

Section VI-A analyzes the effect of reduction on dimensional persistence intervals when comparing to the baseline data set. The experiments provide a comprehensive comparison of the different algorithms presented. Section VI-B focuses on the two methods that produced the best overall HKD results (k-means++ and Agglomerative Ward) and performs a deeper analysis of the HKD comparisons as the persistence intervals are filtered. Each algorithm's ability to preserve the salient topological features of the original point cloud are the focus and motivation for the HKD comparisons with the filtered persistence intervals.

A. HKD for All Results and Data Sets

Testing data sets used for the comparison include real-world and synthetic data. Several triangulated mesh models were used, including an $\mathbb{R}^3$ model of the klein bottle, denoted $klein$, and an $\mathbb{R}^3$ model of a $lion$ from the publicly available triangulated shapes database [54]. Both of these models are beyond the limitations of persistent homology on the original point cloud and are reduced using k-means++ to 900 and 1000 points, respectively. These reduced data sets are then used as the baseline "original" data for these tests.

In addition, the $seeds$ data set from the UCI machine learning repository and the Water Treatment Plant data set ($water$) were analyzed for higher-dimensional topological features and the effects of data reduction beyond $\mathbb{R}^3$. Both of these data sets are traditional classification and clustering data sets that provide a measure of the approach in general classification, specifically H_1 and H_2 intervals.

Two lower dimensional $\mathbb{R}^2$ point clouds, $twoMoons$ and $twoCircles$, were examined to understand the scale based on n, as features in these two data sets are only relevant up to H_1. Each point cloud is synthetically generated and are comprised of 2000 points that model, respectively, two separated half moons and two separated circles.

The test data sets were tested at several different reductions including extreme reductions down to fewer than 100 points. HKD results by dimension for each of these data sets is displayed in Table I. Each column represents the HKD for the reduction algorithm used against the various data sets. The columns are further separated by dimension of the persistence intervals. Rows represent each data set at a specific reduction.

In general the clustering algorithms perform slightly better than both the Maxmin and random approaches. k-means++ and Agglomerative Ward (AggWard) can identify some features further in reduction, such as twoCircles in H_1, but in other cases random performs well on the lion point cloud sampled from the original 4999 vectors for H_0 and H_1 features, and only bested by k-means++ and Maxmin in H_2. While Agglomerative Single Link provides somewhat comparable results, it also lost more dimensional features in some of the tests. The results provide some evidence that k-means and Agglomerative ward consistently provide suitable results at significant reductions. However, the amount of preservation

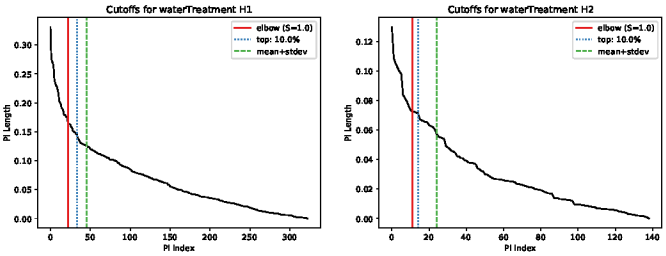


Fig. 2. Knee-based approach on the waterTreatment data set for filtration of longer persistence intervals.

of large topological features can be difficult to identify while examining entire dimensions of persistence intervals for the heat kernel distance. Understanding of the preservation of large topological features motivates further methods to characterize preservation of large topological features.

B. HKD for Filtered Results from k-means++ and AggWard

Large topological features are often the focus of persistent homology; long persistence intervals indicate the importance of the feature. Classification of large topological features can be difficult without domain experience to provide static cutoff lengths of persistence intervals that constitute significance. In order to consider the same accuracy analysis from Section VI-A on longer persistence intervals a filter needs to be applied to remove short persistence intervals. These short persistence intervals are often considered noise when analyzing data with persistent homology.

Several different approaches to this filtration of the persistence intervals were analyzed. The filtering is always done by examining the results from the original, baseline point cloud results and is set by selecting a cutoff length from the persistence intervals at each dimension using various filtering methods. These methods included computing the mean and standard deviation of the persistence intervals, filtering by the lengths of the top percentages of the persistence intervals, and using the kneedle algorithm [55] at various sensitivities to estimate the (convex) knee of the sorted (in decreasing order) persistence intervals. While each of these approaches provided useful filtering for results analysis, the kneedle algorithm, the sum of the mean and standard deviation, and the (shortest of the) persistence intervals of the longest 10% in that dimension provide the most suitable filtering lengths. Variations of the sensitivity of the kneedle algorithm did not produce significant differences in the results; the results with sensitivity set at 1.0 are reported.

Figure 2 depicts the persistence interval lengths (sorted in decreasing order) for homology groups H_1 and H_2 and the cutoff positions for three of the filtering algorithms studied. Specifically the cutoff filters for the elbow of the kneedle algorithm (elbow (S=1.0)), the shortest of the first 10% persistent interval lengths (top: 10.0%) and for the sum of the mean and standard deviation (mean+stdev). While the mean and standard deviation ($\overline{x}+\sigma$) may be a first choice for filtering the longer persistence intervals, the knee provides a more robust

Data	n	k-means++			aggWard			aggSingle			MaxMin			random		
		H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2
klein (900 pts)	600	3.40	0.29	0.02	3.20	0.29	0.06	3.51	0.38	0.06	4.68	0.98	0.11	6.36	1.13	0.24
	300	11.16	2.09	0.18	10.87	2.05	0.13	12.83	2.48	0.36	13.77	2.76	0.35	13.94	2.79	0.36
	200	14.60	2.75	0.29	14.11	2.68	0.28	16.74	3.48	0.43	17.11	3.40	0.39	20.36	3.42	0.38
	100	18.41	3.47	0.40	18.46	3.56	0.35	20.94	4.18	–	21.22	4.16	0.41	24.54	3.89	0.41
	50	21.41	3.92	0.42	21.48	3.95	0.41	23.32	4.44	–	23.47	4.42	0.43	27.12	4.30	0.42
lion (1000 pts)	750	2.51	0.32	0.01	2.58	0.49	0.00	2.47	0.33	0.02	2.45	0.32	0.01	1.96	0.44	0.02
	500	5.67	0.95	0.04	5.68	1.05	0.02	5.57	0.79	0.06	5.41	0.78	0.04	3.93	0.60	0.01
	250	9.89	1.63	0.05	9.94	1.76	0.10	9.73	1.46	0.07	9.41	1.25	0.05	6.65	1.02	0.05
	100	13.45	2.17	0.06	13.45	2.19	0.11	13.44	2.05	0.09	12.28	1.76	0.06	9.26	1.32	0.05
	50	15.03	2.40	0.06	15.12	2.54	0.14	14.72	2.24	0.10	13.68	1.91	0.06	10.19	1.37	0.07
	10	17.25	2.68	–	17.37	2.83	–	16.49	–	–	14.98	2.05	–	11.75	1.50	–
seeds (210 pts)	200	0.20	0.01	0.01	0.20	0.01	0.01	0.20	0.01	0.01	0.40	0.03	0.01	0.63	0.05	0.01
	150	2.03	0.08	0.03	2.07	0.10	0.03	1.92	0.13	0.04	2.36	0.19	0.04	3.30	0.46	0.04
	100	4.72	0.48	0.07	4.85	0.46	0.05	4.70	0.32	0.05	4.91	0.49	0.05	6.17	0.79	0.05
	50	8.50	0.75	0.06	8.44	0.83	0.07	8.41	0.85	–	8.49	0.85	0.07	9.48	0.97	–
	20	11.40	1.00	–	11.45	1.01	–	11.25	1.05	–	11.54	1.01	–	12.86	1.04	–
water (527 pts)	400	22.62	0.91	0.43	22.81	0.98	0.30	21.85	0.97	0.39	21.57	0.67	0.24	25.49	0.80	0.13
	300	44.32	2.23	0.63	44.45	2.45	0.51	42.05	2.07	0.56	42.09	1.45	0.30	48.46	2.52	0.58
	200	67.23	3.31	0.85	68.11	3.42	0.81	65.42	3.25	0.89	65.66	3.10	0.70	68.45	3.41	0.85
	100	91.61	4.20	0.94	91.71	4.34	0.88	91.35	4.53	–	91.91	4.66	0.96	90.97	3.99	0.86
	50	103.27	4.50	0.94	103.33	4.59	0.95	104.03	–	–	104.34	–	–	102.59	4.44	0.95
twoMoons (2000 pts)	1500	0.27	0.02		0.26	0.01		0.27	0.02		0.54	0.04		0.97	0.10	
	1000	0.99	0.07		0.94	0.07		1.01	0.07		1.44	0.13		2.11	0.24	
	500	2.48	0.28		2.45	0.25		2.53	0.24		2.84	0.31		3.52	0.38	
	250	3.85	0.42		3.81	0.39		3.74	0.38		3.94	0.41		4.64	0.49	
	100	5.13	0.49		5.16	0.51		4.67	0.45		4.79	0.46		5.29	0.53	
	50	5.68	0.55		5.70	0.53		5.24	0.46		5.21	0.46		5.64	0.52	
twoCircles (2000 pts)	1500	0.00	0.00		0.00	0.00		0.00	0.00		0.00	0.00		0.00	0.00	
	1000	0.00	0.00		0.00	0.00		0.00	0.00		0.01	0.00		0.01	0.01	
	500	0.01	0.00		0.01	0.00		0.01	0.00		0.03	0.02		0.03	0.02	
	250	0.02	0.01		0.02	0.01		0.03	0.02		0.05	0.04		0.06	0.04	
	100	0.06	0.03		0.06	0.03		0.10	0.06		0.19	0.12		0.16	0.10	
	50	0.13	0.06		0.13	0.06		0.21	0.12		0.31	0.18		0.25	0.14	

TABLE I

HKD OF THE PERSISTENT DIAGRAMS AT VARIOUS REDUCTIONS BY EACH ALGORITHM UNDER TEST. ENTRIES MARKED "–" ARE WHERE THE REDUCED DATA LOST ALL THE FEATURES IN THAT DIMENSION. BLANK ENTRIES ARE POINT CLOUDS WITH NO TOPOLOGICAL FEATURES IN THAT DIMENSION.

and mechanized approach to filtering the top-most intervals. Unfortunately with filtering the top 10% of persistence intervals, the resultant intervals may include shorter intervals based on the number of persistence intervals generated.

All three filters refine the compared persistence intervals provides a measurement of preservation of the salient topological features in the point cloud. An analysis of all data sets chosen for VI-A are included in Figure II to present a further comparison of k-means++ and Agglomerative Ward and their preservation of these longer persistence intervals at various levels of reduction.

There are several notable findings in the filtered comparison as shorter barcodes contribute less noise to the heat kernel distance metric. In the water treatment dataset, the H_1 features in all three filterings displays a significant loss of identified features when reducing from 200 to 100 points. This dropoff indicates a large feature has been lost during the reduction and may indicate the reduction's limits for that specific dataset. Utilizing the persistence interval filtering can provide insight to the accuracy of each algorithm in extreme cases of reduction to quantify only the large persistence intervals for comparison.

Overall partitioning algorithms perform with similar accuracy up to a significant percentage of data reduction. However, several of the algorithms better preserve the salient topological features at more significant reduction percentages, namely k-means++ and aggWard. These algorithms provide similar accuracy results and can give the most accurate persistence intervals under significant reduction. Between k-means++ and aggWard, there are slight differences in their ability to preserve long persistence intervals depending on the structure of the point cloud. Both are nearly identical in the twoCircles and twoMoons dataset; aggWard seems to perform better with triangulated mesh point clouds, while k-means++ has higher accuracy with the categorical point clouds seeds and water.

VII. CONCLUSIONS

Persistent homology presents a novel approach to analyzing data. Unfortunately the computation of persistent homology on big data is not currently possible due to its exponential complexity. Data reduction is a classic approximation technique used by the data mining/machine learning communities to attack computational complexity issues and various explorations to use data reduction for the computation of persistent homology have been performed. This paper has explored how well different data reduction strategies functioned to preserve the large topological features present in a point cloud.

While the experiments performed in this analysis focused on several notable clustering algorithms, only slight improvements in some algorithms were identified to provide better

Data	n	k-means++ (S=1.0)			k-means ($\overline{x}+\sigma$))			k-means (top 10%)			aggWard (S=1.0)			aggWard ($\overline{x}+\sigma$)			aggWard (top 10%)		
		H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2	H_0	H_1	H_2
klein (900 pts)	600	0.06	0.33	0.03	2.10	0.35	0.03	0.12	0.32	0.02	0.00	0.22	0.01	2.55	0.18	0.01	0.22	0.18	0.01
	300	3.55	0.58	0.01	8.58	0.37	0.01	6.10	0.57	0.01	4.66	0.46	0.04	9.54	0.31	0.04	6.41	0.51	0.03
	200	7.72	0.31	0.06	7.11	0.25	0.06	8.23	0.28	0.05	9.31	0.28	0.04	8.16	0.24	0.05	9.60	0.26	0.05
	100	7.87	0.26	0.13	3.95	0.64	0.13	6.99	0.28	0.11	7.93	0.30	0.07	3.89	0.73	0.07	6.93	0.33	0.07
	50	4.99	0.51	0.14	1.11	0.97	0.14	4.00	0.55	0.12	4.92	0.55	0.13	1.04	1.01	0.13	3.93	0.59	0.12
lion (1000 pts)	750	0.09	0.24	0.01	1.89	0.22	0.00	0.09	0.21	0.00	0.67	0.18	0.02	2.23	0.10	0.02	0.13	0.08	0.00
	500	0.83	0.09	0.00	4.35	0.10	0.02	0.80	0.09	0.00	2.90	0.11	0.01	4.27	0.12	0.01	0.93	0.12	0.00
	250	3.91	0.10	0.03	4.28	0.13	0.04	3.91	0.14	0.02	5.28	0.10	0.04	4.25	0.13	0.04	3.79	0.13	0.03
	100	4.20	0.37	0.05	2.07	0.42	0.06	4.20	0.43	0.04	4.67	0.12	0.05	2.23	0.29	0.05	4.33	0.28	0.04
	50	3.50	0.55	0.07	0.67	0.60	0.09	3.50	0.61	–	3.31	0.35	0.08	0.68	0.61	0.08	3.49	0.60	–
	10	1.35	–	–	1.72	–	–	1.35	–	–	1.12	0.58	–	1.73	0.86	–	1.29	0.85	–
seeds (210 pts)	200	0.00	0.03	0.00	0.00	0.01	0.01	0.00	0.03	0.01	0.00	0.03	0.00	0.00	0.01	0.01	0.00	0.03	0.01
	150	0.14	0.04	0.00	0.32	0.02	0.00	0.12	0.04	0.01	0.14	0.05	0.01	0.21	0.04	0.00	0.12	0.05	0.01
	100	0.45	0.11	0.02	2.34	0.09	0.02	1.13	0.11	0.02	0.60	0.16	0.01	2.06	0.16	0.00	1.23	0.16	0.01
	50	0.61	0.23	0.02	2.80	0.22	0.02	2.49	0.23	0.02	1.83	0.29	0.03	2.78	0.24	0.01	2.26	0.29	0.02
	25	2.18	0.33	–	0.57	0.27	–	1.76	0.35	–	1.62	0.35	–	0.52	0.30	–	1.71	0.35	–
water (527 pts)	400	–	0.53	0.07	0.62	0.28	0.11	0.58	0.38	0.20	–	0.29	0.04	0.92	0.24	0.11	0.87	0.37	0.13
	300	–	0.20	0.13	1.52	0.16	0.18	1.76	0.27	0.28	–	0.26	0.10	1.52	0.42	0.18	1.78	0.45	0.24
	200	–	0.31	0.20	2.16	0.59	0.26	2.39	0.74	0.39	–	0.24	0.20	3.04	0.53	0.26	1.71	0.76	0.40
	100	–	0.75	0.23	1.78	1.09	0.28	7.87	1.39	0.43	–	0.90	0.22	1.07	1.22	0.28	7.15	1.42	0.40
	50	–	0.91	–	8.19	1.22	–	15.91	1.56	–	–	0.89	–	7.59	1.24	–	15.59	1.55	–
twoMoons (2000 pts)	1500	0.00	0.01		0.17	0.01		0.00	0.00		0.01	0.01		0.19	0.00		0.00	0.00	
	1000	0.01	0.01		1.75	0.01		0.00	0.01		0.03	0.01		1.72	0.00		0.00	0.00	
	500	0.40	0.02		2.99	0.05		0.00	0.02		0.35	0.03		3.06	0.03		0.02	0.01	
	250	1.73	0.03		1.73	0.11		0.29	0.08		1.96	0.02		1.77	0.09		0.29	0.05	
	100	1.99	0.04		0.45	0.13		1.72	0.10		1.96	0.05		0.42	0.15		1.64	0.11	
	50	1.44	0.09		0.14	0.20		1.50	0.15		1.42	0.07		0.15	0.18		1.48	0.13	
twoCircles (2000 pts)	1500	0.00	0.00		0.02	0.00		0.00	0.00		0.00	0.00		0.02	0.00		0.00	0.00	
	1000	0.02	0.00		0.20	0.00		0.00	0.00		0.01	0.00		0.19	0.00		0.00	0.00	
	500	0.20	0.00		0.99	0.00		0.00	0.00		0.18	0.00		1.00	0.00		0.00	0.00	
	250	1.32	0.01		1.02	0.01		0.00	0.01		1.34	0.01		1.02	0.01		0.00	0.01	
	100	1.51	0.03		1.00	0.03		0.85	0.03		1.51	0.03		1.00	0.03		0.85	0.03	
	50	1.47	0.06		0.96	0.06		1.48	0.06		1.46	0.06		0.96	0.06		1.47	0.06	

TABLE II

HKDs OF "FILTERED" PERSISTENCE INTERVALS TO OBSERVE THE ABILITY TO PRESERVE "SIGNIFICANT TOPOLOGICAL FEATURES". ENTRIES MARKED "–" ARE WHERE THE REDUCED DATA LOST ALL THE FEATURES IN THAT DIMENSION. BLANK ENTRIES ARE POINT CLOUDS WITH NO TOPOLOGICAL FEATURES IN THAT DIMENSION.

partitioning results. Algorithm complexity can play a large role in determining what partitioning algorithm is suitable to an application. In general, k-means++ and Agglomerative Ward tend to consistently provide better overall data reduction results. While Agglomerative Single Link provides comparable results, at larger reductions it tended to lose more persistence intervals in the higher dimensions than k-means++ and Agglomerative Ward. Random sampling and the Maxmin [30] sampling sometimes provide good results, but they tended to be unpredictable and sometimes produce wildly inaccurate results. An additional benefit from the use of clustering is that data around a sampled point can be restored to support the concept of upscaling [31] to restore more accurate persistence intervals from the reduced point cloud.

Big data continues to be on the horizon for persistent homology as techniques for data reduction, simplicial complex collapses and optimizations, and boundary matrix reduction continue to increase performance. Approximations in the hundreds of thousands of points should be possible with a well designed partitioning and upscaling library, even those in higher dimensions. Bringing TDA to big data analysis will provide automated tools for analyzing the connectivity of point clouds beyond current applications and should be continued to be explored in all domains.

REFERENCES

[1] F. Chazal and B. Michel, "An introduction to topological data analysis: Fundamental and practical aspects for data scientists," Oct. 2017.

[2] R. Ghrist, "Barcodes: The persistent topology of data," *Bulletin of the American Mathematical Society*, vol. 45, no. 1, pp. 61–75, 2008.

[3] P. Y. Lum, G. Singh, A. Lehman, T. Ishkanov, M. Vejdemo-Johansson, M. Alagappan, J. Carlsson, and G. Carlsson, "Extracting insights from the shape of complex data using topology," *Scientific Reports*, vol. 3, Feb. 2013.

[4] G. Singh, F. Memoli, and G. Carlsson, "Topological methods for the analysis of high dimensional data sets and 3D object recognition," in *Eurographics Symposium on Point-Based Graphics*, M. Botsch, R. Pajarola, B. Chen, and M. Zwicker, Eds. The Eurographics Association, 2007, pp. 91–100.

[5] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington, "A roadmap for the computation of persistent homology," *EPJ Data Science*, vol. 6, no. 1, Aug. 2017.

[6] R. Ghrist, *Elementary Applied Topology*. Createspace, 2014.

[7] G. Petri, M. Scolamiero, I. Donato, and F. Vaccarino, "Topological strata of weighted complex networks," *PLOS ONE*, vol. 8, no. 6, pp. 1–8, Jun. 2013.

[8] D. Horak, S. Maletić, and M. Rajković, "Persistent homology of complex networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2009, no. 3, p. P03034, Mar. 2009.

[9] M. Hajij, B. Wang, C. E. Scheidegger, and P. Rosen, "Visual detection of structural changes in time-varying graphs using persistent homology," in *IEEE Pacific Visualization Symposium*, ser. PacificVis 2018. USA: IEEE Computer Society, Apr. 2018, pp. 125–134.

[10] P. Bendich, J. S. Marron, E. Miller, A. Pieloch, and S. Skwerer, "Persistent homology analysis of brain artery trees," *The Annals of Applied Statistics*, vol. 10, no. 1, pp. 198–218, Mar. 2016.

[11] T. Kaczynski, K. Mischaikow, and M. Mrozek, *Computational Homology*, ser. Applied Mathematical Sciences. New York: Springer-Verlag, 2004, vol. 157.

[12] H. Wagner, C. Chen, and E. Vuçini, "Efficient computation of persistent homology for cubical data," in *Topological Methods in Data Analysis and Visualization II: Theory, Algorithms, and Applications*, R. Peikert, H. Hauser, H. Carr, and R. Fuchs, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 91–106.

[13] P. Bendich, H. Edelsbrunner, and M. Kerber, "Computing robustness and persistence for images," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1251–1260, Nov. 2010.

[14] K. Mischaikow and V. Nanda, "Morse theory for filtrations and efficient computation of persistent homology," *Discrete & Computational Geometry*, vol. 50, no. 2, pp. 330–353, 2013.

[15] V. Nanda, "Discrete morse theory for filtrations," Ph.D. dissertation, Department of Mathematics, Rutgers University, Oct. 2012. [Online]. Available: http://people.maths.ox.ac.uk/nanda/source/Thesis.pdf

[16] G. Carlsson, T. Ishkhanov, V. de Silva, and A. Zomorodian, "On the local behavior of spaces of natural images," *International Journal of Computer Vision*, vol. 76, no. 1, pp. 1–12, Jan. 2008.

[17] Z. Cang, L. Mu, K. Wu, K. Opron, K. Xia, and G.-W. Wei, "A topological approach for protein classification," *Molecular Based Mathematical Biology*, vol. 3, no. 1, pp. 140–162, Nov. 2015.

[18] T. K. Dey and S. Mandal, "Protein classification with improved topological data analysis," in *18th International Workshop on Algorithms in Bioinformatics*, ser. WABI 2018, vol. 113. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Aug. 2019, pp. 6:1–6:13.

[19] K. Xia and G.-W. Wei, "Persistent homology analysis of protein structure, flexibility, and folding," *Int Journal for Numerical Methods in Biomedical Engineering*, vol. 30, no. 8, pp. 814–844, 2014.

[20] P. G. Camara, D. I. S. Rosenbloom, K. J. Emmett, A. J. Levine, and R. Rabadan, "Topological data analysis generates high-resolution, genome-wide maps of human recombination," *Cell systems*, vol. 3, no. 1, pp. 83–94, 2016.

[21] J. M. Chan, G. Carlsson, and R. Rabadan, "Topology of viral evolution," *Proceedings of the National Academy of Sciences*, vol. 110, no. 46, pp. 18 566–18 571, 2013.

[22] A. Moitra, N. O. Malott, and P. A. Wilsey, "Persistent homology on streaming data," in *2020 International Conference on Data Mining Workshops (ICDMW)*, ser. ICDMW '20. USA: IEEE, Nov. 2020, pp. 636–643.

[23] M. Gao, C. Chen, S. Zhang, Z. Qian, D. Metaxas, and L. Axel, "Segmenting the papillary muscles and the trabeculae from high resolution cardiac CT through restoration of topological handles," in *International Conference on Information Processing in Medical Imaging*, ser. Lecture Notes in Computer Science, vol. 7917. Heidelberg, Berlin: Springer, 2013, pp. 184–195.

[24] F. Belchi, M. Pirashvili, J. Conway, M. Bennett, R. Djukanovic, and J. Brodzki, "Lung topology characteristics in patients with chronic obstructive pulmonary disease," *Scientific reports*, vol. 8, no. 1, March 2018.

[25] U. Bauer. (2018) Ripser. The Technical University of Munich. [Online]. Available: http://www.cs.umd.edu/ mount/ANN/

[26] C. Maria, J.-D. Boissonnat, M. Glisse, and M. Yvinec, "The gudhi library: Simplicial complexes and persistent homology," INRA, Tech. Rep. RR-8548, 2014. [Online]. Available: https://hal.inria.fr/hal-01005601v2

[27] G. Henselman and R. Ghrist, "Matroid filtrations and computational persistent homology," 2016.

[28] F. Chazal, B. T. Fasy, F. Lecci, B. Michel, A. Rinaldo, and L. Wasserman, "Subsampling methods for persistent homology," in *International Conference on Machine Learning*, ser. ICML 2015, Lille, France, Jul. 2015.

[29] A. Moitra, N. Malott, and P. A. Wilsey, "Cluster-based data reduction for persistent homology," in *2018 IEEE International Conference on Big Data*, ser. Big Data 2018, Dec. 2018, pp. 327–334.

[30] V. de Silva and G. Carlsson, "Topological estimation using witness complexes," in *Eurographics Symposium on Point-Based Graphics*, ser. SPBG '04, M. Gross, H. Pfister, M. Alexa, and S. Rusinkiewicz, Eds. Goslar, DEU: The Eurographics Association, 2004, pp. 157–166.

[31] N. O. Malott and P. A. Wilsey, "Fast computation of persistent homology with data reduction and data partitioning," in *2019 IEEE International Conference on Big Data*, ser. Big Data 2019, Dec. 2019, pp. 880–889.

[32] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '07. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[33] D. Müllner, "Modern hierarchical, agglomerative clustering algorithms," *arXiv preprint arXiv:1109.2378*, Sep. 2011. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2011arXiv1109.2378M

[34] F. Murtagh and P. Legendre, "Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion?" *Journal of Classification*, vol. 31, no. 3, pp. 274–295, 2014.

[35] G. Carlsson and F. Memoli, "Classifying clustering schemes," 2010.

[36] A. Zomorodian, "Fast construction of the vietoris–rips complex," *Computer and Graphics*, vol. 34, pp. 263–271, Jun. 2010.

[37] H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, and L. Ziegelmeier, "Persistence images: A stable vector representation of persistent homology," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 218–252, Jan. 2017.

[38] P. Niyogi, S. Smale, and S. Weinberger, "A topological view of unsupervised learning from noisy data," *SIAM Journal on Computing*, vol. 40, no. 3, pp. 646–663, 2011.

[39] G. Carlsson, "Topology and data," *Bulletin of the American Mathematical Society*, vol. 46, no. 3, pp. 255–308, Apr. 2009.

[40] D. I. Spivak, "Category theory for scientists (old version)," 2013.

[41] S. M. Lane, *Categories for the Working Mathematician*, 2nd ed., ser. Graduate Texts in Mathematics. Springer-Verlag, 1998, vol. 5.

[42] D. R. Sheehy, "Linear-size approximations to the vietoris–rips filtration," *Discrete & Computational Geometry*, vol. 49, no. 4, pp. 778–796, Jun. 2013.

[43] T. K. Dey, D. Shi, and Y. Wang, "Simba: An efficient tool for approximating rips-filtration persistence via simplicial batch-collapse," in *24th Annual European Symposium on Algorithms (ESA 2016)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), P. Sankowski and C. Zaroliagis, Eds., vol. 57. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, pp. 35:1–35:16.

[44] T. K. Dey, F. Fan, and Y. Wang, "Computing topological persistence for simplicial maps," in *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*, ser. SOCG'14. New York, NY, USA: ACM, Jun. 2014, pp. 345–354.

[45] B. Brehm and H. Hardering, "Sparips," 2018. [Online]. Available: https://arxiv.org/abs/1807.09982

[46] A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," in *Proceedings of the 23rd international conference on Machine learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, pp. 97–104.

[47] W. B. Johnson and J. Lindenstrauss, "Extensions of lipschitz mappings into a hilbert space," *Contemporary Mathematics*, vol. 26, pp. 189–206, 1984.

[48] D. R. Sheehy, "The persistent homology of distance functions under random projection," in *Proceedings of the Thirtieth Annual Symposium on Computational Geometry*, ser. SOCG'14. New York, NY, USA: ACM, 2014, pp. 328–334.

[49] K. N. Ramamurthy, K. R. Varshney, and J. J. Thiagarajan, "Computing persistent homology under random projection," in *IEEE Workshop on Statistical Signal Processing*, Jun. 2014, pp. 105–108.

[50] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density–based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, Aug. 1996, pp. 226–231.

[51] L. McInnes and J. Healy, "Accelerated hierarchical density based clustering," *2017 IEEE International Conference on Data Mining Workshops*, pp. 33–42, 2017.

[52] F. Chazal, B. Fasy, F. Lecci, B. Michel, A. Rinaldo, A. Rinaldo, and L. Wasserman, "Robust topological inference: Distance to a measure and kernel distance," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 5845–5884, 2017.

[53] J. M. Phillips, B. Wang, and Y. Zheng, "Geometric inference on kernel density estimates," *arXiv preprint arXiv:1307.7760*, pp. 1–18, 2013.

[54] R. W. Sumner and J. Popovic, "Mesh data from deformation transfer for triangle meshes," 2004. [Online]. Available: https://people.csail.mit.edu/sumner/research/deftransfer/data.html

[55] V. Satopa, J. Albrecht, D. Irwin, and B. Raghavan, “Finding a ‘kneedle’ in a haystack: Detecting knee points in system behavior,” in *31st International Conference on Distributed Computing Systems Workshops*, ser. ICDCSW. IEEE Computer Society, 2011, pp. 166–171.