International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

Signal Processing with Computational Topology

[1]*Kamala Shirin Oghuz, [2]Fazil Tarlan Safarov

[1]Baku Higher Oil School, Information Technology Department, Baku, Azerbaijan, & Azerbaijan Technical University, Computer Technology Department, Baku, Azerbaijan

[2]Baku Higher Oil School, Information Technology Department, Baku, Azerbaijan

Authors E-mail: [1]*kamala.pashayeva@bhos.edu.az, [2]fazil.safarov.std@bhos.edu.az

***Abstract* - The objective of this research is to analyze the mathematical methods of signal processing on abstract geometric spaces of the given sensor network, develop an algorithm on a tangible software development environment, and visualize the usefulness of those algorithms. Specifically, the scope of this research covers the notions of persistence in computational topology and how this notion can be observed in certain mathematical features of the given network. Moreover, we will be analyzing sheaves constructed over a network of sensors in the topological space, from the point of persistence modules and persistent sheaf cohomology, visualizing the persistent sheaf cohomology over barcode diagrams. The research objective is analyzed in detail in the introduction. In the first part of the main body, we have analyzed the mathematical background and previous research works in this domain to derive insight into developing an algorithm. In the second section of the main body, discussed and developed the algorithms, based on the computational topology theoretic approaches. In this paper, we will develop the algorithms to construct simplicial complexes, obtain filtration, construct sheaves, and compute persistent sheaf cohomology. The resulting barcode diagrams are analyzed in terms of certain notions, e.g. robustness, stability, and the Betti numbers. In the end, the primary findings will be succinctly expressed in several points.**

Keywords: Sheaves, Cohomology, Persistent Sheaf Cohomology, Voronoi Complex, Weight-based Filtration, Geometrical Grid.

I. INTRODUCTION

In recent decades, the deficiency in the statistical methods in data analysis and signal processing made mathematicians seek alternatives for miscellaneous purposes. The methods derived by using the notions from computational geometry, specifically computational topology have been a significant leap forward in understanding the data in high dimensions, as it is nearly impossible to differentiate between noise and signal with computational methods.

This research emphasizes the applications of sheaves and their cohomology, which are the notions that are utilized to explore the locality in sensor networks. This is valuable as we do not merely focus on the features of the network on a global metric, making the analysis more intricate.

Although Topological Signal Processing has been a significant area of research over the course of the past few decades, there is a dearth of pragmatic algorithms in certain programming languages that are able to calculate the persistent sheaf cohomology and carry out additional analysis on the results of the calculation.

Because this is the situation, the major objective here is to construct algorithms inside an integrated development environment and then to provide an output for further visualization and comparison, respectively. We will be able to provide an overview of the effectiveness of the methods that were used inside the algorithm if we go through the process of comparing and assessing the outcomes. The process of analyzing the results will be how this objective will be achieved. As mentioned, sheaves will be an integral part of this research, for a number of reasons. First and foremost, in most cases in practicality, we have multidimensional datasets provided by signal sources (e.g. sensors) in a network; therefore, we will be researching sheaves to handle the signal at the local level, providing us the flexibility to work with complex signals.

Additionally, it enables us to express several other features of the computational topology (persistence, invariance, etc.) on the signal processing domain and integrate those branches of mathematics and engineering, further facilitating the topological approaches to the analysis and filtering of the signals. By combining the strength of the sheaves with the abstract algebraic representation (cohomology), the second primary objective of this study is to build the method for persistent cohomology computation of the produced sheaves. This will be accomplished after the sheaves have been formed.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

II. RESEARCH METHODOLOGIES

In this chapter, the research will be conducted on the implementation of sheaves on sensor networks and data sources, and algorithms will be developed to visualize the persistent cohomology of those sheaves from sole mathematical logic to make a comparative analysis with the method of persistent homology. Moreover, the concept of construction of a filter over cell complex will be scrutinized with a comparative analysis with the conventional filters on non-stationary signals.

2.1 Overview of the Algorithms before Sheaf Construction

The first step of the research is to investigate the mathematical structure of the sheaves and to build the algorithm in pseudocode for sheaf construction over the given datasets.

Before diving into the analysis of sheaves, the notion of locality and globality over topological spaces and networks should be taken into account, as sheaves are the structures that enable us to do elaborate analysis on local data, connecting to the other sheaves on the global level. For example, as given in [1] two different topological filtrations of binary images may indicate the same persistence diagram, despite the variances between the relations of connection being different in each four images.

Certain methods deriving from the Nerve Theorem such as Mapper analyzed in (Kraft, 2016) and (Mona, 2023), or computing local persistent homology and cohomology, elaborated on [4] over the local Vietoris-Rips Complexes. In [4] the branch numbers b1 and the first Betti numbers βp were calculated, where the pair (b1, β1) pairs are utilized to provide the heat map of branch numbers and local loop structures. However, we will approach this problem with methods from the sheaf theory, an approach that has not been researched in an elaborated manner, and there is no particular algorithm developed for sheaf construction, let alone sheaf (co) homology.

To construct sheaves, we have to specify what kind of signal or data source will be provided, as the case studies implemented in this research will be utilizing the data provided in Taxi Trips Chicago, a dataset that comprises the pickup and dropoff locations as well as several specific quantities such as fares, trip miles, and trip time. The second dataset is the signals from a network of sensors generating signals of temperature and humidity measurements, together with light, motion, and smoke detection over the time domain. To get the raw dataset to have a topology, it has to be preprocessed into some structure [5].

Based on the content of each dataset, we have to choose over which kind of base space (discrete set of points, manifolds, or graphs) and open sets we will construct the sheaves (metric spaces, nodes of a graph, or regions defined over a function), i.e., we should define the topological space which we will be working with [6].

Based on the content of each dataset, we have to choose over which kind of base space (discrete set of points, manifolds, or graphs) and open sets we will construct the sheaves (metric spaces, nodes of a graph, or regions defined over a function), i.e., we should define the topological space which we will be working with.

To construct the sheaves for the dataset on taxi trips, we have to preprocess the data to get it ready to work on as a topo-logical space. That is why we built an algorithm to get the space represented by the geographical bins, regularly shaped cells that is used in spatial analysis. Creating such kind of a grid is necessary to calculate the mean of the numerical data that was provided from miscellaneous points in the spatial domain, to have an overall view of the selected values in the regions with predetermined radii, bringing simplicity to comprehend the preprocessed data, regularity to the calculations with their relatively consistent structure, and alignment for the points in the dataset.

To generate the geometrical bins, the given geographic region should be divided into smaller regions with the given longitude and latitude data columns provided in the data source. Then, the data to construct sheaves would be created in the form of dictionaries, a tuple to exemplify the geographical boxes on the grid, and the list of values that occurred in that box.

To do a comparative analysis, we can divide the region into the Voronoi cells, with which the regions are divided by per- pendicular bisectors, each of which is equidistant from two of the selected node pair vi and vj for all i/ = j. Consequently, the collection of all the convex polyhedrons which were generated as a result of the intersection of all the half-spaces defined by the perpendicular bisectors between vi's and vj's.

Moreover, we have to choose in which way the simplicial complex will be represented, as graphs constructed with the tools provided in the NetworkX library are used in this research, with the node IDs assigned to each face of the simplex. With such constructions, a subbasis will be generated to build a topology (τ), to capture the concept of space on the elements of the given dataset based on the extraneous data contained in the subbasis. In this research, the determined topologies bolster and comprise all the unspecified sheaves.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

Although the obtained datasets are huge, it is finite; therefore, the pre-meditated topology will be finite.

We need ID for the nodes in the graph, which can be assigned with an algorithm taking the specific latitude and longitude data from the preprocessed grid dataset. The pseudocode for the algorithm is indicated below.

Algorithm 1: Finding closest node

1:	**Generate a latitude index with the specified minimum latitude coordinate and the predetermined bin size**
2:	**Generate a longitude index with the specified minimum latitude coordinate and the predetermined bin size**
3:	**Create a Node ID using the calculated indices (Maybe floor of the sum of the indices)**

We also need to specify the simplicial complex to which the generated sheaf will assign the selected columns from the data to construct the sheaves. For this purpose, we need to create edges between the identified nodes, to have proper faces for the simplicial complex for further calculations in persistent homology and cohomology [7, 8].

To build a simplicial complex and get a proper filtration on the simplicial complex, we can use the algorithm provided in [9] in our research. Generally, *filtration* is a nested sequence of subcomplexes in a simplicial complex K, enabling us to generate the sets for structure for the spatial analysis based on "proximity". Moreover, we need back-ground on persistent homology and filtration to construct the sheaves on the lattice like form of the created topology τ. Theoretically, we have created an undirected graph G_1 for the simplicial complex, consisting of the set V of vertices and E of edges (both are finite), representing the nodes and links between them respectively. To do the analysis with the given fare, trip time, and trip miles values, the edges of the graph must be weighted. Each edge or the perpendicular bisectors of the Voronoi diagram has a nonnegative weight based on the aforementioned values which are intended to be analyzed.

For the sake of regularity, we will assume that all the vertices will be a face of the edges. It can be explained by the poset structure of the topology τ , which is constructed on the operation of set inclusion as a relation for the partially ordered set, i.e., $V, E \in \tau \Rightarrow V \leq E \Longleftrightarrow V \subseteq E$. It is also clear that the two subsets V and E have an intersection and union since the topology has a lattice structure [10].

Since the generated graph is finite, we may assert that a subset V' is path-connected if for two vertices $v_a^{'}$ and $v_b^{'}$in the subset $V^{'}$, there is a sequence $v_{n1}^{'N}$ which can be represented as $v_1^{'} \sim v_a^{'}$ and $v_N^{'} \sim v_b^{'}$such that $(v_a^{'}, v_b^{'})$ is in the subset $E^{'}$of edges, meaning that there is a path connecting the given vertices $v_a^{'}, v_b^{'}$ along the edges in $E^{'}$. Employing this proposition, we can express the subset V' in the form of connected components.

Once the weights are determined, we can use the notion of level on the graphs and subgraphs, which are also called the *persistence parameter* in [9]. For each computed weight of the intended value (fare, trip time, and trip miles), we divide the subgraphs based on the weight differences, i.e., for each level l the subgraph is generated by the edges which has weights greater than l.(Ibid)

$$G(l) \sim \big(V(l), E(l)\big) \qquad (2.1.1)$$

$$E(l) = e \in E : W_e > l \qquad (2.1.2)$$

$$V(l) = \bigcup_{e \in E(l)} v : V \leq e \qquad (2.1.3)$$

By defining the level notion, we may express topological persistence through the weight values by defining the change of the graph with the change of level [9]. As indicated above, we may assume that

$$G(l_1) \subseteq G(l_2) \Leftarrow\Rightarrow l_1 \geq l_2 \qquad (2.1.4)$$

Which means the graph may comprise more edges and vertices as the level decreases by the definition of level.

In this research, we will define the aforementioned geometrical "change" in mathematical terms. In Layman's terms, we may take l as the time domain in the reverse direction. For $l_a \geq l_b$ $(a \geq b)$ the graphs $G(l_a)$ and $G(l_b)$ at that specified levels will be compared, as miscellaneous variations may occur; for example, an edge may be "born", or several edges may "die" by the birth of a loop. Moreover, a 2-simplex may be generated by the growth of a connected component.

Such "evolution" of the connected components can be recorded by barcodes by inscribing a separate line for each connected component, marking birth and death of them at each the proper level, as a bar starts with the birth of the component, noted with the birth level in the domain of graph, ends with the death, which is indicated with the merge of the bar into another component's bar. The arrangement of the bars will indicate the state of connected components with the weighted values of fare, trip time, and trip miles, to make an analysis.

Every one of the bars can be regarded as a key-value dictionary that changes in response to the parameter l. The bar is made up of three values: the start value, which indicates the barcode's birth; the end value, which indicates the bar's merger and death (if any), and the state, which is made up of the linked component that is now in place. The information

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

required to compare the present (i.e., current value of l) connected set to a connected set later on is stored in the state of the bar. The list of bars is the barcode [11].

Keeping the mortality of bars in mind, we must decide which bar to extend in the event that two linked components combine. When bars merge, we state that the oldest bar–i.e., the one with the highest start value (recall that we are reversing time, so "older" means greater starting values)–survives, while the younger ones (the ones with lower starting value) die [12].

We define the persistence of each bar as the absolute difference between its start and end l-values. We represent these bars in a graphic with the length of each bar being proportional to their persistence, arranging the bars by overall length with the longest bars on the bottom. This plot name is the barcode diagram, or simply the barcode.

If a component occurs for a substantial range of levels l i.e. its bar has a high persistence, we may view it as robust; if a component exists only for a narrow range of parametric values, one can understand it as a consequence of noise.

For the open sets E and V, we may take the ideals λ_E and Λ_V into account generated from E and V. Since the topological space τ is finite, the ideals Λ's are finite, and there is a filtration on those ideals [10]

$$[V] = \Lambda_E^0 \subseteq \Lambda_E^1 \subseteq \Lambda_E^2 \subseteq \ldots \subseteq \Lambda_E^k = \Lambda_E \qquad (2.1.5)$$

Where Λ_E^i comprises all the $V \in \Lambda_E$ such that there is a maximal chain in Λ_E,$V = V_l \subseteq V_{l-1} \subseteq V_l \subseteq \ldots V_1 \subseteq U$, where l is the levels of the simplicial complex.

To construct sheaves, we can work with classes for several structures in the analysis, based on the parameters of the dataset [13]. First and foremost, a base object for the sheaf will be generated, for our algorithm to be flexible while working with miscellaneous types of complexes, as we will be creating filtrations on a graph, a geometrical grid, and the Voronoi complex.

We may define a base class in Python programming language with an initial function taking a geometry parameter, to create an instance of the base complex class, as we will need flexibility when working a geographical grid or a Voronoi complex. An additional geometry definition is required inside the initial function to create an attribute in the objects that will be generated by calling the base complex class, being assigned to a value in the input parameter. By writing such an algorithm to create the object, we will be able to store the spatial data about the simplicial complex within the particularly generated object. In the subsequent steps, we will use the geometry object to calculate the neighbor nodes or the filtration functions.

In the next step, we will develop the algorithm for juxtaposition in the grid structure and Voronoi complex, categorizing the neighbor types and generating the geometrical structure based on the type of the neighbor. In the case of rectangular cells, it is easy to locate the neighbor cells based on their row and column index in the given rectangular grid. The function will return a series object by taking an implicit geometrical object of neighbor cells. On the note of the Voronoi complex, two cells are adjacent if two vertices have the same distance from a perpendicular bisector and share a common edge. The pseudocode for the algorithm will be like the following:

Algorithm 2: Base Complex Pseudocode

```
class BASE COMPLEX
function INITIAL(geometry attribute)
Assign the geometry attribute to the Base Complex object
end function
function GET ADJACENT OBJECTS FOR GRID(face, type)
Declare a list object for the adjacent cells
Get the index of the face (rows and columns as the dimensions of the object with a ready function
if type = edge then
Add adjacent cells considering the dimensions of the face
else if type = vertex then
Add a cell diagonal to the current cell, considering the dimensions
end if StateReturn the adjacent cells list
end function
function GET ADJACENT OBJECTS FOR THE VORONOI COMPLEX(face)
Declare a list object for the adjacent cells
for vertex in vertices do
if the index of the vertex is valid then
Append the vertex to the complex an adjust its index
end if
end for
end function
end class
```

Based on the theoretical analysis, we may construct the algorithm to decompose the simplicial complex into connected components and compute topological persistence.

However, we have to compute weights for the levels of the filtration, which is the primary obstruction in deriving the algorithm to decompose the simplicial complex into connected components and obtain a filtration, as we distinguish the levels of the filtration corresponding to the weight values. As we will be constructing sheaves after getting the filtration, we should determine the weight values assigned in the sheaf structure. Once we have the topology τ, we need the second and third components: the raw sheaf data set where the data from the

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

given dataset is saved as a selection of segments, and a model presheaf which determines the conceptual framework we want to analyze.

As we know from the definition of sheaves, we can realize a sheaf as a function from the index of our dataset to another set in a category, which takes the node ID as an input in our dataset, and returns the measured value, i.e., fare distribution, trip time, or trip miles. The identical structure will hold for the subsets of the ID index set in our category. Consequently, there will be a map or a function for any open subset I' of the index set, which takes the data sheaf D with the set of node ID's to a section of $D(I')$, $f_{D,I'}: I' \to U$. Generally, we can define this map as

$$D \to \{f_{D,I'}: I' \to U | I' \in \tau\} \qquad (2.1.6)$$

Helping us to distinguish the global section of the data sheaf $D(I)$.

Thus, we may define a (pre) sheaf on the space with topology τ. We may also define the consistency of the an assignment $A = (a_{I'})_{I' \in \tau}$ if $a_V = res_{I',V(a_{I'})}$. Conversely, we may assume that the assignment is in consistent.

While the raw data sheaf D assigns the specific columns of the dataset collected on each node, the constructed presheaf will handle the local assignments of the raw data sheaf. Hence, we define the model presheaf on I over the index set as a set of maps from the set from the data sheaf to the space of the model presheaf, such as a set of averaging maps, assigning the vector spaces, or node ID's to a real number.

Algorithm 3: Obtaining connected components

1:	**Generate an empty set to represent unvisited vertices**
2:	**Generate an array for the arrays representing the connected components at each filtration level**
3:	**for each vertex in the unvisited set do**
4:	**Select a new connected component in which the vertex is present**
5:	**function SEARCH(vertex)**
6:	**for Each vertex in the univisited set and the connected components above threshold filtration level do**
7:	**Exclude the vertex from the unvisited set**
8:	**Include the vertex to the connected component**
9:	**Search the vertex**
10:	**end for**
11:	**end function**
12:	**Search(v)**
13:	**Add the final arrays**
14:	**end for**

From here, we may start sheaf construction with the derived simplicial complexes and their filtration. As we derived the algorithm to extract the connected components for the adjacency operations to obtain the geometric grid and the Voronoi complex, we may develop the algorithms for the filtration of the structures as mentioned earlier.

The first thing to do is to create the function for constructing the grid structure and the Voronoi complex in the separate classes generated for them. Initially, let's analyze the logic of the grid construction algorithm. It is a rather simple iterative algorithm to append the given polygon object of a certain type over the predefined row and column quantities computed by the given specific boundary and cell-size values. The function will return a series of objects.

Thereafter, we will build the Voronoi Complex in its corresponding class. In the pseudocode, we will define an initial function similar to the grid construction algorithm, which takes a geometry object and Voronoi regions as input, to be a feature to be used when called in a main program.

Algorithm 4: Creating Grid Structure

1:	**function GRID(thresholds, width, height)**
2:	**Thresholds are the list-like objects that define the scale of the grid**
3:	**Minx, Miny, Maxx, Maxy = Thresholds**
4:	**Generate an evenly spaced array of X coordinates (columns)**
5:	**Generate an evenly spaced array of Y coordinates (rows)**
6:	**Generate a polygon object (array or list)**
7:	**for Every column in columns do**
8:	**for Every row in rows do**
9:	**Create a multidimensional object of ordered pairs using the coordinate indexes, width, and length**
10:	**Append the given object**
11:	**end for**
12:	**end for**
13:	**end function**

Then, we develop the logic to construct the Voronoi complex, which may take node ID's or the pickup points from data as an input. Inside the function, we can define the regions divided by the bisectors as a set or any iterable object. Iterating through the elements of the region, the vertices of the given region may be extracted by specific functions, and taking the possibility of infinite regions into account and excluding them through a filter command should be included in the algorithm.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

Algorithm 5: Getting connected components

```
class VORONOI COMPLEX
function INITIAL(geometry, Voronoi cells)
Adjust the geometry input and cells to the object of the class
end function
function CONSTRUCT VORONOI COMPLEX(reference points)
Define the class object
Define an empty list for extracting polygons
for Cell in the Voronoi Cells do
Extract vertices of each Voronoi cell
if Cell is not infinite then
Append the Polygon generated by the vertices
end if
end for
Adjust the regions of the Voronoi as an object of the class
end function
end class
```

After constructing the Voronoi Complex, we should define the algorithm to get the filtration on the Voronoi complex. There are several strategies to implement to obtain the structure of proper filtration, but we will employ filtration based on edge weight like the aforementioned filtration on the graph. Although the logic to assign the weights will be fundamentally analogous to the previous method, as we will assign the weight values to the edges between the nodes representing the specific coordinates on the given region, we have to change our approach to determine the shared edges as the geometric object we are dealing with is different from the graph; therefore, we have iterate for every geometric intersection to determine the shared edges before assigning the weight values.

After completing the edge weight calculation, we are ready to develop the logic for deriving filtration. For the function to determine the shared edges, we should declare a dummy iterable object (a list or a set), and append the edges into the object with a conditional algorithm.

The calculation of average weight values is a simple algorithm; however, we should assign the calculated values to the shared edges. There we use the particular object generated for all edge weight values.

Thereafter, we will develop the filtration with a similar logic to the filtration of graphs with levels. The complex should be decomposed into subcomplexes by a comparison with a threshold value before being appended into the subcomplex that will comprise it based on the condition.

Algorithm 6: Getting connected components

```
class VORONOI
function CALCULATING EDGE WEIGHTS
for i, j in the shared edges do
Calculate average fare, trip time, trip miles
Generate a common weight tuple with three components
Assign Weight for each edge (i,j)
end for
end function
end class
```

After getting the filtration, we are ready to develop the algorithm for sheaf construction and further analysis logic with barcodes, as well as computing persistent cohomology. The various kinds of filtration we created with the sheaves attached to the corresponding complex storing data will enable us to observe the patterns on a local scale. For example, let's suppose that we will be analyzing the sheaves constructed over the Voronoi complex with a filtration we constructed. Each Voronoi complex consists of cells centered on a vertex, facilitating the identification of the area of interest for analysis. The stored data corresponding to each node will reveal the specific changes in geometric alterations upon visualization with barcodes.

As the program iterates within the sub-complexes in the weight-based filtration, the sheaf object in the algorithm will help us to determine how the external data (fare, trip time, and trip miles) how data is encompassed within the simplices in the neighborhood generated and merged through the filtration.

The critical point here is the restriction maps which will assign the data to the specific nodes when the dimension of the simplices in a neighborhood changes from smaller to larger ones, i.e. we can express it as conducting the local evolution. For example, as an open neighborhood of a node expands through filtration, the trip miles or trip time data incorporated from the newly included adjacent cells will affect the calculated average data stored in the sheaf of the corresponding node. To visualize the distribution of the fare values, we may use histograms, to compare the fare distribution of specific regions with its neighbors as the threshold value of the weight increases in the filtration.

We have to allocate a function within the base object of the sheaf for the restriction function, to grasp the evolution of the data from a simplex to a subcomplex through the filtration. The logic will be merely the averaging of the numerical data stored in the sheaf of the given simplex and in a bigger face that comprises the given simplex. Let's clarify with an example. In the filtration of the given Voronoi Complex, there is a mean fair value stored in the corresponding sheaf. As we

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

iterate within the filtration to the larger geometric structures, the simplices get merged into those larger structures, and the restriction function helps to express the impact of the values in the smaller structure on the values stored in the dictionaries of the according sheaf numerically. For distribution values, we might employ the comparison of the histograms.

2.2 Computing Sheaf Persistent Cohomology

The theoretical groundwork of the persistence of sheaves was developed and scrutinized from various points of view; however, there is no tangible algorithm to compute the persistent homology and cohomology of sheaves, so we will at- tempt to develop a logic for the algorithm, taking insight from [14] and visualize the persistence over barcode diagrams.

In the current step, we have constructed our sheaves and we have the proper filtration over several geometical objects; therefore, the primary concern here will be the suitability of the given topological space τ to be mapped onto a persistence module over a category (Ibid.).

First and foremost, we should attach the data stored in the sheaf to the generated base object, which is a simple classification function that consists of a few conditionals.

From [14] we choose type T mathematical computation algorithm of the persistent cohomology as we want to visualize the persistence with barcodes. Initially, we should create sheaves with the aforementioned attachment function. For the sake of simplicity and efficient calculation, we will compute only the zero-dimensional cohomology, using the calculations carried out over F_2, namely 0 and 1. Another reason to choose the calculations done in a particular way is to ignore the torsion elements [15] as they are more abstract algebraic elements rather than topological ones. By calculation over the field with two elements, we will be able to indicate the persistence of the connected components (e.g. vertices, edges, and holes). Employing such kind of a calculation allows us to have a brief representation on barcode. Thus, the overall algorithm will be similar to the code for decomposing the simplex into connected components; however, we will be appending the unvisited components into the empty list of components if the cells in the sheaf are not in the visited components. In other words, algorithm searches for the connected components in the dataset that is specified by the sheaf.

Then, we should develop the algorithm to calculate pullbacks, which induce inclusion functions, or morphisms in co-homology, into the next step of the corresponding filtration. Finally, we may compute the cohomology for the obtained pullbacks (see Algorithm 7).

Algorithm 7: Compute zero-dimensional cohomology

```
function 0-COHOMOLOGY (Sheaf data, Voronoi structure)
Generate an empty array for the distinguished connected components
Generate an empty set to distinguish the iterated connected components
for Cell in sheaf data do
if The cell is not the set of visited components then
Find and extract that connected component (See Algorithm 3)
Extract the connected component
end if
end for
end function
```

Finally, we can compute the persistence modules for the type T persistent cohomology. The algorithm may consist of a function taking input as a geometric structure that we constructed (Voronoi complex, graph, or geometrical grid) and its filtration. The function will return the persistence modules in the form of a dictionary, which will append the computed cohomology into the data in the form dictionary, according to each datatype (average fare, average trip time, and average trip miles) (see Algorithm 8).

Algorithm 8: Type T persistent sheaf cohomology computation

```
function CALCULATE TYPE T PERSISTENT SHEAF COHOMOLOGY(Voronoi diagram, filtration) State Generate an empty dictionary with keys corresponding to the parameters (Fare, trip time, trip miles)
for i the range number of the levels of filtration do
Assign variables to store ith and (i+1) th filtration levels
for Data type in the dictionary do
Construct sheaf for each data type
Calculate pullback of the generated sheaf
Compute Cohomology of the sheaf
Dictionary appends the persistent cohomology
end for
end for
The function returns the persistence modules dictionary
end function
```

III. RESULTS AND DISCUSSION

We will discuss the results of each given algorithm, as well as the direct implementation of that logic in Python and C++.

Each barcode diagram corresponding to the various geometric structures will be examined individually before

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 8, Issue 7, pp 89-99, July-2024
https://doi.org/10.47001/IRJIET/2024.807009

performing a comparative study of the findings obtained from applying the same approach for computing persistent cohomology on different geometric structures.

3.1 The Barcode Visualizations of Sheaf Persistence on Each Parameter (Geographical Grid)

In the following diagrams, we can see the barcode diagram indicating persistent cohomology of the sheaf that is constructed over the geographical grid with a distance-based filtration.

A diagram in figure 3.1.1 indicates the persistence diagram of the cohomology of the fare values in the sheaf.

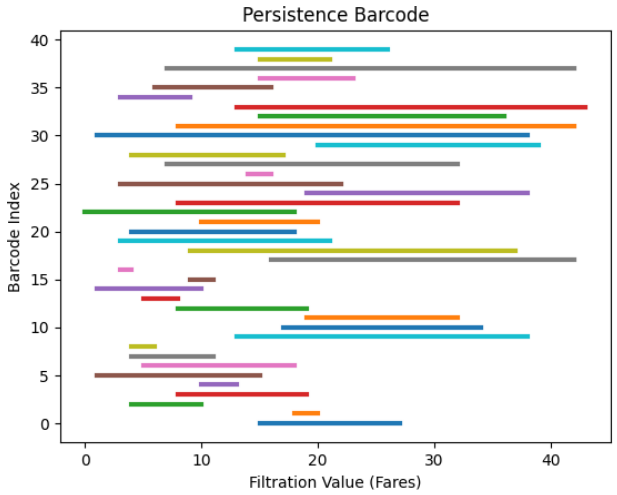

Figure 3.1.1: Persistence of the average fare values over the region

Here, we can see the Betti numbers in the diagram for each dimension of the filter, which reveal the number of connected components of the given dimension. From the barcode, diagram we may infer that we have more connected components when the weight filtration is just below 15, meaning that we have more connected components that did not get merged into, or persisted to high-dimensional simplices in the particular sublevels of the filtration.

Furthermore, the length of each bar represents the persistence of the weight value assigned to a geometrical object, meaning that the particular weight value assigned to a simplex or a subcomplex persisted in a specific geographical grid. The longer bars indicate that the data provided in the specific node is less likely to be noise, as it has been robust over several levels of filtration.

One more time, displayed in figure 3.1.2 are the Betti numbers for each dimension of the filter, indicating the number of linked components in each dimension. Based on the barcode diagram, we can deduce that there are a greater number of linked components when the weight filtration is slightly below 5. This indicates that there are more connected components that were not merged in high-dimensional simplices in the specific sublevels of the filtration. It may be inferred that this information aligns with the data depicted in the histogram.

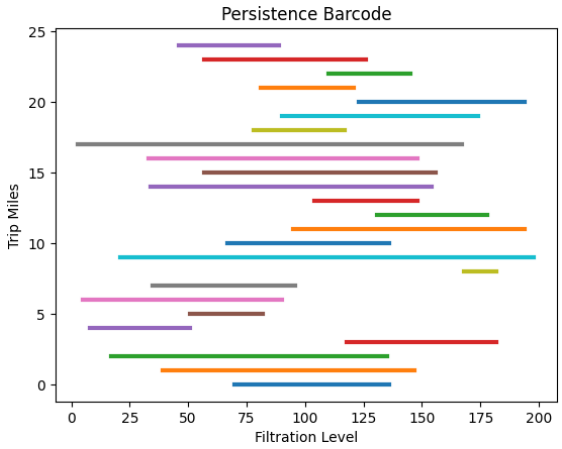

Figure 3.1.2: Persistence of the average trip miles over the region may be deduced that his information corresponds to the data shown in the histogram

Moreover, the size of each bar indicates the duration of the weight value allocated to a geometric object. This means that the specific weight value assigned to a simplex or a subcomplex remained constant inside a certain geographic grid.

Additionally, the increased length of the bars indicates that the data shown in the particular node is more resistant to interference since it has remained consistent even after several stages of filtering.

The diagram 3.1.3 presents the Betti numbers for each dimension of the filtration regarding distance, which represents the number of connected components in each dimension. From the barcode diagram, it can be seen that there is a higher quantity of interconnected components when the weight filtering is just below 5. This suggests that there exist more related components that were not combined in high-dimensional simplices inside the specified sublevels of the filter. It

Furthermore, the magnitude of each bar corresponds to the length of time that the weight value is assigned to a geometric object. This indicates that the allocated weight value for a simplex or subcomplex remained consistent within a specified geographic grid.

Due to the dimensions of the trip seconds data column, we have more connected components for the trip seconds in each dimension, making the barcode diagram contain more bars than the aforementioned diagrams.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 7, pp 89-99, July-2024

https://doi.org/10.47001/IRJIET/2024.807009

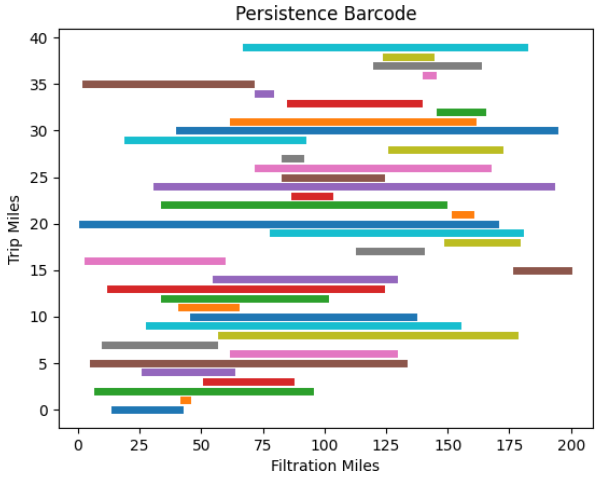

Figure 3.1.3: Persistence of the average trip time over the region

3.2 The Barcode Visualizations of Sheaf Persistence on Each Parameter (Voronoi Complex)

The figure 3.2.1 indicates the persistent sheaf cohomology over the Voronoi sheaf with average weight values. Overall, we may infer that the sheaf constructed over Voronoi resulted in more noise sensitive representation of the signals provided by the structure, as there are longer and more persistent bars representing the cohomology over the barcode diagram of the sheaf cohomology of the geometrical grid.

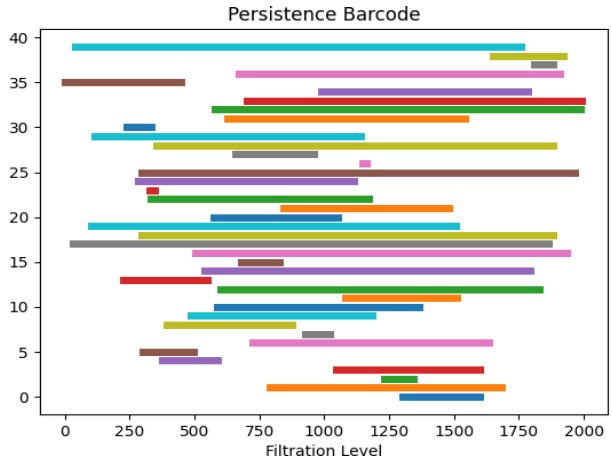

Figure 3.2.1: Persistence of the average fares over the region (Voronoi Complex)

Generally, the results are almost similar on the note of the Betti numbers, compared to the persistence diagram over the geometrical grid, as both of the diagrams display that the highest number of the connected components are below 15 in the filtration.

Certainly, the discrepancies between the graphs can be elaborately analyzed by calculating the bottleneck distance, but it beyond the scope of this research.

The barcode diagram 3.2.2 display the barcode diagram for the persistent sheaf cohomology over the Voronoi Complex in the filtration with weights of trip miles. Although the diagram is not very much analogous to the one of the geometrical grid, we can make several comparisons based on the observations on the diagram.

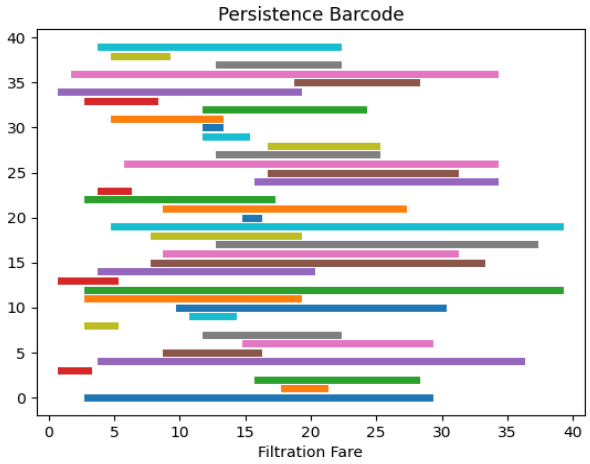

Figure 3.2.2: Persistence of the average trip miles over the region (Voronoi Complex)

The diagram 3.2.2 shows that there are more connected components overall; however, their distribution is almost identical, which are between 100 and 125 in the filtration. The difference between the overall numbers of the connected compo- nents of those geometrical structures is probably based on the size of the cells and the variations of the filtration of those geometrical structures.

Nevertheless, there is no definitive answer in terms of robustness: there are no remarkable difference in the persistence of the geometrical features.

The figure below (figure 3.2.3) demonstrates the barcode diagram for the persistent sheaf cohomology over the Voronoi com- plex in a filtration with weights of trip time.

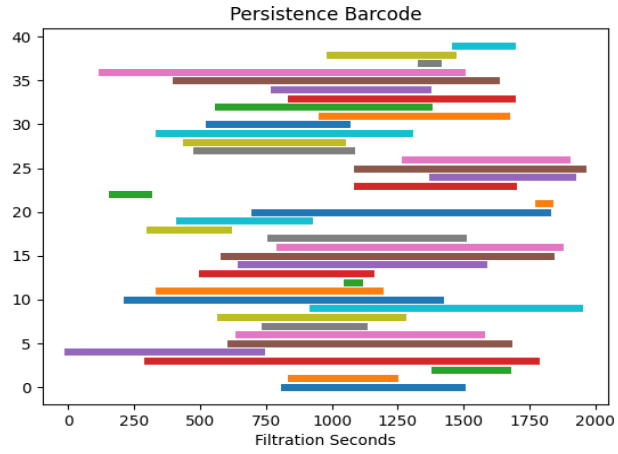

Figure 3.2.3: Persistence of the average trip miles over the region (Voronoi Complex)

ISSN (online): 2581-3048
Volume 8, Issue 7, pp 89-99, July-2024
https://doi.org/10.47001/IRJIET/2024.807009

Because of the size of the trip seconds data column, there are more related components for trip seconds in each dimension. As a result, the barcode diagram has more bars compared to the diagrams stated earlier.

Analyzing Betti numbers, we can deduce that the connected components are more distributed than the aforementioned ones, which corresponding to the histogram representing the trip seconds.

IV. CONCLUSION

In a nutshell, we may analyze the algorithm to compute persistent sheaf cohomology over different geometrical structures by comparing them over several points:

1) The algorithm should specifically address both the topological and the algebraic construction of the sheaves, to compute the type T persistent sheaf cohomology (Please see 1.2).

Topological Construction is similar to the method of how persistent modules of the persistent homology are constructed, as the changes in the sheaf cohomology groups over the level of the filtration in the given topological space will be recorded in the diagrams.

The algebraic part of the algorithm comprises the control of the topological space and alteration of the particular parts of the sheaf in the system of sheaves. As the stalks of the sheaf evolve over the levels, the changes in the algebraic structure of the cohomology groups are inscribed into the barcode diagrams.

2) The differences between the geometrical structures are key for the comparison:

The Voronoi complexes are constructed by partitioning the given topological space by perpendicular bisectors which are equivalently distanced from the nodes, which makes all points cumulated in equal proximity of the particular node.

On the other hand, the geometrical grid enables us to average out the local information in a simpler context, but the Voronoi-based sheaf provides a more robust and complex representation of the signal. Moreover, Voronoi-based construction captures the local information

3) In terms of noise sensitivity, we observed from the example that the sheaves constructed over Voronoi complexes are less robust and more noise sensitive as minor perturbations in the point cloud of the given dataset can drive noticeable changes in the structure of the Voronoi complex, thus in the sheaf structure and its cohomology. However, the geographical grid will be more robust as it has a more fixed structure compared to the Voronoi complex, meaning that small disturbances won't affect the sheaf cohomology.

4) In the context of Betti numbers, namely, the number of connected components, the sheaves constructed over Voronoi complexes will represent more complicated geometrical objects in the structure, as they have more adaptability when compared to the sheaves over the geographical grid. However, high-dimensional simplices are filtered as the grid has an established structure.

REFERENCES

[1] C.-S Hu, & Y.-M. A Chung, Sheaf and Topology Approach to Detecting Local Merging Relations in Digital Images. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 4391–4400. https://doi.org/10.1109/CVPRW53098.2021.00496

[2] R. Kraft. Illustrations of Data Analysis Using the Mapper Algorithm and Persistent Homology. Illustrations of Data Analysis Using the Mapper Algorithm and Persistent Homology. Stockholm.2016.

[3] M. Mona, The Nerve Theorem and its Applications in Topological Data Analysis. The Nerve Theorem and its Applications in Topological Data Analysis. Zurich, Switzerland: Swiss Federal Institute of Technology (ETH) Zurich. 2023.

[4] R. Vandaele, T. De Bie & Y. Saeys, Local Topological Data Analysis to Uncover the Global Structure of Data Approaching Graph-Structured Topologies [Series Title: Lecture Notes in Computer Science]. In M. Berlingerio, Bonchi F., Gartner T., Hurley N., & Ifrim G. (Eds.), Machine Learning and Knowledge Discovery in Databases (2019, pp. 19–36, Vol. 11052). Springer International Publishing. https://doi.org/10.1007/978-3-030-10928-82

[5] S. Arya, J. Curry, & S. Mukherjee. A Sheaf-Theoretic Construction of Shape Space [arXiv:2204.09020 [cs, math, stat]]. Retrieved May 13, 2024, from http://arxiv.org/abs/2204.09020

[6] L. Edelsbrunner, & Zomorodian. Topological Persistence and Simplification. Discrete & Computational Geometry, 2022, 28(4), 511–533. https://doi.org/10.1007/s00454-002-2885-2

[7] M. Robinson. Understanding networks and their behaviors using sheaf theory [arXiv:1308.4621 [math]]. 2013. http://arxiv.org/abs/1308.4621

[8] M. Robinson. Sheaves are the canonical data structure for sensor integration [arXiv:1603.01446 [math]]. 2016. http://arxiv.org/abs/1603.01446

[9] Y. Wu, G. Shindnes, Karve, V. Yager, D. D. B. Work, A. Chakraborty & R. B. Sowers. Congestion Bar-

ISSN (online): 2581-3048
Volume 8, Issue 7, pp 89-99, July-2024
https://doi.org/10.47001/IRJIET/2024.807009

codes: Exploring the Topology of Urban Congestion Using Persistent Homology. 2017. [arXiv: 1707.08557 [physics]]. http://arxiv.org/abs/1707.08557

[10] H. Kvinge, B. Jefferson, C. Joslyn & E. Purvine. Sheaves as a Framework for Understanding and Interpreting Model Fit 2021. [arXiv :2105.10414 [cs, math]]. http://arxiv.org/abs/2105.10414

[11] R. Ghrist. Elementary applied topology: Edition 1.0. Createspace. 2014.

[12] G. Carlsson & A. Zomorodian. The Theory of Multidimensional Persistence. Discrete & Computational Geometry, 2009, 42(1), 71–93. https://doi.org/10.1007/s00454-009-9176-0

[13] F. Russold. Persistent sheaf cohomology 2022. [arXiv:2204.13446 [math]]. http://arxiv.org/abs/2204.13446

[14] J. Curry. Sheaves, Cosheaves and Applications. 2014. [arXiv: 1303.3255 [math]]. http://arxiv.org/abs/1303.3255

[15] G. F. Casas, F. Marchesano, & M. Zatti. Torsion in cohomology and dimensional reduction 2023, [arXiv: 2306.14959 [hep-th]]. http://arxiv.org/abs/2306.14959

Citation of this Article:

Kamala Shirin Oghuz, & Fazil Tarlan Safarov. (2024). Signal Processing with Computational Topology. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 8(7), 89-99, Article DOI https://doi.org/10.47001/IRJIET/2024.807009

****

