JOURNAL OF LEARNING ANALYTICS
SOLAR SOCIETY for LEARNING ANALYTICS RESEARCH
(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

A User's Guide to Topological Data Analysis

Elizabeth Munch
Department of Mathematics and Statistics
University at Albany – SUNY, Albany, NY, USA
emunch@albany.edu

ABSTRACT. Topological data analysis (TDA) is a collection of powerful tools that can quantify shape and structure in data in order to answer questions from the data's domain. This is done by representing some aspect of the structure of the data in a simplified topological signature. In this article, we introduce two of the most commonly used topological signatures. First, the persistence diagram represents loops and holes in the space by considering connectivity of the data points for a continuum of values rather than a single fixed value. The second topological signature, the mapper graph, returns a 1-dimensional structure representing the shape of the data, and is particularly good for exploration and visualization of the data. While these techniques are based on very sophisticated mathematics, the current ubiquity of available software means that these tools are more accessible than ever to be applied to data by researchers in education and learning, as well as all domain scientists.

Keywords: Topological data analysis, persistent homology, mapper

1 INTRODUCTION

With the introduction of sensors in everything and online systems with click by click data on all user activity, data science now touches nearly every field of study. However, the traditional techniques of data analysis have not always kept up with the exploding quantity and complexity of data since they often rely on overly simplistic assumptions. The field of topological data analysis (TDA) has attempted to fill this void by producing a collection of techniques stemming from the idea that data has shape that can be rigorously quantified in order to investigate data. This quantification takes the form of a topological signature: a representation of some aspect of the shape in a simplified form for study. As with any summary, producing a topological signature from data is a lossy process; that is some information will be lost during the creation of the summary. However, the art of using TDA is to put the data in a form that both fits into the standard TDA pipelines, and where the lossy-ness of the method serves to remove high-dimensionality rather than important structure. The user also needs to understand what features of the data can be found with (and which are ignored by) the methods available in order to choose the right hammer for the nail of interest.

In this spirit, this article will serve as an introduction to the standard TDA methods for domain scientists interested in using these tools. This will focus on how to view real data in the TDA context in order to use the available open source software. In particular, we will investigate two commonly used topological signatures from TDA. The first is the persistence diagram in Section 3. This topological signature gives a

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

concise description of the structure of the data by only keeping track of loops in the space. Second, we will investigate the mapper graph in Section 4 which provides a graph representing the underlying structure of the data.

2 DATA WITH DISTANCE

The first hurdle to clear is to have a universal understanding for what we mean by the word "data" as this term means many different things in many different domains. Here, we will take data to mean a collection of data points with each arising from, say, different people, different students, different times, and so on.

Much of TDA is based around the notion that there is an idea of proximity between these data points. So, for example, if each data point $x = \{x_1, \cdots, x_n\}$ consists of n numerical values, we have an easy definition of proximity that comes from the standard Euclidean distance: this is the generalization of the standard distance in the plane $d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$. Euclidean distance gives a good intuitive starting place for the requirements of a generalized distance in the mathematical sense. A distance[1] is a function on two inputs $d(x,y)$ which satisfies the commonly used properties of the Euclidean distance, namely:

- (Positivity) $d(x,y) \geq 0$, and $d(x,y) = 0$ if and only if $x = y$. The distance between two points is a positive number, and the distance is 0 if and only if the points are the same.
- (Symmetry) $d(x,y) = d(y,x)$. The distance from one point to another is the same as going in the opposite direction.
- (Triangle inequality) $d(x,z) \leq d(x,y) + d(y,z)$. The distance between two points x and z is always no longer than taking a detour through point y.

When we have a collection of data points with a definition of a distance, we often refer to this collection as a point cloud. See, for example, the black dots of Figure 1, which inherit a distance from being inside of the plane.

Note that the Euclidean distance assumes that the inputs are numeric. There are, of course, other distances that can be defined on numeric data as well (i.e., the Minkowski distance); however, having entirely numeric data is not a requirement to define a generalized distance. One can also define many different distances when the data is categorical rather than numeric. This might simply be done by looking at matches (define the distance between data points by the number of entries which are the same), or by including a more nuanced view of the categorical entries. See Boriah, Chandola, and Kumar (2008) for an overview of metrics on categorical data.

[1] NB: In this paper, the terms distance and metric are used interchangeably.

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

3 PERSISTENT HOMOLOGY

The first topological signature comes from persistent homology, a powerful tool in TDA for investigating the structure of data (Edelsbrunner, Letscher, & Zomorodian, 2002; Zomorodian & Carlsson, 2004). The persistence diagram can show a great deal of information about a given point cloud such as clustering without an expert-chosen connectivity parameter, which is usually necessary. It can also describe more complicated structure such as loops and voids that are not visible with other methods. Persistent homology has found success in the investigation of data from many different domains; these include image processing (Carlsson, Ishkhanov, de Silva, & Zomorodian, 2008; Perea & Carlsson, 2014; Adcock, Carlsson, & Carlsson, 2016), time series analysis (Perea, Deckard, Haase, & Harer, 2015; Khasawneh & Munch, 2016; Emrani, Gentimis, & Krim, 2014), phylogenetics (Chan, Carlsson, & Rabadan, 2013), neuroscience (Giusti, Pastalkova, Curto, & Itskov, 2015; Dabaghian, Mémoli, Frank, & Carlsson, 2012), and sensor networks (de Silva & Ghrist, 2007; Adams & Carlsson, 2015; Munch, Shapiro, & Harer, 2012).

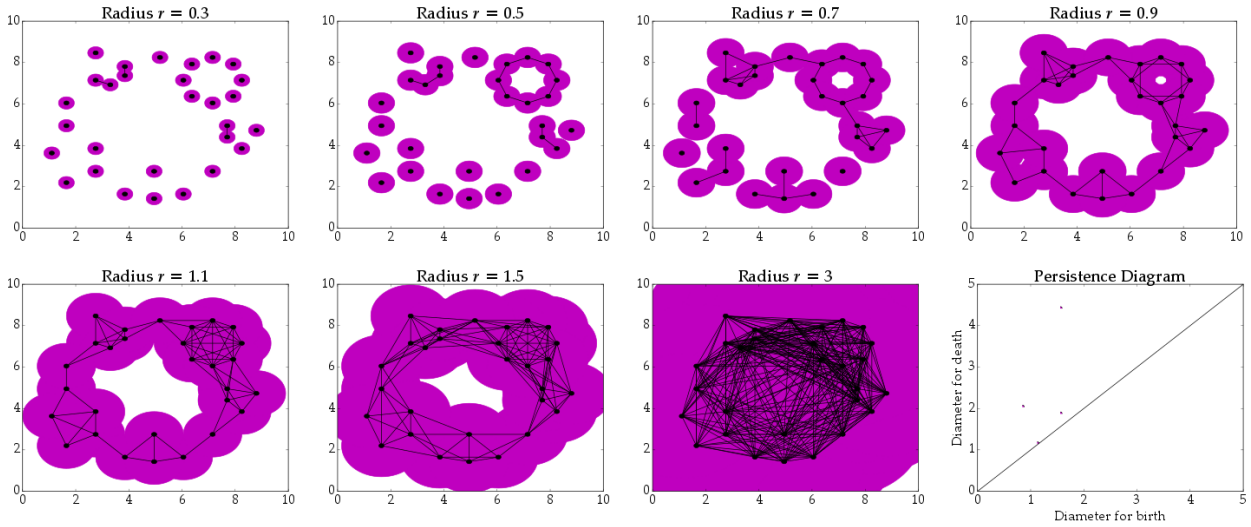

Figure 1: An example using persistent homology to investigate a point cloud data set by constructing the Rips complex, whose edge set is drawn in black on each figure. The Rips complex includes any higher dimensional simplex if all the edges are present, so these are not explicitly drawn. The persistence diagram drawn at the bottom right gives a summary of the appearance and disappearance of loops in the space as the Rips complex parameter changes.

3.1 Simplicial Complexes

The main goal of TDA is to investigate the intrinsic shape of the data using a provided distance. However, the data as provided is nothing more than a collection of individual points, often with too many coordinates each to be fully visualizable. For instance, the point cloud in Figure 1 seems to be sampled from some sort of circular structure, but how can that structure be found or represented, particularly if the data came with, say, 73 coordinates instead of 2 as drawn? Thus, we need a structure that can be used as a proxy for the shape during our investigations.

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

Graphs are a commonly used structure in many data analysis applications since they can store relationships between data points. In a way, graphs encode a 1-dimensional skeleton of the data. That is, the vertices can be thought of as 0-dimensional objects, and the edges as 1-dimensional objects. But, like a skeleton, there are higher dimensional relationships that are lost when we can only see the skeleton. Think of the human forearm: if we could only see the radius and ulna bones, we would think the human arm had a gaping hole. When we can pass to understanding not only the human skeleton but the muscle, too, we realize that arm-hole is filled in and so it is not an inherent part of the body's structure. Simplicial complexes generalize the notion of graphs by allowing for 2-, 3-, and higher dimensional building blocks, called simplices.

This can be seen in the following way. Let us start building a simplicial complex with each data point representing a vertex. A vertex, or "0-dimensional simplex," consists of, obviously enough, one vertex. An edge, or "1-dimensional simplex," is defined by its two endpoint vertices. A 2-dimensional simplex is a triangle, given by its three vertices, and so on. Generally, a d-dimensional simplex is defined by $d + 1$ vertices.

A face of a simplex is one defined by a subset of its vertex set. So, the faces of an edge (given by 2 vertices) are the two endpoint vertices (and technically the edge itself). The faces of a 2-simplex (triangle given by 3 vertices) are the three vertices, the three edges, and the triangle itself. A collection of simplices is called a simplicial complex if all faces of any simplex in the collection are also in the collection. For example, a 2-simplex cannot be in the simplicial complex unless all its edges are also. Figure 2 gives an example of a simplicial complex which has 0-, 1-, and 2-dimensional simplices.

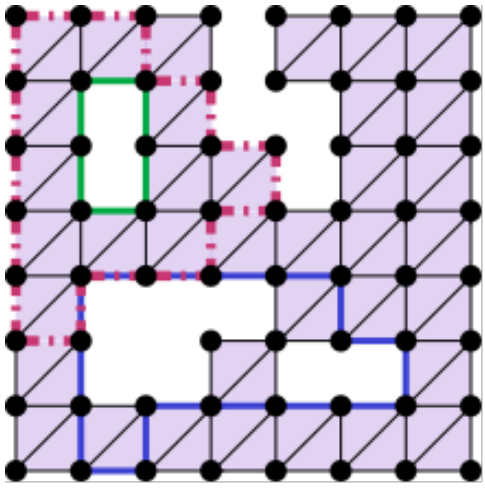

Figure 2: An example of a simplicial complex. The green, pink, and blue collections represent 1-dimensional homology classes, where the pink and green classes are said to be equivalent since they encompass the same hole. The rank of the 1-dimensional homology (that is, the 1st Betti number β_1) is 3 due to the three holes in the space.

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

3.2 The Rips Complex

The next task is to build a useful simplicial complex representing the structure of the data and which uses the original data as the vertex set. Assume we have decided on a metric for the data points and pick a number $t \geq 0$ to start. The Vietoris-Rips complex (sometimes called Rips complex) for parameter t is constructed in the following way. The vertex set is given by the data set itself. For each pair of points x, y in the data set, we include the edge xy if the distance between them is at most t: $d(x, y) \leq t$. For a higher dimensional simplex given by vertices $x_0, \cdots, x_d$, we include the simplex if the complex has all possible edges; explicitly, this means that every vertex $x_0, \cdots, x_d$ is within distance t of every other vertex in the simplex.

The Rips complex is shown in Figure 1 for several different choices of t. Note that saying that two points are within distance t of each other is the same as saying that the pair of disks of radius $r = t/2$ centred at each point touch. So in the figures, we have an edge whenever two of the disks intersect, and we assume that we have triangles and higher dimensional simplices whenever possible although these are not explicitly drawn. The data of this example appears to have come from a circular structure. We can see that there is a range of parameter values for which the Rips complex has this circular structure, namely from approximately $t = 1.8$ (radius $= .9$) to $t = 4.2$ (radius $= 2.1$). The question remains, however, how to do a good job of choosing the t parameter so that the Rips complex reflects the structure of the underlying data set. It is exactly this question that leads us use the persistence diagram as a topological signature of the data.

3.3 What Can Persistence Find?

As we have seen, the Rips complex is particularly useful for seeing structure in the data as long as the connectivity parameter t is chosen well. But, how can we do a good job of choosing t? The answer is to not choose t, but instead to look at the continuum of possible t values looking for ranges that seem to represent something interesting in terms of structure. There are two pieces to the idea of persistent homology. Homology is a tool from classical algebraic topology that can measure certain structures of a simplicial complex. The "persistent" part comes from looking at all possible t values to see where structure appears and disappears; that is, the collection of t values for which the structure persists.

Homology is divided into different dimensions representing the dimension of the structure being measured; see Figure 3 for information on homology for some commonly studied topological spaces. Here, 0-dimensional homology measures clusters; 1-dimensional homology measures loops; and 2-dimensional homology measures voids (air bubbles). There are, of course, definitions of k-dimensional homology for higher k, but they are beyond the scope of this article. We lose the ability to intuitively visualize the structures we are capturing for higher k, thus these have not yet been commonly used in applications. Figure 3 gives the so-called Betti number β_k for different example spaces. The Betti number is the rank of the k-dimensional homology group; in particular, it counts the number of structures seen in that dimension. For the purposes of concreteness, we will discuss 1-dimensional homology and persistent homology here, and defer to Edelsbrunner and Harer (2010), Hatcher (2002),

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

and Munkres (1993) for more rigorous and thorough introductions to general homology and persistent homology.

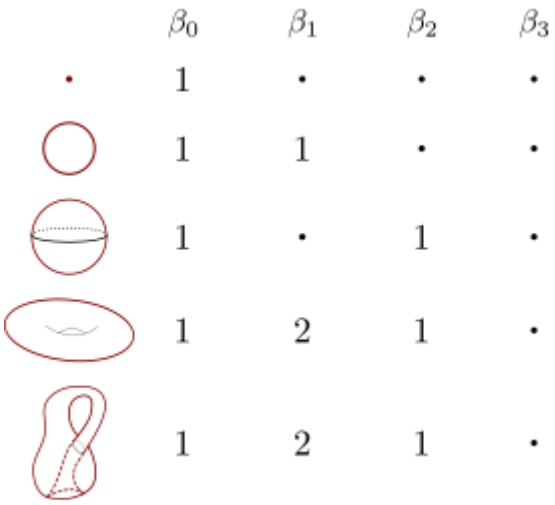

Figure 3: Betti numbers β_k for different topological spaces: a point, a circle, a sphere, a torus (donut), and a Klein bottle. The k^{th} Betti number gives the rank of the k-dimensional homology group, thus measures different properties of the space in each dimension. For example, $k = 0$ measures connectivity, $k = 1$ measures loops, and $k = 2$ measures voids.

Structures in homology are given by "classes." A class in 1-dimensional homology is represented by a collection of 1-simplices (edges) that have an even number of edges touching each vertex.[2] For example, a collection of edges that form a closed loop, as in the example of Figure 2, satisfies this requirement. The reason for using the word "represented" is that different closed loops can represent the same class. Essentially, for two loops to represent the same class, they must encircle the same hole, such as the dashed pink and solid green loops in Figure 2. Working with homology means that rather than studying the incredibly large collection of all such loops, we can divide them into groups where all elements of a group represent the same structure in the space.

While homology measures the structure of a single, stagnant space, persistent homology watches how this structure changes as the space changes. Consider the example in Figure 1. In this case, we have a point cloud and can build the Rips complex for several different choices of the parameter t. As t grows, more and more edges and higher dimensional simplices are added. So, we can choose a representative for a 1-dimensional homology class at one t, and see if it still represents a class at a larger t. The t for which a class is first seen is called the birth diameter, and the t for which a class is no longer different from the previously seen classes is called the death diameter. In the example of Figure 1, we see a 1-dimensional class (a loop) is born when the radius gets to $r = 0.5$ ($t = 1$); however, this fills in by the

[2] Note for experts: we compute homology using $\mathbb{Z}_2$ coefficients.

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

time $r = 1$ ($t = 2$). On the other hand, there is a large loop born when $r = 0.9$ ($t = 1.8$), which does not die until $r = 2.1$ ($t = 4.2$).

For each class, we have a pair of numbers (a, b) for the birth diameter and death diameter. These pairs of numbers are drawn as a point in the persistence diagram. For example, there are points at $(1,2)$ and $(1.8,4.2)$ representing those two classes discussed in the previous paragraph. Note that if a class dies very soon after it is born (i.e., if it has a short lifetime), it will be represented by a point close to the diagonal. If the class has a long lifetime, then it will be represented as a point far from the diagonal. In many applications, the existence of a few points far from the diagonal relative to the rest of the points in the persistence diagram can be taken to mean that these classes come from the inherent structure of the data, while the rest of the points are artifacts of noise.

Of great use in the case of data with noise is the existence of a distance on the persistence diagrams themselves. This way, we have a computable measure of just how similar two persistence diagrams are. Take, for example, the two point clouds in the left of Figure 4. The red circle point cloud is a noisy version of the black square point cloud. The persistence diagrams for the two point clouds are drawn overlaid in the right of Figure 4. While the red diagram has quite a few more points due to noise, both diagrams have a single point far from the diagonal, thus they represent similar underlying structures.

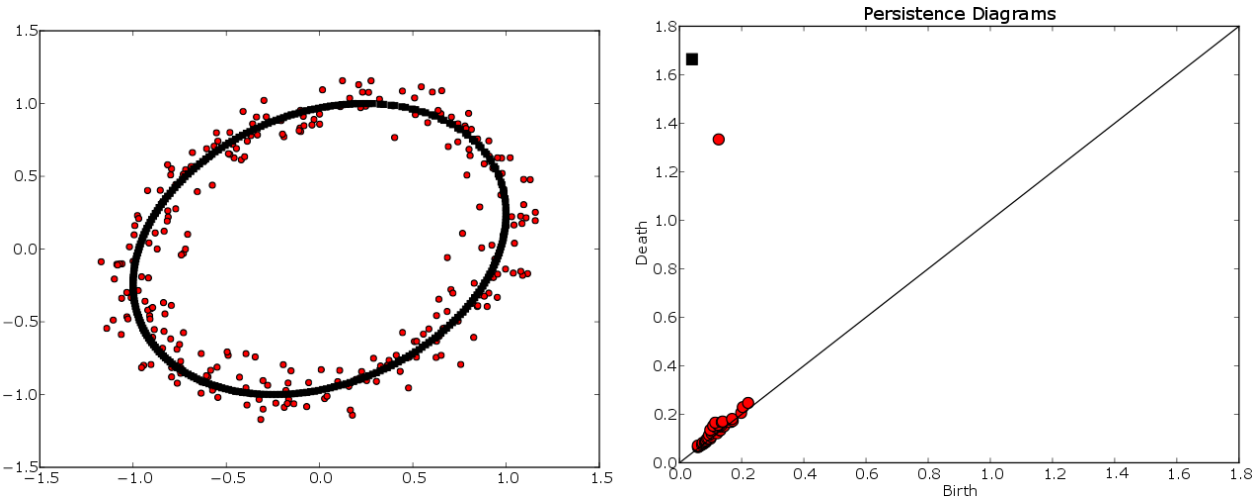

Figure 4: Two example point clouds are overlaid at left, and their persistence diagrams are overlaid at right. Notice that the point clouds are close in some sense. The fact that the persistence diagrams are also close is a result of the stability theorem for persistence.

Two metrics are commonly used to measure the similarity of these objects: the bottleneck and Wasserstein distances. Each works by matching points of one diagram with points of another diagram while allowing the match to be done with the diagonal if necessary. Bottleneck distance is the maximum distance between any pair of points, and thus gives a measure for the most work that must be done to push one diagram into the configuration of the other. Wasserstein distance sums powers of the distances between the pairs; unlike the bottleneck distance, it takes all of the points into account,

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

including the noisy diagonal points. The former is better for a simple test of proximity of diagrams; the later is better when the noisy classes on the diagonal hold useful information about the data.

Arguably the most important theorem for justifying the use of persistent homology as a tool for data analysis is the stability theorem (Cohen-Steiner, Edelsbrunner, & Harer, 2007). It says that the distance between two persistence diagrams (using bottleneck distance) cannot be larger than the distance between the two data sets (using the so-called Hasudorff distance) used to obtain them. When we view this result in the context of data, it says that even if we are given a data set infected by a little noise, the persistence diagram obtained from this data is approximately correct because it is close to the diagram we would have from the noise-free data.

3.4 Further Reading

This idea of understanding persistent homology with respect to noisy data has led to a great deal of work combining persistence with statistical methods. Many of the standard statistics techniques do not immediately apply, however, because unlike the standard literature, the statistic used to represent the data is not a real number, but a much more complicated object: the persistence diagram. Several methods have been proposed for looking at the mean of a collection of diagrams, including the Fréchet mean (Turner, Mileyko, Mukherjee, & Harer, 2014; Munch et al., 2015) and the persistence landscape (Bubenik, 2015). There are confidence intervals for persistence diagrams (Fasy et al., 2014b), as well as intersections with the machine learning framework (Niyogi, Smale, & Weinberger, 2011).

It is important to note that the pipeline described above (point cloud to Rips complex to persistence diagram) is not the only way to use persistent homology to study data. One particularly useful version looks at the so-called sublevelset filtration of an image, which builds up the simplicial complexes using the values at the pixels rather than starting with a collection of points (Kaczynski, Mischaikow, & Mrozek, 2006; Robins, Wood, & Sheppard, 2011). There are also many variants of persistent homology that may be of interest depending on the application. These include persistent cohomology (de Silva, Morozov, & Vejdemo-Johansson, 2011), zigzag persistence (de Silva et al., 2011), and multidimensional persistence (Carlsson & Zomorodian, 2009).

3.5 Available Software

There is an extensive library of open source software available for computation of persistence. Dionysus (Morozov, 2015) is one of the most versatile packages available, with functionality for not only persistent homology, but also persistent cohomology, zigzag persistence, and the bottleneck and Wasserstein distances. Kerber, Morozov, and Nigmetov (2016) also have software for computing the bottleneck and Wasserstein distances. Perseus (Nanda, 2015; Mischaikow & Nanda, 2013) utilizes discrete Morse theory to speed up persistence computation, and provides easy access to persistence for point clouds as well as for images. There is also a TDA package for R that additionally brings in the newly developed statistical methodology (Fasy, Kim, Lecci, & Maria, 2014a). The recently released Ripser (Bauer, 2016) does persistence for point clouds. Rivet provides a first step towards visualizing and

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

understanding multidimensional persistence (Lesnick & Wright, 2015, 2016). Readers looking for an easy way to test out some of these ideas and computations should visit the website (Tralie, 2016), which does computation of persistent homology in the browser for example point clouds. This is not by any means an exhaustive list; see Otter, Porter, Tillmann, Grindrod, and Harrington (2015) for a good survey on software packages.

4 MAPPER

Another very powerful signature arising from TDA is mapper (Singh, Mémoli, & Carlsson, 2007). The idea is to represent the 1-dimensional structure of a data set using a graph. Unlike the persistence diagram, data points have associated locations to points in the mapper graph. Since these graphs are much easier to visualize than the possibly high dimensional data used to construct it, mapper is an excellent tool for investigation and visualization of the structure of a data set. It has been used extensively in data analysis, particularly in the biology and health domains (Nicolau, Levine, & Carlsson, 2011; Li et al., 2015; Torres et al., 2016; Nielson et al., 2015; Yao et al., 2009).

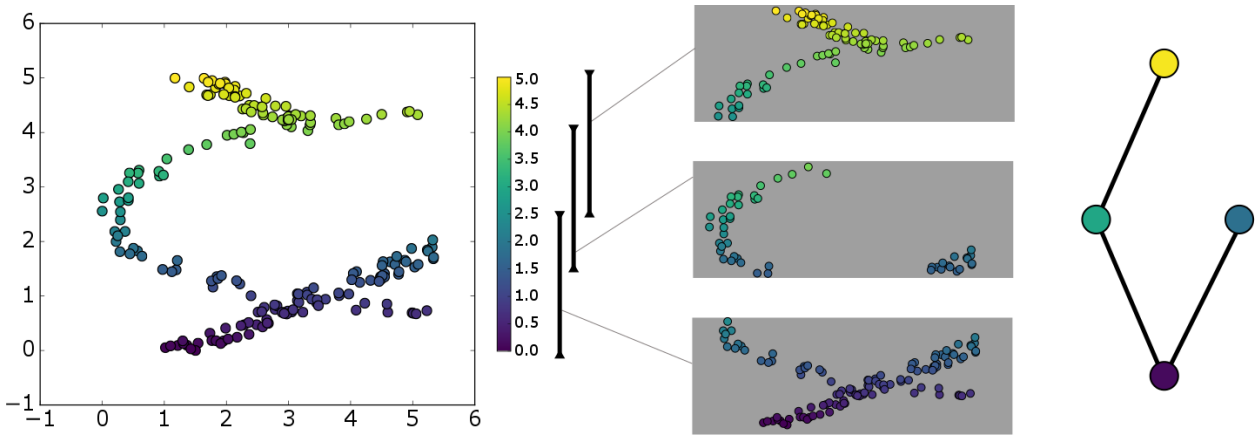

Figure 5: A small example of mapper. The filter function for the given data is given by the y-coordinate. A cover is chosen for the function values, and then the point cloud is clustered within each portion defined by the cover. The resulting graph at right is the mapper graph for the chosen parameters.

4.1 What Can It Find?

Like with the persistence diagrams, we start with a point cloud and a choice of distance on the points. However, unlike the persistence diagram, we have a few more choices to make along the way in order to construct the mapper graph.

The main additional piece of data required for the mapper graph construction is called a filter function. This filter function is simply an assignment of a real number for each data point and is used to spread

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

out the data. Some examples include using one or a combination of the coordinates for each point (as in Figure 5), eccentricity (as in Figure 6), or density.

Then, we decide on a cover of the filter function values. A cover for the interval $[a, b]$ is a collection of overlapping sets such that each number between a and b is included in at least one set; see the black bars in Figure 5 for an example of a cover with three sets. Finally, we consider the subset of points with function values in a single set from the cover, and then cluster the points. Each cluster becomes a node in the mapper graph as in the far right of Figure 5. Edges are included based on the intersection information from overlapping sets in the cover.

An example of the mapper graph can be seen in the right of Figure 6. The important thing to note about the mapper graph is that each node represents a subset of the data points. This makes it a particularly useful signature for data investigation since it can separate data points with different properties even if standard clustering cannot differentiate them. For instance, the example of Figure 6 would be seen as a single cluster, but the mapper graph allows us to differentiate between points on the circle and points on the different flares.

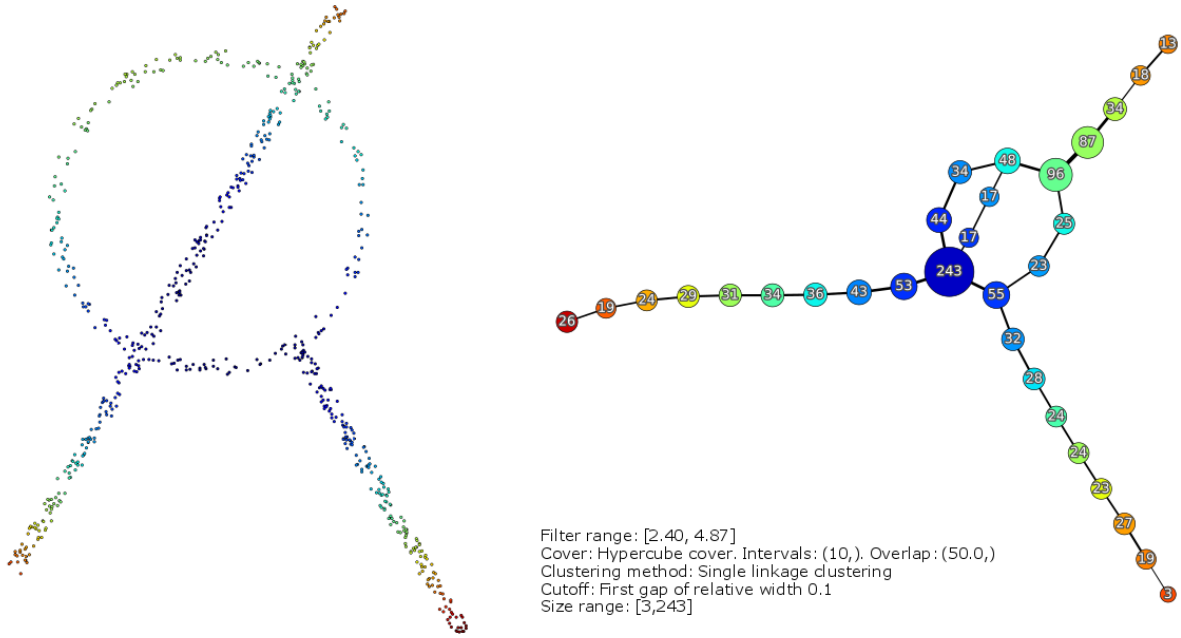

Figure 6: An example point cloud with 700 points is drawn at top left. The eccentricity filter function is shown by the colouring of the points. The mapper graph for this example is shown at the top right. Note how the structure of the mapper graph reflects the intrinsic structure of the point cloud. This example was computed using the Python Mapper code (Müllner & Babu, 2013).

4.2 Further Reading

Mapper can be thought of as an approximation of the Reeb graph (Biasotti, Giorgi, Spagnuolo, & Falcidieno, 2008; Munch & Wang, 2016; de Silva, Munch, & Patel, 2016), another commonly used tool in TDA. Unlike mapper, the Reeb graph starts not with a dataset of points, but instead a full space (for

(2017). A user's guide to topological data analysis. *Journal of Learning Analytics*, *4*(2), 47–61. http://dx.doi.org/10.18608/jla.2017.42.6

example a simplicial complex) with a filter function. However, like mapper, it then studies the clusters over the function values without use of a choice of cover. For this reason, it has been used largely for 3D graphics applications such as data skeletonization (Chazal & Sun, 2014; Ge, Safa, Belkin, & Wang, 2011), shape comparison (Escolano, Hancock, & Biasotti, 2013; Hilaga, Shinagawa, Kohmura, & Kunii, 2001), and surface denoising (Wood, Hoppe, Desbrun, & Schröder, 2004).

4.3 Available Software

The main open source code is Python Mapper (Müllner & Babu, 2013). This code comes with a user-friendly GUI, many built-in options for the mapper input parameters (metrics, filter functions, cover choice, clustering methods), and interactive highlighting of the data with respect to choices of nodes in the mapper graph, all of which make exploratory data analysis particularly easy. In addition, Ayasdi,[3] a Silicon Valley company specializing in applications of TDA, provides educational licenses for their software.

5 DISCUSSION

TDA provides powerful tools that can find structure in data when other methods may fail. The persistence diagram gives information about the underlying structure, particularly loops and holes in the space, without the need for a user-determined proximity parameter. Mapper provides a one-dimensional representation of the data, which is an excellent interface for data exploration and visualization. While this increase in strength comes with a high mathematical barrier to entry (the mathematics behind the tools discussed in this article constitute several semesters of graduate level mathematics courses), the ever-expanding toolkit of freely available code has made TDA more accessible than ever. This article was only able to scratch the surface of the math involved. The interested reader should look to the many excellent surveys and books available on the subject (Ghrist, 2008; Carlsson, 2009; Edelsbrunner & Harer, 2010; Vejdemo-Johansson & Skraba, 2016; Ghrist, 2014). The expanding availability of these powerful tools makes the potential for application of TDA an exciting new direction for research on student learning.

ACKNOWLEDGEMENTS

The author gratefully acknowledges the anonymous reviewer feedback that greatly improved the final paper. This material is based upon work supported by the National Science Foundation under Grant No. NSF CMMI-1562012 and NSF DMS-1622320.

REFERENCES

Adams, H., & Carlsson, G. (2015). Evasion paths in mobile sensor networks. *International Journal of Robotics Research, 34*(1), 90–104. https://dx.doi.org/10.1177/0278364914548051

Adcock, A., Carlsson, E., & Carlsson, G. (2016). The ring of algebraic functions on persistence bar codes.

[3] http://www.ayasdi.com/

Homology, Homotopy and Applications, 18(1), 381–402.

Bauer, U. (2016). Ripser. https://github.com/Ripser/ripser.

Biasotti, S., Giorgi, D., Spagnuolo, M., & Falcidieno, B. (2008). Reeb graphs for shape analysis and applications. *Theoretical Computer Science: Computational Algebraic Geometry and Applications, 392*(13), 5–22.

Boriah, S., Chandola, V., & Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. In *Society for Industrial and Applied Mathematics — 8th SIAM International Conference on Data Mining 2008, Proceedings in Applied Mathematics 130* (Vol. 1, pp. 243–254).

Bubenik, P. (2015). Statistical topological data analysis using persistence landscapes. *Journal of Machine Learning Research, 16*, 77–102.

Carlsson, G. (2009). Topology and data. *Bulletin of the American Mathematical Society, 46*(2), 255–308. https://dx.doi.org/10.1090/S0273-0979-09-01249-X

Carlsson, G., Ishkhanov, T., de Silva, V., & Zomorodian, A. (2008). On the local behavior of spaces of natural images. *International Journal of Computer Vision, 76*, 1–12. http://dx.doi.org/10.1007/s11263-007-0056-x

Carlsson, G., & Zomorodian, A. (2009). The theory of multidimensional persistence. *Discrete & Computational Geometry, 42*(1), 71–93. http://dx.doi.org/10.1007/978-3-642-10631-6_74

Chan, J. M., Carlsson, G., & Rabadan, R. (2013). Topology of viral evolution. *Proceedings of the National Academy of Sciences*, *110*(46), 18566–18571.

Chazal, F., & Sun, J. (2014). Gromov-Hausdorff approximation of filament structure using Reeb-type graph. *Proceedings of the 30th Annual Symposium on Computational Geometry* (SoCG'14), 8–11 June 2014, Kyoto, Japan (pp. 491–500). New York: ACM.

Cohen-Steiner, D., Edelsbrunner, H., & Harer, J. (2007). Stability of persistence diagrams. *Discrete & Computational Geometry, 37*(1), 103–120. http://dx.doi.org/10.1007/s10208-008-9027-z

Dabaghian, Y., Mémoli, F., Frank, L., & Carlsson, G. (2012). A topological paradigm for hippocampal spatial map formation using persistent homology. *PLoS Computational Biology, 8*(8), e1002581.

de Silva, V., & Ghrist, R. (2007). Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology, 7,* 339–358. http://dx.doi.org/10.2140/agt.2007.7.339

de Silva, V., Morozov, D., & Vejdemo-Johansson, M. (2011). Persistent cohomology and circular coordinates. *Discrete & Computational Geometry, 45*(4), 737–759. http://dx.doi.org/10.1007/s00453-015-9999-4

de Silva, V., Munch, E., & Patel, A. (2016). Categorified Reeb graphs. *Discrete & Computational Geometry*, *55*(4), 854–906.

Edelsbrunner, H., Letscher, D., & Zomorodian, A. (2002). Topological persistence and simplification. *Discrete & Computational Geometry, 28*(4), 511–533. http://dx.doi.org/10.1145/1064092.1064133

Edelsbrunner, H., & Harer, J. (2010). *Computational topology: An introduction*. American Mathematical Society.

Emrani, S., Gentimis, T., & Krim, H. (2014). Persistent homology of delay embeddings and its application to wheeze detection. *Signal Processing Letters, IEEE, 21*(4), 459–463. http://dx.doi.org/10.1109/LSP.2014.2305700

Escolano, F., Hancock, E., & Biasotti, S. (2013). Complexity fusion for indexing Reeb digraphs. In R. Wilson, E. Hancock, A. Bors, & W. Smith (Eds.), *Computer analysis of images and patterns*, vol. 8047 of *Lecture Notes in Computer Science* (pp. 120–127). Berlin/Heidelberg: Springer. http://dx.doi.org/10.1007/978-3-642-40261-6_14

Fasy, B. T., Kim, J., Lecci, F., & Maria, C. (2014a). Introduction to the R package TDA. *arXiv:1411.1830*.

Fasy, B. T., Lecci, F., Rinaldo, A., Wasserman, L., Balakrishnan, S., & Singh, A. (2014b). Confidence sets for persistence diagrams. *Annals of Statistics, 42*(6), 2301–2339.

Ge, X., Safa, I. I., Belkin, M., & Wang, Y. (2011). Data skeletonization via Reeb graphs. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, & K. Weinberger (Eds.), *Advances in Neural Information Processing Systems 24*, 837–845.

Ghrist, R. (2008). Barcodes: The persistent topology of data. *Bulletin of the American Mathematical Society, 45*, 61–75. http://dx.doi.org/10.1090/S0273-0979-07-01191-3

Ghrist, R. (2014). *Elementary Applied Topology*, 10th ed. Createspace.

Giusti, C., Pastalkova, E., Curto, C., & Itskov, V. (2015). Clique topology reveals intrinsic geometric structure in neural correlations. *Proceedings of the National Academy of Sciences, 112*(44), 13455–13460. http://dx.doi.org/10.1073/pnas.1506407112

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

Hilaga, M., Shinagawa, Y., Kohmura, T., & Kunii, T. L. (2001). Topology matching for fully automatic similarity estimation of 3D shapes. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH '01), 12–17 August 2001, Los Angeles, California, USA (pp. 203–212). New York: ACM. http://dx.doi.org/10.1145/383259.383282

Kaczynski, T., Mischaikow, K., & Mrozek, M. (2006). *Computational homology*, vol. 157. Springer Science & Business Media.

Kerber, M., Morozov, D., & Nigmetov, A. (2016). Geometry helps to compare persistence diagrams. *Meeting on Algorithm Engineering and Experiments* (ALENEX16), 10–12 January 2016, Arlington, Virginia, USA (pp. 103–112). arXiv:1606.03357 [cs.CG] https://dx.doi.org/10.1137/1.9781611974317.9

Khasawneh, F. A., & Munch, E. (2016). Chatter detection in turning using persistent homology. *Mechanical Systems and Signal Processing, 70–71,* 527–541. https://dx.doi.org/10.1016/j.ymssp.2015.09.046

Lesnick, M., & Wright, M. (2015). Interactive visualization of 2-D persistence modules. *arXiv preprint arXiv:1512.00180*

Lesnick, M., & Wright, M. (2016). Rivet: The rank invariant visualization and exploration tool. http://rivet.online/.

Li, L., Cheng, W.-Y., Glicksberg, B. S., Gottesman, O., Tamler, R., Chen, R., Bottinger, E. P., & Dudley, J. T. (2015). Identification of type 2 diabetes subgroups through topological analysis of patient similarity. *Science Translational Medicine*, *7*(311), 311ra174–311ra174. http://dx.doi.org/10.1126/scitranslmed.aaa9364

Mischaikow, K., & Nanda, V. (2013). Morse theory for filtrations and efficient computation of persistent homology. *Discrete & Computational Geometry, 50*(2), 330–353. http://dx.doi.org/10.1007/s00454-013-9529-6

Morozov, D. (2015). Dionysus. http://www.mrzv.org/software/dionysus/

Müllner, D., & Babu, A. (2013). Python mapper: An open-source toolchain for data exploration, analysis and visualization. http://danifold.net/mapper

Munch, E., Shapiro, M., & Harer, J. (2012). Failure filtrations for fenced sensor networks. *The International Journal of Robotics Research, 31*(9), 1044–1056. https://dx.doi.org/10.1177/0278364912451671

Munch, E., Turner, K., Bendich, P., Mukherjee, S., Mattingly, J., & Harer, J. (2015). Probabilistic Fréchet means for time varying persistence diagrams. *Electronic Journal of Statistics, 9*, 1173–1204.

Munch, E., & Wang, B. (2016). Convergence between categorical representations of Reeb space and Mapper. In S. Fekete & A. Lubiw (Eds.), *32nd International Symposium on Computational Geometry* (SoCG 2016), vol. 51 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 53:1–53:16. Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Munkres, J. R. (1993). *Elements of algebraic topology*. Addison Wesley.

Nanda, V. (2015). Perseus. http://www.sas.upenn.edu/ ~ vnanda/perseus/

Nicolau, M., Levine, A. J., & Carlsson, G. (2011). Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival. *Proceedings of the National Academy of Sciences, 108*(17), 7265–7270. http://dx.doi.org/10.1073/pnas.1102826108/-/DCSupplemental.

Nielson, J. L., Paquette, J., Liu, A. W., Guandique, C. F., Tovar, C. A., Inoue, T., Irvine, K.-A., Gensel, J. C., Kloke, J., Petrossian, T. C., et al. (2015). Topological data analysis for discovery in preclinical spinal cord injury and traumatic brain injury. *Nature Communications, 6. http://dx.doi.org/10.1038/ncomms9581*

Niyogi, P., Smale, S., & Weinberger, S. (2011). A topological view of unsupervised learning from noisy data. *SIAM Journal on Computing, 40*, 646–663. https://dx.doi.org/10.1137/090762932

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., & Harrington, H. A. (2015). A roadmap for the computation of persistent homology. *arXiv: 1506.08903*

Perea, J. A., & Carlsson, G. (2014). A Klein-bottle-based dictionary for texture representation. *International Journal of Computer Vision, 107*(1), 75–97. http://dx.doi.org/10.1007/s11263-013-0676-2

Perea, J. A., Deckard, A., Haase, S. B., & Harer, J. (2015). SW1PerS: Sliding windows and 1-persistence scoring; discovering periodicity in gene expression time series data. *BMC Bioinformatics, 16*(1).

Robins, V., Wood, P. J., & Sheppard, A. P. (2011). Theory and algorithms for constructing discrete Morse complexes from grayscale digital images. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 33*(8), 1646–1658.

Singh, G., Mémoli, F., & Carlsson, G. (2007). Topological methods for the analysis of high dimensional data sets and 3D object recognition. In *Eurographics Symposium on Point-Based Graphics. Retrieved from* https://research.math.osu.edu/tgda/mapperPBG.pdf

Torres, B. Y., Oliveira, J. H. M., Thomas Tate, A., Rath, P., Cumnock, K., & Schneider, D. S. (2016). Tracking resilience to infections by mapping disease space. *PLoS Biology, 14*(4), 1–19.

Tralie, C. (2016). Fast Rips in the browser. http://www.ctralie.com/Software/jsTDA/

Turner, K., Mileyko, Y., Mukherjee, S., & Harer, J. (2014). Fréchet means for distributions of persistence

diagrams. *Discrete & Computational Geometry, 52*(1), 44–70.

Vejdemo-Johansson, M., & Skraba, P. (2016). Topology, Big Data and Optimization. In *Big Data Optimization: Recent Developments and Challenges* (pp. 147–176). Springer. http://dx.doi.org/10.1007/978-3-319-30265-2_7

Wood, Z., Hoppe, H., Desbrun, M., & Schröder, P. (2004). Removing excess topology from isosurfaces. *ACM Transactions on Graphics, 23*(2), 190–208. http://dx.doi.org/10.1145/990002.990007

Yao, Y., Sun, J., Huang, X., Bowman, G. R., Singh, G., Lesnick, M., Guibas, L. J., Pande, V. S., & Carlsson, G. (2009). Topological methods for exploring low-density states in biomolecular folding pathways. *The Journal of Chemical Physics, 130*(14). http://dx.doi.org/10.1063/1.3103496

Zomorodian, A., & Carlsson, G. (2004). Computing persistent homology. *Discrete & Computational Geometry, 33*(2), 249–274. http://dx.doi.org/10.1007/s00454-004-1146-y