THE PENNSYLVANIA STATE UNIVERSITY
SCHREYER HONORS COLLEGE

DEPARTMENT OF MATHEMATICS

TOPOLOGY AND DATA ANALYSIS

HANGYU ZHOU
SPRING 2017

A thesis
submitted in partial fulfillment
of the requirements
for baccalaureate degrees
in Mathematics
with honors in Mathematics

Reviewed and approved* by the following:

John Roe
Professor of Mathematics
Thesis Supervisor

Sergei Tabachnikov
Professor of Mathematics
Honors Adviser

*Signatures are on file in the Schreyer Honors College.

Abstract

Topological data analysis is a recently developed technique to analyze datasets in Euclidean space. This new technique enables us to analyze datasets which are high-dimensional, incomplete and noisy. The motivation of topological data analysis is to study the shape of the data. To see the shape from a discrete set of data points, many algorithms require a choice of proximity. However, this parameter is usually hard to decide and we need some other information to determine what proximity to use. The main insight of persistent homology is that we should be looking at all proximities altogether, but it is hard to transform this large amount of information into an understandable and easy-to-present form. In topological data analysis, the idea of *homology* solves this problem. Briefly, we assume that structures that persist over a long range of proximities are real structures of the dataset, while structures which only persist for a short range are considered to be noise. In this thesis, we will discuss the mathematical tools that are necessary to understand topological data analysis, introduce how a particular algorithm works, and apply this technique to analyze some real-world data.

Acknowledgements

I feel lucky to work with Professor John Roe, who motivated me to study topological data analysis. This project is impossible without his teachings and patience. Thank you for your understanding and encouragement when I feel depressed. Thank you for being my thesis advisor even though I am an undergraduate who is still learning about doing research.

Table of Contents

List of Figures

1 Mathematical Introduction

1.1 From Data to Simplicial Complexes

In this paper, by "data" we mean a finite set of points in some Euclidean space; we call the set of data **the point cloud**. As an easy example, consider the figure below.

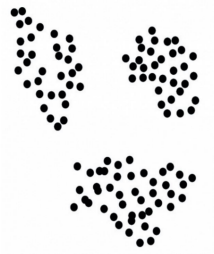

Figure 1: A point cloud

We see that the data points seem to break into three different clusters. Even without noise, the clustering is only visible on a certain range of scales. And in real life, the data are not as nice as shown in figure 1. We always encounter some noise as shown below and we would like to be able to see this clustering behavior even with noise.

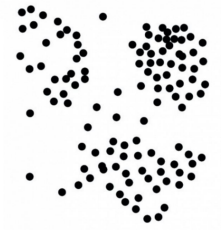

Figure 2: A point cloud with noise

For our purpose, we introduce a parameter ϵ which help us complete the point cloud to a simplicial complex with data points as vertices. The edges are determined by proximity and ϵ is the measure of proximity.

Definition 1. Given a colection of points $\{x_n\}$ in Euclidean space $\mathbb{R}^n$, the **Rips complex**, $\mathcal{R}_\epsilon$, is the simplicial complex whose k-simplices correspond to unordered $(k+1)$-tuples of points $\{x_n\}_0^k$ which are pairwise within distance ϵ.

There are some other ways to define a simplicial complex from a point cloud, but from a computation point of view, the Rips complex is less expensive because it is the maximal

among all simplicial complexes with a given 1-skeleton. Thus, the combinatorics of the 1-skeleton completely determines the Rips complex.

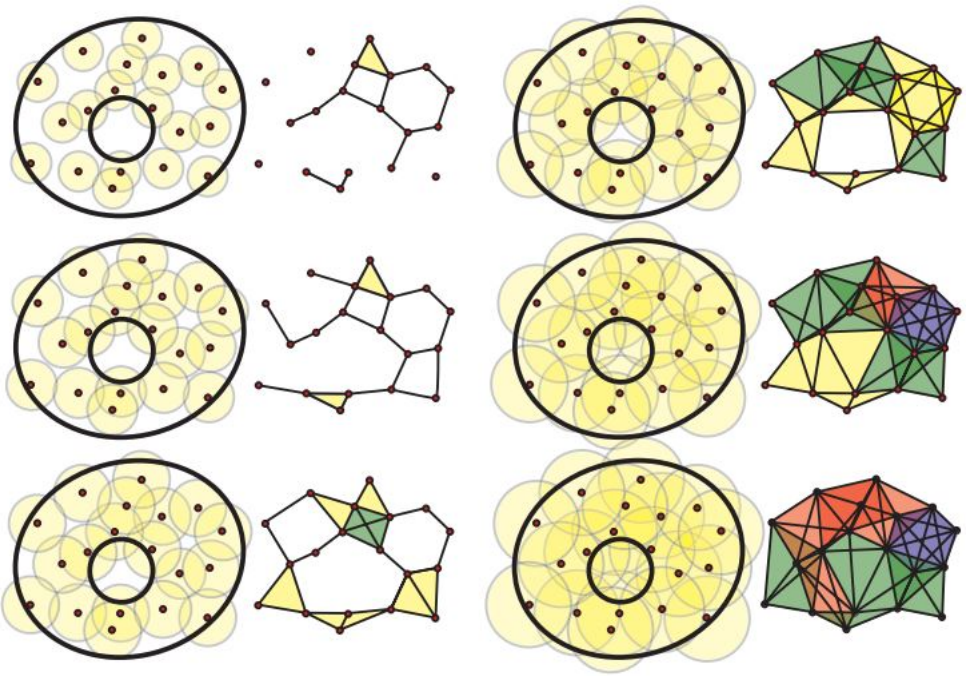

Figure 3: Example of Rips complex with different choice of parameter ϵ from [2]

The structure of the Rips complex varies as we vary ϵ. For sufficiently small ϵ, the complex is a discrete set; for sufficiently large ϵ, the complex is a single high-dimensional simplex. So a natural question to ask is: which parameter ϵ should we choose in order to capture the right structure? Consider the example in figure 3 above. The point cloud is a sampling of points in a planar annulus. As shown in the figure, it seems that it is really hard to decide which ϵ to choose. When ϵ is large enough to remove the gap from within the structure of the annulus, the large hole in the middle is also filled in.

1.2 Homology

Definition 2. A set $\{v_0, \cdots v_n\}$ of points of $\mathbb{R}^n$ is said to be **geometrically independent** if for any scalars t_i, the equations

$$\sum_{i=0}^{n} t_i = 0 \qquad \text{and} \qquad \sum_{i=0}^{n} t_i v_i = 0$$

imply that $t_0 = t_1 = \cdots = t_n = 0$.

Clearly, a one-point set is always geometrically independent. It is an easy exercise to show that in general the set $\{v_0, \cdots v_n\}$ is geometrically independent if and only if the vectors

$$v_1 - v_0, \cdots, v_n - v_0$$

are linearly independent. Thus, two distinct points form a geometrically independent set in $\mathbb{R}^n$, so do three non-collinear points, four non-coplanar points, and so on.

Definition 3. Given a geometrically independent set $\{v_0, \cdots v_n\}$ in $\mathbb{R}^n$, we define the **n-simplex** Δ spanned by $v_0, \cdots v_n$ to be the set of all points x of $\mathbb{R}^n$ such that

$$x = \sum_{i=0}^{n} t_i v_i \text{ , where } \sum_{i=0}^{n} t_i = 1$$

and $t_i \geq 0$ for all i. The numbers t_i are uniquely determined by x and are called the **barycentric coordinates** of the point x of Δ with respect to $v_0, \cdots v_n$.

For example,a 0-simplex is a point; a 1-simplex is a line segment; a 2-simplex is a triangle; a 3-simplex is a tetrahedron.

Definition 4. The points $v_0, \cdots v_n$ that span Δ are called the **vertices** of Δ. The number n is called the **dimension** of Δ. Any simplex spanned by a subset of $\{v_0, \cdots v_n\}$ is called a **face** of Δ. The faces of Δ different from Δ itself are called the **proper faces** of Δ and the union of all proper faces is called the **boundary** of Δ, denoted by $\partial\Delta$. The **open simplex** Δ° is $\Delta - \partial\Delta$, the interior of Δ.

Now we are ready to define a simplicial complex.

Definition 5. A **simplicial complex** $\mathcal{K}$ in $\mathbb{R}^n$ is a collection of simplices in $\mathbb{R}^n$ such that:

1. Every face of a simplex of $\mathcal{K}$ is in $\mathcal{K}$.
2. The intersection of any two simplices of $\mathcal{K}$ is a face of each of them.

In figure 4, $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_4$ are all simplicial complexes, while $\mathcal{K}_3$ is not.

Definition 6. A **subcomplex** of $\mathcal{K}$ is a subcollection of $\mathcal{K}$ that contains all faces of its elements. A particular subcomplex of $\mathcal{K}$ is the collection of all simplices of $\mathcal{K}$ of dimension at most p; it is called the **p-skeleton** of $\mathcal{K}$.

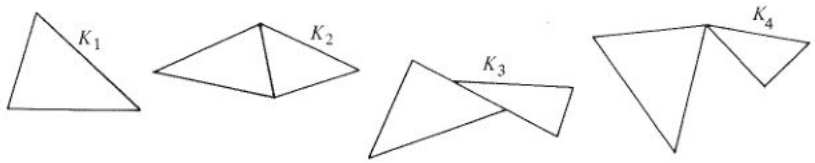

Figure 4: Examples of simplicial complexes ($\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_4$)

Definition 7. Let Δ be a simplex. Define two orderings of its vertex set to be **equivalent** if they differ from one another by an even permutation. If $\dim\Delta > 0$, the orderings of the vertices fall into two equivalence classes. Each of these classes is called an **orientation** of Δ. If $\dim\Delta = 0$, then there is only one class and hence only one orientation of Δ. A simplex Δ together with an orientation is called an **oriented simplex**.

From this point on, we will use $v_0 \cdots v_n$ to denote the simplex spanned by $\{v_0, \cdots, v_n\}$, use $[v_0, \cdots, v_n]$ to denote the oriented simplex, and use $(v_0, \cdots, v_n)$ to denote the equivalence class of the particular ordering.

Definition 8. Let $\mathcal{K}$ be a simplicial complex. A **p-chain** on $\mathcal{K}$ with values in a ring R is a function c from the set of oriented p-simplices of $\mathcal{K}$ to R, such that:

1. $c(\Delta) = -c(\Delta')$ if Δ and Δ' are same simplex with opposite orientation.
2. $c(\Delta) = 0$ for all but finitely many oriented p-simplices.

The group of p-chains of $\mathcal{K}$ is denoted by $C_p(\mathcal{K})$, with the addition defined by adding the values. if $p < 0$ or $p >$dim$\mathcal{K}$, we let $C_p(\mathcal{K})$ be the trivial group.

If Δ is an oriented simplex, the **elementary chain** c corresponding to Δ is the function such that: $c(\Delta) = 1$ and $c(\Delta') = -1$ if Δ' and Δ are the same simplex with opposite orientation and 0 otherwise. By abuse of notation, we sometimes also use Δ to denote the elementary p-chain corresponding to the oriented simplex Δ. Then we can write $\Delta' = -\Delta$.

Theorem 1. $C_p(\mathcal{K})$ is free abelian. A basis for $C_p(\mathcal{K})$ can be obtained by orienting each p-simplex and using the corresponding elementary chains as a basis.

Now we have a sequence of free abelian groups, and we can define a map along the chain by sending each simplex to its "boundary".

Definition 9. Define the homomorphism

$$\partial_p : C_p(\mathcal{K}) \to C_{p-1}(\mathcal{K})$$

by

$$\partial_p(\Delta) = \partial_p[v_0, \cdots, v_p] = \sum_{i=0}^{p} (-1)^i [v_0, \cdots, \hat{v_i}, \cdots, v_p],$$

where $\hat{v_i}$ means that the vertex v_i is removed from the list. ∂_p is called a **boundary operator**.

Since $C_p(\mathcal{K})$ is the trivial group when $p < 0$, the map ∂_p is trivial for $p \leq 0$. It is an easy exercise to check $\partial_p(-\Delta) = -\partial_p(\Delta)$ and thus ∂_p is well-defined.

Theorem 2. $\partial_{p-1} \circ \partial_p = 0$.

Proof.

$$\begin{aligned}
\partial_{p-1}\partial_p[v_0, \cdots, v_n] &= \sum_{i=0}^{p} (-1)^i \partial_{p-1}[v_0, \cdots, \hat{v_i}, \cdots, v_n] \\
&= \sum_{j<i} (-1)^i (-1)^j [\cdots, \hat{v_j}, \cdots, \hat{v_i}, \cdots] + \sum_{j>i} (-1)^i (-1)^{j-1} [\cdots, \hat{v_i}, \cdots, \hat{v_j}, \cdots] \\
&= 0
\end{aligned}$$

□

The last step is true because each term appears twice, with opposite signs. The sequence of chains together with the boundary maps is call a **chain complex**. Now we can define the homology group.

Definition 10. The group $H_p(\mathcal{K}) = \text{Ker}\partial_p/\text{Im}\partial_{p+1}$ is the p^{th} **homology group** of $\mathcal{K}$.

For example, suppose that $\mathcal{K}$ is the boundary of a triangle, with three 0-simplices v_1, v_2, v_3 and three 1-simplices e_1, e_2, e_3. Note that the triangle itself is not part of $\mathcal{K}$, i.e. $\mathcal{K}$ does not contain 2-simplices. Since $\partial_1(e_1) = v_1 - v_2$, $\partial_1(e_2) = v_2 - v_3$, $\partial_1(e_3) = v_3 - v_1 = -\partial_1(e_1) - \partial_1(e_2)$ and $\{v_1 - v_2, v_2 - v_3, v_2\}$ is a basis for C_0, it follows that $H_0(\mathcal{K})$ is isomorphic to $\mathbb{Z}$ and is generated by v_2. Since there are no 2-simplices, $H_1(\mathcal{K})$ is equal to $\ker(\partial_1)$, which is infinite cyclic generated by $e_1 + e_2 + e_3$ since $\partial_1(ae_1 + be_2 + ce_3) = (a-c)v_1 + (b-a)v_2 + (c-b)v_3 = 0$ only if $a = b = c$. The groups $H_n(\mathcal{K})$ are 0 for $n \geq 2$ since there are simplices in these dimensions. Thus,

$$H_n(\mathcal{K}) = \begin{cases} \mathbb{Z} & \text{for } n = 0, 1 \\ 0 & \text{for } n \geq 2. \end{cases}$$

1.3 Persistent Homology

Although homology is a strong tool, it is not sufficient to consider only the homology of a complex associated to a point cloud at a particular ϵ. Thus, it is a mistake to ask which value of ϵ is optimal. In fact, the correct thing to do is to consider all ϵ together. This leads to the idea of persistent homology.

Definition 11. Given a simplicial complex $\mathcal{K}$, a **filtration** is a totally ordered set of subcomplexes $\mathcal{K}_i$ of $\mathcal{K}$, indexed by the nonnegative integers, such that if $i \leq j$ then $\mathcal{K}_i \subseteq \mathcal{K}_j$. The total ordering itself is called a **filter**.

In our case, the filtration is given by the sequence of Rips complexes with different proximity ϵ and we have the natural inclusion map defined along the sequence.

Let $D \subseteq \mathbb{R}^n$ be a point cloud. For $\epsilon > 0$, let $\mathcal{R}_\epsilon(D)$ be the Rips complex of D at proximity ϵ. If we take coefficients in some field k, then we can view the chain complex of $\mathcal{R}_\epsilon(D)$ as a chain complex of k-modules, in other words, vector spaces over k. Let $\mathcal{R}_\epsilon^m(D)$ denote the m-chains of the complex $\mathcal{R}_\epsilon(D)$. To define persistent homology, we introduce the idea of varying ϵ. Let

$$\epsilon_1 < \epsilon_2 < \cdots$$

be a sequence of ϵ-values tending to ∞. We abbreviate $\mathcal{R}_n(D)$ for $\mathcal{R}_{\epsilon_n}(D)$. Then we have a sequence of vector spaces

$$\mathcal{R}_1^m(D) \to \mathcal{R}_2^m(D) \to \mathcal{R}_3^m(D) \to \cdots$$

and also a sequence of chain complexes

$$\begin{array}{ccccccc}
\vdots & & \vdots & & \vdots & & \\
\partial\downarrow & & \partial\downarrow & & \partial\downarrow & & \\
\mathcal{R}_1^{m-1}(D) & \xrightarrow{f_1} & \mathcal{R}_2^{m-1}(D) & \xrightarrow{f_2} & \mathcal{R}_3^{m-1}(D) & \xrightarrow{f_3} & \cdots \\
\partial\downarrow & & \partial\downarrow & & \partial\downarrow & & \\
\mathcal{R}_1^{m}(D) & \xrightarrow{f_1} & \mathcal{R}_2^{m}(D) & \xrightarrow{f_2} & \mathcal{R}_3^{m}(D) & \xrightarrow{f_3} & \cdots \\
\partial\downarrow & & \partial\downarrow & & \partial\downarrow & & \\
\mathcal{R}_1^{m+1}(D) & \xrightarrow{f_1} & \mathcal{R}_2^{m+1}(D) & \xrightarrow{f_2} & \mathcal{R}_3^{m+1}(D) & \xrightarrow{f_3} & \cdots \\
\partial\downarrow & & \partial\downarrow & & \partial\downarrow & & \\
\vdots & & \vdots & & \vdots & &
\end{array}$$

The diagram above commutes. In the diagram, each column is a chain complex, and the chain maps f_i connect chain complexes of successively larger simplicial complexes in the filtration together. Since the filtration of Rips complexes is linked by inclusion, the chain maps are induced by the inclusion maps.

Definition 12. A sequence of vector spaces $V_1 \to V_2 \to \cdots$ together with maps $f_n \colon V_n \to V_{n+1}$ is called a **persistence vector space**. A sequence of chain complexes connected by chain maps is called a **persistence complex**.

In the next section, we will show how a persistence vector space can be regarded as a graded module over the polynomial ring $k[x]$. This will allow us to use the structure theory for modules over $k[x]$ to classify persistence vector spaces. As an immediate consequence of this discussion, we have the following lemma.

Lemma 1. The homology spaces of a persistence complex are persistence vector spaces.

We therefore want to seek a classification of persistence vector spaces (which we abbreviate as persistence spaces from this point on). This can be done if we add some finiteness conditions.

Definition 13. A persistence space $V_1 \xrightarrow{f_1} V_2 \xrightarrow{f_2} \cdots$ is of **finite type** if

1. all the vector spaces V_i are finite dimensional
2. all but finitely many of the maps f_i are isomorphisms

Lemma 2. The chain spaces, and therefore the homology spaces, associated with the Rips complex of finite point cloud D are of finite type.

Proof. This is clear for Rips complexes themselves because the Rips complexes become stay for sufficiently large ϵ. Thus, for sufficiently large ϵ, the homology of the Rips complex is that of a point which completes the proof. □

Definition 14. For $n_1 \leq n_2$, $n_1, n_2 \in \mathbb{N}$, the persistence space $Q(n_1, n_2)$ has vector spaces

$$Q(n_1, n_2)_i = \begin{cases} k \text{ if } n_1 \leq i \leq n_2 \\ 0 \text{ otherwise} \end{cases}$$

with the maps between the k's being isomorphisms and others zero. We allow $n_2 = +\infty$.

Theorem 3. Every persistence space of finite type is isomorphic to a unique finite direct sum of spaces of the type $Q(n_1, n_2)$.

Proof. This is a direct result of the structure theorem which will be proved in the next section. □

The **barcode** representation of such a space has one bar starting at n_1 and ending at n_2 for each $Q(n_1, n_2)$ summand.

To give an example, the figure above shows four points (0,0), (0,1), (2,1), (2,0) and the Rips complex with different ϵ (1,2,$\sqrt{5}$ respectively).

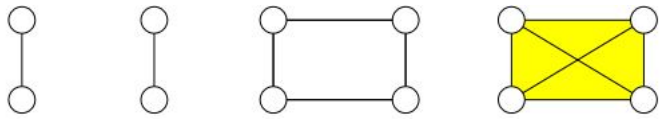

Figure 5: Rips complex with $\epsilon = 1, 2, \sqrt{5}$

The barcode representation of the homology groups for the figure above is shown below. The x-axis is ϵ. Each horizontal bar represents the birthdeath of a separate homology class. Longer bars correspond to more robust topological structure in the data.

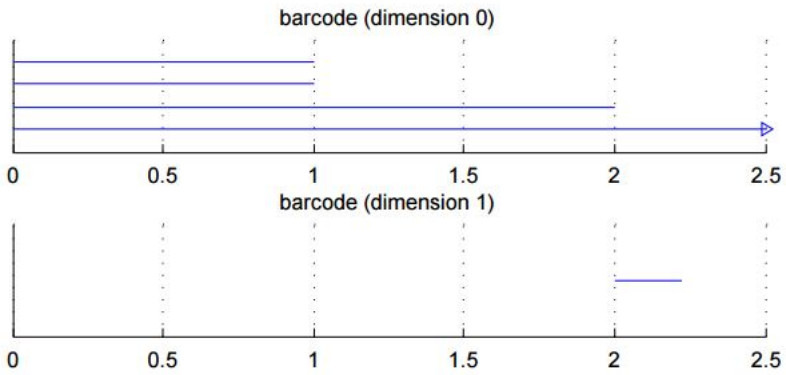

Figure 6: Example of barcodes

The top panel shows H_0. At $\epsilon = 0$ there are four bars for the four disconnected vertices. At $\epsilon = 1$ two edges appear, reducing the number of connected components to two. This is why the top two bars die. At $\epsilon = 2$, the vertices forms a rectangle and becomes fully connected, so one more bar dies. The remaining bar represents the one vertex that grabs everything to eventually become the fully connected component. It never dies. The bottom panel shows H_1. In the example above, a homology class corresponding to the hole is born at $\epsilon = 2$, when the rectangle becomes connected. It persists until $\epsilon = \sqrt{5}$ and dies because the Rips complex becomes the solid tetrahedron. This is represented by the single short bar.

1.4 Structure Theorem for Persistence Spaces

Definition 15. Let R be a ring, M,N be R-modules. The **direct sum** $A \oplus B = \{(a,b) \colon a \in A, b \in B\}$ is a module under component wise operations: for any $a \in A$,$b \in B$ and $r \in R$, $(a_1,b_1)+(a_2,b_2) = (a_1+a_2, b_1+b_2)$ and $r(a,b) = (ra,rb)$. This extends to a direct sum of finitely many R-modules. However, for a direct sum of infinitely many R-modules, all elements have all but finitely many components equal to 0.

Definition 16. A ring R is a **graded ring**, if there is a given family of subgroups $\{R_n\}_{n\in\mathbb{Z}}$ of R such that

1. $R = \bigoplus_n R_n$,
2. $R_n R_m \subseteq R_{n+m}$ for all n, m.

Note that any ring R is a graded ring with the trivial grading $R_0 = R$ and $R_n = 0$, for all $n \neq 0$. As another example, let k be a field and $x_1, \cdots, x_d$ be variables over k. For $p = \{p_1, \cdots, p_d\} \in \mathbb{N}^d$, let $x^p = x_1^{p_1} \cdots x_d^{p_d}$. Then the polynomial ring $R = k[x_1, \cdots, x_d] = \bigoplus_n R_n$ is a graded ring, where

$$R_n = \{\sum_{p\in\mathbb{N}^d} r_p x^p \colon r_p \in R \text{ and } p_1 + \cdots + p_d = n\}.$$

This is called the standard grading on the polynomial ring $k[x_1, \cdots, x_d]$.

Definition 17. Let R be a graded ring and M an R-module. We say that M is a **graded R-module** if there is a given family of subgroups $\{M_n\}_{n\in\mathbb{Z}}$ of M such that

1. $M = \bigoplus_n M_n$,
2. $R_n M_m \subseteq M_{n+m}$ for all n, m.

An nonzero element $m \in M$ is **homogeneous** of degree n if $m \in M_n$, for some $n \in \mathbb{Z}$. We denote the degree of m by $\deg m$. If $m \in M$ is a nonzero element, then we can express it uniquely as a finite sum $\sum_i m_i$, where each m_i is homogeneous. These m_i are called the **homogeneous components** of m.

Theorem 4. let k be a field. A persistence space over k is a graded $k[x]$-module.

Proof. If we have a persistence space:

$$V_1 \xrightarrow{f_1} V_2 \xrightarrow{f_2} \cdots$$

Let $V = V_1 \oplus V_2 \oplus \cdots$. If we let k act by scalar multiplication is the usual way and $x \cdot (v_1, v_2, \cdots) = (0, f_1(v_1), f_2(v_2), \cdots)$, then V becomes a graded module over $k[x]$ with homogeneous parts $V_1, V_2, \cdots$. For the other directions, if we have a graded $k[x]$-module V with homogeneous parts $V_1, V_2, \cdots$, then we can get a sequence of vector spaces $V_1 \to V_2 \to \cdots$ and the maps between the vector spaces are defined by the action of x on V. Because x is a homogeneous element in $k[x]$ of degree one, multiplication by x must necessarily be represented by linear maps from V_j to V_{j+1}. □

Definition 18. Let M be a graded R-module and n an integer. M **shifted** by n, denoted by $M(n)$, is defined to be equal to M as an R-module, but with its grading defined by $M(n)_k = M_{n+k}$.

Definition 19. Let R be a graded ring and M, N graded R-modules. Let $f\colon M \to N$ be an R-module homomorphism. Then f is a **graded homomorphism** if $f(M_n) \subseteq N_n$ for all n. In addition, f is a **graded isomorphism** if f is a graded homomorphism and an isomorphism.

From now on, when we say two graded modules are isomorphic, we always mean graded isomorphic.

Proposition 1. Let R be a graded ring and M, N graded R-modules. Let $f\colon M \to N$ be an R-module graded isomorphism, then the following are true:

1. $f(M_n) = N_n$ for all n,
2. f^{-1} is also a graded isomorphism.

Proof. 1. Pick an arbitrary element $b \in N_n$. Since f is an isomorphism, there exists $a \in M$ such that $f(a) = b$. Write a as a finite sum $a = \sum_i a_i$ where each a_i is homogeneous.

Since f is a graded homomorphism, $f(a) = \sum_i f(a_i) \in N_n$ and $f(a_i) \in N_i$. Since $N_i \cap N_j = \{0\}$ if $i \neq j$, we get that $f(a_i) = 0$ and thus $a_i = 0$ for all $i \neq n$. Hence, $a = a_n \in M_n$.

2. We want to show that that $f^{-1}(N_n) \subseteq M_n$. Pick an arbitrary element $a \in f^{-1}(N_n)$. Then $f(a) \in N_n = f(M_n)$ by (1). Then $f(a) = f(a')$ for some $a' \in M_n$. Since f is an isomorphism and thus one-to-one, $a = a' \in M_n$.

□

Definition 20. Let R be a ring, M be a graded R-module. If there is a function from I to $\mathbb{Z}$ which maps i to n_i and a graded isomorphism

$$f\colon \bigoplus_{i\in I} R(n_i) \to M,$$

then M is **free**.

Definition 21. Let R be a ring, M be a finitely generated graded R-module. Let $\{m_1, \cdots, m_n\}$ be a generating set of M. If this set is also linearly independent, then it is a **basis** of M. If in addition, every element in the set is homogeneous, we call the set a **homogeneous basis** of M. The **rank** of M is the cardinality of a basis of M (i.e. the number of elements in M).

Proposition 2. Let R be a ring, M be a graded R-module. Then M is free if and only if it has a homogeneous basis.

Proof. " $\implies$ " Assume that M if free. Then by definition there exists an isomorphism $f\colon \bigoplus_{i\in I} R(n_i) \to M$. Let e_i be the element which has 1 in $R(n_i)$-coordinate and 0 elsewhere and $m_i = f(e_i)$. Clearly, $\{e_i\colon i \in I\}$ is a basis for $\bigoplus_{i\in I} R(n_i)$. We will show that $\{m_i\colon i \in I\}$ is a basis for M. Since f is graded and e_i are homogeneous for all i, m_i are homogeneous for all i. If $\sum_i r_i m_i = 0$ where $r_i \in R$ for all i. Since f is a graded isomorphism, f^{-1} is also a graded isomorphism by proposition 1. Thus,

$$f^{-1}(\sum_i r_i m_i) = \sum_i r_i f^{-1}(m_i) = \sum_i r_i e_i = f(0) = 0.$$

Since the set of all e_i is linearly independent, we get that $r_i = 0$ for all i which means that $\{m_i\colon i \in I\}$ is also linearly independent. To show that $\{m_i\colon i \in I\}$ generates M, take an arbitrary element $m \in M$. Since f^{-1} is an isomorphism and $\{e_i\colon i \in I\}$ generates $\bigoplus_{i\in I} R(n_i)$, we have

$$m = f\left(f^{-1}(m)\right) = f\left(\sum_i r_i e_i\right) = \sum_i r_i f(e_i) = \sum_i r_i m_i.$$

Therefore, $\{m_i \colon i \in I\}$ is a linearly independent generating set, namely, a basis for M.

" $\Longleftarrow$ " Suppose that M is generated by a linearly independent collection of homogeneous elements $\{m_i \colon i \in I\}$. Consider the space $\bigoplus_{i\in I} R(-\deg m_i)$. Let e_i be the element which has 1 in $R(-\deg m_i)$-coordinate and 0 elsewhere. Clearly, $\{e_i \colon i \in I\}$ is a basis for $\bigoplus_{i\in I} R(-\deg m_i)$. Define a homomorphism f by

$$f \colon \bigoplus_{i\in I} R(-\deg m_i) \to M$$
$$e_i \mapsto m_i.$$

We first show that f is an isomorphism. Take distinct elements $r, r' \in \bigoplus_{i\in I} R(-\deg m_i)$. Then since $\{e_i \colon i \in I\}$ is a basis, we have $r = \sum_i r_i e_i$ and $r' = \sum_i r'_i e_i$ where $r_i \neq r'_i$ for some i. Then

$$f(r) = f(\sum_i r_i e_i) = \sum_i r_i f(e_i) = \sum_i r_i m_i,$$
$$f(r') = f(\sum_i r'_i e_i) = \sum_i r'_i f(e_i) = \sum_i r'_i m_i.$$

Since $\{m_i \colon i \in I\}$ is a basis for M and $r_i \neq r'_i$ for some i, $f(r) \neq f(r')$. Therefore f is injective. Take an arbitrary element $m \in M$. Then has a unique decomposition $m = \sum_i r_i m_i$. And we have $f(\sum_i r_i e_i) = \sum_i r_i f(e_i) = \sum_i r_i m_i = m$. Therefore, f is surjective. To sum up, f is both injective and surjective which implies that f is a isomorphism.

And by the way we define the homomorphism f, f is automatically graded. Therefore, f is a graded isomorphism and thus M is free. □

Note that although a free graded R-module M has a homogeneous basis, the rank of a M is not necessarily well-defined. The rank is well-defined if and only if every basis of M has the same cardinality. This is true if R is commutative. Before proving this fact, we need to define what a maximal ideal is.

Definition 22. Let R be a ring and I an ideal of R such that $I \neq R$. I is a **maximal ideal** of R if for any ideal J with $I \subsetneq J$, either $J = I$ or $J = R$.

we will also take for granted Zorn's lemma without proof. The lemma implies the following.

Lemma 3. Every nontrivial ring R contains a maximal ideal.

Proposition 3. Let R be a commutative graded ring, M be a finitely generated and free graded R-module. Then every basis of M has the same cardinality.

Proof. Let I be an ideal in R. Then M/IM is a free R/I module. Moreover, by Zorn's lemma, we can take I to be a maximal ideal in R. Now we will show that R/I is a field. If $[x] \neq 0$ in R/I, this means that $x \notin I$. Therefore $\langle I, x\rangle$ is an ideal in R which contains I. By definition of a maximal ideal, $\langle I, x\rangle = R$ and thus contains the 1. Hence, there exists $y \in R$ such that $xy - 1 \in I$. This means that $[y]$ is the inverse of $[x]$ in R/I. Therefore, R/I is a field. Therefore, M/IM is a vector space over R/I and the rank of M/IM is well-defined. Since M is free and finitely generated, $M \cong R^n$ for some n. Thus, $M/IM \cong (R/I)^n$. Therefore, the rank of M is also well-defined and is the same as the rank of the vector space M/IM. □

Definition 23. Let R be a ring, $M = \oplus_n M_n$ be a graded R-module and N a submodule of M. For each $n \in \mathbb{Z}$, let $N_n = N \cap M_n$. If the family of subgroups $\{N_n\}$ makes N into a graded R-module, we say that N is a **graded submodule** of M. Note that for any submodule N of M, $R_n N_m \subseteq N_{n+m}$. Thus, N is graded if and only if $N = \bigoplus_n N_n$.

Proposition 4. Let R be a graded ring, M a graded R-module and N a graded submodule of M. Then M/N is a graded R-module, and graded by

$$M/N = \bigoplus_n (M/N \cap M_n).$$

Lemma 4. Let R be a graded principal ideal domain, M be a graded, finitely generated, and free R-module of rank 1 and N be a graded submodule of M. Then N is also free of rank 1.

Proof. By definition, M is generated by a single element m of degree t for some $t \in \mathbb{Z}$. Define $S = \{s \in R \colon sm \in N\}$. Note that if $s_1, s_2 \in S$, then $s_1 m, s_2 m \in N$ and $(s_1 + s_2)m \in N$ which means that $s_1 + s_2 \in S$. Also if $s \in S$ and $r \in R$, then $sm \in N$ and $rsm \in N$ which means that $rs \in S$. Hence, S is an ideal in R. Because R is a pid, $S = \langle s\rangle$ for some $s \in R$. Thus, N generated by sm. Thus N is generated by sm. Next we will show that s is homogeneous. Assume that s is not homogeneous, then we can write s as a sum of at least two homogeneous components of different degree $s = s_1 + \cdots + s_k$. Suppose $\deg s_1 < \cdots < \deg s_k$. Every element $r \in R$ is also of similar form $r = r_1 + \cdots + r_l$ where $\deg r_1 < \cdots < \deg r_l$. So every element in N has the form $rsm = r_1 s_1 m + \cdots + r_l s_k m$ and $\deg r_1 s_1 m < r_l s_k m$. Therefore $N_n = M_n \cap N = \{0\}$ and $N = \bigoplus_n N_n = \{0\}$ which is a contradiction. Hence s is homogeneous and N is isomorphic to $R(-\deg sm)$. □

Lemma 5. Let R be a ring. If $f \colon M \to N$ is a graded homomorphism of graded R-modules, then $\ker f$ is a graded submodule of M and $\mathrm{im} f$ is a graded submodule of N.

Proof. Since f is a homomorphism of R-modules, $\ker f$ is submodule of M and $\mathrm{im}f$ is a submodule of N. It suffices to show that $\ker f = \bigoplus_n (\ker f \cap M_n)$ and that $\mathrm{im}f = \bigoplus_n (\mathrm{im}f \cap N_n)$.

Let $m \in \ker f$. We can write m as a sum $m = \sum_{i=1}^k m_i$ where $m_i \in M_i$. Note that

$$0 = f(m) = f(\sum_{i=1}^k m_i) = \sum_{i=1}^k f(m_i).$$

Since the components m_i are homogeneous of different degrees and f is graded, $f(m_i)$ are also homogeneous of different degrees in N. Therefore, we must have $f(m_i) = 0$ for all i which means that $m_i \in \ker f$ for all i. Therefore, $\ker f \subseteq \bigoplus_n (\ker f \cap M_n)$. For the other direction, let $m \in \bigoplus_n (\ker f \cap M_n)$. Then $m = \sum_{i=1}^k m_i$ where each $m_i \in (\ker f \cap M_n)$. Since f is a homomorphism,

$$f(m) = f(\sum_{i=1}^k m_i) = \sum_{i=1}^k f(m_i) = 0$$

which means that $m \in \ker f$. Therefore, $\ker f = \bigoplus_n (\ker f \cap M_n)$.

Let $n \in \mathrm{im}f$. We can write n as a sum $n = \sum_{i=1}^k n_i$ where $n_i \in N_i$. As $n \in \mathrm{im}f$, there exists $m \in M$ such that $f(m) = n$. Say $m = \sum_{j=1}^l m_j$ where $m_j \in M_j$. Then,

$$n = f(m) = f(\sum_{j=1}^l m_j) = \sum_{j=1}^l f(m_j).$$

Since the decomposition is unique, we must have $k = l$ and (after reordering) $f(m_i) = n_i$ for all i. Thus $n_i \in \mathrm{im}f$ for all i which implies that $\mathrm{im}f \subseteq \bigoplus_n (\mathrm{im}f \cap N_n)$. For the other direction, let $n \in \bigoplus_n (\mathrm{im}f \cap N_n)$. Then $n = \sum_{i=1}^k n_i$ where each $n_i \in (\mathrm{im}f \cap N_n)$. Then for each i there exists m_i such that $f(m_i) = n_i$. Let $m = \sum_{i=1}^k m_i$. Since f is a homomorphism,

$$f(m) = f(\sum_{i=1}^k m_i) = \sum_{i=1}^k f(m_i) = \sum_{i=1}^k n_i = n$$

which means that $n \in \mathrm{im}f$. Therefore, $\mathrm{im}f = \bigoplus_n (\mathrm{im}f \cap N_n)$. □

Lemma 6. Let R be a commutative ring, $\{M_i \colon i = 1, \cdots, n\}$ be a finite collection of graded R-modules. Then $M = \bigoplus_{i=1}^n M_i$ is a graded R-module. Moreover, if each M_i if free of rank r_i, then M is free of rank $\sum_i r_i$.

Proof. M is obviously a R-module. We need to check that M is graded. For each i, let M_i be graded by $M_i = \bigoplus_j M_{i_j}$. Then $M = \bigoplus_i \left(\bigoplus_j M_{i_j}\right) = \bigoplus_j \left(\bigoplus_i M_{i_j}\right)$. If we fix j and pick

$m \in \bigoplus_i M_{i_j}$, then $m = (m_1, \cdots, m_n)$ where $m_i \in M_{i_j}$ for each i. Take an arbitrary $r \in R$, $rm = (rm_1, \cdots, rm_n)$ and for each i, $rm_i \in M_{i_{j+1}}$. Thus $rm \in \bigoplus_i M_{i_{j+1}}$ which means that M is graded by $M = \bigoplus_j \left(\bigoplus_i M_{i_j}\right)$.

Moreover, if each M_i if free of rank r_i, then for each i, there exists a basis B_i of M_i. Then $\cup B_i$ is a basis of M and hence M is free of rank $\sum_i r_i$. □

Lemma 7. Given a short exact sequence with maps, q and r as follow:

$$0 \longrightarrow A \xrightarrow{p} B \xrightarrow{r} C \longrightarrow 0$$

If there exists a map $u\colon C \to B$ such that ru is the identity on C, then B is isomorphic to the direct sum of A and C.

Proof. Every element in B is in the set $\ker r + \mathrm{im} u$ since for all b in B, $b = (bur(b)) + ur(b)$. Since if $r(b) = 0$ and $u(c) = b$, then $0 = ru(c) = c$, the intersection of $\ker r$ and $\mathrm{im} u$ is 0.

By exactness, $\mathrm{im} q = \ker r$, and since q is injective, $\mathrm{im} q$ is isomorphic to A, so A is isomorphic to $\ker r$. Since ru is a bijective, u is injective, and thus $\mathrm{im} u$ is isomorphic to C. So B is the direct sum of A and C. □

Lemma 8. Let k be a field and $R = k[x]$ be a polynomial ring with the standard grading, M be a finitely generated free R-module of rank n and N be a graded submodule of M. Then

1. N is free of rank $d \leq n$,
2. There exist a homogeneous basis $\{m_1, m_2, \cdots, m_n\}$ of M, and nonzero homogeneous elements $\{r_1, r_2, \cdots, r_d\} \in R$ with $r_i | r_{i+1}$ for $i = 1, \cdots, d-1$ such that $\{r_1 m_1, r_2 m_2, \cdots, r_d m_d\}$ is a homogeneous basis of N.

Proof. We prove the theorem by induction on n. The case when $n = 1$, the theorem holds by lemma 2. Assume the theorem holds for all modules of rank $\leq n - 1$. We have $M \cong \bigoplus_{i=1}^{n} k[x](-t_i)$ where all t_i are integers. Therefore, $\{m_1, \cdots, m_n\}$ is a basis for M where each m_i is homogeneous of degree t_i. If $N = \{0\}$, we are done. If $N \neq \{0\}$, consider the projections

$$\pi_i\colon M \to k[x](t_i).$$

It is easy to check that all these projections are graded homomorphisms. Since $N \neq \{0\}$, $\pi_i(N) \neq \{0\}$ for some i. Thus,by lemma 3, $\pi_i(N)$ is a nonzero graded submodule of $k[x](t_i)$, which by lemma 2, is free of rank ≤ 1 and generated by some homogeneous element $x^{d_i} \in k[x](t_i)$. Consider the exact sequence:

$$0 \longrightarrow \ker \pi_i \cap N \longrightarrow N \xrightarrow{\pi_i|_N} \pi_i(N) \longrightarrow 0$$
$$\varphi$$

Note that $\pi_i(x^{d_i} m_i) = x^{d_i}$. Define the homomorphism φ such that $\varphi(x^{d_i}) = x^{d_i} m_i$. Since $\pi_i(N)$ is generated by x^{d_i}, φ is well-defined and $\varphi(\pi_i(N))$ is generated by $x^{d_i} m_i$. It is easy to check that $\pi_i \circ \varphi$ is the identity on $\pi_i(N)$. Thus, by the splitting lemma, N is isomorphic to $(\ker \pi_i \cap N) \oplus \pi_i(N)$.

We can check that $\ker \pi_i \cap N$ is a graded submodule of $\ker \pi_i$. Since $\ker \pi_i$ has rank $\leq n-1$, by the induction hypothesis, $\ker \pi_i \cap N$ is free of rank $\leq n-1$. By lemma 4, N is free of rank $\leq n$. Note that $\ker \pi_i$ is a free graded R-module generated by $\{m_1, \cdots, m_{i-1}, m_{i+1}, \cdots, m_n\}$. Thus by our induction assumption, there exist homogeneous elements $x^{d_1}, \cdots, x^{d_{i-1}}, x^{d_{i+1}}, \cdots, x^{d_n} \in R$ such that $\{x^{d_1} m_1, \cdots, x^{d_{i-1}} m_{i-1}, x^{d_{i+1}} m_{i+1}, \cdots, x^{d_n} m_n\}$ is the homogeneous basis for $\ker \pi_i \cap N$. Since $\varphi(\pi_i(N))$ is generated by $x^{d_i} m_i$, we get that $\{x^{d_1} m_1, \cdots, x^{d_n} m_n\}$ is the homogeneous basis of N. By reordering, the divisibility condition is satisfied. □

Note that the homogeneous elements $\{r_1, r_2, \cdots, r_d\} \in R$ are just powers of x.

Lemma 9. Let R be a graded ring, M, M' graded R-modules. Let $\varphi\colon M \to M'$ be a surjective graded homomorphism with kernel N. Let $\pi\colon M \to M/N$ the graded quotient homomorphism. Then there exists a graded isomorphism $\tilde{\varphi}\colon M/N \to M'$ satisfying $\varphi = \tilde{\varphi} \circ \pi$. In other words, $M/N \cong M'$.

Proof. The homomorphism theorem for modules gives us an isomorphism of R-modules $\tilde{\varphi}\colon M/N \to M'$ satisfying $\varphi = \tilde{\varphi} \circ \pi$. We only need to check that $\tilde{\varphi}$ is graded. Let M be graded by $M = \bigoplus_n M_n$,M' be graded by $M' = \bigoplus_n M'_n$. Then

$$\tilde{\varphi}(M_n/N \cap M_n) = \tilde{\varphi}(\pi(M_n)) = \varphi(M_i) \subseteq M'_i$$

□

Lemma 10. Let R be a ring, $M_1, \cdots, M_n$ be graded R-modules and $N_i \subseteq M_i$ graded submodules. Then

$$(M_1 \oplus \cdots \oplus M_n)/(N_1 \oplus \cdots \oplus N_n) \cong M_1/N_1 \oplus \cdots \oplus M_n/N_n.$$

Proof. Consider the homomorphism of $M_1 \oplus \cdots \oplus M_n$ onto $M_1/N_1 \oplus \cdots \oplus M_n/N_n$ defined by $(m_1, \cdots, m_n) \mapsto (m_1 + N1, \cdots, m_n + N_n)$. The kernel of this map is $N_1 \oplus \cdots \oplus N_n \subseteq M_1 \oplus \cdots \oplus M_n$, so by the lemma 5,

$$(M_1 \oplus \cdots \oplus M_n)/(N_1 \oplus \cdots \oplus N_n) \cong M_1/N_1 \oplus \cdots \oplus M_n/N_n.$$

□

Theorem 5. (Structure Theorem for Graded Modules Over Polynomial Rings) Let $k[x]$ be a polynomial ring with the natural grading and M a finitely generated graded $k[x]$-module of rank n. Then

$$M \cong k[x](t_1)/\langle x^{d_1}\rangle \oplus \cdots \oplus k[x](t_m)/\langle x^{d_m}\rangle \oplus k[x](t_{m+1}) \oplus \cdots \oplus k[x](t_n)$$

where $x^{d_i} | x^{d_{i+1}}$ for $i = 1, \cdots, m-1$. In other words, $d_i \leq d_{i+1}$ for $i = 1, \cdots, m-1$.

Proof. Let $\{m_1, \cdots, m_n\}$ be a homogeneous generating set of M. We can find such a set by taking the generators to be the elements of M_n. Then there exists a surjective graded homomorphism $f\colon \bigoplus_{t=1}^n k[x](t_i) \to M$. By lemma 3, $\ker f$ is as submodule of $\bigoplus_{t=1}^n k[x](t_i)$. Since $\bigoplus_{t=1}^n k[x](t_i)$ is a free $k[x]$-module, by lemma 6, there exists a basis $\{e_1, \cdots, e_n\}$ of $\bigoplus_{t=1}^n k[x](t_i)$ and homogeneous elements $x^{d_1}, \cdots, x^{d_m} \in k[x]$ such that $\{x^{d_1}e_1, \cdots, x^{d_m}e_m\}$ is a basis of $\ker f$. By lemma 7, we have $M \cong \bigoplus_{t=1}^n k[x](t_i)/\ker f$. The theorem follows from lemma 8, taking $r_i = 0$ for $m < i \leq n$. □

Using Theorem 4, this translates immediately yo the structure theorem of persistence space of finite type, which is Theorem 3 in the previous section.

2 Computational Methods

2.1 Smith Normal Form Codes

While learning the technique of topological data analysis, we have developed an program in C++ which computes the Smith normal form of matrices over polynomial rings.

Definition 24. Let A be a nonzero $m \times n$ matrix over a principal ideal domain R. There exist invertible $m \times m$ and $n \times n$-matrices S, T so that the product SAT is

$$\begin{pmatrix} a_1 & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & a_2 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & \ddots & & & & 0 \\ \vdots & \vdots & & a_r & & & \vdots \\ \vdots & \vdots & & & 0 & & \vdots \\ \vdots & \vdots & & & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & \cdots & 0 \end{pmatrix}$$

and the diagonal elements α_i satisfy $\alpha_i \mid \alpha_{i+1}\ \forall\ 1 \leq i < r$. This matrix is the **Smith normal form** of the matrix A.

Why is the computation of Smith norm form involved in topological data analysis? Consider an R-module M given by n generators and m relations. Then M is a quotient of R^n by $T(R^m)$, where T is the matrix of relations. Putting this matrix in Smith normal form can be achieved by row and column operations, i.e. by basis changes in R^n and R^m, which do not affect the isomorphism class of the module. Thus, the existence of Smith normal form gives a proof of the ungraded analog of the structure theorem, and its computation is the central algorithmic step in computing homology with coefficients in a principal ideal domain.

The procedure of computing the Smith normal form over a Euclidean domain such as $k[x]$ involves three steps.

Step I: Suppose we take a non-zero matrix A as the input. By interchanging the rows and columns, we can move an element of smallest degree, as the $(1,1)^{th}$ element. Now bring the first row, to the form $(a_{11}, 0, \cdots, 0)$ as follows. If an element in the first row is a multiple of a_{11}, we subtract a suitable multiple of the first column from that column. Otherwise, if

a_{1k} is not a multiple of a_{11}, then we write $a_{1k} = qa_{11} + r$ where $\deg(r) < \deg(a_{11})$. Then we subtract p times of the first column from the k^{th} column and interchange the first column and the k^{th} column. Now repeat the process, until we get all zeroes in the first row, except the first one. We can do similar row operations on the first column to obtain a matrix of the form,

$$\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & b_{m1} & \cdots & b_{mn} \end{pmatrix}$$

Step II: Now, if all the entries b_{ij} are multiples of a_{11}, we apply step I to the matrix $B = [b_{ij}]$. Otherwise, if the entry b_{ij} is not a multiple of a_{11}, then we add the i^{th} row to the first row. Repeat the step I to make all other entries in the first row and column zero. Since, each time we are reducing the degree of a_{11}, we must get a matrix of the form above such that $a_1 1$ divides all b_{ij} in finite steps.

Step III: Repeat steps I and II to the matrix $B = [b_{ij}]$.

To give an example, let us consider the matrix

$$A = \begin{pmatrix} x^2 & x-1 \\ x & x^2 \end{pmatrix}.$$

Pick one of the polynomials with the lowest degree. In our case, a_{12} and a_{21} both have lowest degrees, so we can pick either one. Let's Pick A_{12}.

Swap columns so that the pivot becomes the $(1,1)^{th}$ element. In our case, we need to swap the first and the second columns. We now have

$$\begin{pmatrix} x-1 & x^2 \\ x^2 & x \end{pmatrix}.$$

Write all the polynomials in the first row as $a_{1k} = qa_{11} + r$ where $\deg(r) < \deg(a11)$. In our case:

$$\begin{pmatrix} x-1 & (x-1)(x+1)+1 \\ x^2 & x \end{pmatrix}.$$

Then subtract $x+1$ times the first column from second column and then interchange those two columns. Now we have

$$\begin{pmatrix} 1 & x-1 \\ -x^3-x^2+x & x^2 \end{pmatrix}.$$

Then we subtract $x-1$ times the first column from second column to bring a_{12} to 0.

$$\begin{pmatrix} 1 & 0 \\ -x^3-x^2+x & -x^4+x^2-x \end{pmatrix}.$$

Then we subtract $-x^3-x^2+x$ times the first row from second row to bring a_{21} to 0.

$$\begin{pmatrix} 1 & 0 \\ 0 & -x^4+x^2-x \end{pmatrix}.$$

Finally, we need to check that a_{11} divides a_{22}, which is does in our case. Therefore the matrix above is the Smith normal form for A.

The code that I wrote to implement this algorithm is shown in the appendix.

2.2 Algorithms

In this section, we will first explain the procedure of an algorithm which computes the persistent homology, then give an example and finally explain why this algorithm works. The procedure involves three steps.

Step I: We first list all simplices that appear at some point in the filtration. Then sort them in increasing degree and dimension. The degree of a simplex is the proximity parameter ϵ at which it appears in the filtration of Rips complex. It does not matter if we sort by degree first or dimension first. Label all the simplices starting with index 0. Every simplex is associated with a boolean variable which is called "pivot mark" and is initialized to all False. Every simplex is also associated with a pointer which is initialized to Null.

Step II: Repeat the following steps for every simplex in index order:

1. Write out its boundary.

2. "Remove pivots" from the boundary, this has two stages:

   (a) Look through the boundary. If any simplex appears whose "pivot mark" is False, replace it by 0 in the boundary.

   (b) If the resulting reduced boundary is 0, we are done. Otherwise, iterate the following process until te reduced boundary becomes 0 or the algorithm tells us to stop.

      i. Select the simplex in the reduced boundary whose index is greatest. Call this simplex z.

      ii. Look at the pointer for z. If the pointer is Null, then exit the loop. If the pointer is not Null, subtract an appropriate multiple of the reduced boundary of the the simplex to which the pointer points to zero out the coefficient of z in the reduced boundary. Now go back to step i.

3. The "remove pivots" step returns a reduced boundary for the current simplex. If this is 0, set the "pivot mark" to True. If z is the simplex of highest index in the reduced boundary, set the pointer of z to the index of the current simplex.

Step III: Go through the simplices whose pointer mark is True. Let k be the current simplex. If the pointer of k is Null, add to the homology in dimension $\dim k$ an interval from $\deg k$ to ∞. If the pointer of k points to a simplex l, then add to the homology in dimension $\dim k$ an interval from $\deg k$ to $\deg l$.

Let's consider an example which we have seen in section 1.3.

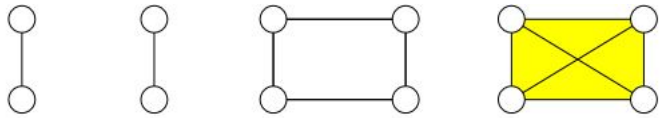

Figure 7: Example of the algorithm

Label the upper left vertex by a, the upper right vertex by b, the lower left vertex by c and the lower right vertex by d. We first go through step I and write out all the information we have in the table below. In this example, sorting by degree first and by dimension first give us the same ordering. In general, when the algorithm is computing a simplex, all its subsimplices have already been considered be the algorithm no matter we sort by degree first or dimension first, because if a simplex appear at time t, all of its subsimplices must appear at times smaller than or equal to t.

Index	Simplex	Degree	Pivot Mark	Boundary	Pointer
0	a	0	F	0	N
1	b	0	F	0	N
2	c	0	F	0	N
3	d	0	F	0	N
4	ac	1	F	$a-c$	N
5	bd	1	F	$b-d$	N
6	ab	2	F	$a-b$	N
7	cd	2	F	$c-d$	N
8	ad	$\sqrt{5}$	F	$a-d$	N
9	bc	$\sqrt{5}$	F	$b-c$	N
10	abc	$\sqrt{5}$	F	$ab-ac+bc$	N
11	abd	$\sqrt{5}$	F	$ab-ad+bd$	N
12	acd	$\sqrt{5}$	F	$ac-ad+cd$	N
13	bcd	$\sqrt{5}$	F	$bc-bd+cd$	N
14	$abcd$	$\sqrt{5}$	F	$abc-abd+$ $adc-bcd$	N

Table 1: Initial state

The boundaries of simplices a, b, c, d are already 0, so we set the "pivot" mark to True and we are done with these simplices.

For simplex ac, its boundary $a-c$ contains no simplices with "pivot mark" False. The simplex with the greatest index is c. The "pivot mark" of c is True and the pointer is Null, so we are done with the "remove pivots" step. Then we set the pointer of c to 4. Similar procedures happen for simplices bd and ab. The boundaries of bd, ab are unchanged and we set the pointer of d to 5 and b to 6.

For simplex cd, its boundary $c-d$ contains no simplices with "pivot mark" False. The simplex with the greatest index is d. The "pivot mark" of d is True and the pointer is 5. The reduced boundary of simplex number 5 is $b-d$. So we subtract $b-d$ from $c-d$ to eliminate d. The reduced boundary of cd is now $c-b$. The simplex with the greatest index is now c. The pointer of c is 4, so we subtract -1 times $a-c$ from $c-b$. The reduced boundary is now $a-b$. The simplex with the greatest index is now b. The pointer of b is 6, so we subtract $a-b$ from $a-b$. The reduced boundary of cd is now 0, so we need to set the "pivot mark" of cd to True. Similar procedures happen for simplices ad and bc. Their

boundaries are reduced to 0 and we need to set their "pivot mark" to True. The updated table is presented below.

Index	Simplex	Degree	Pivot Mark	Boundary	Reduced Boundary	Pointer
0	a	0	T	0	0	N
1	b	0	T	0	0	6
2	c	0	T	0	0	4
3	d	0	T	0	0	5
4	ac	1	F	$a-c$	$a-c$	N
5	bd	1	F	$b-d$	$b-d$	N
6	ab	2	F	$a-b$	$a-b$	N
7	cd	2	T	$c-d$	0	N
8	ad	$\sqrt{5}$	T	$a-d$	0	N
9	bc	$\sqrt{5}$	T	$b-c$	0	N
10	abc	$\sqrt{5}$	F	$ab-ac+bc$		N
11	abd	$\sqrt{5}$	F	$ab-ad+bd$		N
12	acd	$\sqrt{5}$	F	$ac-ad+cd$		N
13	bcd	$\sqrt{5}$	F	$bc-bd+cd$		N
14	$abcd$	$\sqrt{5}$	F	$abc-abd+adc-bcd$		N

Table 2: State after consider 0- and 1-simplices

For simplex abc, its boundary is $ab-ac+bc$ and ab, ac both have "pivot mark" False, so we replace them with 0. The reduced boundary is now bc. The simplex with the greatest index is bc. The "pivot mark" of bc is True and the pointer is Null, so we are done with the "remove pivots" step. Then we set the pointer of bc to 10. Similar procedures happen for simplices abd and ab. The boundaries of bd, ab are reduced to $-ad, cd$ and we set the pointer of ad to 11 and cd to 12. For simplex bcd, if we follow the steps correctly, the boundary will be reduced to 0 and we need to set its "pivot mark" to True.

For the simplex $abcd$, its boundary is $abc-abd+adc-bcd$. We reduce the boundary to $-bcd$ because all the other simplices appeared in the boundary have "pivot mark" True. The simplex with the greatest index is bcd. The "pivot mark" of bcd is True and the pointer is Null, so we are done with the "remove pivots" step. Then we set the pointer of bcd to 14. The updated table is presented below.

Index	Simplex	Degree	Pivot Mark	Boundary	Reduced Boundary	Pointer
0	a	0	T	0	0	N
1	b	0	T	0	0	6
2	c	0	T	0	0	4
3	d	0	T	0	0	5
4	ac	1	F	$a-c$	$a-c$	N
5	bd	1	F	$b-d$	$b-d$	N
6	ab	2	F	$a-b$	$a-b$	N
7	cd	2	T	$c-d$	0	12
8	ad	$\sqrt{5}$	T	$a-d$	0	11
9	bc	$\sqrt{5}$	T	$b-c$	0	10
10	abc	$\sqrt{5}$	F	$ab-ac+bc$	bc	N
11	abd	$\sqrt{5}$	F	$ab-ad+bd$	$-ad$	N
12	acd	$\sqrt{5}$	F	$ac-ad+cd$	cd	N
13	bcd	$\sqrt{5}$	T	$bc-bd+cd$	0	14
14	$abcd$	$\sqrt{5}$	F	$abc-abd+adc-bcd$	$-bcd$	N

Table 3: State after completion of step II

Finally, we can compute the homology. In dimension 0, we have four intervals because there are four simplices in dimension 0 with "pivot mark" True. According the step III of the algorithm, the four intervals are $[0,\infty], [0,2], [0,1], [0,1]$. In dimension 1, we have three intervals which are $[2,\sqrt{5}], [\sqrt{5},\sqrt{5}], [\sqrt{5},\sqrt{5}]$. In dimension 2, we have one interval which is $[\sqrt{5},\sqrt{5}]$. The interval $[\sqrt{5},\sqrt{5}]$ is trivial. The result we get is consistent with the previous barcode representation which is shown below.

So why does this algorithm work? Our goal is to compute the homology of the chain complex

$$\cdots C_{k+1} \xrightarrow{\partial_{k+1}} C_k \xrightarrow{\partial_k} C_{k-1} \cdots$$

as a graded $k[x]$-module where k is a field. The process is to inductively find homogeneous bases for each C_k such that, with respect to these bases, for C_k and C_{k-1}, the boundary matrix has the Smith normal form

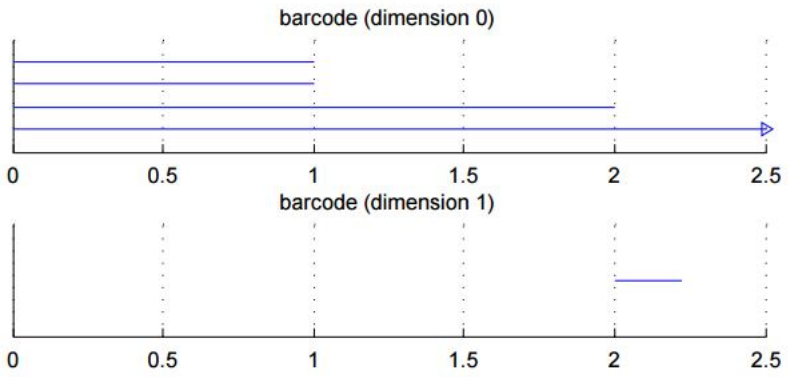

Figure 8: Barcode for the Example

$$\begin{pmatrix} x^{d_1} & 0 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & x^{d_2} & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 0 & \ddots & & & & 0 \\ \vdots & \vdots & & x^{d_m} & & & \vdots \\ \vdots & \vdots & & & 0 & & \vdots \\ \vdots & \vdots & & & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & \cdots & 0 \end{pmatrix}.$$

It will follow that the homology in dimension $k-1$ has the form

$$k[x](t_1)/\langle x^{d_1}\rangle \oplus \cdots \oplus k[x](t_m)/\langle x^{d_m}\rangle \oplus \text{free terms}$$

where $t_1, t_2, \cdots$ are the degrees of the generator and the number of free terms is determined by the rank of $\ker \partial_{k-1}$.

However, because of the special properties of homogeneous ideals in $k[x]$ (namely, they are totally ordered by inclusion), it is enough to reduce the matrices successively to lower triangular form by column operations. The row operations to get rid of the entries below the diagonal will automatically exist without changing the pivot. The algorithm described before is actually a shorthand for certain elementary row or column operations on the boundary matrix:

Step II:

1. No operations happen in this step.

2. (a) Since $\partial^2 = 0$, if a simplex is a pivot column in the boundary matrix running from

that simplex, then it is not map to a generator of $\ker \partial_k$ where k is dimension of the simplex.

(b) i. No operations happen in this step.

ii. This step is a shorthand for a elementary column operation. We are subtracting an appropriate multiple of the column, which corresponds to pointer of the simplex with the greatest index, from another column, which correspond to the current simplex.

After step II and possibly some swaps between columns and row, we are guaranteed to get a lower triangular boundary matrix and the elementary row and column operations only change the bases for C_k, but leave the homology groups unchanged.

After we have finished step II, the information in the table represents that there are bases for the chain groups $C_0, \cdots, C_n$ such that:

1. each row of the table corresponding to a k-simplex $k \leq n$ represents a basis vector consisting of that simplex plus a linear combination of k-simplices that are of lower index in the table,

2. the boundary matrices $\partial : C_k \to C_{k+1}$ are in Smith normal form,

3. if the "pivot mark" for a given k-simplex σ is False, then the corresponding basis vector is part of the basis for $\mathbb{Z}_k$ (the k-cycle), i.e. corresponds to a zero diagonal in the Smith normal form. When this is the case, the pointer points to a $(k+1)$-simplex whose corresponding basis vector has boundary containing σ plus simplices of lesser index,

4. if the "pivot mark" for a given k-simplex σ is True, then the boundary of the associated basis vector is nonzero. In this case the Smith normal form of the boundary matrix going from C_k to C_{k-1} contains a diagonal element x^l sending the basis vector associated to σ to the basis vector associated to the "reduced boundary" simplex with the highest degree. The value of l is the difference between the degrees of these two simplices.

We can verify by induction on n that, if the above statementsare true up to dimension n, and then after we excute the algorithm on $(n+1)$-simplices only, these statements will be true up to dimension $n+1$ also. Therefore, the homology is computed from the data by the process described in step III.

2.3 Persistent Homology Codes

Because the limited time for this project, instead of writing the complete package of codes which compute barcodes from data sets, we use one of the available packages in Matlab — javaPlex written by Computational Topology workgroup at Stanford University. There are two main functionalities of javaPlex: the automated construction of filtered complexes from geometric data and the computation of the persistent homology of filtered chain complexes of vector spaces, implementing the algorithm discussed in the previous section. We first create a .mat file which contains a $m \times n$ matrix which represents our data set — m points in $\mathbb{R}^n$. Then we apply a function to construct filtered complexes from the .mat file. Then we apply a second function which computes the persistent homology of filtered chain complexes of vector spaces which we just got. Finally, we plot the barcodes using the plot function in matlab.

3 Examples

3.1 Students' Performance in a Math Course

We have used the javaplex codes to analyze the students performance in an introductory math course Math110 in Penn State. The data set contains 1000 data points in $\mathbb{R}^{30}$. Each dimension corresponds to performance of a particular assignment. First we consider only the good students (i.e. students who finished all the assignments) in this data set. The obtained barcode is as follow.

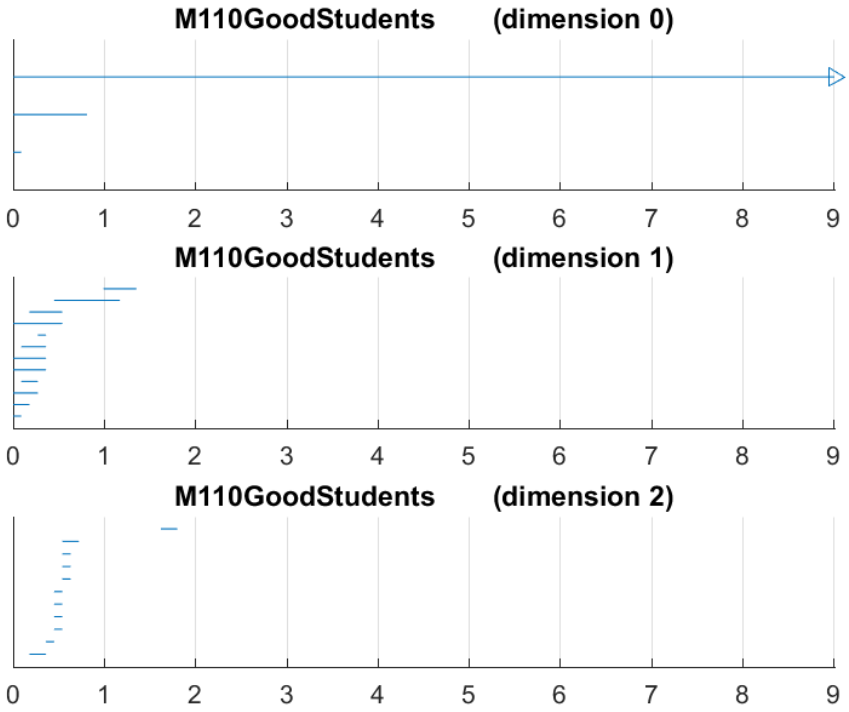

Figure 9: Barcodes for good students

The barcode in dimension 0 shows that there are two connected components until around 0.8. Then we did a cluster analysis on this data set and the centroid of the two clusters that we observed are very close in coordinates except for the last one. When looking back to the original data set, we noticed that a lot of low scores appear in the last coordinate which is unusual. One possible interpretation is that many good students were already guaranteed a grade A even without doing the last assignment, so they didn't care about the last assignments. Moreover, we thought we might detect a long barcode in dimension

1 which means that there is a circle in the data set. A potential explanation is that there exists two groups of assignments and students randomly prioritize one of the two groups, but then, only if the have mastered the priority group, do they work on the other one. But the evidence for a barcode in dimension 1 is not strong.

The next thing we did is normalizing the assignments scores and then running the analysis again. By normalizing,we mean that we set the full score of each assignment to 1, and compute each student's score percentage. Since every assignment has a different full score, we may be able to capture more interesting structure by doing so.

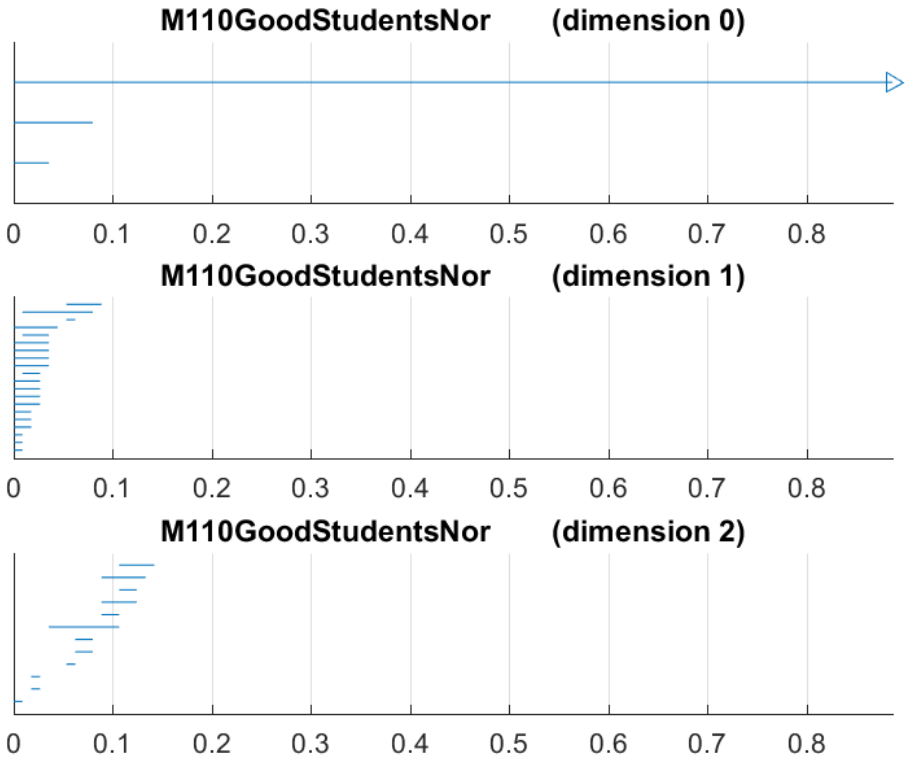

Figure 10: Barcodes for good students after normalization

However, it turned out that normalization made no difference. The barcodes before and after normalization is all most the same except for some random noise. The results are shown below.

The last thing we did is filling the no shows with the average score of the all students' scores on that assignments. However, the barcodes show that all the interesting structures are smoothed out by replacing the no shows by averages. The result is shown below.

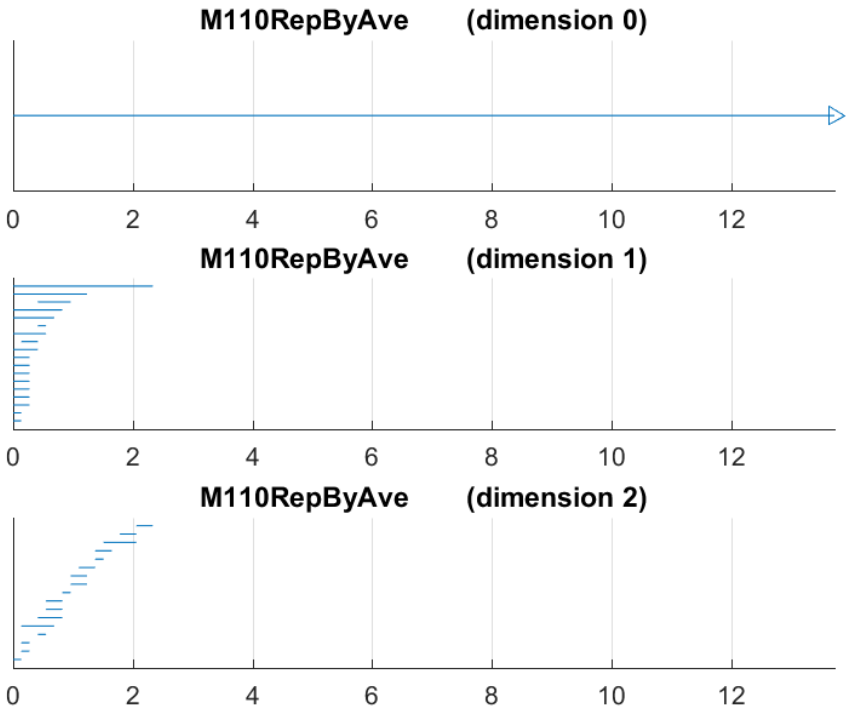

Figure 11: Barcodes after replacing no shows by averages

3.2 Cyclo-octane Molecule Conformations

First we need some terms to understand this example. The conformation of a molecule is specification of the relative positions of all atoms in $\mathbb{R}^3$. Typical parameterizations include coordinates of atom centers or torsional angles. The second parametrization is usually used for proteins, but in our example, we will use the coordinates of atom centers. The conformation space is the space of all conformations.

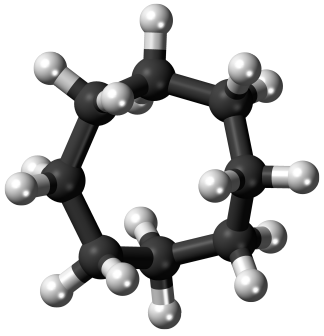

Figure 12: Example of a conformation of cyclo-octane from [8]

The cyclo-octane molecule C_8H_{16} consists of a ring of 8 carbons atoms, each bonded to a pair of hydrogen atoms. The locations of the carbon atoms in a conformation determine the locations of the hydrogen atoms via energy minimization. Each conformation is represented by a point in $\mathbb{R}^{24}$. The conformation space of cyclo-octane is the union of a sphere with a Klein bottle, glued together along two circles of singularities. In a paper published in 2012, Zomorodian used persistent homology efficiently recover the homology groups of the conformation space of cyclo-octane molecules. Another similar example is shown in [8]. In this example, they begin with a sample of 6,040 experimental points on the conformation space (this data is publicly available at Shawn Martin's webpage http://www.sandia.gov/ smartin/software.html). The resulting barcode is as follow.

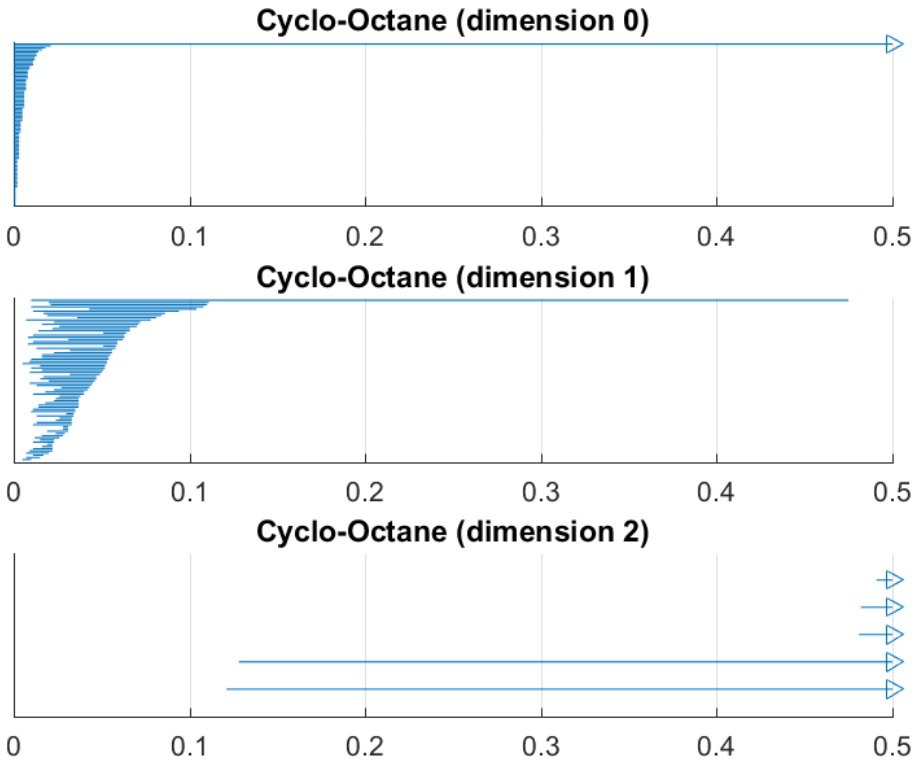

Figure 13: Barcodes for the conformation space of cyclo-octane

The homology groups as implied by the dominating barcodes is the same as the homology groups of the union of a sphere with a Klein bottle, glued together along two circles of singularities.

4 Conclusion

Although we have not found anything interesting for students performance in the introductory math course, we have learned, through reading extensive materials and experimenting with datasets that, topological data analysis is really a powerful tool. We saw its powerfulness in detecting structures and discovering insights from really complex data. Also, the ability to combining theoretical math knowledge in various fields such as algebra and topology into a program which can extract information from a dataset is an exciting experience.

Appendix

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#define BASEPRIME 7
#define MAXDEG 99
#define MACRO 5


//BASEPRIME - order of coefficient field;
//MAXDEG - highest degree we can handle

int invert(int n)
// Computes the inverse of n in the field
{
int t, t1, r, r1, q, x;
t = 0; t1 = 1; r = BASEPRIME; r1 = n % BASEPRIME;
while (r1 != 0)
{
q = r / r1;
x = t; t = t1; t1 = x - q*t1;
x = r; r = r1; r1 = x - q*r1;
}
if (r>1)
{
printf("\n Modular division by zero");
exit(2);
}
if (t<0) { t += BASEPRIME; }
return(t);
}


struct poly  // Structure to store polynomials
{
```

```
int degree;
int coeff[MAXDEG];
};

struct matrix
{
int col;
int row;
struct poly entry[MACRO][MACRO];
};


void initialize(struct poly *p)
{
int i;
p->degree = -1;
for (i = 0; i < MAXDEG; i++)
{
p->coeff[i] = 0;

}
}


void readpoly(struct poly *result)
//Read a polynomial from the keyboard.
{
int i, m, d;
initialize(result);
for (i = 0; i < MAXDEG; ++i)
{
result->coeff[i] = 0;
}
printf("\n Enter degree of polynomial: ");
scanf_s("%i", &d);
if (d>MAXDEG)
{
printf("\n Degree too large");
exit(5);
}
result->degree = d;
for (i = 0; i <= d; ++i)
{
printf("\n Enter coefficient of X to the power %i:", i);
scanf_s("%i", &m);
```

```
result->coeff[i] = m;
}
}

void writepoly(struct poly p1)
// Write a polynomial to the screen.
{
int i;
if (p1.degree == -1)
{
printf("0");
}
else
{
for (i = 0; i <= p1.degree; ++i)
{
if (p1.coeff[i] != 0)
{
if (i == 0)
{
printf("%i", p1.coeff[i]);
}
else
{
printf("%ix^%i", p1.coeff[i], i);
}
if (i<p1.degree) { printf("+"); };

}
}
}

}

void writepolyfile(struct poly p1, FILE *out)
// Write a polynomial to file.
{
int i;
if (p1.degree == -1)
{
fprintf(out, "0");
}
else
{
for (i = 0; i <= p1.degree; ++i)
```

```
{
if (p1.coeff[i] != 0)
{
if (i == 0)
{
fprintf(out, "%i", p1.coeff[i]);
}
else
{
fprintf(out, "%ix^%i", p1.coeff[i], i);
}
if (i < p1.degree) { fprintf(out, "+"); };

}
}
}

}


void neg(struct poly *result, struct poly p1)
{
int i;
initialize(result);
result->degree = p1.degree;
for (i = 0; i < MAXDEG; i++)
{
result->coeff[i] = BASEPRIME - p1.coeff[i];
}
}


void add(struct poly *result, struct poly p1, struct poly p2)
//Add two polynomials.  We pass a pointer to where the result is stored.
{
int i, d;
initialize(result);
if (p1.degree > p2.degree) d = p1.degree;
else d = p2.degree;
result->degree = -1;
for (i = 0; i <= d; ++i)
{
result->coeff[i] = (p1.coeff[i] + p2.coeff[i]) % BASEPRIME;
if (result->coeff[i] != 0)
{
```

```
result->degree = i;
}
}
}


void mult(struct poly *result, struct poly p1, struct poly p2)
// Multiply two polynomials.
{
int i, j, d;
initialize(result);
d = p1.degree + p2.degree;
if (d>MAXDEG)
{
printf("\n Degree overflow error\n");
exit(1);
}
for (i = 0; i <= d; ++i)
{
result->coeff[i] = 0;
for (j = 0; j <= i; ++j)
{
result->coeff[i] = (result->coeff[i] + p1.coeff[j] * p2.coeff[i - j])
% BASEPRIME;
}

if (result->coeff[i] != 0)
{
result->degree = i;
}
}
}

void divide(struct poly *quot, struct poly *remd, struct poly a, struct poly b)
// Divide two polynomials (long division).
{
int i, j, d, q, v;
for (i = 0; i < MAXDEG; ++i)
{
quot->coeff[i] = 0;
remd->coeff[i] = a.coeff[i];
}
remd->degree = a.degree;
if (b.degree < 0)
{
```

```
printf("\n Polynomial division by zero");
exit(3);
}
q = invert(b.coeff[b.degree]);
quot->degree = (remd->degree - b.degree);
while (remd->degree >= b.degree) //while division is possible
{
v = q*(remd->coeff[remd->degree]) % BASEPRIME;
quot->coeff[remd->degree - b.degree] = v;
for (i = 0; i <= b.degree; ++i)
{
remd->coeff[i + (remd->degree - b.degree)] =
(remd->coeff[i + (remd->degree - b.degree)] - (b.coeff[i])*v) % BASEPRIME;
if (remd->coeff[i + (remd->degree - b.degree)] < 0) {
remd->coeff[i + (remd->degree - b.degree)] += BASEPRIME;
}
}
// Find the degree of the remainder
d = -1;
for (j = 0; j <= remd->degree; ++j)
{
if ((remd->coeff[j])>0)
{
d = j;
}
}
remd->degree = d;

// Find the degree of the quotient
d = -1;
for (j = 0; j <= quot->degree; ++j)
{
if ((quot->coeff[j])>0)
{
d = j;
}
}
quot->degree = d;
}
}

void tracematrix(matrix M)
{
int i, j;
printf("\n");
```

```
for (i = 0; i < M.row; i++)
{
for (j = 0; j < M.col; j++)
{
writepoly(M.entry[i][j]);
printf(";");
}
printf("\n");
}
printf(".\n");
}


int main(void)
{



//initialization
struct matrix M;
struct poly pivot, p, rem, quo, nquo, m;
int i, j, k, t, a, pivotrow, pivotcol;
bool allzero, alldivides;
FILE *in, *out;

M.row = 0;
M.col = 0;

for (i = 0; i < MACRO; i++)
{
for (j = 0; j < MACRO; j++)
{
M.entry[i][j].degree = -1;
for (k = 0; k < MAXDEG; k++)
{
M.entry[i][j].coeff[k] = 0;
}
}
}

pivot.degree = MAXDEG;
p.degree = -1;
rem.degree = -1;
quo.degree = -1;
nquo.degree = -1;
```

```
m.degree = -1;

for (i = 0; i < MAXDEG; ++i)
{
pivot.coeff[i] = 0;
p.coeff[i] = 0;
rem.coeff[i] = 0;
quo.coeff[i] = 0;
nquo.coeff[i] = 0;
m.coeff[i] = 0;
}

i, j, k, t, a, pivotrow, pivotcol = 0;

allzero = false;
alldivides = false;

fopen_s(&in, "matrix1.txt", "r");
fopen_s(&out, "SNF1.txt", "w");

//input
fscanf_s(in, "%i", &M.row);
fscanf_s(in, "%i", &M.col);

for ( i = 0; i < M.row; i++)
{
for ( j = 0; j < M.col; j++)
{
fscanf_s(in, "%i", &M.entry[i][j].degree);
for ( k = 0; k <= M.entry[i][j].degree; k++)
{
fscanf_s(in, "%i", &M.entry[i][j].coeff[k]);
a = (M.entry[i][j].coeff[k]+100*BASEPRIME) % BASEPRIME;
M.entry[i][j].coeff[k] = a;
}
}
}



//start computing

if (M.row > M.col)
{
a = M.col;
}
```

```
else
{
a = M.row;
}


for (t = 0; t < a; t++)
{

do
{
//choosing the pivot
pivot.degree = MAXDEG;
for (i = t; i < M.row; i++)
{
for (j = t; j < M.col; j++)
{
if (M.entry[i][j].degree < pivot.degree && M.entry[i][j].degree!=-1)
{
pivot = M.entry[i][j];
pivotrow = i;
pivotcol = j;
}
}
}

//swaping rows and columns so that pivot is at (t,t)-th position
for (i = t; i < M.row; i++)
{
p = M.entry[i][t];
M.entry[i][t] = M.entry[i][pivotcol];
M.entry[i][pivotcol] =  p;
}

for (j = t; j < M.col; j++)
{
p = M.entry[t][j];
M.entry[t][j] = M.entry[pivotrow][j];
M.entry[pivotrow][j] = p;
}


allzero = false;
//eliminating entries
while (!allzero)
```

```
{
//eliminate entries to the right of the pivot
for (j = t+1; j < M.col; j++)
{
do
{
divide(&quo, &rem, M.entry[t][j], M.entry[t][t]);

for (i = t; i < M.row; i++)
{
neg(&nquo, quo);
mult(&m, nquo, M.entry[i][t]);
add(&p, M.entry[i][j], m);
M.entry[i][j] = p;
}

if (rem.degree != -1)
{
for (i = t; i < M.row; i++)
{
p = M.entry[i][t];
M.entry[i][t] = M.entry[i][j];
M.entry[i][j] = p;
}
}
} while (rem.degree != -1);
}


//eliminate entries under the pivot
for (i = t+1; i < M.row; i++)
{
do
{
divide(&quo, &rem, M.entry[i][t], M.entry[t][t]);

for (j = t; j < M.col; j++)
{
neg(&nquo, quo);
mult(&m, nquo, M.entry[t][j]);
add(&p, M.entry[i][j], m);
M.entry[i][j] = p;
}

if (rem.degree != -1)
```

```
{
for (j = t; j < M.col; j++)
{
p = M.entry[t][j];
M.entry[t][j] = M.entry[i][j];
M.entry[i][j] = p;
}
}
} while (rem.degree != -1);
}

//check if all entries to the right of the pivot are still zero
allzero = true;
for (j = t+1; j < M.col; j++)
{
if (M.entry[t][j].degree != -1)
{
allzero = false;
}
}
}

alldivides = true;

if (t != a - 1)
{
for (i = t + 1; i < M.row; i++)
{
for (j = t + 1; j < M.col; j++)
{
divide(&quo, &rem, M.entry[i][j], M.entry[t][t]);
if (rem.degree != -1)
{
for (k = t; k < M.col; k++)
{
p = M.entry[t][k];
M.entry[t][k] = M.entry[i][k];
M.entry[i][k] = p;
}
alldivides = false;
break;
}
}

if (!alldivides)
```

```
{
break;
}
}
}
} while (!alldivides);
}


//output
fprintf(out, "The diagonal entries are:\n");
for (i = 0; i < a; i++)
{
writepolyfile(M.entry[i][i], out);
fprintf(out, "\n");
}


fclose(in);
fclose(out);

exit(0);
}
```

Bibliography

[1] Carlsson, G., Ishkhanov, T., de Silva V., and Zomorodian A. (2006). *On the local behavior of spaces of natural images.*

[2] Ghrist, R. (n.d.) *Barcodes: The Persistent Topology of Data.*
Retrieved from `https://www.math.upenn.edu/~ghrist/preprints/barcodes.pdf`

[3] Hatcher, A. (2002). *Algebraic topology.* Cambridge: Cambridge University Press.

[4] Lesnick, M. (2013). *Studying the Shape of Data Using Topology.*
Retrieved from `https://www.ias.edu/ideas/2013/lesnick-topological-data-analysis`

[5] Munkres, J. R. (1984) *Elements of Algebraic Topology.*

[6] Simplicial complex. (n.d.). *Wikipedia.*
`https://en.wikipedia.org/wiki/Simplicial\_complex`

[7] Shastri, P. (May 2014). *Lectures on Modules over Principal Ideal Domains.*
Retrieved from `http://www.imsc.res.in/~knr/14mayafs/Notes/ps.pdf`

[8] Tausz, A., & Vejdemo-Johansson, M., & Adams, H. (2014). JavaPlex: A research software package for persistent (co)homology. Hong, H., & Yap, C. *Proceedings of ICMS 2014* (pp. 129-136). Software available at `http://appliedtopology.github.io/javaplex/`

[9] Zomorodian, A., Carlsson, G. (2005). *Computing Persistent Homology.*
Retrieved from `http://dl.acm.org/citation.cfm?id=1044846`

[10] Zhu, X. (n.d.). *Persistent Homology: An Introduction and a New Text Representation for Natural Language Processing.*
Retrieved from `http://pages.cs.wisc.edu/~jerryzhu/pub/homology.pdf`

Hangyu Zhou

Education

2013–2017 **Pennsylvania State University**.
Bachelor of Science in Mathematics, Minor in Physics
Schreyer Honor College

Honor Thesis

Title Topology and Data

Supervisors John Roe

Description This thesis describes how topological techniques are used in data analysis, includes codes which demonstrates these techniques, and contains application to an students' performance in an introductory math course.

Experience

2015,2016 **MASS Program**, *Pennsylvania State University*.
Mathematics Advanced Study Semester

Course Detail:
- Introduction to Applied Algebraic Geometry
- Classic Mechanics and Calculus of Variations
- Lie Groups in Two Three and Four Dimensions
- Hypercomplex Numbers
- An Introduction to Dynamics From a Topological Geometric Viewpoint
- Colloquium Lectures
- Interdisciplinary Seminar

Awards

2015,2016 Graduate with Distinction in MASS Program

2016,2017 William B. Forest Honors Schclarship In Mathematics

Computer skills

Basic Mathematica

Intermediate LaTeX, Office, C++

850 Toftrees Ave – State College, PA 16803
☎ *646-512-0053* • ✉ *robinzzhy@gmail.com*