Topological Inference for Modern Data Analysis

An Introduction to Persistent Homology

Giancarlo Sanchez

A project presented for the degree of
Master's of Science in Mathematics

Department of Mathematics and Statistics
Florida International University
Graduate Advisor: Dr. Miroslav Yotov
April 29, 2016

Abstract

Persistent homology has become the main tool in topological data analysis because of it's rich mathematical theory, ease of computation and the wealth of possible applications. This paper surveys the reasoning for considering the use of topology in the analysis of high dimensional data sets and lays out the mathematical theory needed to do so. The fundamental results that make persistent homology a valid and useful tool for studying data are discussed and finally we compute examples using the package TDA available in the CRAN library for the statistical programming language R.

Data & Geometry

The modern data analyst's biggest challenge is dealing with tremendous amounts of complex data. Not only is it difficult to deal with the enormous amounts of observations being stored, but also in the amount of attributes or features being measured in the process. Advances in computing power have made it easier to analyze high dimensional data and it has been proven very beneficial for companies and scientists alike to extract useful information from these high dimensional data sets. Many companies even collect data without knowing beforehand if the data will ever be analyzed, let alone how. This has pushed forward the rise of *data science* as an extremely interesting and necessary field of study as it stands at the intersection of modern mathematics, statistics, computer science and an enormous amount of applications.

Typically, a data scientist's work flow begins with a particular question or hypothesis in mind. The researcher experiments and collects samples, preprocesses (cleans) the data, and then tests to see if the sampled data fits an expected model. There is usually a substantial amount of preprocessing involved before common algorithms can be used effectively. One of the difficulties in analyzing this high dimensional data arises as a result of the bizarre geometric properties of high dimensional spaces. This is commonly referred as the *curse of dimensionality*, a phrase used as an umbrella term for many of the unfortunate consequences that arise when analyzing high dimensional data. Take, for instance, the volume of the n dimensional sphere. The volume increases from $n = 1$

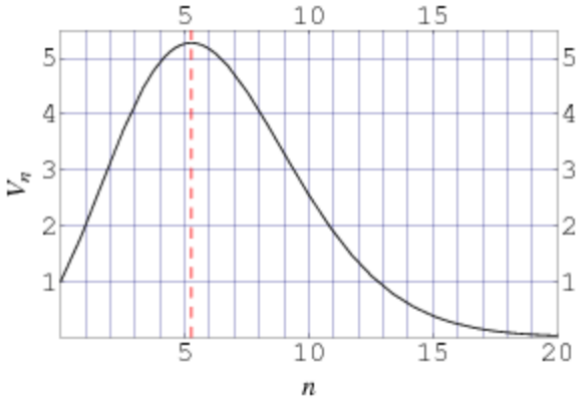


Figure 1: Volume of n -dimensional sphere.

to $n = 5$ and then rapidly decreases to 0 from $n = 20$ and beyond. Consider also the multivariate standard Gaussian distribution scaled to have integral equal to 1. It is well know that 90% of the volume falls within a radius of 1.65 of the mean for the single variate case. This percentage rapidly decreases to 0 as the dimension of the Gaussian function increases. This means that much of the data is found in the tails of the distribution, far, far away from center. More trouble arises when attempting to learn models in the machine learning sense.[1] Since the models used for learning are only valid in the range or volume in which the data are available, as the dimensions increase we must collect more and more data to fill up the space. For example, for a simple linear regression model with one explanatory variable, we might be satisfied with collecting 10 observations before we try to learn the model. A linear regression model with 2 explanatory variables must then need 100 observations collected and 1000 for a 3-dimensional model and so on. The number of samples needed grows exponentially as the dimension of the feature variables increases. Models and algorithms generally do not behave well when scaling these high dimensions. However, while these data sets do seem large at first glance, it is common to see data sparsely floating around in these higher dimensions while actual artifacts tend to be living within a simpler, lower dimensional structure. Therefore, a big component of the preprocessing phase is the time spent transforming data into a simpler space or selecting unimportant features to disregard in the analysis.

Geometric intuition has paved the way for various methods in data analysis which aim at reducing this excess in dimensionality. Traditional tools like Principal Component Analysis, while maintaining the largest variance among points, linearly projects points onto a lower dimensional subspace. While other linear techniques are useful, much modern research has been devoted to nonlinear dimensionality techniques (collectively known as manifold learning) and follow similar geometric frameworks to data reduction. In it, we assume that the data were originally sampled from a sub-manifold that is of lower dimension than the ambient space. We try to find a nonlinear space that describes the data well and is in bijection with smaller sub-manifold. The problem of dealing with high dimensions typically gets worse with nonlinear methods since they involve much more parameters to be estimated.

Nevertheless, geometry has been very influential in the design of many data analysis algorithms and techniques. Our intuition guides us well in 2 or 3 dimensions, yet the counter-intuitive geometry of higher dimensions causes us to lack insight. To understand the basic structure of the data, more qualitative information is needed. Geometry itself is concerned with questions of size, shape, and relative positions of objects. For example, it provides the tools to be able to mathematically differentiate a circle from a square - which can be unnecessary in data analysis if all we needed to know was that there was a *hole* in the middle of this data for some circle-like data set. Information about the topology of the underlying space in which the data were sampled from provides us with qualitative information that is useful in this analysis

If our data consisted of noisy samples from a circle with a uniform distribution as the one pictured in Figure 2, we would first have to realize the data in some coordinate system and then recover the geometry of the circle from our sample as close as we can on a given scale. On the other hand, topological tools such as homology, disregard fine scale curvature of the and describe the circle as being a single connected component with a 1-dimensional loop. Homology provides us with the tools for distinguishing spaces and

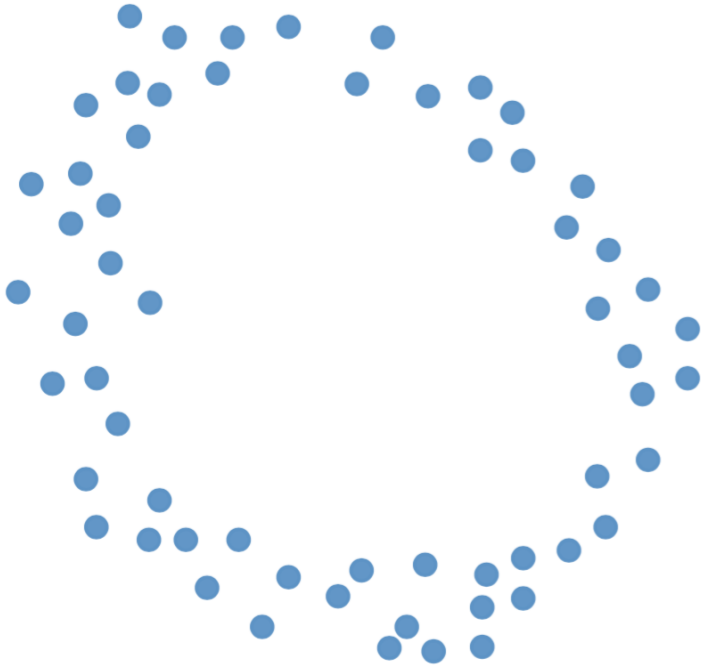


Figure 2: A point cloud sampled from a circle with noise.

shapes by counting these connected components, holes, voids, and other higher dimensional analogs. Although information about the rigid geometric shape is lost, what we do obtain is a *topological property* which tells us the inherent connectivity properties of the data. These properties do not change if we continuously change the coordinate or if we scale or deform the space. This makes topology a powerful tool when trying to find a low dimension representation for high dimensional data and analyzing data in general.

The use of topological techniques in understanding high dimensional data sets has gained a huge following over the course of these recent years and has motivated the development of the field of topological data analysis or TDA. This is mainly because of the success of TDA's main computing tool, persistent homology. Simply put, persistent homology is a multi-scale generalization of homology. Of course, while a bunch of discrete points might not have any interesting topology, we can associate to these points a sequence of nested topological spaces. This sequence, called a filtration, will encode the connectivity information needed to understand the underlying topological space which the data was initially sampled from space. As we see how the homological features change through filtration by computing the homology at each level, we witness the birth and death each topological feature. Though any particular level of the filtration might not be an accurate approximation of the underlying space the data was sampled from, the topological features which *persist* through the filtration for a significantly long life regarded as topological invariants of the space.

To associate any flavorful topological information to our data, we need a way to convert our data into a topological space so that we accurately convey the relative connectivity of the points in our data. As is common in algebraic topology when trying to represent continuous objects, we use simplicial complexes to represent our data's connectivity and study the topological properties of their geometric representations.

From Data to Simplcial Complexes

Characters and landscapes seen in video games are a great example of how simplicial complexes provide great approximations of continuous surfaces, as anyone who has ever played a video game would attest. In TDA, we try to approximate two distinct points' similarity by using simplicial complexes. If points are similar, we keep them in the same complex, if they are different, we keep them as separate ones. Similarity can be defined by euclidean distance though many other metrics are used. Typically, a similarity measure is chosen by a domain expert. The data set $\mathbb{X}$, sometimes referred to as a point cloud, consists of samples from a space $X \subset \mathbb{R}^d$. We try to infer the topological information of X using only the sample points and their relative distances. To approximate X though, we first turn convert our data into a simplicial complex. This produces an easy to work with topological object that contains needed connectivity information of $\mathbb{X}$ and if our sample is dense enough, provides a real good approximation of the topological invariants of the original space X. There are many ways of constructing a simplicial complex from a discrete point cloud. Each construction provides a similar approximation of the underlying structure of the data. They differ when it comes to the computational considerations. In this section we introduce simplical complexes and two fundamental constructions used in persistent homology, the $\check{C}$ech complex and the Vietoris-Rips complex.

Simplicial Complexes & The Nerve

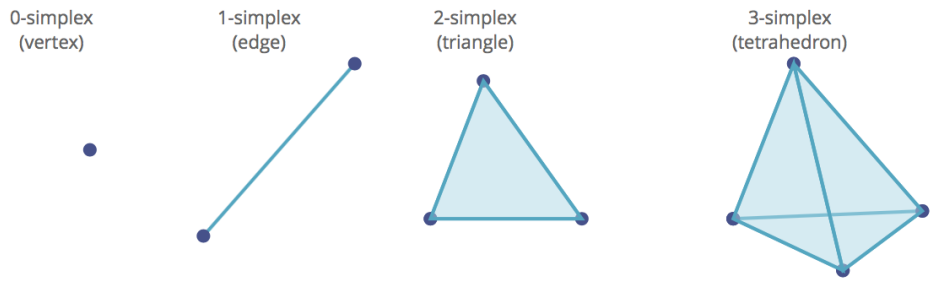


Figure 3: Geometric representation of low dimensional simplex.

Simplexes are the building blocks of simplicial complexes. A point is a 0-dimensional simplex, a line is a 1-dimensional simplex, and a solid triangle is a 2-dimensional simplex. A 3-dimensional simplex can be thought of as a tetrahedron but it's difficult to imagine any higher dimensional n-simplex. Geometrically one can define simplicial complexes by gluing together simplicies along their faces in a nice way (in some ambient space $\mathbb{R}^d$). There is an equivalent combinatorial representation of any simplicial complex which makes discussing some of their properties much smoother.

Definition. Given a finite vertex set $V = \{1, ...n\}$, a **simplicial complex** Σ is a collection of subsets of V such that if $\sigma \in \Sigma$ and $\sigma' \subset \sigma$ then $\sigma' \in \Sigma$.

We'll normally consider our point cloud as the finite vertex set $V = \mathbb{X}$ and try to find simplicial complex with a topology which closely approximates the underlying space X.

We call an element $\sigma \in \Sigma$ a simplex and the dimension of σ is one less than the cardinality, $dim(\sigma) = |\sigma| - 1$. The dimension of the simplicial complex Σ is the highest dimension among all it's simplex. A nonempty proper subset of a simplex $\gamma \subset \sigma$ is called a face of σ and by definition is also a simplex. A simplicial complex that is a subset of a simplicial complex $\Gamma \subset \Sigma$ is called a simplicial subcomplex. There exists a function that sends each simplex in Γ to itself in Σ known as the inclusion map $\iota : \Gamma \to \Sigma$. More generally, we can define a function between any two arbitrary simplicial complexes so long the vertices of each n-dimensional simplex are mapped to the verticies of a target simplex. That is, if Σ_1 and Σ_2 are 2 simplicial complexes, a function f from the vertex set of Σ_1 to the vertex set of Σ_2 is a *simplical map* if for all $\sigma \in \Sigma_1$, we have that $f(\sigma) \in \Sigma_2$. It is important to note that any simplical map could only send n-simplicies to m-simplicies where $m \leq n$.

We can construct a simplical complex for any collection of sets based on how the sets intersect. More specifically, for a given open cover $\mathcal{U} = \{U_\alpha\}_{\alpha \in A}$ of a topological space X, the *nerve* of $\mathcal{U}$ denoted by $\mathcal{N}(\mathcal{U})$, is defined to be the simplicial complex with vertex set A and a k dimensional simplex $\sigma = \{\alpha_0, ...\alpha_k\} \in \mathcal{N}(U)$ if and only if $U_{\alpha_0} \cap ... \cap U_{\alpha_k} \neq \emptyset$. There is a condition that guarantees when the nerve of the cover will stay faithful to the topology of the underlying space.

Nerve Lemma. Suppose that X and $\mathcal{U}$ are as above, and suppose that the covering is numerable. Suppose further that for all $\emptyset \neq S \subset A$, we have that $\bigcap_{s \in S} U_S$ is either contractible or empty. Then $\mathcal{N}(\mathcal{U})$ is homotopy equivalent to X.

The Nerve Lemma is a basic result which guarantees that $\mathcal{N}(U)$ is of the same homotopy type as the underlying space X.

Čech & Vietoris-Rips Complexes

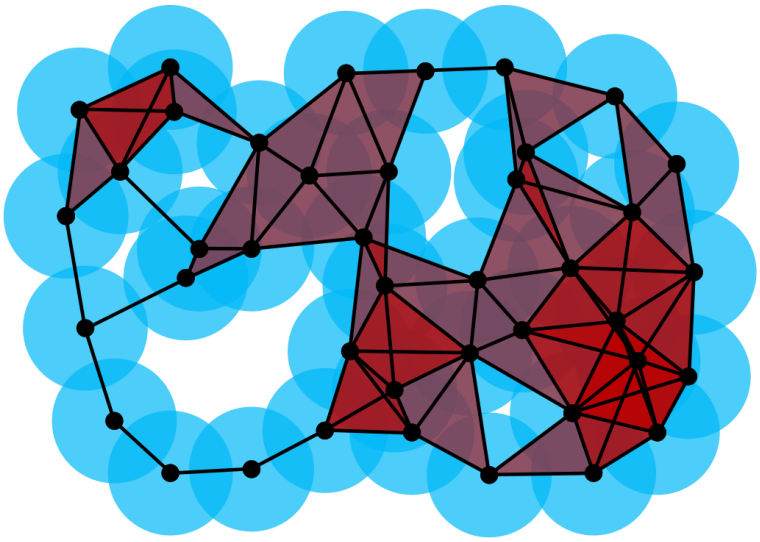


Figure 4: Example of a Čech Complex.

The Čech complex, named after the Czech mathematician Eduard Čech, is the first example of a simplicial complex which we can use for out data. Intuitively, since each

point should lie on or around the space X, if we make our points bigger and have enough of them, we should be able to give a good approximation of X. Mathematically, we can consider each point as the center of an open ball of fixed radius $\epsilon > 0$ denoted by$B_\epsilon(x) := \{y : d(x, y) < \epsilon\}$. The collection of these ϵ balls makes a cover $\mathcal{B}_\epsilon = \{B_\epsilon(x) : x \in \mathbb{X}\}$ of our point cloud.

Definition. A *Čech complex* is the nerve of the covering of balls,
$\mathcal{C}_\epsilon(\mathbb{X}) := \mathcal{N}(\mathcal{B}_\epsilon) = \{\sigma \subset \mathbb{X} : \bigcap_{x\in\sigma} B_\epsilon(x) \neq \emptyset\}$.

That is, $\mathcal{C}_\epsilon(\mathbb{X})$ is the simplicial complex obtained by considering $\mathbb{X}$ as our vertex set and $\{x_1, ..., x_{k+1}\}$ span a k-simplex if and only if $\bigcap_{i=1}^{k+1} B_\epsilon(x_i) \neq \emptyset$. Since the ϵ-balls are each convex, their intersection is convex (hence contractible) and thus the Nerve Lemma applies. Čech complex is homotopy equivalent to the union of balls, which should be a rough estimation of our space. Therefore if we want to learn about our point cloud we can pick an ϵ and study the topology of it's corresponding Čech complex. Notice that for a smaller radius $\epsilon' < \epsilon$, the smaller Čech Complex $\mathcal{C}_{\epsilon'}$ is a subcomplex. In turn, we get an inclusion map $\iota : \mathcal{C}_{\epsilon'} \to \mathcal{C}_\epsilon$ of complexes. For a small enough value, we'll have a Čech complex consisting of n 0-dimensional simplices. For a large enough value, we'll have single (n-1)-dimensional simplex. As ϵ increases we will obtain a finite nested sequence of Čech complexes and inclusion maps. Each of these spaces approximates the underlying space at different scales. The problem comes when finding the right scale to consider out data. The idea of persistent homology is to study all scales simultaneous and record how topological features evolve as the scale changes. The Čech complex filtration is a good way to imagine how evolution at different scales occurs and, moreover, why important topological features should persist.

A problem with the Čech complex is that it requires that computers store simplicies of all dimensions, making it computationally expensive. It is also redundant in the sense that with a little ingenuity we can come up with a smaller simplicial complex of the same homotopy type. An idea for dealing with this is to construct a simplicial complex characterized completely by it's 1-dimensional simplicies.

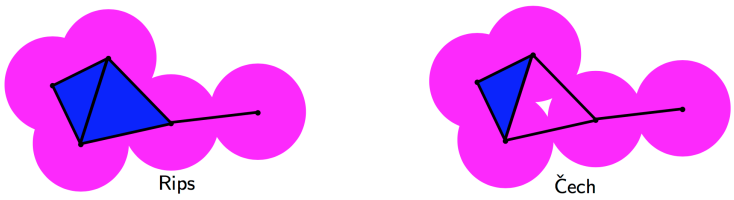


Figure 5: Difference between Rips complex and Čech complex.

This suggests the following variant of the Čech construction, referred to as the Vietoris-Rips complex (sometimes shortened to just Rips complex). While the theoretical properties of the Čech complex make it mathematically desirable, for computational reasons, it is more common to approximate it with the *Vietoris - Rips complex* (sometime abbreviated as VR or Rips complex) which includes a simplex in the complex every time pairwise distances between vertices are small enough.

Definition. Given $\epsilon > 0$, a ***Vietoris-Rips complex*** on $\mathbb{X}$ is the simplicial complex

$$\mathcal{R}_\epsilon(\mathbb{X}) := \{\sigma \subset \mathbb{X} : \forall x, y \in \sigma, B_\epsilon(x) \cap B_\epsilon(y) \neq \emptyset\}$$

That is, we include a k-simplex if for points $\{x_1, ..., x_{k+1}\}$, the distance between any two points is $d(x_i, x_j) < 2\epsilon$ for all $1 \leq i, j, \leq k$. Notice that we also get an inclusion $\mathcal{R}_{\epsilon'} \subset \mathcal{R}_\epsilon$ for $\epsilon' \leq \epsilon$. The VR complex is easier to compute since only pairwise distances need to be calculated in order to obtain our simplicial complex. Moreover, it's easily seen that we have the inclusion $\mathcal{C}_\epsilon(\mathbb{X}) \subset \mathcal{R}_\epsilon(\mathbb{X})$. In fact, the collection of 1-simplicies is equal for both complexes so the VR complex is the largest complex that can made given the collection of 1 simplicies from a $\check{C}$ech complex. Even though the VR complex will be generally larger than the $\check{C}$ech complex, they satisfy the sandwich relation,

$$\mathcal{C}_\epsilon(\mathbb{X}) \subset \mathcal{R}_\epsilon(\mathbb{X}) \subset \mathcal{C}_{2\epsilon}(\mathbb{X}).$$

This means that if the $\check{C}$ech complex is a good approximation of X, then so will the VR complex - even though we don't have a guarantee that the VR complex is of the same homotopy type of any particular space.

Other Complexes

The $\check{C}$ech and VR complexes are generally very large simplicial complexes and there are many ways that we can reduce some of the redundancy of information. Much research in topological data analysis is directed towards finding efficient simplicial complexes to represent to the data. The following are some alternatives usually considered.

Delaunay Complex

For each point $x \in \mathbb{X}$, we can decompose the ambient space $\mathbb{R}^d$ into a collection of subspaces $\{V_x\}_{x \in \mathbb{X}}$ known as the *Voronoi Cells*, named after the Russian mathematician Georgy Voronoy. They are defined for each $x \in \mathbb{X}$ as

$$V_x = \{y \in \mathbb{R}^n : d(x, y) \leq d(x', y), \forall x' \in \mathbb{X}\}..$$

Note that V_x is a convex polyhedron since it is the intersection of all the half-spaces of points that are at least closer to x than they are to x'. Also, note that any two cells either have an empty intersection or they intersect at a boundary. The collection of all these Voronoi Cells is called the *Voronoi Diagram*. We say the set $\mathbb{X}$ is in general position if no $k + 2$ points lie on a common $(k1)$-sphere.

Definition. The nerve of the Voronoi diagram is a simplicial complex known as the ***Delaunay Complex*** $\mathcal{D}(\mathbb{X}) = \{\sigma \subset \mathbb{X} : \bigcap_{x \in \sigma} V_x \neq \emptyset\}$

This construction is named after Boris Delaunay (or Delone), a student of Voronoi. The points must be in general position to obtain a natural realization in $\mathbb{R}^d$. This implies that any simplex in the Delaunay complex has dimension lower than or equal to d.

Alpha Complex

We can further constrain the Delaunay complex to obtain a smaller complex. Since the intersection of the Voronoi region of a point $x \in \mathbb{X}$ and an ϵ-ball covering x is a contractible space, we can intersect these two subspaces and obtain a simplicial complex realizable in $\mathbb{R}^d$ of the correct homotopy type by invoking the Nerve lemma. Together, the regions defined by the intersections $R_x(\epsilon) := V_x \cap B_x(\epsilon)$ make a covering of our point cloud.

Definition. The ***Alpha complex*** is the nerve of these regions
$$\mathcal{A}_\epsilon(\mathbb{X}) = \{\sigma \subset \mathbb{X} : \bigcap_{x\in\sigma} R_x(\epsilon) \neq \emptyset\}.$$

Alpha complexes are closely related to the idea of Alpha shapes which were first introduced by Edelsbrunner et al. in [7] as a generalization of the convex hull of points in $\mathbb{R}^2$. Alpha complexes and Alpha shapes have been used to successfully model biological molecules such as proteins and DNA. Edelsbrunner, an Austrian computer scientist, has been an avid researcher in the field of computation geometry and computational topology. He was the first to propose an algorithm for computing the persistence of a sequence of spaces.

There are other constructions which reduce the overall size of the complex while still remaining close to the original shape. Weighted versions of the constructions above, weighted Rips complex and weighted Alpha complex, provide interesting tweaks. Also, instead of using all the points in the data set, we can pick specific landmark points and build a complex on them. This is called the Witness complex. Constructing efficient simplicial complexes is an important hurdle to overcome when trying to reduce computational complexity and is a crucial part of effectively using TDA for high dimensional data analysis.

Filtrations and Sublevel Sets

The simplical complex that we obtain from either the Čech or Rips complex depends entirely on our choice of ϵ. Rather than finding an optimal fixed value, we focus on a range of values for the scale parameter ϵ. As ϵ increases between these extremes, we obtain a nested a sequence of complexes

$$\Sigma_0 \subset \Sigma_1 \subset ...\Sigma_{d-1} \subset \Sigma_d$$

known as a filtration, each corresponding to a value of ϵ at which our complex adds a simplex.

There is another way to construct simplical complexes from our data - by using functions defined on the data. For example, given a point cloud $\mathbb{X} \subset \mathbb{R}^d$ consider the function $f : \mathbb{R}^d \to \mathbb{R}$ that sends any point to $p \in \mathbb{R}^n$ to it's minimal distance from our cloud, i.e the distance function

$$f(p) = \min_{x_i\in\mathbb{X}}(\|x_i - p\|).$$

The sublevel sets $X_r := f^{-1}(-\infty, t]$ form a filtration for increasing values in $\mathbb{R}$. Meaning, we get the inclusions of topological spaces indexed by $\mathbb{R}$

$$X_r \hookrightarrow X_s \hookrightarrow X_t \hookrightarrow ...$$

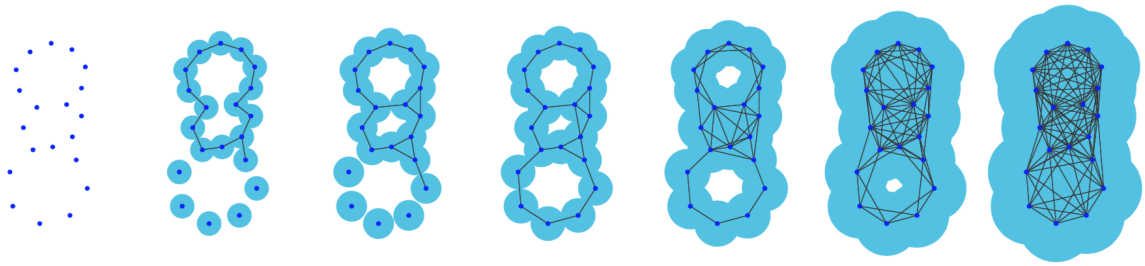


Figure 6: As we increase the size of the balls around our point cloud, we get a nested sequence of simplicial complexes. The true homology sampled space is given for some ϵ value somewhere in between the to extremes.

In fact, for this specific function f, the sub-level sets are just the union of balls around our data. That is,

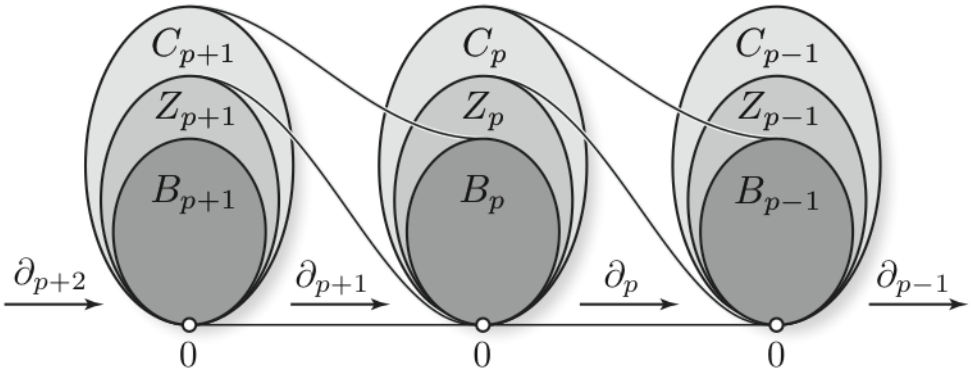
$$X_t = \{p : f(p) \leq t\} = \bigcup_{x_i \in \mathbb{X}} B_t(x)$$

so we can easily recover a $\check{C}$ech complex from the sublevel sets of a distance function. Sublevel sets (also, superlevel set) let us study topological properties related to functions defined on our point cloud. While the $\check{C}$ech complex arises as a result of a function, the Vietoris Rips complex unfortunately does not and neither do many other constructions. It is, therefore, worthwhile to study both sublevel set filtrations and other simplical complexes not derived from functions.

Topological Persistence

Homology provides us with a language to talk about the structure of a space in a precise way by associating algebraic structures to topological spaces. Homology describes qualitative features of the space that are invariant under continuous transformations. These features are graded by dimension and are represented by associating homology groups H_0, H_1, H_2, etc. to each dimension. The rank of H_0 measures the number of connected components of our space. The rank of H_1 measures whether loops in the space can be contracted to a single point or not. Similarly, higher dimensional connectivity features can be computed and quantified with the use of homology.

Homology and Persistence



We'll switch the notation for a simplicial complex from Σ to K to emphasize that we are dealing with simplicial complexes as topological spaces now instead of their combinatorial counterpart. For any simplicial complex K we associate a chain complex

$$... \xrightarrow{\partial_{p+2}} C_{p+1} \xrightarrow{\partial_{p+1}} C_p \xrightarrow{\partial_p} C_{p-1} \xrightarrow{\partial_{p-1}} ...$$

which we denote by $C_\bullet(K)$, where each chain group C_p is the free k-module generated by all p-dimensional simplices in K. The elements of C_p, called p-chains, are formal sums of the p-dimensional simplicies. The boundary homomorphism $\partial_p : C_p \to C_{p-1}$ maps a p-chain to the formal sum of it's (p-1)-dimensional faces. The elements of $Z_p := Ker(\partial_p)$ are called cycles and the images $B_p := Im(\partial_{p+1})$ are the boundaries. Since $\partial_{p+1} \circ \partial_p = 0$ for chain complex, we get that $Z_p \subset B_p$. The p-th homology group is defined to be the quotient group

$$H_p := Z_p / B_p.$$

That is, we consider two cycles identical if they differ by a boundary. The elements of H_p are equivalence classes of p-dimensional features.

The *Functoriality* of homology is the property that lets us study topological space with algebra in a well defined manner. Since we want to compute the homology of each space in our filtration, we need to be sure that the inclusion maps provide us with well defined maps between the associate homology groups. Taking the p-th homology of a simplicial complex is functorial in the sense that for a simplicial map $f : K \to L$, it

induces a linear map $C(f) : C_p(K) \to C_p(L)$ for all dimensions p, which defines a map linear map $f_* : H_p(K) \to H_p(L)$. In other words, a map of spaces defines a map of homological features.

To see how homological features change as we filter through different approximations, we apply homology functorially to the filtration to see which features persist the longest. For example, let $K_t = \mathcal{C}_t(\mathbb{X})$ be the $\check{C}$ech complex (or any other simplicial complex described above) on a point cloud indexed by $t \in \mathbb{R}_{\geq 0}$. We define $K_0 := \emptyset$ and define $K := K_t$ where t is the smallest real number such that if $t' \geq t$ then $K_t = K_{t'}$. We obtain the following filtration

$$\emptyset = K_0 \to ... \to K_i ... \to K_t = K.$$

Because of the functorial properties of associating a chain complex to a simplicial complex, we obtain the sequence

$$C_p(K_0) \to ... \to C_p(K_i) \to ... \to C_p(K).$$

Computing the p-th homology groups gives us the sequence with induced map

$$H_p(K_0) \to ... H_p(K_i) \to ... \to H_p(K)$$

This last sequence is an example of a *persistence module* and we get one for every dimension. Persistence modules are the basic algebraic objects used in the theory of persistent homology. In categorical terms, if **Vec** is the category of vector spaces over a fixed field and $\mathbb{R}$ considered as a total order category (with objects real numbers $r \in \mathbb{R}$ and morphisms $\varphi \in Hom(r, s)$ if and only if $r \leq s$, then a persistence module can be defined as a functor $\mathcal{P} : \mathbb{R} \to$ **Vec** . We'll rigorously define a persistence module in the following section.

In our example above, the filtration was indexed by $\mathbb{R}$. Since our data is finite, we will actually only produce a finite amount of simplicial complexes in our filtration which differ. Therefore, we can index the filtration by $\mathbb{N}$ and add a new simplicial complex to the filtration every time a simplex is added to the construction. Actually, any totally ordered set will give us a persistence module. There is much research in the study multidimensional persistence with filtrations indexed by $\mathbb{R}^n$. Despite the seemingly complex process of obtaining the persistence module, there are methods to visualize the persistence modules which make the information attributed to the data set easier to understand.

Barcodes, Diagrams and Landscapes

Our first two visualizations, persistence *barcodes* and *diagrams*, are equivalent representations of persistence modules and are sometimes used interchangeably in literature. These are what makes persistent homology a useful tool for the every day data analyst since they are good visualizations of the topological information we seek. The third representation, the persistence *landscape*, comes from the need to incorporate statistics and machine learning techniques to the theory. The most important theorems in persistent homology thus far - the structure and stability theorems - can be easily understood with these representations as well. First, we define the persistence module.

A persistence module is the pair $(\mathcal{V}, \varphi)$ where, $\mathcal{V} = \{V_t\}_{t \in \mathbb{R}}$ is a family of vector spaces over some k indexed by $\mathbb{R}$, together with linear maps $\rho_{t,s} : V_s \to V_t$ for every $s, t \in \mathbb{R}$ such

that $s \leq t$. These maps must respect the composition in the sense that $\rho_{t,r} = \rho_{t,s} \circ \rho_{s,r}$. Again, the definition can be generalized for vector spaces indexed on any totally ordered set as many authors will switch between a finite total order $[n]$ to $\mathbb{N}$ or $\mathbb{Z}$.

Structure

The barcode first appeared in 2004 for persistence modules indexed over $\mathbb{Z}$ using the standard structure theorem of finitely generated $\mathbb{Z}$-modules over a principal ideal domain. William Crawley-Boevey later generalized the proof to intervals over $\mathbb{R}$. The barcode provides an explicit view of when homological features are born and when they die within the filtration. This information is encoded in simpler persistence modules called *interval modules*.

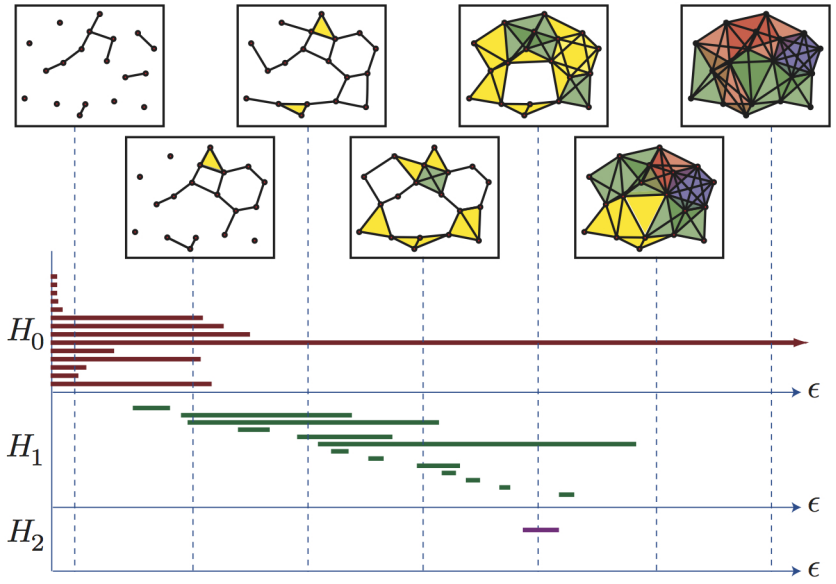


Figure 7: An example of a persistence barcode associated to the filtration shown above. A barcode is given for each dimension of the homology computed.

Recall that an interval $I = [s, t] \subset \mathbb{R}$ is a set such that $x \in \mathbb{R}$ whenever $s, t \in \mathbb{R}$ for $s \leq x \leq t$. An interval module k_I is a persistence module such that every vector space indexed by elements of I is k, every vector space indexed by $\mathbb{R} - I$ is the zero vector space, and every $\varphi_{t,s}$ is the identity map for $s \leq t$ in I and zero map other wise. A persistence module is called point-wise finite dimensional if each vector space in the family is finite dimensional. This is the case when studying finite data, but the theory has been generalized to include infinite dimensional spaces.

The fundamental structure theorem given by Crawley-Boevey in [4] lets us decompose a point-wise finite dimensional persistence module into a sum of interval modules indexed by a multi-set of intervals B, i.e.

$$\mathcal{V} \cong \bigoplus_{I \in B} k_I.$$

We can, therefore, express persistence module as the sum of these simpler interval modules. The barcode is this family of intervals in $\mathbb{R}$, where we regard each interval as the

lifespan of a topological feature of the data and the length of the interval determines a measure of significance of the feature. For each homological dimension we obtain a barcode.

Stability

To have any chance of analyzing noisy data, it is important that persistence modules be stable under small changes in the data. The *algebraic stability of persistence theorem*, a central theorem in persistent homology as shown by Chazal et al. in[5], shows exactly. They showed that if there exists a δ-interleaving (a type of isomorphism) between persistence modules then there exists a δ-matching (another type of isomorphism) between their corresponding barcodes.

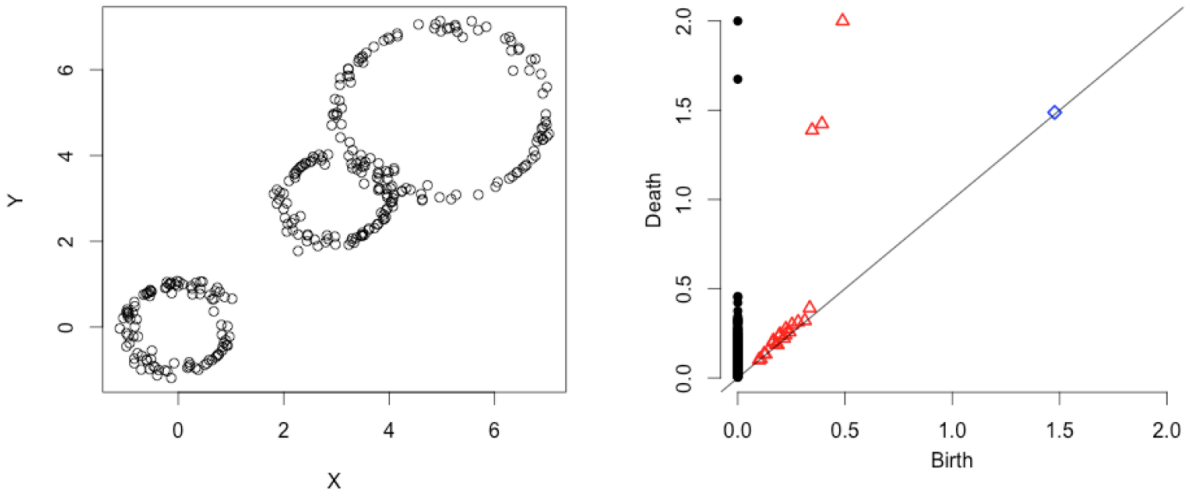


Figure 8: The black points represent the 0-th homology. The two points towards the top represent the 2 connected components of the point cloud. The three red triangles represent the 3 1-dimensional holes found in the data. The single blue point is a 2-dimensional hole which had a very short life span.

Stability results were shown first by D. Cohen-Steiner et al. [12] for *persistence diagrams* obtained from the level sets of a continuous function $f : \mathbb{X} \to \mathbb{R}$. The persistence diagram $Dgm(f)$ is a multi-set of points in the extended 2d-plane $\bar{\mathbb{R}}^2 = [-\infty, \infty]^2$. To define the persistence diagram, for each interval $I = [s, t] \in B$ we add a point at (s, t). The persistence diagram of a function is the multi-set

$$Dgm(f) = \{(s_I, t_I) : I \in B\}.$$

In figure 8 we show an example of a persistence diagram. The points close to the $y = x$ line are considered topological noise and the points farther away are relics of significant topological information. Since the points are all above the line, sometimes the diagram is rotated to have the $y = x$ line horizontal. For two continuous functions $f, g : \mathbb{X} \to \mathbb{R}$, we can compare the distance between the persistence diagrams of their sublevel level sets by using the Wasserstein distance

$$W_q(Dgm(f), Dgm(g)) = \inf_{\gamma}(\sum_{x \in Dgm(f)} \|x - \gamma(x)\|_{\infty}^{q})^{1/q}$$

as γ ranges through all possible bijections between multisets, $\gamma : Dgm(f) \to Dgm(g)$ and where $\|x - \gamma(x)\|_{\infty} = \max\{|b - b'|, |d - d'|\}$. For these definitions to work, one has to add infinitely many copies of points on the line $y = x$ in the plane. As q goes to infinity, we obtain what is known as the bottleneck distance. Formally, we consider $f \to Dgm(f)$, as a map of metric spaces and define stability to be the Lipschitz continuity of the map. We get the stability result

$$W_{\infty}(Dgm(f), Dgm(g)) \leq \|f - g\|_{\infty}$$

if functions f and g have finitely many values where the homology changes. The algebraic stability theorem provides a generalization of this result which lets us compare invariant of functions defined on different domains and invariants of filtrations which don't arise as level set filtrations such as the Rips complex.

Statistics

To properly be able to use persistent homology for data analysis, we need to address the quality of the diagrams we obtain. We would like to be able to answer the questions: What is the mean of a set of diagrams? Standard deviation? Can we use this to compute confidence intervals? Unfortunately, the space of persistence diagrams along with the (appropriate) Wasserstein distance gives us a metric space that is not easy to work with. We can define a Frechet mean and variance on the space of persistence diagrams with this metric but it is not, complete so the mean is not always unique. [6]

We can circumvent this by mapping persistence diagrams or barcodes to different space where classical statistics and machine learning techniques can be readily applied. The leading work around to this problem is to consider the *persistence landscape* of a data set as defined by Peter Bubenik in his 2012 paper *Statistical Topological Data Analysis using Persistence Landscapes*. The persistence landscape is the an injective mapping to the space of 1-Lipschitz functions. For a point $p = (x, y)$ in some diagram D, we consider the function

$$\Lambda_p(t) = \begin{cases} t - x + y & t \in [x - y, x] \\ x + y - t & t \in (x, x + y] \\ 0 & otherwise \end{cases} = \begin{cases} t - b & t \in [b, \frac{b+d}{2}] \\ d - t & t \in (\frac{b+d}{2}, d] \\ 0 & otherwise. \end{cases}$$

The persistent landscape of D is a summary of these functions given by

$$\lambda_D(k, t) = k \max_{p \in D} \Lambda_p(t), t \in [0, T], k \in \mathbb{N}$$

where $kmax$ is the k-th largest value in the set. Bubenik showed that there is a strong law of large numbers in the space of these landscapes and thus we can compute a unique mean. However, this mean may fail to be the image of a persistence diagram. In [8], Chazal et al., show that it is possible to define $(1 - \alpha)$- confidence sets for persistence diagrams as well as confidence bands around persistence landscapes.

Since the space of diagrams with the Wasserstein metric isn't a Hilbert space, it is impossible to readily employ current machine learning techniques such as PCA or SVM. Nevertheless, the 'kernel trick' can be used on $\mathbb{X}$ to implicitly define a Hilbert space by using persistence landscapes as is shown in [11]. Whether or not we can find can find other mappings besides the landscape that allow for meaningful results is still an open question. The interplay between persistent homology and statistics/machine learning is key for fully utilizing topological information in data analysis.

The *TDA* package in R

There are many libraries available for computing simplical complexes and persistent homology. These are mostly written mostly in C++ or Java for efficiency. Among the fastest are the libraries DIONYSUS and GUDHI, which can handle simplicies of size up to 1 billion. The statistical programming language R is a popular programming language for many data analyst. A group at Carnegie Mellon University has developed a package TDA available in the CRAN library which provides an R interface to use the algorithms available in the C++ libraries DIONYSUS, GUDHI, and PHAT. [10]For high dimensional data, computations should be done on a powerful CPU with more computing power than an average personal computer. Nevertheless there are still many things that one can do with smaller data sets. We show a few computations of geometric examples and finally on real world data sets.

The usual setting in TDA is that we are given a finite point cloud $\mathbb{X}$ and assume that the points in $\mathbb{X}$ lives on or 'near' some object X, usually a compact set, in some ambient space $\mathbb{R}^d$. We can mathematically formalize this by using small values of ϵ for the Hausdorff distance

$$d_H(X, \mathbb{X}) = \max\{\sup_{x \in X} \inf_{p \in \mathbb{X}} \|x - p\|, \sup_{p \in \mathbb{X}} \inf_{x \in X} \|p - x\|\} = \epsilon.$$

Torus Data Set

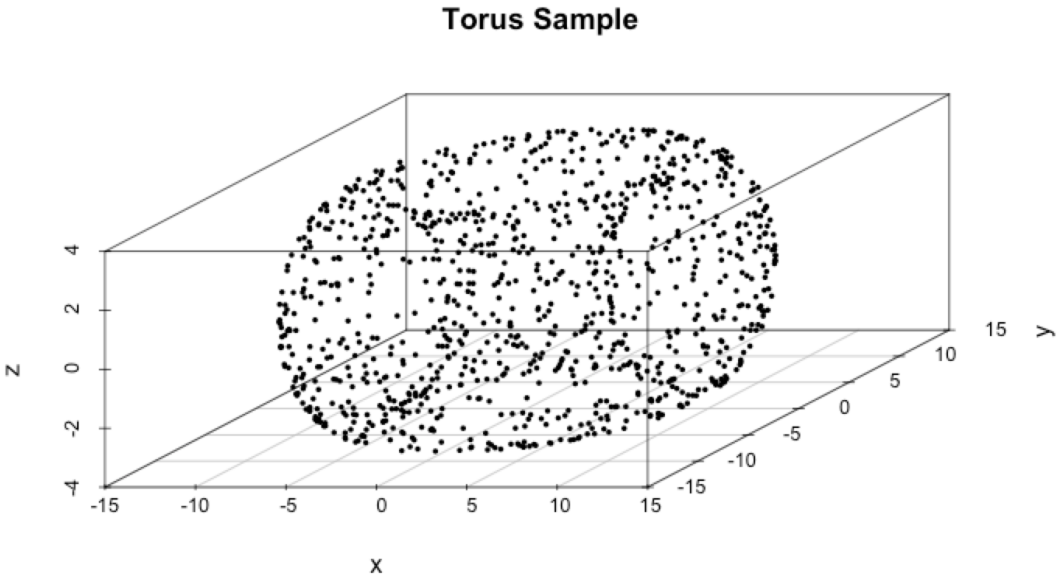


Figure 9: 1000 points sampled from a torus with t he radius from the center of the hole to the center of the torus tube = 4, and radius of torus tube = 2.

Our first data set consists of 1,000 points sampled from a uniform distribution on a torus show in figure 9. Our sample should be dense enough to reflect the features of the underlying torus. This example serves is a good reality check as many homological feature are to be detected. Using the function *ripsDiag* in the package TDA, we can

compute the Rips filtration of our point cloud using our choice of either the GUDHI or DIONYSUS libraries. Each library uses a different collection of algorithms to compute the filtration. Algorithms can be significantly faster depending on the size and type of the data set. The *ripsDiag* function with the DIONYSUS algorithm was relatively faster than GUDHI in this computation.

One can clearly see all the homological features of the torus contained in the summaries as shown in figure 10. We use color to help distinguish the homological dimensions. The 0th homological dimension tells us that there is one component. The 1st homological dimension, as shown in red, shows us the two 1 dimensional loops which are present in the torus. And finally the blue diamond, and the highest bar, show us the 2-dimensional void that is present in the tube of the torus. The features are clearly separated from the topological noise which is represented by all the small intervals and points near the $y = x$ line of the diagram.

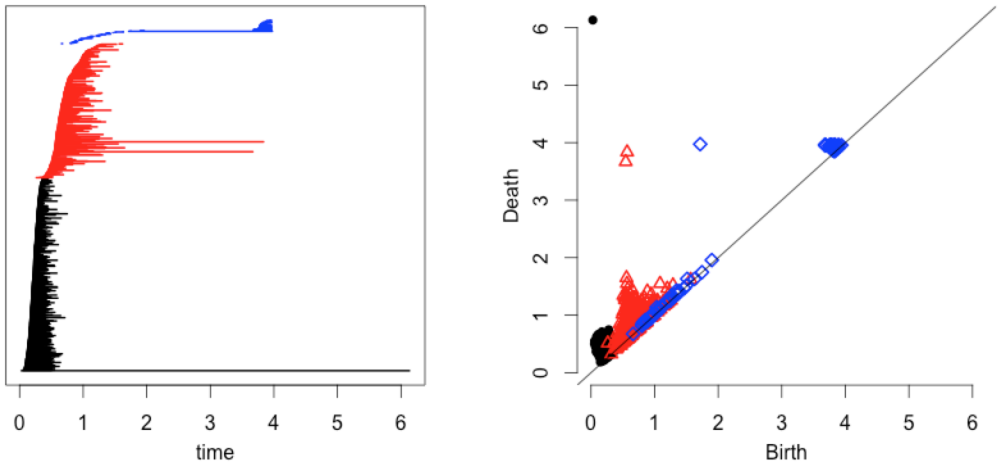


Figure 10: Barcode and Persistence Diagram for our torus sample

Three Rings Data Set

Our next data set is a collection of 300 points sampled with Gaussian noise from 3 adjacent rings in $\mathbb{R}^2$. Our two summaries indeed tell us a lot of information about the underlying space. In the diagram, the black points representing the 0th homology show us that our object is indeed one component yet the two smaller circles that appear for a short time suggest the presence of the two smaller adjacent rings. Also looking at the red triangles for the 1st homology, we see that these smaller components are loops of relatively the same size.

The biggest loop is represented by the third triangle and it's size relative to the others can be see by it's life span in the barcode. We can use the *gridDiag* function in order to compute the sublevel sets of which ever function we'd like. Here, we used the sublevel sets $X_r = f^{-1}(-\infty, t]$ that correspond to the distance function

$$f(p) = \min_{x_i \in \mathbb{X}}(\|x_i - p\|)$$

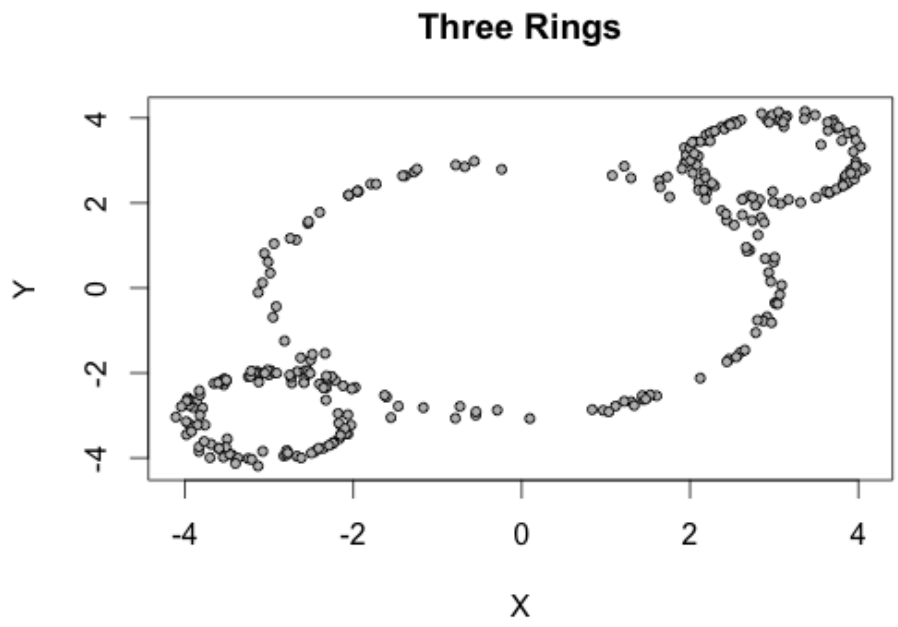


Figure 11: 300 points sampled from 3 adjacent rings in$\mathbb{R}^2$

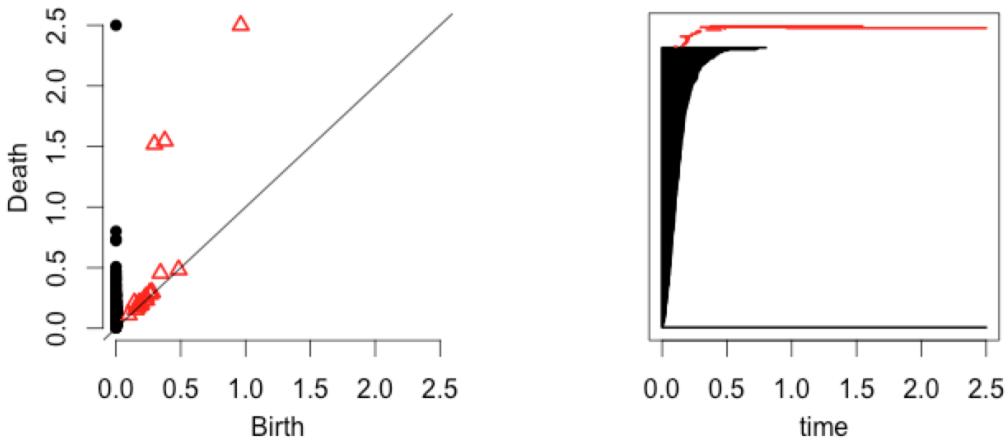


Figure 12: Persistence Diagram and Barcode for the Viertoris Rips filtration of the Three Rings point cloud data.

evaluated over a grid of points which is meant to resemble $\mathbb{R}^2$. We can visualize the sub-level sets in $\mathbb{R}^3$.

We obtain a obtain similar summaries to the one produced by the Rips filtration, but due to the scale of the grid over which the points where evaluated over (in this example the range divided the the x-axis [-5,5] and the y-axis [-5,5] by discrete points each with a distance of .025), we were limited by our computing power to produces any results which could point our the two smaller rings. We still obtain clear evidence of 3 1-dimensional

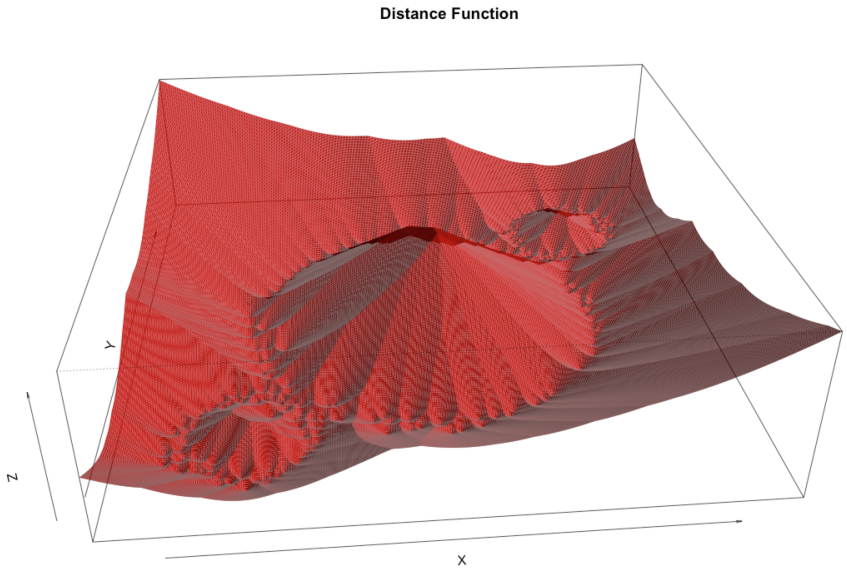


Figure 13: Visualization of the distance function to the Three Rings point cloud evaluated over a grid in $\mathbb{R}^2$.

loops as can be seen instantly in the barcode.

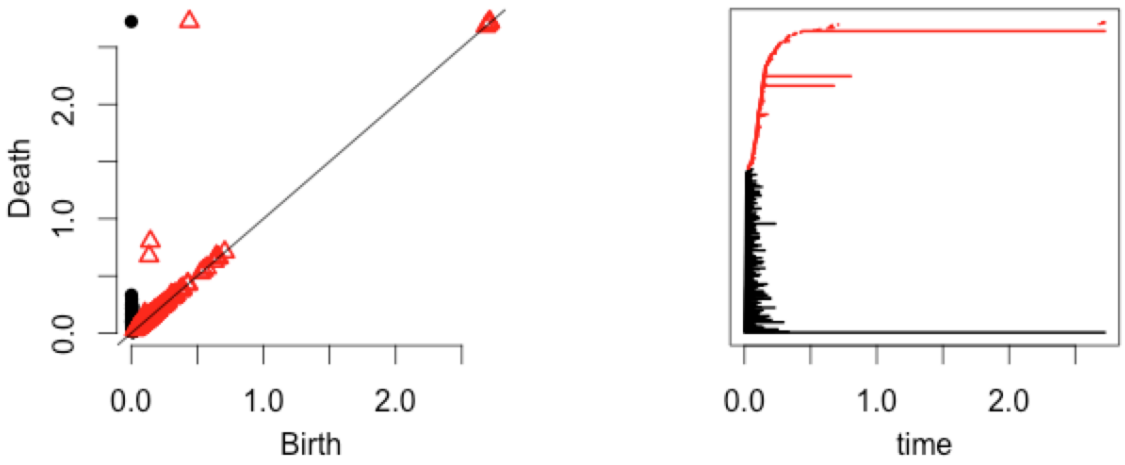


Figure 14: Persistence Diagram and Barcode for level sets of the distance function from the Three Rings point cloud data.

This distance function however is not a good estimator when the data set we are analyzing includes outliers. For instance, consider the same Three Rings point cloud we were analyzing with the addition of just one point as an outlier in the center of the big circle. Again we plot the distance function and compute the level set filtration.

The single point in the center is certainly a problem and one can see that by just adding just a few more strange points, we can miserably fail in finding the right homology of the underlying space. Instead, one can use the *distance to measure* function as define by [9] .

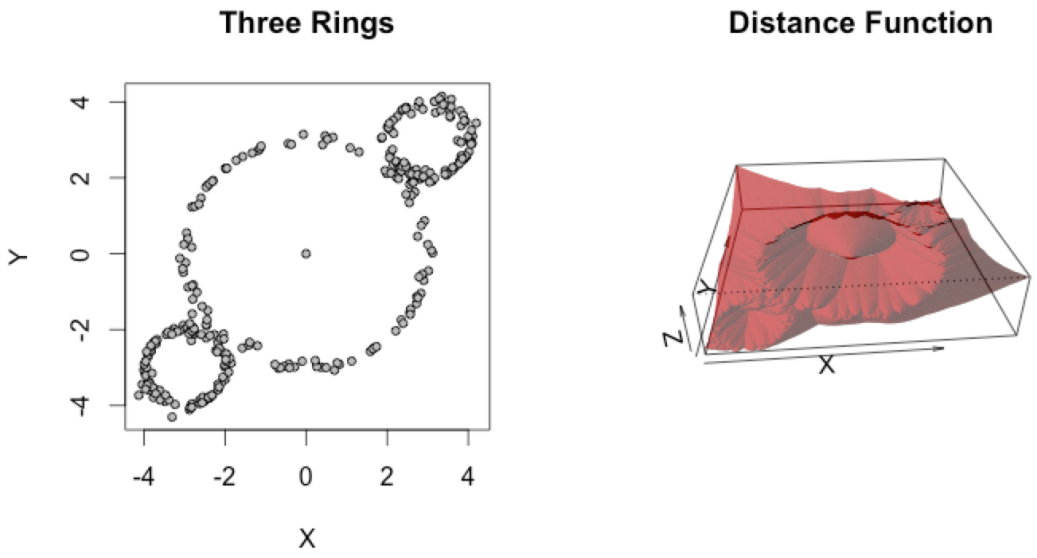


Figure 15: Three Rings With Outlier point cloud data and distance function.

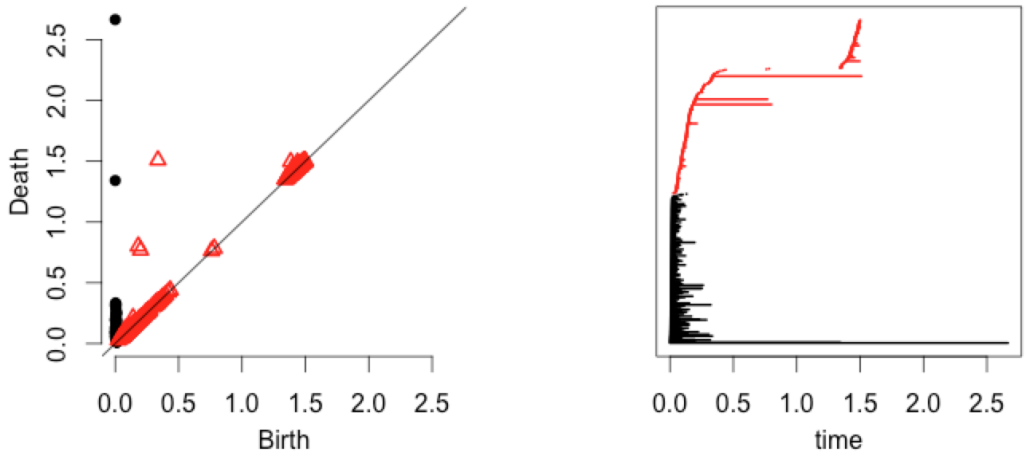


Figure 16: Persistence Diagram and Barcode for level sets of the distance function from the Three Rings With Outlier point cloud data.

This is a smoother version of the distance functions which is more robust to noise. Given $X = \{x_1, , x_n\}$, the empirical version of the distance to measure is given by

$$\hat{d}_{m_0}(y) = \sqrt{\frac{1}{k} \sum_{x_i \in N_k(y)} \|x_i - y\|^2},$$

where $k = \lceil m_0 n \rceil$ for some smoothing parameter $m_0$and $N_k(y)$ is the set containing the k nearest neighbors of y among $x_1, \ldots, x_n$.

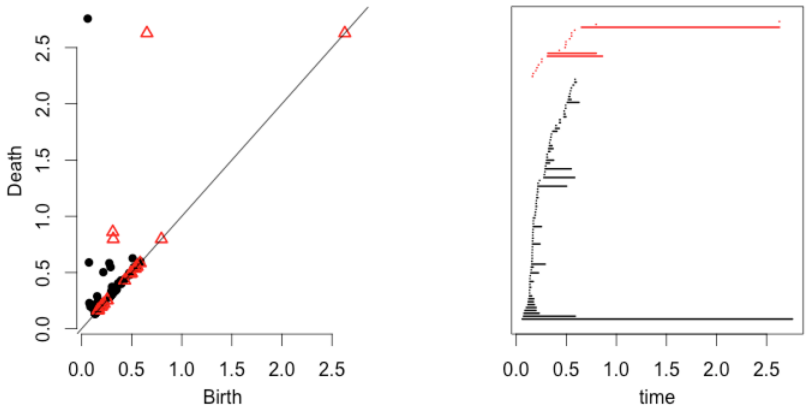


Figure 17: Persistence Diagram and Barcode for level sets of the Distance to Measure function from the Three Rings With Outlier point cloud data.

Iris Data Set

The Iris Flower data set is a famous data set introduce by Ronald Fisher. It contains 50 samples of each different species of plants: Setosa, Virginica and Versicolor. Four measurements are taken from each sample, the length and width of the sepals and also of the petals. Based only these four features, it is difficult to cluster the observations in an unsupervised manor. The Setosa is easily contained in it's own cluster but the Virginica and the Versicolor are difficult to separate without the labels of the species information as we can see from the scatter plot matrix in figure 19.

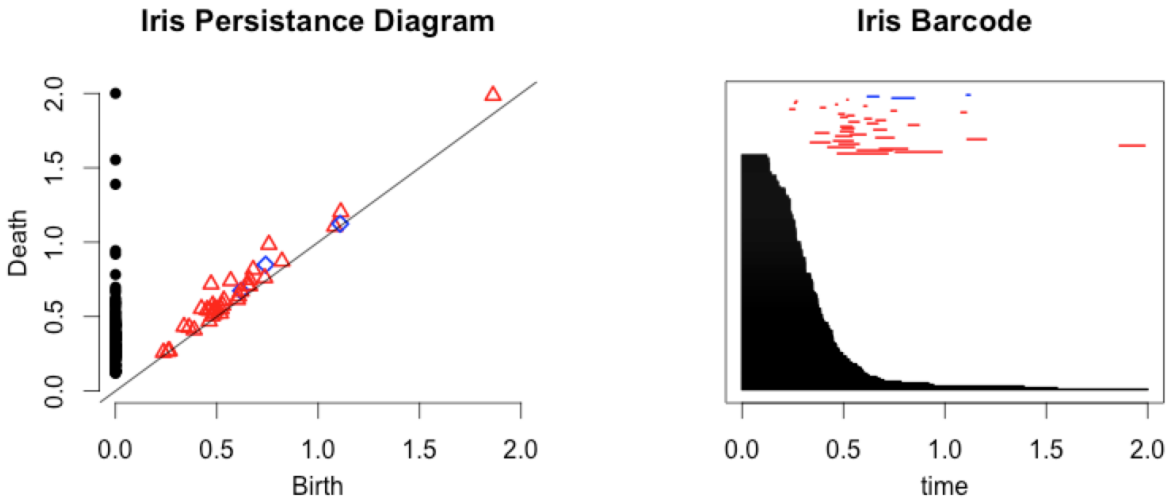


Figure 18: Topological summaries of the Rips filtration on the Iris data set.

The Setosa species can be easily separated because lower petal length and width. Computing the persistent homology of the data set, we obtain a summary which clearly depicts the existence of 3 clusters within the data set as we can tell by the 3 0-dimensional components present in the following summaries. The connected components of a graph

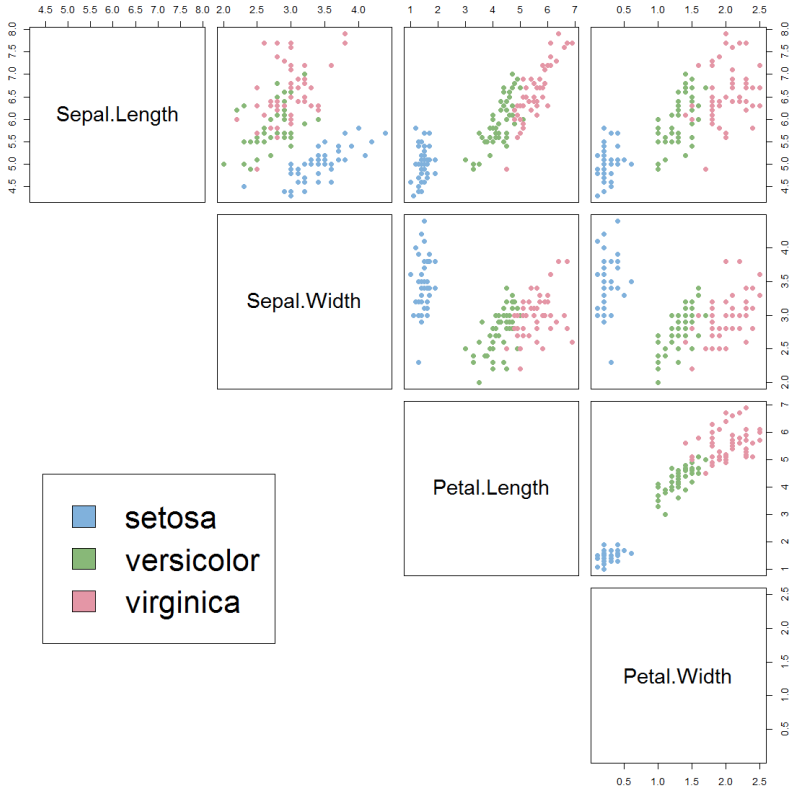


Figure 19: The scatter plot matrix of the Iris data set.

are naturally tied in with the ideas in single linkage cluster analysis. Yet single linkage clustering has no way of expressing any circular behavior of data points. With these persistent summaries, we can see that this data set indeed does not exhibit this circular behavior allowing us to further understand the 'shape' of the cluster and the data set in general.

Discussion

Persistent homology is an exciting new idea for modern mathematics and data analysis. In this paper we have shown how one can combinatorially encode the similarity of points within a data set and use this information to study the underlying topological properties of the data set. The topological properties of a data set become useful when trying to understand the intrinsic structure of the space from which the data is being sampled from. Qualitative information, such as homology, provides insight to the data set which is key for further exploration and analysis. The rich theory of persistence has many further directions to explore. Popular one being multidimensional persistence and *zig-zag persistence* which considers, instead of a filtration where functions only go 'to the right', functions in either direction. Persistence is being studied not only in an applied sense but theoretical as well. The development of the theory of persistent homology, along with the necessary integration of statistics and machine learning, will hopefully continue to show the benefits of topological information in data analysis as researchers and practitioners alike delve deep into the new field of topological data analysis.

References

[1] Michel Verleysen and Damien Franois. *The Curse of Dimensionality in Data Mining and Time Series Prediction*. IW ANN 2005, LNCS 3512, pp. 758 770, 2005. Springer-Verlag Berlin Heidelberg 2005.

[2] Carlsson, G (2009). *Topology and Data*. Bull. Amer. Math. Soc. 46, 255-308.

[3] Peter Bubenik, Vin de Silva, and Jonathan Scott. *Metrics for generalized persistence modules.* Foundations of Computational Mathematics, pages 131. http://arxiv.org/abs/1312.3829.

[4] William Crawley-Boevey *Decomposition of pointwise finite-dimensional persistence modules.* arXiv:1210.0819 [math.RT]

[5] Frederic Chazal, Vin de Silva, Marc Glisse, and Steve Oudot. *The structure and stability of persistence modules.* arXiv:1207.3674 [math.AT]

[6] Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, Larry Wasserman, Sivaraman Balakrishnan, Aarti Singh. *Confidence sets for persistence diagrams* Ann. Statist. Volume 42, Number 6 (2014), 2301-2339.

[7] Edelsbrunner, Herbert; Kirkpatrick, David G.; Seidel, Raimund (1983), *On the shape of a set of points in the plane* IEEE Transactions on Information Theory 29 (4): 551559, doi:10.1109/TIT.1983.1056714

[8] Frederic Chazal, Brittany Terese Fasy, Fabrizio Lecci, Alessandro Rinaldo, Aarti Singh, and Larry Wasserman. *On the Bootstrap for Persistence Diagrams and Landscapes.* arxiv:1311.0376.

[9] Frdric Chazal, Brittany T. Fasy, Fabrizio Lecci, Bertrand Michel, Alessandro Rinaldo, Larry Wasserman. *Robust Topological Inference: Distance To a Measure and Kernel Distance* arXiv:1412.7917 [cs.MS]

[10] Brittany Terese Fasy, Jisu Kim, Fabrizio Lecci, Clment Maria. *Introduction to the R package TDA*, arXiv:1411.1830 [cs.MS]

[11] Jan Reininghaus, Stefan Huber, Ulrich Bauer, Roland Kwitt *A Stable Multi-Scale Kernel for Topological Machine Learning* http://arxiv.org/pdf/1412.6821.pdf

[12] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. *Stability of persistence diagrams.* Discrete and Computational Geometry, 37(1):103120, 2007.