Homology and Data

Austin Eide

April 8, 2016

Abstract

Topological Data Analysis (TDA) is a growing field in applied mathematics. TDA encompasses a wide variety of topological methods which can be applied to the problem of analyzing large or noisy point-cloud data sets. One of the most important of these methods is persistent homology, which characterizes the shape of a data set by finding holes of various dimensions in the data. This paper covers the algebraic construction of simplicial homology groups, gives efficient methods for computing them, and provides some intuition into what these groups mean. It then exnteds these ideas to a data analysis setting by discussing the algebraic theory behind persistent homology and also giving some examples computed in the R package TDA, which efficiently computes the persistent homology of data.

1 Introduction

The power of topology comes from its abstraction; a topologist is interested in examining the most general features that characterize a space. These features are said to be *invariant* under homeomorphisms and allow the topologist to conclude that spaces which appear radically different are actually one and the same, a concept that gives rise to an often repeated joke about breakfast pastries and a certain type of mug.

On a more elegant note, topological invariance may also have been the concept which led Henri Poincaré to conclude that "mathematics is the art of giving the same name to different things." Poincaré is often credited as the mathematician who laid the foundations for modern algebraic topology, and important tools in the subject such as homotopy theory can be traced back to his work. Homotopy uses algebra to encode information about spaces in a simple way. *Homotopy equivalence* is a very strong topological relation, and, while it does not imply that spaces are homeomorphic, many important topological properties such path-connectdness and simple-connectedness are also homotopy invariant. However, while homotopy is a powerful theory, it is generally very difficult to compute explicitly the homotopy groups of a given space.

This fact renders homotopy difficult to use in particular settings, especially in applied or computational topology. Another theory attributed to Poincaré, though, proves to be nearly as powerful as homotopy while also allowing for efficient computation. This

theory—called homology—uses holes to classify space. To give an often-used example: a sphere is not a solid ball because a sphere bounds a hole while the ball does not; similarly, a circle is not a disk because the circle is a hollow loop while the disk is solid. This type of classification is quite similar to the type produced by homotopy. However, a key distinction is that where homotopy groups of a space X are constructed based on functions with certain properties from the n-dimensional sphere S^n to X, the homology groups of X rely solely upon information encoded in what is called a chain complex on X, making them more easily computable in most cases.

In this paper, we are interested in *simplicial homology*, or the homology groups of simplicial complexes. Working on simplicial complexes is advantageous in that a given complex can be described combinatorially (i.e., in terms of sets and subsets), meaning that simplicial complexes of reasonable size are relatively easy to store and do computations on. Moreover, simplicial homology can be used to reach broader results in topology, as many topological spaces can be triangulated by a simplicial complex that is homeomorphic to the original space. In Section 2 I give a brief overview of simplexes and simplicial complexes. From there, I describe a general construction of simplicial homology groups. Then, in Section 3, I discuss prevailing methods and algorithms for computing the homology of a simplicial complex.

Homology (and in particular, simplicial homology) is a valuable tool in large part because the homology of a given space can be easily computed. To this end, much of the recent work in applied topology has dealt with efficient methods for computing homology groups, especially as it pertains to data analysis. In this kind of topological data analysis, we begin with a set of vectors $X = \{x_1, x_2, \ldots, x_n\}$ in $\mathbb{R}^m$ (point-cloud data) and suppose that this data has been sampled from some underlying space P also in $\mathbb{R}^m$ [9]. By examining the homological features of the data X, we hope to gain some insight into what P might look like. To do this, we construct a simplicial complex K on X according to certain parameters and conditions for connecting two points by an edge, and then compute the homology of this complex. Analyses like this are especially useful in examining high-dimensional data which cannot be plotted or viualized directly: we can detect the presence of homological features in such data without ever actually seeing them. Recent work in image recognition and protein analysis speaks to this [3,7].

Analyzing data topologically is beneficial because the topological features of the data X and the underlying space P do not depend on coordinates or metrics which "may not be natural [to the data] in any sense," as Carlsson puts it in [3]. Rather, a topological analysis directly examines the shape and fundamental geometric properties of a space, which are impervious to metric or coordinate choices. However, in the setup mentioned in the previous paragraph, the reader might notice that in constructing a complex on a set of data we are nonetheless forced to make a choice of parameter which may or may not be optimal (how do we decide where to draw an edge, and where not to?). Rather than attempting to optimize parameters, topological data analysis has generally turned

to the idea of *persistence* to solve this problem. In persistent homology, we construct a sequence of nested complexes $K_1 \subset K_2 \subset \cdots \subset K_m$ on a set of point-cloud data X, where each K_i is a simplicial complex corresponding to a certain parameter choice. We then compute homology groups for each K_i to determine which homological features persist through multiple choices of the parameter and which are merely particularities of a small range of choices. In Section 4, I give a more thorough treatment of persistent homology. This section also contains some results from experimenting with sampled data in the R package TDA, which uses efficient algorithms to compute the persistent homology of point-cloud data.

2 Simplicial Complexes and Homology

Intuitively, a k-simplex is the k-dimensional analogue of a triangle. A 2-simplex consists of 3 affinely independent points and the interior of the triangle which they comprise; a 3-simplex, or a tetrahedron, is constructed similarly with 4 points. Simplicial complexes (of dimension n) are simply collections of simplexes (of dimensions $0, \ldots, n$) which satisfy certain intersection conditions. These concepts are laid out more formally below. Note that the constructions in this section are fairly graph theoretic, as I believe this provides the most straightforward introduction to simplicial homology.

2.1 Simplexes

We define a k-simplex as follows:

Definition Let $\{v_0, \ldots, v_k\}$ be a set of affinely independent vectors in $\mathbb{R}^n$. The k-dimensional simplex with vertices $\{v_0, \ldots, v_k\}$ is the set

$$\Delta_k = \{c_0 v_0 + c_1 v_1 + \cdots + c_k v_k \in \mathbb{R}^n : c_0 + c_1 + \cdots + c_k = 1, c_i \geq 0\}.$$

The vectors $\{v_0, \ldots, v_k\}$ are affinely independent if $v_1 - v_0, v_2 - v_0, \ldots, v_k - v_0$ are linearly indepedent. For any point in Δ_k, the real numbers $c_0, c_1, \ldots, c_k$ are uniquely determined and are called the *barycentric coordinates* of the point [2].

I don't go into much detail here because for the purposes of this paper we will be working with simplicial complexes in a general topological space, rather than a real vector space. However, a nice explanation of affine subspaces and affine independence is given in Giblin's *Graphs, Surfaces, and Homology* [10] for the reader who would like a more rigorous grounding in this. Here, a solid intuition will suffice. Essentially, the condition that $\{v_0, \ldots, v_k\}$ be affinely independent means that these vectors do not all lie on a single $(k-1)$-dimensional plane. Simplexes are analogous to triangles, and we could not construct a triangle from 3 points that all lie on a single line (nor a tetrahedron from 4 points on a single plane). A more illuminating way to think of the definition is that the k-simplex on a set of $k+1$ affinely independent vertices is the convex hull of those

vertices, or the smallest closed set containing all of the vertices. Naturally, for 2 vertices this set is the closed line segment between them; for 3 vertices it is the closed triangle defined by the vertices; and so on.

Note that in the definition, a k-simplex is determined entirely by the given $k+1$ vertices. So, while the realization of the 2-simplex with vertices v_0, v_1, and v_2 as a triangle is helpful for visualization, we could describe it equally well with the set $\{v_0, v_1, v_2\}$. From here on, we'll refer to such a set as a *vertex set*. In general, we'll refer to simplexes with the notation $(v_0 v_1 \dots v_k)$, and the vertex set containing the v_i will be considered a separate object.

This new notation gives rise to another important aspect of simplexes and simplicial complexes: *orientation*. Describing simplexes by their vertex sets imposes a natural ordering on the vertices. For any k-simplex, then, we have $(k+1)!$ distinct orderings of vertices. A 1-simplex (which from the definition is a line segment) has two distinct orderings, as its vertex set contains two elements. In this case, the two orderings also give two orientations of the simplex, which we can visualize by placing a direction on the line segment defined by the two vertices. For $(v_0 v_1)$, we have

and for $(v_1 v_0)$

The association between ordering and orientation in the case of the 1-simplex is clear: we have two orderings of vertices which each give a distinct orientation of the line segment between the vertices. But what about higher dimensional simplexes? $(v_0 v_1 v_2)$ and $(v_2 v_0 v_1)$ are two distinct orderings of the vertices of a 2-simplex, yet they correspond to the same orientation given that the vertices of the resulting triangle are read in a "clockwise" direction. On the other hand, $(v_0 v_2 v_1)$ gives an opposite, or "counterclockwise" orientation. Thus, although there are six possible orderings of vertices for a 2-simplex, it isn't hard to see that there are still only two possible orientations, each corresponding to the "direction" in which the vertices of the triangle are read. In general, any k-simplex with $k \geq 1$ has two possible orientations. Showing this rigorously requires use of some details about affine subspaces which we have foregone. The main idea is that, given an ordered vertex set $\{v_0, v_1, \dots, v_n\}$ any *even* permutation of these elements gives one orientation, while any *odd* permutation gives the opposite orientation. Geometrically, we can assign one rotational direction to the even permutations and the opposite rotational direction to the odd permutations. Orientation becomes very important once we place an algebraic structure onto simplicial complexes.

A final piece of important information regarding simplexes is the notion of a *face*. The definition is as follows:

Definition A *face* of a simplex with vertex set $V = \{v_0, v_1, \ldots, v_n\}$ is a simplex whose vertices form a nonempty subset of V.

In this sense, then, v_0 and v_1 are both faces of the simplex (v_0v_1) above.

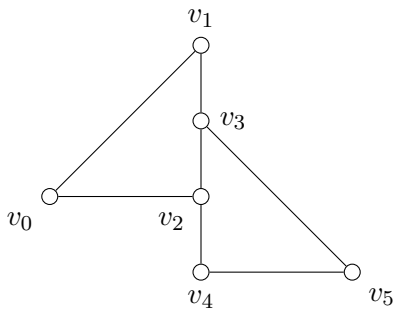
2.2 Simplicial Complexes

From simplexes, we are able to construct simplicial complexes, which are the structures needed in order to approximate surfaces. We define a simplicial complex:

Definition A *simplicial complex* is a finite set K of simplexes for which the following are true:

1. If $s \in K$ and t is a face of s, then $t \in K$
2. If $s, t \in K$, then $s \cap t$ is either empty or is a face of both s and t

The second condition is often called the *intersection condition*, and it delineates precisely how simplexes can be combined into a simplicial complex. Simply, put, simplexes must either share an entire common face (be it a point, an edge, a triangle, etc.) or not be connected at all. So, for example, the figure below is not a simplicial complex because the 2-simplexes $(v_0v_1v_2)$ and $(v_3v_4v_5)$ do not satisfy the intersection condition, as the edge $(v_2v_3) = (v_0v_1v_2) \cap (v_3v_4v_5)$ is not a face of either 2-simplex.



The first condition of the definition gives us a connection between simplicial complexes and graphs. Consider the following definition.

Definition The *r-skeleton* of a simplicial complex K, denoted K^r, is the set of subcomplexes of K with dimension $\leq r$.

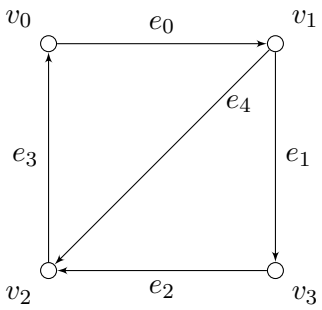
Note that a subcomplex of K is simply a subset of K which is itself a simplicial complex. In particular, we will be interested in the 1-skeletons of simplicial complexes. Take any k-simplex with $k \geq 1$; using the definition of a face, it is easy to decompose the simplex into a union of 1-simplexes which can naturally be expressed as a graph. Here, we will take some informal definitions regarding graphs. A *graph* is a vertex set V along with an edge set E, where each edge in E corresponds to exactly two vertices in V. A *connected*

graph is a graph in which there exists a path between any two vertices in V. Note that any graph G may be the union of multiple *connected components*, or subgraphs which are disconnected from one another. These definitions will allow us to draw an important conclusion about how we should interpret certain homological features (see Propositions 2.3 and 2.4).

Before moving on to these results, though, we need to describe an algebraic structure on simplicial complexes, which I handle in the next section.

2.2.1 Algebra on Simplicial Complexes

With homology, we formalize the notion of holes and voids of a given dimension in a space. Homology accomplishes this by identifying the boundaries of these holes and voids. On a simplicial complex, the structures that bound holes are simplexes, and, in particular, combinations of simplexes. In order to compute homology, we first must formalize the way we describe these combinations. This follows in a natural way. Consider the following oriented complex:



The cycle $v_1 \to v_2 \to v_3 \to v_1$ can be expressed as an integral linear combination of its edges. Note that, following the order of the vertices listed, we would travel "backwards" along edges e_2 and e_1, while going forwards on e_4. We express these directions with a coefficient of ± 1 associated to each edge in the combination. For the cycle $v_1 \to v_2 \to v_3 \to v_1$, the proper expression is

$$e_4 + -e_2 + -e_1.$$

Call such a combination a 1-chain, as it is a combination of 1-simplexes. In general, a p-chain on a simplicial complex is an integral linear combination of p-simplexes. Note that a p-chain need not be a cycle in general, but clearly only p-cycles are capable of bounding holes (this is akin to the graph theoretic notion of a cycle on a graph bounding a region). In general, a chain of p-simplexes $\sigma_1, \ldots, \sigma_k$ is written

$$\lambda_1\sigma_1 + \cdots + \lambda_k\sigma_k$$

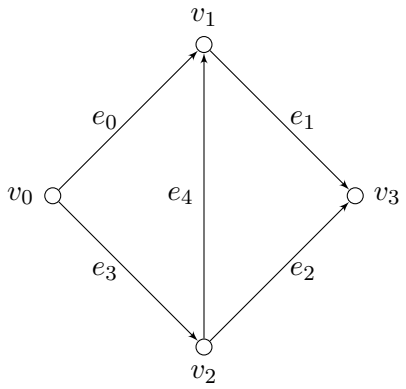
where the λ_i are coefficients from a ring R. In the context of this paper, these coefficients will always come from $\{-1, 0, 1\}$.

With this, we can construct homology groups.

2.3 Homology Groups

Up to this point, the notion of what a homology group really is, and what it actually measures, has been left somewhat vague. This section will be devoted to explaining how these groups function.

Recall the previous section's discussion on holes and boundaries in a simplicial complex. With the notion of a p-chain, we can formally describe the combinations of simplexes which bound holes—namely, p-cycles. However, this is not sufficient to give the homology of a simplicial complex, as we aren't yet able to distinguish between p-cycles that bound holes and ones that don't. Consider the following simplicial complex:



Take $(v_0v_1v_2)$ to be a 2-simplex, but do *not* consider the triangle on v_1, v_2, and v_3 a 2-simplex. While these are both "triangles" in the geometric sense, they are actually quite different as parts of an overall simplicial complex. The triangle with vertices v_0, v_1, and v_2 is a 2-simplex, meaning it contains the points in its interior. The triangle with vertices v_1, v_2, and v_3 is not a 2-simplex, and thus does not contain interior points. Thus, the cycle $e_1 - e_2 + e_4$ goes around a hole, while the cycle $e_0 - e_4 - e_3$ does not.

The ability to distinguish bounding from non-bounding cycles is the final piece we need to construct homology groups. The next sections are devoted to giving the algebraic structure of this concept.

2.3.1 Chain Groups

Definition Let K be an n-dimensional oriented simplicial complex and α_p be the number of p-simplexes of K; these simplexes will be given by $\sigma_p^1, \ldots, \sigma_p^{\alpha_p}$. The p^{th} *chain group of* K, denoted $C_p(K)$, is defined as the free abelian group on the set $\left\{\sigma_p^1, \ldots, \sigma_p^{\alpha_p}\right\}$.

From the definition, the set $\left\{\sigma_p^1, \ldots, \sigma_p^{\alpha_p}\right\}$ is a *basis* for $C_p(K)$, so that any $c \in C_p(K)$ can be expressed uniquely by some $\sum_{i=1}^{a_p} \lambda_i \sigma_p^i$ where $\lambda_i \in \mathbb{Z}$ for all i. These chain groups allow us to treat p-simplexes differently in different dimensions. The trouble in the bounding/non-bounding triangle example above arose because we were unable to formally state that the 1-chain $e_0 - e_4 - e_3$ was actually the *boundary* of a 2-simplex, while the chain $e_1 - e_2 + e_4$ was not. Chain groups, along with the next definitions, are able to make this distinction.

Definition The *boundary operator* ∂_p of an oriented p-simplex $\sigma_p = (v_0 v_1, \ldots, v_p)$ gives the $(p-1)$-chain

$$\partial_p(\sigma_p) = \sum_{i=0}^{p} (-1)^i (v_0 \ldots \widehat{v_i} \ldots v_p)$$

where $\widehat{v_i}$ denotes the omission of vertex v_i.

To work out a quick example, consider the first figure in this section. The boundary of the 2-simplex $(v_0 v_1 v_2)$ is

$$\partial_2(v_0 v_1 v_2) = (v_1 v_2) + -(v_0 v_2) + (v_0 v_1) = -e_4 - e_3 + e_0$$

which is how we naturally described the boundary before. We generalize ∂_p with the next definition.

Definition The *boundary homomorphism* $\partial_p : C_p(K) \to C_{p-1}(K)$ for an n-dimensional simplicial complex K is defined by

$$\partial_p \sum \lambda_i \sigma_p^i = \sum \lambda_i \partial_p(\sigma_p^i)$$

for $0 \leq p \leq n$ and is defined to be the trivial homomorphism elsewhere.

Perhaps the most important observation to make about the boundary homomorphism is that $\ker \partial_p$ is precisely the set of p-cycles of the complex K, which are the only structures on a simplicial complex that might capture holes. Seeing this for the case $p = 1$ is straightforward. If the 1-simplexes $(v_0 v_1)$, $(v_1 v_2)$, $(v_2 v_1)$ are subjected to the boundary homomorphism as a 1-chain, we get

$$\partial_1(v_0 v_1 + v_1 v_2 + v_2 v_1) = (v_1 - v_0) + (v_2 - v_1) + (v_0 - v_2) = 0.$$

This result movitates the following theorem, which gives a relationship between the image of a boundary homomorphism and the kernel of the successive boundary homomorphism.

Theorem 2.1. *The composition of any two consecutive boundary homomorphisms maps all elements to* 0*. That is, for any* p*,* $\partial_{p-1} \circ \partial_p : C_p(K) \to C_{p-2}(K)$ *is trivial.*

Proof. When for $p = 0$ and $p = 1$, this follows immediately, as the ∂_p is defined to be trivial for $p < 0$.

Take p to be greater than or equal to 2. For some p-simplex $\sigma = (v_0 v_1 \dots v_p)$ we have

$$\partial_{p-1}(\partial_p \sigma) = \sum_{i=0}^{p} (-1)^i \partial_{p-1}(v_0 \dots \widehat{v_i} \dots v_p).$$

This is true because $\partial_p(\sigma)$ is given as the $(p-1)$-chain $\sum_{i=0}^{p}(-1)^i(v_0 \dots \widehat{v_i} \dots v_p)$. We then use the definition of the boundary homomorphism to get the right-hand term of the above equation.

Now, in general, applying the boundary operator to a p-simplex gives a $(p-1)$-chain with $(p+1)$ distinct summands (one summand for each omission of a vertex). Thus, for each i in the outer sum above, we have an inner summation (from ∂_{p-1}) of $(p-1)+1 = p$ distinct $(p-2)$-chains. Because i ranges from 0 to p, this gives $p(p+1)$ summed terms in the composition $\partial_{p-1} \circ \partial_p$. This is important, as it gives an even number of summands. We propose that each $(p-2)$-dimensional face of σ occurs exactly twice in this sum. In general, such a face is given by $(v_0 \dots \widehat{v_i} \dots \widehat{v_j} \dots v_p)$ where $i < j$ and will appear in terms

$$(-1)^i \partial_{p-1}(v_0 \dots \widehat{v_i} \dots v_p) = (-1)^i \sum_{j=0}^{p} (v_0 \dots \widehat{v_i} \dots \widehat{v_j} \dots v_p)$$

and

$$(-1)^j \partial_{p-1}(v_0 \dots \widehat{v_j} \dots v_p) = (-1)^j \sum_{i=0}^{p} (v_0 \dots \widehat{v_i} \dots \widehat{v_j} \dots v_p).$$

The coefficient of $(v_0 \dots \widehat{v_i} \dots \widehat{v_j} \dots v_p)$ in the first sum is actually $(-1)^i(-1)^{j-1}$, because once we remove v_i to form $(v_0 \dots \widehat{v_i} \dots \widehat{v_j} \dots v_p)$, every index $k > i$ must be decreased by one to account for the shift. Because $j > i$, the coefficient becomes $(j-1)$. The coefficient in the second sum is simply $(-1)^j(-1)^i$, giving a total coefficient of $(-1)^i(-1)^{j-1} + (-1)^j(-1)^i = 0$, thus proving that $\partial_{p-1} \circ \partial_p = 0$. □

From this theorem, it follows immediately that $\operatorname{img} \partial_p \subseteq \ker \partial_{p-1}$. We say that the chain groups $C_i(K)$ along with boundary homomorphisms ∂_i form the *chain complex*

$$\dots \to 0 \xrightarrow{\partial_{n+1}} C_n(K) \xrightarrow{\partial_n} C_{n-1}(K) \xrightarrow{\partial_{n-1}} \dots \xrightarrow{\partial_1(K)} C_0(K) \xrightarrow{\partial_0} 0 \to \dots .$$

2.3.2 Boundary, Cycle, and Homology Groups

The relation $\operatorname{img} \partial_p \subseteq \ker \partial_{p-1}$ provides a natural way to distinguish between bounding and non-bounding p-cycles: non-bounding p-cycles are elements of $\ker \partial_p$ that are not elements of $\operatorname{img} \partial_{p+1}$.

Denote $\ker \partial_p$ as $Z_p(K)$ (the pth *cycle group*) and $\text{img } \partial_{p+1}$ as $B_p(K)$ (the pth *boundary group*). Note that $B_p(K)$ is a subgroup of $Z_p(K)$. (Each of these groups, as subgroups of the free and finitely generated group $C_p(K)$, are also free and finitely generated, which will be important later.) The quotient group $H_p(K) = Z_p(K)/B_p(K)$ is the pth *homology group* of K. We are working solely with abelian groups, so $Z_p(K)$ and $B_p(K)$ are both normal subgroups of $C_p(K)$.

By taking the quotient of $Z_p(K)$ and a subgroup $B_p(K)$ we are essentially modding out all p-cycles of K by the p-cycles that are actually boundaries of $(p+1)$-simplexes. The quotient group $H_p(K)$ gives us what is left over—in particular, the non-bounding p-cycles. The rank of the free part of $H_p(K)$, sometimes called the pth Betti number β_p, gives the number of p-dimensional holes in K. A couple of basic results surrounding these groups follow.

Proposition 2.2. *For K an n-dimensional simplicial complex, the n^{th} homology group $H_n(K)$ is a free and finitely generated abelian group.*

Proof. If K is an n-dimensional simplicial complex, then $C_{n+1}(K) = 0$ and thus we have $\text{img } \partial_{n+1} = B_n(K) = 0$. We then have $H_n(K) = Z_n(K)$, and $Z_n(K)$ is a free and finitely generated abelian group, as it is a subgroup of $C_n(K)$. □

Elements of $H_p(K)$ are cosets of the form $z + B_p(K)$, where $z \in Z_p(K)$. Let each coset be denoted by the *homology class* $[z]$, so that any cycle $z' \in [z]$ is equal to $z + b$ for some $b \in B_p(K)$. Two cycles z and z' are *homologous* if and only if $z - z' \in B_p(K)$. We can now reach some results about 0th homology groups.

Proposition 2.3. *Two vertices v and v' of a simplicial complex K are homologous if and only if they lie in the same connected component of the graph K^1 (the 1-skeleton of K).*

Proof. Suppose v and v' lie in the same connected component of K^1. Then there exists a path between then, say from v to v'. This path can be thought of as a 1-chain in the context of the simplicial complex K; call this 1-chain c. Then $\partial_1(c) = v' - v$, which means $v' - v \in \text{img } \partial_1 = B_0(K)$, so the two vertices are homologous. Now, suppose v is homologous to v'. We need to construct a path in K^1 between the two vertices, and can proceed by induction. For the case $n = 1$, suppose $v' - v = \partial_1(c)$, where the number of 1-simplexes in c is $|c| = 1$. Then c is the required path between v and v'. Now, suppose $|c|$ is an arbitrary positive integer and let the hypothesis hold for all 1-chains of length less than or equal to $|c|$. If $\partial_1(c) = v' - v$ then the 1-chain c must contain an edge e such that e has v' as an endpoint. Let $e = (v''v')$. Then the 1-chain $c - e$ is a path from v to v'' by the inductive hypothesis, and thus $(c - e) + e = c$ is a path from v to v'. □

Proposition 2.4. *$H_0(K)$ is freely generated by the distinct homology classes $[v_i]$ of vertices of K.*

Proof. Note that because $\partial_0 : C_0(K) \to 0$, we have $C_0(K) = Z_0(K)$ and thus it follows that $H_0(K) = C_0(K)/B_0(K)$. From Proposition 2.3, there is a single homology class for each connected component of K[1]. Thus, the group of homology classes $H_0(K)$ must be generated by the distinct homology classes of the vertices of K.

Now, to show that $H_0(K)$ is freely generated by these classes. Let $v_1, \ldots, v_k$ be representative vertices of each distinct homology class and suppose there exist integers λ_i for $1 \leq i \leq k$ such that $\lambda_1 v_1 + \cdots + \lambda_k v_k = b$ for some $b \in B_0(K)$ (that is, the combination of v_i is equal to 0 in the homology group). We can write $b = \partial_1(c)$ for some 1-chain $c \in C_1(K)$. Thus $\lambda_1 v_1 + \cdots + \lambda_k v_k = \partial_1(c)$. Write $c = c_1 + \cdots + c_k$ so that c_i only contains 1-simplexes in the connected component K_i corresponding to vertex v_i, so $\lambda_1 v_1 + \cdots + \lambda_k v_k = \partial_1(c_1) + \cdots + \partial_1(c_k)$. Because $\partial_1(c_i)$ will not contain any vertices that are not in K_i, we may conclude that $\lambda_i v_i = \partial_1(c_i)$ for all i. But this means $\lambda_i v_i \in B_0(K)$, which can only be the case if $\lambda_i = 0$. Thus, having $\lambda_1 v_1 + \cdots + \lambda_k v_k = b$ for some $b \in B_0(K)$ implies that $\lambda_i = 0$ for all i, so the distinct homology classes $[v_i]$ are a basis for $H_0(K)$. □

Using the structure theorem for finitely generated abelian groups, we can conclude that $H_0(K) \simeq \mathbb{Z}^k$, where k is the number of connected components of K[1]. Similarly, the top-dimensional group $H_n(K)$ is isomorphic to $\mathbb{Z}^m$, where m is the rank of $H_n(K)$. (This positive integer m corresponds to the number of distinct homological features of dimension n on K, which may be loops, voids, etc.) These statements follow directly from the result that $H_1(K)$ and $H_n(K)$ are free abelian groups, which implies they contain no torsion elements. Groups of dimension p with $1 < p < n$ are not necessarily free abelian, and so could have torsion; this phenomenon could be interpreted as giving some sense of the "twistedness" of the space, although we don't explore that idea here. Going forward, we will be concerned with finding the structure of each homology group $H_p(K)$, and possibly finding representative elements of these groups.

The next section deals with computations of homology groups. There, we will describe some of the processes by which we can use the results of this section to analyze the topological features of particular simplicial complexes, surfaces, and eventually of data.

3 Computation

In this section, we'll examine two basic methods for computing the homology groups of a simplicial complex. The first method uses the algebraic properties of the chain complex to calculate the ranks of each homology group $H_p(K)$. This problem amounts to doing matrix reduction, and I give a small example to illustrate. The second method is an incremental algorithm which depends only on the particular simplicial complex K. I conclude the section by discussing the relative efficiency of each of these methods.

3.1 The Smith Normal Form Method

To this point, all definitions relating to homology have been formulated around finitely generated abelian groups. In many cases, however, homological algebra is generalized using R-modules rather than abelian groups. A left R-module over a ring R with unity is an abelian group A with a function $\mathcal{P} : R \times A \to A$ written $\mathcal{P}(r, a) = ra$ such that for $r, r' \in R$ and $a, a' \in A$,

1. $(r + r')a = ra + r'a$
2. $(rr')a = r(r'a)$
3. $r(a + a') = ra + ra'$
4. $1_R a = a$ [12]

The definitions for a right R-module and a module over a commutative ring R follow similarly. Note that any abelian group is actually a $\mathbb{Z}$-module, where we natrually interpret na to be a added to itself n times. Thus, the construction of homology groups in terms of abelian groups can equivalently be put in terms of $\mathbb{Z}$-modules. In this paper, I only consider modules over $\mathbb{Z}$. However, some general results about R-modules, like the ones that follow, can be applied to our situation, so I chose to introduce the definition to make things clearer.

3.1.1 The Smith Normal Form and Presentations

Proposition 3.1. *Let G be a free R-module with basis $g_1, \ldots, g_k$ and let H be another R-module. Given $h_1, \ldots h_k \in H$, there exists a unique module homomorphism $\varphi : G \to H$ such that $\varphi(g_i) = h_i$ for all $1 \leq i \leq k$.*

Proof. Define $\varphi : G \to H$ such that $\varphi(g_i) = h_i$. Extend φ linearly so that we have $\varphi(\lambda_i g_i) = \lambda_i \varphi(g_i) = \lambda_i h_i$ and further $\varphi(\lambda_1 g_1 + \cdots + \lambda_k g_k) = \lambda_1 h_1 + \lambda_k h_k$ for all $\lambda_i \in R$. Because G is free, the λ_i of any element $\lambda_1 g_1 + \cdots + \lambda_k g_k \in G$ are unique. This implies φ is well-defined, and thus is a homomorphism. This homomorphism is also unique. Suppose ρ is another homomorphism satisfying the given conditions; then $\rho = \varphi$. □

In our case, Proposition 3.1 implies that, because each $C_p(K)$ is a free abelian group, the boundary homomorphism ∂_p is uniquely determined by the basis of $C_p(K)$. These bases are finite, so each homomorphism ∂_p may be represented by a corresponding matrix, where n_p is the rank of $C_p(K)$. Recall that $Z_p(K) = \ker \partial_p$ and $B_p(K) = \operatorname{img} \partial_{p+1}$. Thus, determining the structure (i.e., rank and basis elements) of these groups amounts to calculating row and column spaces of the boundary matrices corresponding to each ∂_p.

In general, this can be achieved by standard Gaussian elimination. However, recall that in our case each ∂_p corresponds to a matrix over a ring R (here, $R = \mathbb{Z}$) rather than a field; this means that the ususal row reduction algorithm is not necessarily sufficient to reduce these matrices. In particular, without the condition that every entry in a matrix

is a unit, we are not always able to reach a pivot point in each row merely by scaling and adding other rows. There is, however, a standard algorithm that reduces matrices over a principal ideal domain (PID) to a canonical form. This form is called the Smith Normal Form, and it has very nice algebraic properties connected to the structure theorem for finitely generated R-modules. We exploit some of these properties in the computation of simplicial homology groups. I first give a general sketch of the Smith Normal Form as it relates to presentations of abelian groups, and then apply it to our situation with an example.

Let R be a PID and G a finitely generated R-module with generators $\{g_1, \ldots, g_n\}$. We have a surjective module homomorphism $\varphi : R^n \to G$ by $\varphi(r_1, \ldots, r_n) = \sum_i = r_i g_i$. If $K = \ker \varphi$, then the fundamental isomorphism theorem for modules gives $G \simeq R/K$. We say that element of K are *relations* on G, in the sense that if $(a_1 \ldots, a_n) \in K$, then $\varphi(a_1 \ldots, a_n) = \sum_{i=1}^{n} a_i g_i = 0$. If K is finitely generated, we let $\{k_1, \ldots, k_m\}$ be the generators of K, where $k_i = (a_{i1}, \ldots, a_{in})$. We can naturally write the k_i in an $m \times n$ *relation matrix* $A = (a_{ij})$, so that k_i comprises the ith row of A. (Note that doing this fixes the orderings of the generating sets of G and K, so we treat these sets as ordered henceforth.) The structure of this matrix A gives the structure of the R-module (re: abelian group) G. We can always obtain a nice structure for A using the following results.

Proposition 3.2. *Let A be the relation matrix for a finitely generated R-module G corresponding to K generated by $\{k_1, \ldots, k_m\}$. Suppose G is generated by $\{g_1, \ldots, g_n\}$. If P is an $m \times m$ invertible matrix and Q is an $n \times n$ invertible matrix with entries in R, then:*

1. *PA is a relation matrix for G corresponding to a different generating set of K $\{k'_1, \ldots, k'_m\}$.*
2. *AQ is a relation matrix for G corresponding to a different generating set of G $\{g'_1, \ldots, g'_n\}$.*
3. *PAQ is a relation matrix for G correspoding to different generating sets of both K and G.*

I won't prove the above statements here (for a full treatment, see [11]). However, recall that left-multiplying A by an invertible square matrix corresponds to performing row operations on A. Similarly, right-multiplying by an invertible square matrix is tantamount to performing column operations on A. Essentially, Proposition 3.2 gives us the ability to perform both row and column operations on a relation matrix while preserving its relations. It is important to note that, just as row and column operations on a linear transformation matrix correspond to changes of basis in the domain and range spaces, respectively, row and column operations here change the generating sets of K and G.

Proposition 3.3. *Let A be a relation matrix for a finitely generated R-module G. If there exist an invertible $m \times m$ matrix P and invertible $n \times n$ matrix Q over R such that*

$$PAQ = \begin{pmatrix} a_1 & 0 & \cdots & & \\ 0 & a_2 & 0 & \cdots & \\ \vdots & & \ddots & & \\ & & & & a_n \\ 0 & \cdots & & & \end{pmatrix}$$

then $G \simeq R/a_1 \oplus \cdots \oplus R/a_n$.

Proof. From Proposition 3.2, PAQ is a relation matrix for the module G. The rows of PAQ then generate the kernel K of the surjective module homomorphism $\varphi : R^n \to G$, where this homormorphism is the same as stated previously. By the Fundamental Isomorphism Theorem for modules, we have that $G \simeq R/K$. Note that K also generates the kernel of the surjective homormorphism $\gamma : R^n \to R/a_1 \oplus \cdots \oplus R/a_n$ by $\gamma(r_1 \dots r_n) = (r_1 + a_1, \dots, r_n + a_n)$, so $R/a_1 \oplus \cdots \oplus R/a_n \simeq R/K$ as well. This implies $G \simeq R/a_1 \oplus \cdots \oplus R/a_n$. □

A key connection with the Structure Theorem for finitely generated modules comes from the next theorem.

Theorem 3.4. *Let A be a matrix over a principal ideal domain R. Then there exist invertible square matrices P and Q over R such that*

$$PAQ = \begin{pmatrix} a_1 & & & & & & \\ & a_2 & & & & & \\ & & \ddots & & & & \\ & & & a_m & & & \\ & & & & 0 & & \\ & & & & & \ddots & \\ & & & & & & 0 \end{pmatrix}$$

where $a_1 | a_2 | \cdots | a_m$. This matrix is called the Smith Normal Form matrix of A.

I won't state the full proof of this here; the interested reader can see [13]. However, the theorem rests on the following idea: in a PID any increasing chain of ideals (so that the next ideal in the chain always contains the previous) eventually "stops." As stated in [1], if a set of ideals I_n is totally ordered under the inclusion

$$I_1 \subseteq I_n \subseteq \cdots \subseteq I_n \subseteq \cdots$$

then there exists an integer j so that $I_j = I_{j+1} = \cdots$. The algorithm to reduce matrices to Smith Normal Form makes use of this fact by continuously taking the greatest common divisor of entries in the matrix. This operation in a sense creates an increasing sequence

of ideals, which the above fact implies must cease at some point. Interested readers can find a full treatment of the algorithm in [14]. For the purposes of the small examples that follow, we won't need the full power of the algorithm; the entries in the boundary matrices we work with are in $\{-1, 0, 1\}$, and simply applying row and column operations will suffice to reduce them to Smith Normal Form. Moreover, there are efficient packages in Mathematica and R which are capable of reducing more complicated matrices to Smith Normal Form which I use later. First, though, an important corollary.

Corollary 3.5. *For any finitely generated R-module G where R is a PID there exist $a_1, \dots a_m$ with $a_1|a_2|\cdots, |a_m$ and an integer k so that $G \simeq R/a_1 \oplus \cdots \oplus R/a_m \oplus R^k$.*

This is simply the Structure Theorem for finitely generated modules. For any R-module G with R a PID we can construct a relation matrix A for G. Theorem 3.4 implies that this matrix is reducible to Smith Normal Form, and Proposition 3.3 then implies that G is isomorphic to $R/a_1 \oplus \cdots \oplus R/a_m \oplus (R/0)^k = R/a_1 \oplus \cdots \oplus R/a_m \oplus R^k$, where k is the number of 0 rows of the Smith Normal Form of A.

To see the connection of the Smith Normal Form to homology, recall first that the finitely generated module (abelian group) $H_p(K)$ is defined as $Z_p(K)/B_p(K)$. In this module, elements of $B_p(K)$ are effectively 0, so we can think of the matrix correspoding to the boundary homomorphism ∂_{p+1} as a relation matrix for $H_p(K)$. The nonzero elements of the Smith Normal Form of this matrix give the torsion elements of the direct sum decomposition of $H_p(K)$. However, because of the way $H_p(K)$ is defined, we cannot read off the rest of its decomposition from the reduced matrix of ∂_p. The free part of $H_p(K)$ is given by $\text{rank}(Z_p(K)) - \text{rank}(B_p(K))$. Let n_p be the rank of the free module $C_p(K)$, l_p be the rank of the matrix corresponding to ∂_p. Then $\text{rank}(B_p(K)) = l_{p+1}$ and $\text{rank}(Z_p(K)) = n_p - l_p$. This gives

$$H_p(K) \simeq \mathbb{Z}^{n_p - l_p - l_{p+1}} \oplus_{i=1}^{l_{p+1}} \mathbb{Z}/a_i\mathbb{Z}$$

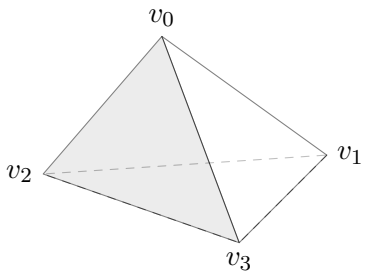
where the a_i are the nonzero entries of the Smith Normal Form matrix corresponding to ∂_{p+1}.

In our case, homology groups will generally by free. Torsion components arise when the simplicial complex K is non-orientable, a case I don't consider in this paper.

Before moving on, note that in general boundary matrices for homology computations are written as $n_{p-1} \times n_p$ matrices. Continuing the analogy to the discussion on the general Smith Normal Form, this means that in our case the relations now correspond to *columns* of the boundary matrix while the generators correspond to *rows*. It is more natural to express boundary matrices this way, and it does not effect the computations: the transpose of an $n_{p-1} \times n_p$ matrix in SNF is the $n_p \times n_{p-1}$ SNF matrix corresponding to the same generators and relations, which is in line with the preceeding discussion.

3.1.2 A Constructed Example

To see this method in action, consider the hollow tetrahedron (a triangulation of the 2-sphere) described by the simplicial complex below.



We set up the boundary matrix D_2 corresponding to ∂_2 as

$$D_2 = \begin{array}{c|cccc} & v_0v_1v_2 & v_0v_1v_3 & v_0v_2v_3 & v_1v_2v_3 \\ \hline v_0v_1 & 1 & -1 & 0 & 0 \\ v_0v_2 & -1 & 0 & 1 & 0 \\ v_0v_3 & 0 & 1 & -1 & 0 \\ v_1v_2 & 1 & 0 & 0 & -1 \\ v_1v_3 & 0 & -1 & 0 & 1 \\ v_2v_3 & 0 & 0 & 1 & -1 \end{array}$$

I've included the bases of C_2 and C_1 as the first row and column, respectively, of D_2. Doing this allows us to keep track of the changes of basis that result from row and column operations on a matrix. In general, performing column operations on a matrix changes the basis of the domain, while row operations give a change of basis for the target space. In general, these changes of basis follow the same pattern as the row or column operation. For instance, if e_i and e_j are basis elements of the domain and $\hat{e}_i$ and $\hat{e}_j$ are basis elements of the range, we have the following:

1. Exchanging r_i and r_j (c_i and c_j) exchanges basis elements $\hat{e}_i$ and $\hat{e}_j$ (e_i and e_j)
2. Scaling r_i (c_i) by k scales the basis element $\hat{e}_i$ (e_i) to $k\hat{e}_i$ (ke_i)
3. Adding kc_i to c_j changes basis element c_j to $c_j + kc_i$
4. Adding kr_i to r_j changes basis element $\hat{e}_i$ to $\hat{e}_i - k\hat{e}_j$

Item 4 above is the only change of basis rule which requires an unnatrual step, which step is probably familiar to linear algebra students. Using these formulas to track changes of basis while reducing the boundary matrices is fairly straightforward, and it allows us to write explicitly the bases of $Z_p(K)$ and $B_p(K)$ (and $H_p(K)$, if it is free). To reduce D_2, we can first perform column operations to clear a linearly dependent column, and then move this column to the last spot:

		$v_0v_1v_3$	$v_0v_2v_3$	$v_1v_2v_3$	$(v_0v_1v_2 + v_0v_1v_3 + v_0v_2v_3 + v_1v_2v_3)$
	v_0v_1	-1	0	0	0
	v_0v_2	0	1	0	0
$\rightarrow$	v_0v_3	1	-1	0	0
	v_1v_2	0	0	-1	0
	v_1v_3	-1	0	1	0
	v_2v_3	0	1	-1	0

Now the first three columns of the matrix are certainly linearly independent, so we may conclude that the final column gives a basis for $\ker \partial_2$. Thus, we can write $Z_2 = \langle v_0v_1v_2 + v_0v_1v_3 + v_0v_2v_3 + v_1v_2v_3 \rangle$. Because we are working on a 2-dimensional complex, we have $H_2 = Z_2$ (see Proposition 2.2) and thus $H_2 \simeq \mathbb{Z}$. This can be intepreted as giving that the only 2-dimensional void in this complex is the one bounded by the 2-cycle of all 2-dimensional faces of the tetrahedron.

To reduce the matrix to Smith Normal Form, we perform row operations and get:

		$v_0v_1v_3$	$v_0v_2v_3$	$v_1v_2v_3$	$(v_0v_1v_2 + v_0v_1v_3 + v_0v_2v_3 + v_1v_2v_3)$
	$v_0v_3 - v_0v_1 - v_1v_3$	1	0	0	0
	$v_0v_2 - v_0v_3 + v_2v_3$	0	1	0	0
$\rightarrow$	$v_1v_3 - v_1v_2 - v_2v_3$	0	0	1	0
	v_0v_3	0	0	0	0
	v_1v_3	0	0	0	0
	v_2v_3	0	0	0	0

From this reduced matrix, we similarly conclude that

$$B_1 = \langle (v_0v_3 - v_0v_1 - v_1v_3), (v_0v_2 - v_0v_3 + v_2v_3), (v_1v_3 - v_1v_2 - v_2v_3) \rangle,$$

which is isomorphic to $\mathbb{Z}^3$. So, in fact, the first boundary group is generated by only 3 elements, which are the boundaries of the 2-simplexes $v_0v_1v_2$, $v_0v_2v_3$, and $v_1v_2v_3$. This makes geometric sense, as the fourth 2-simplex $v_0v_1v_2$ shares exactly one edge with each of the other 2-simplexes, so its boundary can be written as a combination of the boundaries of these other 2-simplexes. We may also conclude that H_1 has no torsion elements, as the diagonal entries of the reduced matrix D_2 are all 1, which gives $(\mathbb{Z}/\mathbb{Z})^3 = 0$ as the torsion component of H_1. To get the rest of H_1, the Smith Normal Form of D_1 is needed. This matrix is

From this reduced matrix, we have $B_0 = \langle v_1 - v_0, v_2 - v_0, v_3 - v_0 \rangle$. We also have

$$Z_1 = \langle (v_0v_3 - v_0v_1 - v_1v_3), (v_0v_2 - v_0v_3 + v_2v_3), (v_1v_3 - v_1v_2 - v_2v_3) \rangle.$$

This group is actually equal to B_1, so we conclude that $H_1 = 0$.

	v_1v_2	v_1v_3	v_2v_3	$(v_0v_3 - v_0v_1 - v_1v_3)$	$(v_0v_2 - v_0v_3 + v_2v_3)$	$(v_1v_3 - v_1v_2 - v_2v_3)$
$v_1 - v_0$	1	0	0	0	0	0
$\rightarrow$ $v_2 - v_0$	0	1	0	0	0	0
$v_3 - v_0$	0	0	1	0	0	0
v_0	0	0	0	0	0	0

We can calculate H_0 without using more matrices. Recall that for any simplicial complex K, $C_0(K) = Z_0(K)$. In our case, $Z_0 \simeq \mathbb{Z}^4$ (for the 4 vertices of the complex) and $B_0 = \langle v_1 - v_0, v_2 - v_0, v_3 - v_0 \rangle \simeq \mathbb{Z}^3$. So $H_0 \simeq \mathbb{Z}^4/\mathbb{Z}^3 \simeq \mathbb{Z}$, which is as we would hope for a simplicial complex with one connected component. In conclusion, from this analysis we would write that the 0th Betti Number β_0 is 1, $\beta_1 = 0$, and $\beta_2 = 1$.

It is not a coincidence that Z_1 is generated by the same set as B_1; in fact, I took care in the process of reducing D_1 to ensure that the basis of Z_1 would contain the basis of B_1. I did this by making the *column* operations on D_1 the inverses of the *row* operations on D_2, a process outlined in [14]. This works in part because in D_2, all entries in a row sum to 0, while in D_1 all entries in a column sum to 0, and we manipulate basis elements of C_1 by row and column operations in D_2 and D_1, respectively. This feature of the reduced matrices is quite nice here, as it allows us to immediately conclude that $Z_1 = B_1$, and thus $H_1 = 0$. In general, we could write a basis for H_p by taking the basis for Z_p without the basis for B_p, which is a subset.

Note that manipulating boundary matrices so that Z_p and B_p are described in terms of the same basis elements is not always easy. While the Smith Normal Form of a matrix A is unique, Theorem 3.4 does not state that the diagonalization matrices P and Q are unique; thus, there are many different bases for C_p and C_{p-1} corresponding to the same Smith Normal Form of the boundary homomorphism matrix D_p. This is a problem when working with programs like Mathematica, which is capable of reducing matrices over PIDs to Smith Normal Form but does not track changes in basis during the reduction process. Some packages in Mathematica will also give the matrices P and Q, but even with these matrices it is not always possible to calculate the correct bases. In large part this is due to the change of basis rule corresponding to adding rows to each other (item 4 above). However, other efficient programs in C++ and R dedicated to homology computations are capable of calculating Smith Normal Forms along with appropriate bases; these will be discussed more in Section 4.

3.2 The Incremental Method

As stated before, the Smith Normal Form method relies heavily on the algebraic structure of the chain complex. It also requires matrix reductions which may not be efficient given the size of the complex. The following algorithm, due to Edelsbrunner and Delfinado, requires little algebraic setup and is much easier to state than the Smith Normal Form method. The algorithm assumes that we have a complex K which is *totally filtered*.

Definition A *filtration* of a simplicial complex K is a a sequence of nested simplicial complexes

$$K_1 \subseteq K_2 \subseteq \cdots \subseteq K_q = K.$$

A filtration is a *total filtration* if $K_{i+1} = K_i \cup \sigma_{i+1}$—that is, if each complex in the filtration differs from the previous and next complex by a single simplex [2].

Because simplicial complexes can be described combinatorially, it is easy to put a total filtration on a simplicial complex. In addition to assuming that K is totally filtered the algorithm also requires that K is a subcomplex of a triangulation of S^3, the three-sphere. Essentially, this condition guarantees that the vertices of K lie in $\mathbb{R}^3$. The reason for this restriction is that the algorithm runs by counting which p-simplexes are a part of a p-cycle, and which aren't. There are only general methods for determining if a simplex is part of a cycle for 1-cycles and $d-1$ cycles, where d is the dimension of the simplicial complex. Thus, we are restricted to at most 3-dimensional complexes while using this method. The incremental algorithm works by keeping a running tally of the Betti numbers β_0, β_1, β_2, and β_3. It begins by setting $\beta_0 = \beta_1 = \beta_2 = \beta_3 = 0$. It proceeds:

For each $0 \leq i \leq m$, with $k = \dim \sigma_i$, set $b_k = b_k + 1$ if σ_i is part of a k-cycle on K_i. Otherwise, set $b_{k-1} = b_{k-1} - 1$.

The algorithm terminates once the process has been repeated for each i. The reason that the algorithm works is fairly straightforward in low-dimensional cases. For example, adding an edge between unconnected vertices would decrease β_0 by 1, while adding an edge between connected vertices (and thus creating a 1-cycle) would increase β_1 by 1. The difficulty in implementing the algorithm, as stated previously, comes in determining which k-simplexes are part of k-cycles and which are not. By definition, 0-simplexes (vertices) are also 0-cycles, so adding a 0-simplex will always increase β_0 by 1 (this is why we needn't worry about the apparent β_{-1} that arises in the description of the algorithm). Determining 1-cycles amounts to the problem of finding cycles on a graph, which is well understood and for which there are a variety of methods. Finding 2-cycles turns out to be somewhat more challenging, and deserves a closer look.

We begin by defining $\overline{K}_i = K/K_i$. We then create a graph G_i as follows: for each 3-simplex in $\overline{K}_i$, create a vertex in G_i; if two 3-simplexes in $\overline{K}_i$ share a 2-dimensional face, connect their corresponding vertices in G_i by an edge. Then a 2-simplex σ_i is part of a 2-cycle if G_{i-1} has one less connected component than G_i. Otherwise, σ_i is not part of a 2-cycle. Thus, the problem of detecting 2-cycles can be reduced to a graph theoretic problem which is well-understood.

To see how 3-simplexes are counted in the algorithm, recall that we require K to be a subcomplex of a triangulation of S^3. Thus, if $\beta_3 = 1$, K must itself be a triangulation of S^3, since S^3 is a single 3-cycle. (We know further that β_3 is not greater than

1 because K is specified as a subcomplex of a triangulation of S^3.) This concept is more easily visualized in lower dimensional cases. For example, if we take the hollow tetrahedron to be a triangulation of S^2, we see that the faces of the tetrahedron, which are 2-simplexes and thus triangulations of S^1, bound S^2. If K is embeddable in $\mathbb{R}^3$, then it has no 3-cycles (which do *not* embed in $\mathbb{R}^3$) and thus $\beta_3 = 0$. In terms of the algorithm, then, the addition of a 3-simplex always decreases β_2 by 1 unless K is a triangulation of S^3, in which case the addition of the final 3-simplex σ_m increases β_3 by 1.

The following table demonstrates how the algorithm runs when applied to the tetrahedron from the earlier example. Each column represents the subcomplex K_i obtained by adding the simplex σ_i to K_{i-1} and the rows give the Betti numbers updated at that subcomplex. Notice how adding a 1-simplex between vertices decreases β_0 by 1, adding a 2-simplex decreases β_1 by 1, and adding the final 2-simplex $v_1v_2v_3$ increases β_2 by 1:

	v_0	v_1	v_2	v_3	v_0v_1	v_0v_2	v_0v_3	v_1v_2	v_1v_3	v_2v_3	$v_0v_1v_2$	$v_0v_1v_3$	$v_0v_2v_3$	$v_1v_2v_3$
β_0	1	2	3	4	3	2	1	1	1	1	1	1	1	1
β_1	0	0	0	0	0	0	0	1	2	3	2	1	0	0
β_2	0	0	0	0	0	0	0	0	0	0	0	0	0	1

3.3 Efficiency

In practice, both of the methods described above are commonly implemented in homology computations. Once the boundary matrices are constructed, the Smith Normal Form method amounts to Gaussian Elimination and requires $O(m^3)$ elementary row operations, where m is the number of simplices in the complex K [14]. It should be noted that the sizes of simplicial complexes, and thus the sizes of the boundary matrices, can become very large very quickly. For instance, even relatively simple examples such as the simplicial torus, which has twenty-seven 1-simplexes, eighteen 2-simplexes, and thus a 27×18 boundary matrix D_2, are difficult to manage. For complexes built on thousands of data points, pure matrix reduction is often infeasible. To this end, much of the current research in computational homology is being directed toward efficient reductions of matrices to Smith Normal Form [6].

The incremental algorithm can actually compute the Betti numbers of K in $O(m)$ given certain restrictions on how the complex is stored [5]. However, the incremental algorithm as I've described it here does not have the ability to compute explicit bases for $B_p(K)$, $Z_p(K)$, and $H_p(K)$, which are very useful to have in applications.

4 Persistent Homology

To this point, we have discussed how to compute the homology of a given simplicial complex. In topological data analysis, however, we generally begin with a set of point-

cloud data $X = \{x_1, x_2, \ldots, x_m\}$ in $\mathbb{R}^n$ *without* any simplicial structure built on it. We assume that the data X has been sampled from an underlying manifold P. Our hope is that by constructing a simplicial complex on X that closely approximates P and then computing its homology, we gain some insight into the the shape of P. In this section, I discuss methods for constructing such complexes, a process which requires choices of parameters. I then discuss the notion of persistent homology, which is a way of dampening the effects of (possibly non-optimal) parameter choices on homology computations of a set of point-cloud data X.

4.1 Constructing Complexes & Filtrations

There are many types of simplicial complexes defined for a set of point-cloud data X, most of which are constructed based on some function of the distances between vertices in X. These complexes can vary greatly in efficiency, both in the time required to construct them and the space needed to store them. For a good introduction to different types of complexes on point-cloud data, see [3]. In this paper, we use the simplest such complex: the *Vietoris-Rips Complex*.

Definition Let $X = \{x_1, x_2, \cdots, x_m\}$ and d be a metric such that (X, d) is a metric space. The *Vietoris-Rips Complex* on X with the parameter ϵ, denoted $VR(X, \epsilon)$, is the simplicial complex whose vertex set is X where a subset σ of $k+1$ vertices in X spans a k-simplex if and only if $d(x_i, x_j) \leq \epsilon$ for all $x_i, x_j \in \sigma$.

In other words, the Vietoris-Rips complex constructs a k-simplex on any set of $k+1$ vertices which have pairwise distances less than or equal to ϵ. From this idea, it is easy to see that if $\epsilon < \epsilon'$, then we have the inclusion $VR(X, \epsilon) \subseteq VR(X, \epsilon')$, as increasing the value of the parameter ϵ in a Vietoris-Rips complex can add new simplexes but will never remove existing simplexes. In this manner we can build a filtration of complexes on X. For $\alpha \in \mathbb{R}$ and $k = 0, 1, 2, \ldots, q$, let $K_k = VR(X, \epsilon + k\alpha)$; then a filtration of simplicial complexes on the vertex set X is given by

$$K_0 \subseteq K_1 \subseteq K_2 \subseteq \cdots \subseteq K_q$$

We see that for small k, the complex K_k will be "finer" than complexes associated to large k. In this sense, we get images of the simplicial complex approximating the underlying space P of X at different resolutions. Note that in practice, if $X \subset \mathbb{R}^n$, we restrict the maximum dimension of simplexes included in $VR(X, \epsilon)$ to n; without this restriction, the complex could include simplexes with dimension greater than n which do not embed in $\mathbb{R}^n$. Thus, we assume that P is at most n-dimensional [9].

The Vietoris-Rips Complex is indicative of methods for constructing simplicial complexes on a data set that depends explicitly on pairwise distances between points. However, we can also construct filtrations without calculating these distances, which becomes computationally costly as the number of data points increases. Consider a set $A \subset \mathbb{R}^m$ such that $X \subset A$ and a real-valued function $f : A \to \mathbb{R}$. The *sublevel set* of f corresponding

to t, denoted $f^{-1}(-\infty, t]$, is the set $\{x \in A : f(x) \leq t\}$. Similarly, the *superlevel set* corresponding to t, written $f^{-1}[t, \infty)$ is the set $\{x \in A : f(x) \geq t\}$. A complex can then be constructed on the points of X in each sublevel and superlevel set. Increasing (in the case of sublevel sets) or decreasing (in the case of superlevel sets) the parameter t then creates a filtration of simplicial complexes. This method relies heavily on Morse Theory, which deals generally with real-valued functions on manifolds. For a more thorough grounding in Morse Theory as it relates to simplicial homology, see [7, pp. $140 - 167$].

Variations on the distance function are common choices for f in this setting. For instance, given $X = \{x_1, x_2, \ldots, x_m\}$, the distance function for $y \in A$ $\Delta(y)$ is defined

$$\Delta(y) = \inf_{x_i \in X} ||y - x||_m$$

where $||y - x||_m$ is the usual Euclidean distance between two points in $\mathbb{R}^m$. Note that when the metric d in the Vietoris-Rips Complex is the usual Euclidean metric, these two constructions are quite similar. Other functions commonly used in this context are the distance to measure function (DTM), the k-nearest neighbor density estimator (kNN), and the Gaussian kernel density estimator (KDE). While these functions vary a great deal in behavior, they all give some sense of where points in X are densely clustered and where they are not, a feature which makes this method of filtration helpful for lessening the effects of data points which may represent statistical noise.

4.2 Persistence

Once we construct a filtration, we can calculate its *persistent homology*. In the filtration

$$K_0 \subseteq K_1 \subseteq K_2 \subseteq \cdots \subseteq K_q$$

for any K_i, K_j with $i < j$ the inclusion map of K_i into K_j induces a homomorphism $I_p : H_p(K_i) \to H_p(K_j)$. This is true generally for continuous maps between topological spaces due to the following proposition.

Proposition 4.1. *A continuous map f from a simplicial complex K to another simplicial complex K' induces a homomorphism between the p^{th} homology groups of the complexes.*

Proof. Let $f : K \to K'$ be a map between simplicial complexes. To define an induced homomorphism f_* between the p-th homology groups of K and K', we begin by letting $f_*(\sigma_p) = f(\sigma_p)$. Extending this linearly to p-chains of $C_p(K)$, we define f_* so that $f_*(\Sigma_i \lambda_i \sigma_p^i) = \Sigma_i \lambda_i f_*(\sigma_p^i)$. Note that $f_* \partial_p = \partial_p f_*$: using the definition of the boundary operator, we see

$$\begin{aligned} f_*(\partial_p(\sigma_p)) =& f_*(\Sigma_i (-1)^i (v_0, \ldots, \widehat{v_i}, \ldots, v_p)) \\ =& \Sigma_i (-1)^i f_*((v_0, \ldots, \widehat{v_i}, \ldots, v_p)) \\ =& \partial_p(f_*(\sigma_p)) \end{aligned}$$

In particular, this means that f_* takes cycles in K to cycles in K', as for some p-cycle $\zeta \in K$, $f_*(\partial_p(\zeta)) = f_*(0) = 0 = \partial_p(f_*(\zeta))$. Because the composition of consecutive boundary homomorphisms is the 0 homomorphism, we can conclude that f_* maps boundaries in K to boundaries in K' by a similar argument. So f_* is a homomorphism between the p-th boundary and cycle groups of K and K' and thus between $H_p(K)$ and $H_p(K')$ [11]. $\square$

The induced homomorphisms between homology groups of a filtration then give us a sequence of homology groups

$$H_p(K_0) \to H_p(K_1) \to \cdots \to H_p(K_q)$$

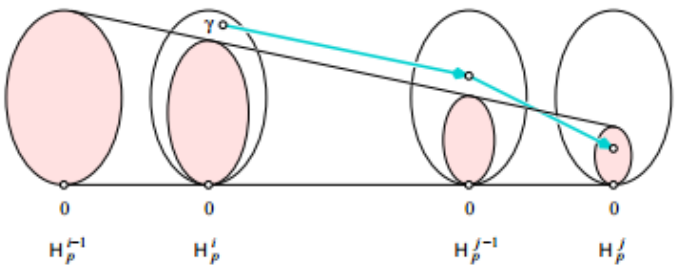
linked by the induced homomorphism between each $H_p(K_i), H_p(K_{i+1})$. As we move up the filtration, we may gain or lose homology classes in the corresponding sequence of homology groups. For example, if K corresponds to a complex consisting of two vertices not connected by an edge and K' is equal to K with an edge added between the vertices, we lose one homology class under the induced homomorphism that takes $H_0(K)$ to $H_0(K')$, as K' has only one connected component while K has two (see Proposition 2.4)

From the sequence above, we define persistent homology groups.

Definition Given a filtration of a simplicial complex as above, for $0 \leq i \leq j \leq q$ the j-th persistent p-th homology group of K_i is the image of $H_p(K_i)$ under the homomorphism $I_*^{i,j}$ induced by the inclusion of K_i in K_j, denoted $H_p^{i,j} = \text{img } I_p^{i,j}$ [7].

We can similarly define the corresponding Betti Number $\beta_p^{i,j}$ as the rank of $H_p^{i,j}$. Computing persistent Betti Numbers involves matrix reductions similar to those displayed in Section 3. For more on computations of persistent homology groups, see [7] and [14].

Let γ be a homology class of $H_p(K_i)$. We say that γ is *born* at K_i if $\gamma \notin \text{img } I_p^{i-1,i}$, that is, if γ is not in the image of the induced homomorphism taking $H_p(K_{i-1})$ to $H_p(K_i)$. If γ is born at K_i *dies* entering K_j if $I_p^{i,j-1}(\gamma) \notin \text{img } I_p^{i-1,j-1}$ but $I_p^{i,j}(\gamma) \in \text{img } I_p^{i-1,j}$. The following diagram from [7] helps to illustrate this definition:



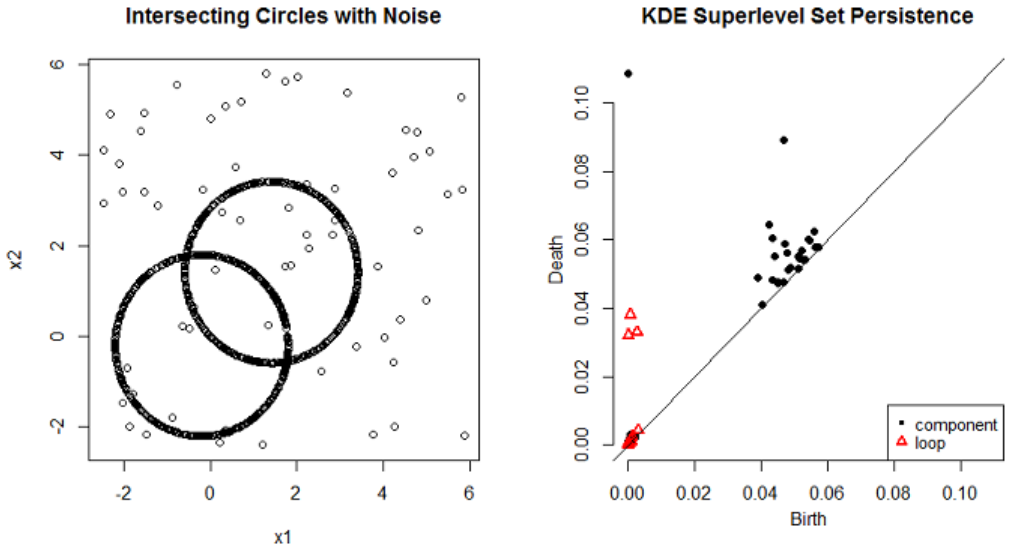
If γ dies entering K_j, it merges with an older homology class; otherwise, γ could not be in the image of $I_p^{i-1,j}$. If γ is born at K_i and dies entering K_j, we say that

γ has persistence $j - i$. This information is much more naturally encoded in the p^{th} *persistence diagram* of the filtration. Persistence diagrams are multisets of points in the first quadrant of the plane; specifically, a homology class that is born at K_i and dies entering K_j is represented in the diagram by the point (i, j). Thus, because we assume $j \geq i$, all points in the persistence diagram lie on or above the diagonal $i = j$ line. A point's height off of the diagonal is the persistence of the corresponding homology class. In practice, we can combine diagrams of various dimensions into a single diagram by assigning a different symbol to each dimension of homological feature present in the filtration, a process that is illustrated in the next section.

4.2.1 Persistent Homology Calculations with R

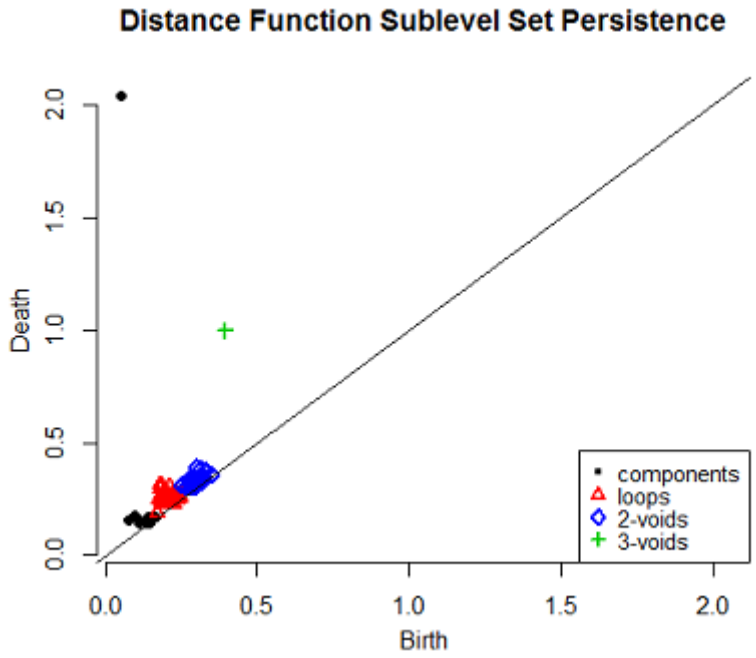
The R package TDA is based on the efficient C++ libraries GUDHI, Dionysus, and PHAT, all of which were designed to compute persistent homology of point-cloud data as well as perform other types of topological data analysis. Specifically, TDA adapts features of these libraries to construct various types of simplicial filtrations on data, compute the persistent homology of these filtrations, and print the resulting persistence diagrams. Two experiments with sampled data are outlined below to display some of these capabilities.

In the first experiment, 1200 points were sampled uniformly from two intersecting circles using TDA's `sphereUnif` function. Additionally, "noise" points were included inside and around the circles. Then, using the `gridDiag` function, a filtration was constructed using the superlevel sets of the kernel density estimator function defined on a grid containing the sample. The persistence diagram of this filtration, as well as a plot of the original sample, is shown in the figure below.



Points in the persistence diagram represent connected components, while red triangles represent loops. The persistence diagram shows strong evidence for the existence of the three distinct loops present in the plot of the data. However, we also see a strong indication of a second distinct connected component, which is not present in the data. This may be the result of a particular parameter choice in constructing the grid and defining the KDE function, or it may be a peculiarity of this particular sample. Overall, however, TDA returns the homology of the data more or less as we would hope. Note that the clustering of points within the interval $[0.04, 0.06]$ on the *birth* axis is most likely a result of the superlevel set method of filtration; discrete or isolated points have relatively low values in the KDE function, and thus would appear toward the middle or end of a superlevel set filtration.

In the second experiment, 1200 points were sampled uniformly from the S^3, again using `sphereUnif`. Again, `gridDiag` constructed a filtration, this time using sublevel sets distance function described previously. The resulting persistence diagram is below.



We see evidence for a single connected component and a single 3-void, which is exactly the homology of S^3.

Generally, TDA handles small-scale calculations such as these very well. However, in my experiments with TDA I've found that it can be quite sensitive to scaling and rounding of data, which in theory should not affect the outcomes of homology computations. TDA also slows down substantially as the number of data points in a sample grows. If anything, this speaks to the fact that most of the difficulty in computing persistent

homology comes in the initial step of creating a filtration of simplicial complexes on the data. Doing this efficiently is generally difficult, and it continues to be a broad area of research. In R, the sub-and-superlevel methods of filtration are the most efficient method available (the alternative is the Vietoris-Rips complex, which for a complex consisting of n simplexes requires the calculation of an $n \times n$ distance matrix).

For larger-scale problems, working directly with the C++ code from GUDHI, Dionysus, or PHAT may be preferable, as it offers the user more control over parameters and over the method utilized to construct a simplicial filtration. Another similarly designed software called JavaPlex is well-documented and also has bindings for use in Matlab. These libraries are updated frequently, and together provide a wide range of tools for topological data analysis to average users.

5 Conclusions

This paper has laid out the basic theory of simplicial persistent homology and has provided some basic methods for applying this theory to data analysis problems. However, the breadth of tools available for topological data analysis in general, and persistent homology in particular, spans far beyond the scope of this paper. Among these tools are statistical frameworks for interpreting persistence diagrams. For example, [8] constructs confidence sets for persistence diagrams, which separate significant homological features from insignifcant ones using a generalization of confidence intervals, while [4] discusses efficient methods of subsampling to compute persistent homology. The work from [8] has been implemented in the package TDA, which is able to compute $(1-\alpha)$ confidence sets for the persistent homology of a given data set and include a corresponding confidence band on the persistence diagram, making interpretation much simpler.

While strides are being made in a statisticial direction, the majority of the difficulty in applying persistent homology to data still comes in efficiently constructing simplicial filtrations. (A good summary of available methods of filtration is given in [3]). A large amount of research is still being devoted to finding filtrations that minimize computational costs while providing accurate approximations of the desired underlying space. The Vietoris-Rips complex gives a good illustration of this trade-off: it genearlly gives accurate approximations, but at such high costs so as to make it infeasible for most real-world situations (as stated previously, computing pairwise distances between n simplexes results in an $n \times n$ distance matrix). This area of study falls at the intersection of topology and computer science, making it a new an interesting topic for research in the years to come.

Most of all, though, more actual examples of topological analyses of data are needed in order for the field to advance. The most commonly cited example of persistent homology applied to data, outlined in [7], is a multi-year study of data taken from black and white photographs. The photographs were broken into 3×3 grids of pixels, with

each pixel containing a grayscale value representing the shade of that pixel in the image. These grids were converted to vectors in $\mathbb{R}^9$, which were then analyzed using persistent homology. While the mathematical analysis of the data is not particularly difficult to follow, the resulting intepretations are. This problem does not seem to be specific to this study, but rather is endemic of shape analyses of high-dimensional data in general. How do we interpret any kind of statement about shape in such an unfamiliar environment as 9-dimensional space? Questions like these lead Gunnar Carlsson to warn in [3] that a qualitative understanding of the type of data being studied must be highly developed before any kind of advanced analysis can take place. Overall, for persistent homology analyses to be applied succesfully, we need to be able to think creatively and unconventionally about what certain features of data might be telling us.

As the field grows, more examples of succesful analyses using persistent homology will undoubtedly arise, which in turn will open new areas for expansion. Until then, research on the theoretical side can continue to advance.

References

[1] Anderson, M., & Feil, T. (2014). *A first course in abstract algebra: rings, groups, and fields*. CRC Press.

[2] Basener, W. F. (2006). *Topology and Its Applications*. A Wiley Series of Texts, Monographs and Tracts.

[3] Carlsson, G. (2009). Topology and Data. *Bulletin of the American Mathematical Society*, 46(2), 255-308.

[4] Chazal, F., Fasy, B. T., Lecci, F., Michel, B., Rinaldo, A., & Wasserman, L. (2014). Subsampling methods for persistent homology. *arXiv preprint arXiv:1406.1901.*

[5] Delfinado, C. J. A., & Edelsbrunner, H. (1995). An incremental sprangeltop algorithm for Betti numbers of simplicial complexes on the 3-sphere. *Computer Aided Geometric Design*, 12(7), 771-784.

[6] Dumas, J. G., Heckenbach, F., Saunders, D., & Welker, V. (2003). Computing simplicial homology based on efficient Smith normal form algorithms. In *Algebra, Geometry and Software Systems* (pp. 177-206). Springer Berlin Heidelberg.

[7] Edelsbrunner, H., & Harer, J. (2010). *Computational topology: an introduction*. American Mathematical Soc..

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

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

[10] Giblin, P. (2013). *Graphs, surfaces and homology: An introduction to algebraic topology*. Springer Science & Business Media.

[11] Hatcher, A. *Algebraic topology*. 2002. Cambridge UP, Cambridge.

[12] MacLane, S. (2012). *Homology*. Springer Science & Business Media.

[13] Morandi, P. J. (2005). The Smith Normal Form of a Matrix. Unpublished.

[14] Zomorodian, A. J. (2005). *Topology for computing* (Vol. 16). Cambridge University Press.