UTILIZING TOPOLOGICAL STRUCTURES OF DATA FOR MACHINE LEARNING

by

Sourabh Pradeep Palande

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computing

School of Computing

The University of Utah

December 2020

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Sourabh Pradeep Palande**

has been approved by the following supervisory committee members:

Bei Wang Phillips ,	Chair(s)	**9/25/2020** Date Approved
Preston Thomas Fletcher ,	Member	**9/28/2020** Date Approved
Suresh Venkatasubramanian ,	Member	**9/25/2020** Date Approved
Jeffrey Phillips ,	Member	**9/24/2020** Date Approved
Brandon Zielinski ,	Member	Date Approved

by **Mary Hall** , Chair/Dean of

the Department/College/School of **Computing**

and by **David B. Kieda** , Dean of The Graduate School.

ABSTRACT

The ever-increasing size and complexity of data pose fundamental challenges to existing machine learning techniques, which are typically designed to work with data in vector forms. We believe that topological data analysis (TDA) can provide a different perspective to address these challenges. TDA is a multidisciplinary field that studies the topological structures of data. TDA techniques can be particularly powerful in handling data modeled as trees, graphs, simplicial complexes, hypergraphs, or ensembles of these objects.

The theme of this dissertation is to bring together the fields of TDA and machine learning. Throughout this dissertation, we describe ways to integrate ideas from TDA into different stages of a machine learning pipeline. We first present unsupervised and semisupervised learning algorithms that leverage the topological structure of the data. Then, we present methods to compare complex objects such as graphs and their ensembles. We describe ways to extract topological summaries from these objects and utilize them as input features in machine learning. Our specific contributions include the following:

- We present a spectral sparsification algorithm for simplicial complexes as well as algorithms for unsupervised and semisupervised learning on simplicial complexes, specifically, spectral clustering and label propagation.
- We present ways to utilize topological features of brain networks in statistical inference and machine learning tasks such as classification and regression.
- We present methods to evaluate the structural variability within an ensemble of graphs arising from graph reduction algorithms.

Our vision is to develop new machine learning frameworks that seamlessly integrate ideas from TDA.

CONTENTS

LIST OF TABLES

LIST OF PUBLICATIONS

Journal Articles

- B. OSTING, S. PALANDE, AND B. WANG[1], *Spectral sparsification of simplicial complexes for clustering and label propagation*, Journal of Computational Geometry (JoCG), 11 (2020), pp. 176–211.

- S. PALANDE, V. JOSE, B. ZIELINSKI, J. ANDERSON, P. T. FLETCHER, AND B. WANG, *Revisiting abnormalities in brain network architecture underlying autism using topology-inspired statistical inference*, Brain Connectivity, 9 (2019), pp. 13–21.

Conference Proceedings

- A. RATHORE, S. PALANDE, J. ANDERSON, B. ZIELINSKI, P. T. FLETCHER, AND B. WANG, *Autism classification using topological features and deep learning: a cautionary tale*, In Medical Image Computing and Computer Assisted Intervention (MICCAI), Springer International Publishing, (2019), pp. 736–744.

- K. L. ANDERSON, J. S. ANDERSON, S. PALANDE, AND B. WANG, *Topological data analysis of functional MRI connectivity in time and space domains*, in Connectomics in NeuroImaging, Springer International Publishing, 2018, pp. 67–77.

- E. WONG, S. PALANDE, B. WANG, B. ZIELINSKI, J. ANDERSON, AND P. T. FLETCHER, *Kernel partial least squares regression for relating functional brain network topology to clinical measures of behavior*, In 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), IEEE, (2016), pp. 1303–1306.

Manuscripts

- A. RATHORE, N. CHALAPATHY, S. PALANDE, AND B. WANG, *TopoAct: Exploring the shape of activations in deep learning*, Submitted, arXiv:1912.06332, 2019.

- F. LAN, S. PALANDE, M. YOUNG, AND B. WANG, *Uncertainty visualization for graph reduction*, In preparation, 2020.

---

[1]Authors are listed alphabetically

CHAPTER 1

INTRODUCTION

We live in the age of data. Various scientific and engineering fields are continually generating large volumes and increasingly complex forms of data. The size and complexity of such data pose fundamental challenges to existing machine learning techniques, which are often designed to work with data in vector forms. We believe that topological data analysis (TDA) can provide a different perspective to address these challenges.

TDA is a multidisciplinary field that focuses on capturing the topological structures of the data. TDA techniques are particularly useful for characterizing and summarizing complex forms of data such as trees, graphs, simplicial complexes, hypergraphs, or ensembles of these objects. The goal of this dissertation is to integrate ideas from TDA into different stages of machine learning pipelines.

1.1 Overview

In this dissertation, we present ways to utilize the topological structures of the data for machine learning with complex data types such as simplicial complexes and collections of graphs. Our contributions are summarized in Figure 1.1. First, in Chapter 2, we work with data modeled as simplicial complexes, which are generalizations of graphs. Graphs capture relationships between pairs of objects in the dataset. Many learning algorithms that take advantage of the structure encoded in these pairwise relationships have been proposed in the literature, such as spectral clustering and label propagation. However, in many instances, encoding only the pairwise relationships is not enough. For example, in the case of social networks or email communication networks, the way information is disseminated from person to person differs from how it is disseminated between different groups.

Simplicial complexes allow us to capture higher order relationships among two, three, or more objects in the dataset simultaneously. However, researchers have only recently started exploring learning algorithms for simplicial complexes. In Chapter 2, we present

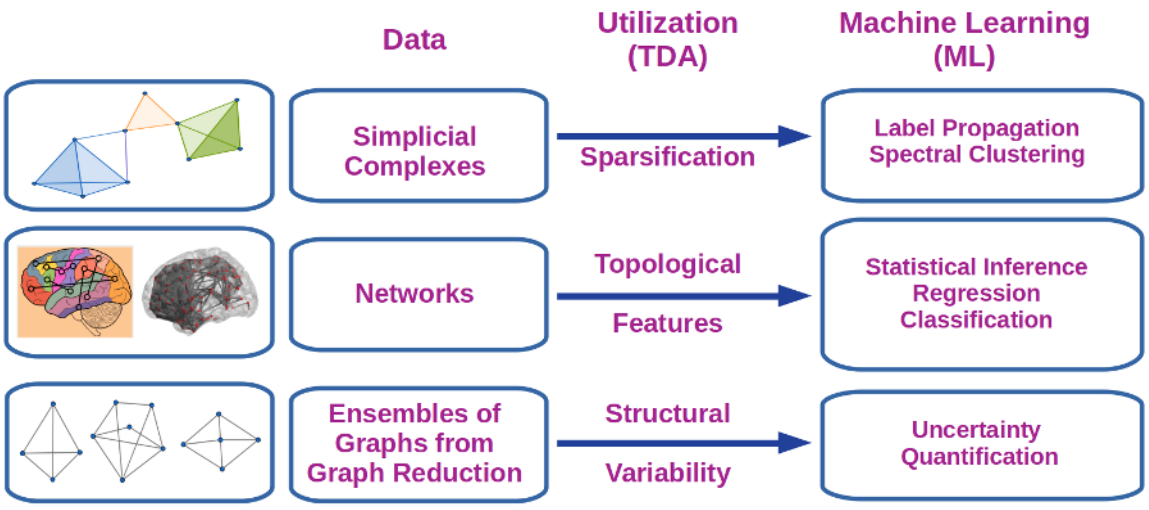

FIGURE. 1.1. Overview of the dissertation.

semisupervised and unsupervised learning algorithms, such as spectral clustering and label propagation, that leverage the topological structures encoded in the higher order relationships encoded in the simplicial complex. Analogous to graph algorithms, the learning algorithms for simplicial complexes scale with the size of the simplicial complex. Therefore, we also present a sparsification algorithm that reduces the size of the simplicial complex while preserving its spectral properties. A large part of this work is based on the following publication:

- B. OSTING, S. PALANDE, AND B. WANG[1], *Spectral sparsification of simplicial complexes for clustering and label propagation*, Journal of Computational Geometry (JoCG), 11 (2020), pp. 176–211.

Next, in Chapter 3, we work with datasets where individual samples are modeled as networks. We apply persistent homology, an essential TDA technique, to extract topological features from these networks and utilize them as inputs in machine learning. We apply these ideas to study structural and functional brain networks in autism spectrum disorders (ASD). The structural or functional relationships between different brain regions are often modeled as networks in ASD studies. We extract the topological features from these networks

[1]Authors are listed alphabetically

and utilize them in statistical inference or machine learning tasks such as regression and classification. The work presented in Chapter 3 is based on the following three publications:

- S. PALANDE, V. JOSE, B. ZIELINSKI, J. ANDERSON, P. T. FLETCHER, AND B. WANG, *Revisiting abnormalities in brain network architecture underlying autism using topology-inspired statistical inference*, Brain Connectivity, 9 (2019), pp. 13–21.

- E. WONG, S. PALANDE, B. WANG, B. ZIELINSKI, J. ANDERSON, AND P. T. FLETCHER, *Kernel partial least squares regression for relating functional brain network topology to clinical measures of behavior*, In 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), IEEE, (2016), pp. 1303–1306.

- A. RATHORE, S. PALANDE, J. ANDERSON, B. ZIELINSKI, P. T. FLETCHER, AND B. WANG, *Autism classification using topological features and deep learning: a cautionary tale*, In Medical Image Computing and Computer Assisted Intervention (MICCAI), Springer International Publishing, (2019), pp. 736–744.

Lastly, in Chapter 4, we once again work with data in the form of an ensemble of graphs obtained using graph reduction. However, unlike brain networks, we no longer have correspondences between vertices of the graphs. Graph reduction algorithms are used to approximate a large graph with a smaller graph, while preserving some basic properties. Different reduction algorithms can produce slightly different output graphs. Sometimes, a reduction algorithm employs randomization to improve its running time. As a result, different runs of the same algorithm can produce slightly different outputs, even with the same inputs and parameter settings. We present methods to evaluate the structural variability in an ensemble of reduced graphs obtained from different graph reduction algorithms or different runs of the same algorithm. These methods have potential applications in uncertainty quantification and visualization for graph reduction. We may also be able to use structural variability to compare different graph reduction algorithms. This part of the dissertation is based on the following manuscript:

- F. LAN, S. PALANDE, M. YOUNG, AND B. WANG, *Uncertainty visualization for graph reduction*, In preparation, 2020.

1.2 Other Contributions

Apart from work presented in this dissertation, we also contribute to two other projects. In the first project, we evaluate the reproducibility of graph-theoretic measures and topological features derived from brain networks across several resting-state functional MRI preprocessing strategies. We also analyze the relationship between these features and cognitive test scores and personality metrics. We show that topological features are reasonably robust to variations in preprocessing strategies. We also show that they are significantly correlated to individual differences in cognition and personality.

- K. L. ANDERSON, J. S. ANDERSON, S. PALANDE, AND B. WANG, *Topological data analysis of functional MRI connectivity in time and space domains*, in Connectomics in NeuroImaging, Springer International Publishing, 2018, pp. 67–77.

In the second project, we present **TopoAct**. This interactive visual analytics system leverages topological summaries called mapper graphs [115] to summarize the overall shape of the activation space for a fixed layer of a trained deep learning classifier. We observe branches and loops in the mapper graphs that correspond to evolving activation patterns. We find that these topological features are correlated with semantically meaningful differences between images from different classes.

- A. RATHORE, N. CHALAPATHY, S. PALANDE, AND B. WANG, *TopoAct: Exploring the shape of activations in deep learning*, Submitted, arXiv:1912.06332, 2019.

These projects fit into the overall theme of integrating ideas from TDA in machine learning. However, we do not include them in this dissertation and restrict our discussion to work mentioned earlier.

CHAPTER 2

SPECTRAL ALGORITHMS FOR SIMPLICIAL COMPLEXES

In this chapter, we present unsupervised and semisupervised learning and data reduction methods for simplicial complexes. Specifically, we focus on spectral learning algorithms such as spectral clustering, label propagation, and spectral sparsification.

In Section 2.1, we give an introduction and describe the motivation behind the work presented in the chapter. In Section 2.2, we introduce the notation used in this chapter and give a brief description of relevant algebraic concepts such as effective resistance and spectral sparsification of graphs. In Section 2.3, we present a spectral sparsification algorithm for simplicial complexes and prove that the sparsified simplicial complex closely approximates the spectral properties of the input complex. In Section 2.4, we state and prove a generalized Cheeger inequality for the weighted simplicial complexes and give a lower bound for the sparsified simplicial complex. In Section 2.5, we present spectral clustering and label propagation algorithms for simplicial complexes by modifying the analogous algorithms for graphs. We also define a random walk on a simplicial complex at a fixed dimension and establish its equivalence to other random walks proposed in the literature. We showcase experimental results validating our algorithms in Section 2.6 and conclude with a discussion and some open questions in Section 2.7.

2.1 Introduction

Understanding large systems with complex interactions and multiscale dynamics is essential in various social, biological, and technological settings. A common approach to analyzing such a system is representing it as a graph where vertices represent objects and (weighted) edges represent *pairwise interactions* between the objects. A vast arsenal of methods has been developed to analyze the properties of graphs, which can then be combined with domain-specific knowledge to infer the system's properties. These tools

include graph partitioning and clustering [96, 126, 127], random processes on graphs [53], combinatorial graph invariants [40], and spectral graph theory [31]. In particular, spectral methods for graph-based learning have had great success due to their efficiency and good theoretical guarantees for applications ranging from image segmentation [82] to community detection [7]. For example, spectral clustering [6, 119] is a graph-based learning method used for the unsupervised clustering task, and label propagation [123, 132] is a graph-based learning method for semisupervised regression.

The computational costs associated with naive implementations of many graph-based algorithms are prohibitive. In this scenario, it is useful to approximate the original graph with one having fewer edges or vertices while preserving specific properties of interest, known as *graph sparsification*. A variety of graph sparsification methods have been developed that allow for both efficient storage and computation [14, 117, 118]; see [13] for a survey.

It is well known from spectral graph theory that the spectrum of the graph Laplacian bounds a variety of properties of interest, including the size of cuts (i.e., bottlenecks), clusters (i.e., communities), distances, various random processes (i.e., PageRank), and combinatorial properties (e.g., coloring, spanning trees). Spielman and Srivastava developed a method for graph sparsification using effective resistances of edges that approximately preserve the spectrum of the graph Laplacian [117]. It follows that this method [13] can be used to produce a sparsified graph that contains a great deal of information about the original graph, and, in the graph-based machine learning setting, about the underlying dataset.

2.1.1 Simplicial Complexes and Data Analysis

Although graphs have been used with great success in many applications, they capture only pairwise interactions between objects in the dataset. Simplicial complexes allow us to capture the *higher order interactions* that occur between three or more objects in complex datasets [55, 91].

Interest in developing learning algorithms that directly operate on simplicial complexes is growing. For example, researchers have begun to develop machine learning methods for simplicial complexes, including methods based on higher order random processes [15, 54, 90], generalized Cheeger and isoperimetric inequalities [56, 121, 98], high-dimensional

expanders [41, 79, 97], and spectral methods [127, 61].

In this chapter, we extend spectral learning algorithms such as spectral clustering and label propagation to simplicial complexes. Analogous to spectral graph algorithms, these algorithms on simplicial complexes are severely limited by the computational costs associated with massive datasets. Since these algorithms rely on the spectral theory for higher order Laplacians, it is desirable to develop methods for sparsifying simplicial complexes that approximately preserve the spectrum of higher order Laplacians. Our main result is a spectral sparsification algorithm for simplicial complex, based on the graph sparsification algorithm of Spielman and Srivastava [117].

2.1.2 Contributions

Our contributions are as follows:

- We introduce a *generalized effective resistance* of simplices by extending the notion of *effective resistance* of edges (e.g., [25, 42, 49]), and extend the methods and analysis of Spielman and Srivastava [117] for sparsifying graphs to simplicial complexes at a fixed dimension. We prove that our sparsification algorithm approximately preserves the spectrum of the up-Laplacian; see Theorem 2.1 and Section 2.3.

- We generalize the Cheeger constant of Gundert and Szedlák for *unweighted* simplicial complexes [56] to *weighted* simplicial complexes and show that the Cheeger constant of the sparsified simplicial complex is bounded below by a multiplicative factor of the first nontrivial eigenvalue of the up-Laplacian for the original complex; see Corollary 2.2.

- We extend spectral learning algorithms such as spectral clustering and label propagation to simplicial complexes. We demonstrate via substantial numerical experiments that preserving the structure of the up-Laplacian via sparsification also preserves the results of spectral clustering and label propagation (Section 2.6). These applications exemplify the utility of our spectral sparsification methods.

- Lastly, we define random walks on simplicial complexes at a fixed dimension. We also discuss other random walks on simplicial complexes proposed in literature [90, 97] and show that they are all equivalent to the random walk we define.

2.2 Background

In this section, we will set out the notation used throughout this chapter. We will first review the relevant definitions and algorithms for graphs and then give their simplicial complex analogues.

2.2.1 Spectral Algorithms for Graphs

Let $G = (V, E, w)$ be a connected weighted undirected graph with n vertices and m edges, with positive edge weights $w : E \to \mathbb{R}^+$. Let W be an $m \times m$ diagonal matrix with $W(e, e) = w(e) = w_e$. Let $A \in \mathbb{R}^{n \times n}$ be the weighted adjacency matrix such that $A(i, j) = w_e$, where w_e is the weight of the edge e. A is also referred to as the *affinity matrix* or the *similarity matrix*, where $A(i, j) \geq 0$ captures the affinity (i.e., measure of similarity) between vertices $i, j \in V$. Let $D \in \mathbb{R}^{n \times n}$ be a diagonal matrix, where $D(j, j) = \sum_i A(i, j)$. In the case of a binary graph (where edge weights are either 0 or 1), D is simply the degree matrix, with diagonal elements $D(j, j)$ being the number of edges incident on vertex v_j.

The matrix $L = D - A$ is called the *graph Laplacian*, and the matrix $L_N = D^{-1/2} L D^{-1/2}$ is called the *normalized graph Laplacian*. Now suppose the edges are oriented arbitrarily. L can be written as $L = B^T W B$, where $B \in \mathbb{R}^{m \times n}$ is the signed edge-vertex incidence matrix defined as

$$B(e, j) = \begin{cases} 0 & \text{if vertex } j \text{ is not on the boundary of edge } e \\ 1 & \text{if } j \text{ is } e\text{'s head} \\ -1 & \text{if } j \text{ is } e\text{'s tail.} \end{cases}$$

2.2.1.1 Spectral clustering. We use the Ng-Jordan-Weiss algorithm [92], given here as Algorithm 1, to perform spectral clustering of graphs. To cluster the vertices of G into k clusters, the algorithm first computes a *spectral embedding* of the vertices of G into a k-dimensional Euclidean space and then uses the k-means algorithm to cluster the points in this Euclidean space into k clusters. The spectral embedding is constructed using the eigenvectors of matrix $M = \Delta^{-1/2} A \Delta^{-1/2} = I - L_N$. The largest eigenvalues of M correspond to the smallest eigenvalues of L_N, and the matrices have the same eigenvectors.

2.2.1.2 Label propagation on graphs. We describe here a simplified version of the iterative label propagation algorithm [131] based on the notion of stochastic matrix (i.e., random walk matrix) $P = AD^{-1}$. It represents the transition probabilities of the labels. Given P and an initial label distribution $\mathbf{y}$, we iteratively multiply the vector $\mathbf{y}$

Algorithm 1: $\mathbf{y} = \mathbf{Cluster}(G, d)$

Data: $G(V, E, W)$: A weighted, undirected graph with $|V| = n$, d: the number of clusters

Result: $\mathbf{y}$: A vector of cluster assignments $l \in \{1, 2, \ldots, d\}$ for the vertices of G.

Construct matrix A where $A(i, j) =$ the weight of edge e_{ij}, $A(i, j) = 0$ otherwise.
Compute diagonal matrix $D \in \mathbb{R}^{n \times n}$, where $D(j, j) = \sum_i A(i, j)$.
$M = D^{-1/2} A D^{-1/2}$.
Construct matrix $X = [u_1 u_2 \cdots u_d] \in \mathbb{R}^{n \times d}$ where $u_1, u_2, \ldots, u_d$ are the eigenvectors corresponding to the d largest eigenvalues of M (chosen to be orthogonal to each other in the case of repeated eigenvalues), respectively.
$Y_{ij} = X_{ij} / \left(\sum_j X_{ij}^2 \right)^{1/2}$ (normalize rows of X to have unit length).
$\mathbf{y} = \mathbf{kMeans}(Y, d)$.
Return $\mathbf{y}$ as cluster assignments for vertices of G.

by P. If the graph is *label-connected* (i.e., we can always reach a labeled vertex from any unlabeled one), then P^t converges to a stationary distribution x, that is, $P^t \mathbf{x} = \mathbf{x}$ for a large enough t.

Suppose there are two label classes $\{+1, -1\}$. Without loss of generality, assume that the first l of the n vertices are assigned labels initially, represented as a length-l vector $\mathbf{y}_l$. Given a graph $G(V, E)$ and labels $\mathbf{y}_l$, the simplified version of label propagation algorithm is outlined in Algorithm 2.

Consider P to be divided into blocks as follows:

$$P = \begin{pmatrix} P_{ll} & P_{lu} \\ P_{ul} & P_{uu}, \end{pmatrix}$$

where l and u index the labeled and unlabeled vertices with the number of vertices $n_0 = l + u$. Let $\mathbf{y} = (\mathbf{y}_l, \mathbf{y}_u)$ be the labels at convergence. Then, $\mathbf{y}_u$ is given by

$$\mathbf{y}_u = (I - P_{uu})^{-1} P_{ul} \mathbf{y_l}.$$

As long as our graph is connected, it is also label-connected and $(I - P_{uu})$ is nonsingular. Therefore, we can directly compute the labels at convergence without going through the iterative process described in Algorithm 2.

2.2.1.3 Graph sparsification. Algorithm 1 and Algorithm 2 both require computing the eigenvectors of the graph Laplacian. The computation scales with the number of edges in the graph. For large input graphs, these computations can be prohibitively expensive. In such cases, it is desirable to approximate the input graph with a smaller, sparser

Algorithm 2: $\mathbf{y} = \textbf{PropagateLabels}(G, \mathbf{y_l})$

Data: A weighted, undirected graph G with n vertices, vector $\mathbf{y_l}$ containing labels $\in \{+1, -1\}$ of first l vertices.

Result: A vector $\mathbf{y}$ of label assignments $l \in \{+1, -1\}$ for all the vertices of G.

Order the vertices of G so that labels $\mathbf{y_l}$ correspond to the first l vertices.
Construct matrix A where $A(i,j) =$ the weight of edge (i,j), $A(i,j) = 0$ otherwise.
Compute diagonal matrix $D \in \mathbb{R}^{n \times n}$, where $D(j,j) = \sum_i A_{ij}$.
$P = AD^{-1}$.
Initialize $\mathbf{y}^{(0)} = (\mathbf{y}_l, \mathbf{0})$, $t = 0$.
Repeat until convergence:

$$\mathbf{y}^{(t+1)} = P\mathbf{y}^{(t)},$$
$$\mathbf{y}_l^{(t+1)} = \mathbf{y}_l^{(t)}.$$

Return $\text{sgn}(\mathbf{y^{(t)}})$ as label assignments for vertices of G.

graph, perform computations on this smaller graph, and use the result to approximate a solution for the input graph. Graph sparsification is the process of approximating a graph with another graph, with fewer edges, while preserving certain properties.

There are several different notions of graph sparsification. Spectral sparsification, as the name suggests, attempts to preserve the spectral properties of the associated graph Laplacian. We say $H = (V, F, u)$ is a *sparse ε-approximation* of $G = (V, E, w)$ if $F \subset E$ and

$$(1 - \varepsilon)L_G \preceq L_H \preceq (1 + \varepsilon)L_G, \tag{2.1}$$

where L_G and L_H are the graph Laplacians of G and H, respectively, and the inequalities are to be understood in the sense of the semidefinite matrix ordering, i.e.,

$$(1 - \varepsilon)x^T L_G x \leq x^T L_H x \leq (1 + \varepsilon)x^T L_G x \qquad \forall x \in \mathbb{R}^n.$$

The "sparsification by effective resistances" algorithm proposed by Spielman and Srivastava [117], described in Algorithm 3, is most relevant to our approach. The effective resistance R_e at an edge e is the energy dissipation (potential difference) when a unit current is injected at one end and removed at the other end of e. Let us define the matrix $R := B(L)^+ B^T = B(B^T W B)^+ B^T$, where L^+ is the Moore-Penrose pseudoinverse of L. The diagonal entry $R(e,e)$ of R, is the effective resistance R_e across e.

In practice, computing R exactly as described in Algorithm 3 would be at least as expensive as performing spectral clustering on the input graph directly. However, Spielman

Algorithm 3: $H = \textbf{SparsifyGraph}(G, q)$

Data: A weighted, undirected graph G, and an integer q.

Result: A weighted, undirected graph H, which is a sparse ε-approximation of G

Sample q edges independently with replacement according to the probability

$$p_e = \frac{w(e)R(e,e)}{\sum_e w(e)R(e,e)},$$

and add sampled edges to H with weight $w(e)/qp_e$. If an edge is chosen more than once, the weights are summed.

and Srivastava [117] showed that a constant factor approximation of the effective resistances is sufficient to obtain a good graph sparsifier. They also provided a nearly linear time algorithm to compute the approximate effective resistances.

2.2.2 Simplicial Complexes

With the various matrices defined for graphs in mind, we will now define their simplicial complex analogues. A *simplicial complex* K is a finite collection of simplices such that every face of a simplex of K is in K, and the intersection of any two simplices of K is a face of each of them [91]. The 0-, 1-, and 2-simplices correspond to vertices, edges, and triangles. An *oriented simplex* is a simplex with a chosen ordering of its vertices. Consider an oriented $(k+1)$-simplex $\tau = [v_0, \ldots, v_{k+1}]$ of K where $v_0 < \cdots < v_{k+1}$ is the vertex ordering. $\sigma = \tau \backslash \{v_j\}$ is the k-simplex obtained from τ by omitting vertex v_j. The *oriented incidence number* $[\tau \colon \sigma]$ of a k-simplex σ of K is defined as $(-1)^j$ if $\sigma = \tau \backslash \{v_j\}$ for some $j = 0, \ldots, k+1$ and 0 if $\sigma \not\subseteq \tau$. For the remainder of this paper, we will assume K is an oriented simplicial complex on a vertex set $V = \{v_1, v_2, \ldots, v_n\}$. $S_k(K)$ denotes the collection of all oriented k-simplices of K and $n_k = |S_k(K)|$. The *p-skeleton* of K is denoted as $K^{(p)} := \bigcup_{0 \le k \le p} S_k(K)$. Let $\dim K$ denote the dimension of K. For a review of simplicial complexes, see [51, 55, 91].

2.2.2.1 Laplace operators on simplicial complexes. The k-th *chain group* of a complex K with coefficient in $\mathbb{R}$ is a vector space over the field $\mathbb{R}$ with basis $S_k(K)$. We denote it by $C_k(K)$ or $C_k(K, \mathbb{R})$. The k-th *cochain group* $C^k(K) = C^k(K, \mathbb{R})$ is the dual of the chain group, defined by $C^k(K) := \mathrm{Hom}(C_k(K), \mathbb{R})$, where $\mathrm{Hom}(C_k(K), \mathbb{R})$ denotes all homomorphisms of $C_k(K)$ into $\mathbb{R}$. The coboundary operator $\delta_k \colon C^k(K) \to C^{k+1}(K)$

is defined as $(\delta_k f)(\tau) = \sum_{\sigma \in S_k} [\tau \colon \sigma] f(\sigma)$. Let $Z^k = \mathrm{Ker}(\delta_k)$ and $B^k = \mathrm{Im}(\delta_{k-1})$ denote the groups of k-dimensional *cocycles* and k-dimensional *coboundaries*, respectively. The coboundary operator satisfies the property $\delta_k \delta_{k-1} = 0$, which implies that $B^k \subseteq Z^k$. The boundary operators, δ_k^*, are the adjoints of the coboundary operators,

$$\cdots \quad C^{k+1}(K) \underset{\delta_k^*}{\overset{\delta_k}{\leftrightarrows}} C^k(K) \underset{\delta_{k-1}^*}{\overset{\delta_{k-1}}{\leftrightarrows}} C^{k-1}(K) \quad \cdots$$

satisfying $(\delta_k a, b)_{C^{k+1}} = (a, \delta_k^* b)_{C^k}$ for every $a \in C^k(K)$ and $b \in C^{k+1}(K)$, where $(\cdot, \cdot)_{C^k}$ denotes the scalar product on the cochain group. We denote by $Z_k = \mathrm{Ker}(\delta_k^*)$ and $B_k = \mathrm{Im}(\delta_{k+1}^*)$, the groups of k-dimensional *cycles* and k-dimensional *boundaries*, respectively.

Following [61], we define three combinatorial Laplace operators that operate on $C^k(K)$ (for the k-th dimension), namely, the *up-Laplacian*,

$$\mathcal{L}_k^{\mathrm{up}}(K) = \delta_k^* \delta_k,$$

the *down-Laplacian*, $\mathcal{L}_k^{\mathrm{down}}(K) = \delta_{k-1} \delta_{k-1}^*$, and the *Laplacian*, $\mathcal{L}_k(K) = \mathcal{L}_k^{\mathrm{up}}(K) + \mathcal{L}_k^{\mathrm{down}}(K)$. All three operators are self-adjoint, non-negative, and compact, and they enjoy a collection of spectral properties, as detailed in [61]. We restrict our attention to the up-Laplacians.

2.2.2.2 Explicit expression for the up-Laplacian. To make the expression of the up-Laplacian explicit, we need to choose a scalar product on the coboundary vector spaces that can be viewed in terms of weight functions [61]. In particular, the weight function w is evaluated on the set of all simplices of K, $w \colon \bigcup_{k=0}^{\dim K} S_k(K) \to \mathbb{R}^+$, where the weight of a simplex f is $w(f)$ (also denoted as w_f). Let $w_k \colon S_k(K) \to \mathbb{R}^+$. Then, $C^k(K)$ is the space of real-valued functions on $S_k(K)$, with inner product $(a, b)_{C^k} := \sum_{f \in S_k(K)} w_k(f) a(f) b(f)$, for every $a, b \in C^k(K)$.

Choosing the natural bases, we identify each coboundary operator δ_k with an incidence matrix D_k. The *incidence matrix* $D_k \in \mathbb{R}^{n_{k+1}} \times \mathbb{R}^{n_k}$ encodes which k-simplices are incident to which $(k+1)$ simplices in the complex, and is defined as

$$D_k(i,j) = \begin{cases} 0 & \text{if } \sigma_j^k \text{ is not on the boundary of } \sigma_i^{k+1} \\ 1 & \text{if } \sigma_j^k \text{ is coherent with the induced orientation of } \sigma_i^{k+1} \\ -1 & \text{if } \sigma_j^k \text{ is not coherent with the induced orientation of } \sigma_i^{k+1} \end{cases}.$$

Let D_k^T be the transpose of D_k. Let W_k be the diagonal matrix representing the scalar product on $C^k(K)$. The k-dimensional up-Laplacian can then be expressed in the chosen

bases as the matrix

$$\mathcal{L}_k^{\text{up}}(K) = W_k^{-1} D_k^T W_{k+1} D_k.$$

We also define the following symmetric positive semidefinite matrix:

$$\mathcal{L}_{K,k} = W_k \mathcal{L}_k^{\text{up}}(K) = D_k^T W_{k+1} D_k.$$

Using the notation for the up-Laplacians, we can write the graph Laplacian as $L = D_0^T W_1 D_0 = W_0 \mathcal{L}_{K,0}$, by setting $B = D_0$ and $W = W_1$. Vertex weights are usually ignored in the graph sparsification literature, which is equivalent to setting the corresponding weight matrix W_0 to the identity. We can now express R as $R_1 = D_0(L)^+ D_0^T = D_0(D_0^T W_1 D_0)^+ D_0^T$.

2.2.2.3 Affinity matrix of a simplicial complex. For a fixed dimension k, we define an $n_k \times n_k$ weighted, oriented *affinity matrix* A_k^{up} such that for all k-dimensional simplices σ_i, σ_j of the simplicial complex K,

$$A_k^{\text{up}}(i,j) = \begin{cases} -w_f & \sigma_i \text{ and } \sigma_j \text{ are both faces of the same } (k+1)\text{-simplex } f \in S_{k+1}(K) \text{ and} \\ & \text{both agree or disagree with the orientation of } f \text{ ,} \\ w_f & \sigma_i \text{ and } \sigma_j \text{ are both faces of the same } (k+1)\text{-simplex } f \in S_{k+1}(K) \text{ and} \\ & \text{either } \sigma_i \text{ or } \sigma_j \text{ (but not both) agree with the orientation of } f \text{ ,} \\ 0 & \text{if } \sigma_i \text{ and } \sigma_j \text{ are not faces of the same } (k+1)\text{-simplex } f \\ & \text{for any } f \in S_{k+1}(K) \text{ .} \end{cases}$$

For example, consider the clockwise oriented triangle f in Figure 2.1. Let w_f be the weight of f. Edges e_1, e_2 of f are both oriented clockwise with respect to f. Therefore, they have a negative affinity of $-w_f$. Edges e_1, e_3 have different orientations with respect to f. Therefore they have a positive affinity of w_f.

The weighted degree of a k-simplex, $\deg(\sigma_i)$, is defined as the sum of weights of all $(k+1)$-simplices incident on σ_i, i.e.,

$$\deg(\sigma_i) = \sum_{f \in S_{k+1};\ \sigma_i \subset f} w_f.$$

We define the $n_k \times n_k$ degree matrix Δ_k indexed by k-simplices $\sigma_i \in K$ as

$$\Delta_k(i,i) = \deg(\sigma_i).$$

With these definitions, we can write the matrix $\mathcal{L}_{K,k}$ as

$$\mathcal{L}_{K,k} = \Delta_k - A_k^{up},$$

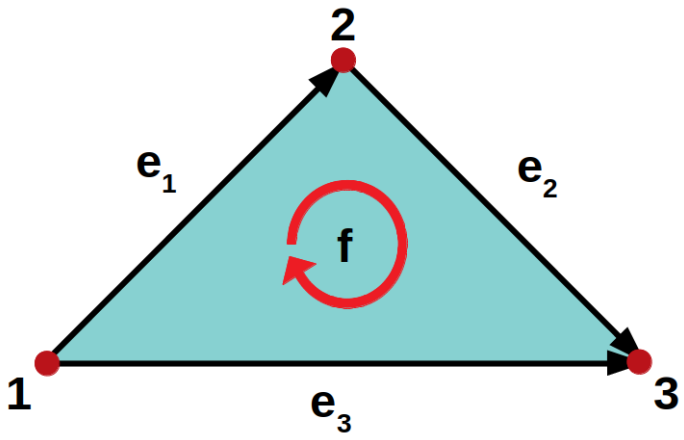

FIGURE. 2.1. Example of oriented affinity. Edges e_1, e_2 of triangle f have the same (clockwise) orientation w.r.t. f. Therefore, they have a negative affinity of $-w_f$, where w_f is the weight of triangle f. Edges e_1, e_3 have opposite orientations. Therefore, they have a positive affinity of w_f.

2.3 Sparsification of Simplicial Complexes

We now describe a sparsification algorithm for simplicial complexes (Algorithm 4) and our main theoretical result (Theorem 2.1) along with a detailed proof.

2.3.1 Sparsification Algorithm and the Main Theorem

To generalize the effective resistance for simplices beyond dimension one (i.e., edges), we consider the operator $R_k \colon C^k \to C^k$, defined by

$$R_k = D_{k-1}(\mathcal{L}_{K,k-1})^{+} D_{k-1}^T = D_{k-1}\left(D_{k-1}^T W_k D_{k-1}\right)^{+} D_{k-1}^T,$$

which is the projection onto the image of D_{k-1}. The *generalized effective resistance* of the k-dimensional simplex, f, is defined to be the diagonal entry, $R_k(f,f)$ (also denoted as R_f). For $k = 1$, the generalized effective resistance reduces to the effective resistance for graphs [49].

2.3.1.1 Sparsification algorithm. Algorithm 4 is a natural generalization of the graph sparsification algorithm (Algorithm 3) given by Spielman and Srivastava [117]. The algorithm sparsifies a given simplicial complex K at a fixed dimension k (while ignoring all dimensions larger than k). The main idea is to include each k-simplex f of K in the sparsifier J with a probability proportional to its generalized effective resistance. Specifically,

Algorithm 4: $J = \textbf{Sparsify}(K, k, q)$

Data: A weighted, oriented simplicial complex K, a dimension k (where $1 \le k \le \dim K$), and an integer q.

Result: A weighted, oriented simplicial complex J that is sparsified at dimension k, with equivalent $(k-1)$-skeleton to K and $\dim J = k$.

$J := K^{(k-1)}$

Sample q k-dimensional simplices independently with replacement according to the probability

$$p_f = \frac{w_k(f) R_k(f,f)}{\sum_f w_k(f) R_k(f,f)},$$

and add sampled simplices to J with weight $w_k(f)/qp_f$. If a simplex is chosen more than once, the weights are summed.

for a fixed dimension k, the algorithm chooses a random k-simplex f of K with probability p_f (proportional to $w_f R_f$) and adds f to J with weight w_f/qp_f. q samples are taken independently with replacement, summing the weights if a simplex is chosen more than once. The following theorem (Theorem 2.1) shows that if q is sufficiently large, the $(k-1)$-dimensional up-Laplacians of K and J are close

Theorem 2.1. *Let K be a weighted, oriented simpicial complex, and for some fixed k, let $J = \textbf{Sparsify}(K, k, q)$, where $1 \le k \le \dim K$. Suppose K and J have $(k-1)$-th up-Laplacians $\mathcal{L}_K := \mathcal{L}^{up}_{k-1}(K)$ and $\mathcal{L}_J := \mathcal{L}^{up}_{k-1}(J)$, respectively. Let n_{k-1} denote the number of $(k-1)$-simplices in K. Fix $\epsilon > 0$ (where $1/\sqrt{n_{k-1}} < \epsilon \le 1$), and let $q = 9C^2 n_{k-1} \log n_{k-1}/\epsilon^2$, where C is an absolute constant. If n_{k-1} is sufficiently large, then with probability at least $1/2$,*

$$(1-\epsilon)\mathcal{L}_K \preceq \mathcal{L}_J \preceq (1+\epsilon)\mathcal{L}_K, \tag{2.2}$$

where the inequalities are to be understood in the sense of the semidefinite matrix ordering. Equivalently,

$$(1-\epsilon)x^T\mathcal{L}_K x \le x^T\mathcal{L}_J x \le (1+\epsilon)x^T\mathcal{L}_K x \qquad \forall\, x \in \mathbb{R}^{n_{k-1}}.$$

A proof of Theorem 2.1 is detailed in the next section.

2.3.2 Proof of the Main Theorem

Our proof closely follows the proof of Spielman and Srivastava [117, Theorem 1]. Following the definitions from Section 2.3, let $\mathcal{L} = W_{k-1}\mathcal{L}_K$ and $\tilde{\mathcal{L}} = W_{k-1}\mathcal{L}_J$. $\mathcal{L}$ and $\tilde{\mathcal{L}}$ are

symmetric positive semidefinite matrices. When our sparsification algorithm is applied to sparsify K at dimension k, all simplices in K of dimensions up to $k-1$ are simply copied to J along with the corresponding weights so that the weight matrix W_{k-1} is the same for K and J. To prove Theorem 2.1, we will first establish that

$$(1-\epsilon)\mathcal{L} \preceq \tilde{\mathcal{L}} \preceq (1+\epsilon)\mathcal{L},$$

and then show that (Equation 2.2) holds if and only if the inequality above holds.

Since $\mathcal{L}$ is symmetric, positive semidefinite, $\mathcal{L}^+$ and R_k are also symmetric positive semidefinite matrices. We define the matrix $\Pi = W_k^{1/2} R_k W_k^{1/2}$.

Lemma 2.1. *The matrix $\Pi = W_k^{1/2} R_k W_k^{1/2}$ defined above has following properties:*

1. Π *is a projection matrix.*

2. $\mathrm{Im}(\Pi) = \mathrm{Im}(W_k^{1/2} D_{k-1})$.

3. $\Pi(f,f) = \|\Pi(\cdot, f)\|^2$.

4. $\mathrm{Rank}(\Pi) = \mathrm{Tr}(\Pi) \leq n_{k-1}$.

Proof. 1. Observe that

$$\begin{aligned}
\Pi^2 &= (W_k^{1/2} D_{k-1} \mathcal{L}^+ D_{k-1}^T W_k^{1/2})(W_k^{1/2} D_{k-1} \mathcal{L}^+ D_{k-1}^T W_k^{1/2}) \\
&= (W_k^{1/2} D_{k-1} \mathcal{L}^+)(D_{k-1}^T W_k^{1/2} W_k^{1/2} D_{k-1})(\mathcal{L}^+ D_{k-1}^T W_k^{1/2}) \\
&= W_k^{1/2} D_{k-1} \mathcal{L}^+ \mathcal{L} \mathcal{L}^+ D_{k-1}^T W_k^{1/2} \qquad \text{since} \quad \mathcal{L} = D_{k-1}^T W_k D_{k-1} \\
&= W_k^{1/2} D_{k-1} \mathcal{L}^+ D_{k-1}^T W_k^{1/2} \\
&= \Pi.
\end{aligned}$$

2. First, note that $\mathrm{Im}(\Pi) = \mathrm{Im}(W_k^{1/2} D_{k-1} \mathcal{L}^+ D_{k-1}^T W_k^{1/2}) \subseteq \mathrm{Im}(W_k^{1/2} D_{k-1})$. Now, for any vector $y \in \mathrm{Im}(W_k^{1/2} D_{k-1})$, there exists a vector $x \perp \mathrm{Ker}(W_k^{1/2} D_{k-1}) = \mathrm{Ker}(\mathcal{L})$ such that $y = W_k^{1/2} D_{k-1} x$. Then,

$$\begin{aligned}\Pi y &= (W_k^{1/2} D_{k-1} \mathcal{L}^+ D_{k-1}^T W_k^{1/2})(W_k^{1/2} D_{k-1} x) \\ &= (W_k^{1/2} D_{k-1} \mathcal{L}^+)(D_{k-1}^T W_k^{1/2} W_k^{1/2} D_{k-1}) x \\ &= W_k^{1/2} D_{k-1} \mathcal{L}^+ \mathcal{L} x \\ &= W_k^{1/2} D_{k-1} x \\ &= y.\end{aligned}$$

Therefore, $y \in \mathrm{Im}(\Pi)$.

3. We have, $\Pi(f,f) = \Pi^2(f,f)$, and since Π is symmetric,

$$\Pi^2(f,f) = \Pi(\cdot,f)^T \Pi(\cdot,f) = \|\Pi(\cdot,f)\|^2 .$$

4. Since $\Pi^2 = \Pi$, all eigenvalues of Π are either 0 or 1. Therefore, $\mathrm{Rank}(\Pi) = \mathrm{Tr}(\Pi)$. Although, since D_{k-1} is an $n_k \times n_{k-1}$ matrix,

$$\dim \mathrm{Im}(\Pi) = \dim \mathrm{Im}(W_k^{1/2} D_{k-1}) \leq n_{k-1}.$$

□

We also define the $n_k \times n_k$ non-negative, diagonal matrix Q_k with entries:

$$Q_k(f,f) = \frac{\tilde{w}_f}{w_f} = \frac{\#\text{ times } f \text{ is sampled}}{q p_f},$$

where the random entry $Q_k(f,f)$ captures the "amount" of k-simplex f included in J by the sparsification algorithm. The weight of simplex f in J is $\tilde{w}_f = Q_k(f,f) w_f$. The weight matrix can be written as $\tilde{W}_k = W_k Q_k = W_k^{1/2} Q_k W_k^{1/2}$. The $(k-1)$-th up-Laplacian $\tilde{\mathcal{L}}$ of the sparse complex can be written as

$$\tilde{\mathcal{L}} = D_{k-1}^T \tilde{W}_k D_{k-1} = D_{k-1}^T (W_k^{1/2} Q_k W_k^{1/2}) D_{k-1}.$$

The scaling by $1/qp_f$ in Q_k ensures that $\mathbb{E}\tilde{W}_k = W_k$. As a result, we have $\mathbb{E}Q_k = I$ and $\mathbb{E}\tilde{\mathcal{L}} = \mathcal{L}$.

Lemma 2.2 (Rudelson and Vershynin [109])**.** *Let* $\mathbf{p}$ *be a probability distribution over* $\Omega \subseteq \mathbb{R}^d$ *such that* $\sup_{y \in \Omega} \|y\|_2 \leq M$ *and* $\left\|\mathbb{E}_{\mathbf{p}} y y^T\right\|_2 \leq 1$*. Let* $y_1, y_2, \ldots, y_q$ *be independent samples drawn from* $\mathbf{p}$*. Then,*

$$\mathbb{E}\left\|\frac{1}{q}\sum_{i=1}^{q} y_i y_i^T - \mathbb{E} y y^T\right\|_2 \leq \min\left\{CM\sqrt{\frac{\log(q)}{q}},\ 1\right\},$$

where C *is an absolute constant.*

The matrix $\Pi Q_k \Pi$ can be expressed as the average of symmetric rank one matrices:

$$\begin{aligned}\Pi Q_k \Pi &= \sum_f Q_k(f,f)\Pi(\cdot,f)\Pi(\cdot,f)^T \\ &= \sum_f \frac{(\#\text{ times } f \text{ is sampled})}{qp_f}\Pi(\cdot,f)\Pi(\cdot,f)^T \\ &= \frac{1}{q}\sum_f (\#\text{ times } f \text{ is sampled})\frac{\Pi(\cdot,f)}{\sqrt{p_f}}\frac{\Pi(\cdot,f)^T}{\sqrt{p_f}} \\ &= \frac{1}{q}\sum_{i=1}^{q} y_i y_i^T.\end{aligned}$$

Vectors y_i are drawn independently with replacement from the distribution

$$y = \frac{1}{\sqrt{p_f}}\Pi(\cdot,f) \quad \text{with probability } p_f.$$

The expectation of yy^T is given by

$$\mathbb{E}yy^T = \sum_f p_f \frac{\Pi(\cdot,f)}{\sqrt{p_f}}\frac{\Pi(\cdot,f)^T}{\sqrt{p_f}} = \Pi\Pi = \Pi.$$

Therefore, $\left\|\mathbb{E}yy^T\right\|_2 = \|\Pi\|_2 = 1$. A bound on the norm of y is given by

$$\frac{1}{\sqrt{p_f}}\|\Pi(\cdot,f)\|_2 = \frac{\sqrt{\Pi(f,f)}}{\sqrt{p_f}} = \sqrt{\sum_f w_f R_k(f,f)} = \sqrt{\mathrm{Tr}(\Pi)} \le n_{k-1}.$$

Now, using Lemma 2.2, with $q = 9C^2 n_{k-1}\ln n_{k-1}/\epsilon^2$, we have

$$\mathbb{E}\,\|\Pi Q_k \Pi\|_2 = \mathbb{E}\left\|\frac{1}{q}\sum_{i=1}^{q} y_i y_i^T\right\|_2 \le C\sqrt{\epsilon^2 n_{k-1}\frac{\ln\left(9C^2 n_{k-1}\ln n_{k-1}/\epsilon^2\right)}{9C^2 n_{k-1}\ln n_{k-1}}} \le \frac{\epsilon}{2},$$

for sufficiently large n_{k-1}, as ϵ is assumed to be at least $1/\sqrt{n_{k-1}}$. Then, by Markov's inequality, we have $\|\Pi Q_k \Pi - \Pi\Pi\|_2 \le \epsilon$ with a probability at least $1/2$.

Lemma 2.3. *Suppose Q_k is non-negative diagonal matrix such that $\|\Pi Q_k \Pi - \Pi\Pi\|_2 \le \epsilon$. Then, for all $x \in \mathbb{R}^{n_{k-1}}$,*

$$(1-\epsilon)x^T\mathcal{L}x \le x^T\tilde{\mathcal{L}}x \le (1+\epsilon)x^T\mathcal{L}x$$

where $\mathcal{L} = D_{k-1}^T W_k D_{k-1}$ and $\tilde{\mathcal{L}} = D_{k-1}^T W_k^{1/2} Q_k W_k^{1/2} D_{k-1}$.

Proof. For a symmetric matrix A, $\|A\|_2 = \sup_{y\neq 0} \frac{|y^T A y|}{y^T y}$. Therefore, the assumption that $\|\Pi Q_k \Pi - \Pi\Pi\|_2 \leq \epsilon$ is equivalent to

$$\sup_{y\in\mathbb{R}^{n_k}, y\neq 0} \frac{|y^T \Pi (Q_k - I)\Pi y|}{y^T y} \leq \epsilon.$$

Note that if $x \in \ker(W_k^{1/2} D_{k-1})$, $x^T \mathcal{L} x = x^T \tilde{\mathcal{L}} x = 0$ and the claim holds trivially. If $x \notin \ker(W_k^{1/2} D_{k-1})$, then there is a vector $y = W_k^{1/2} D_{k-1} x$ in $\mathrm{Im}(W_k^{1/2} D_{k-1})$. Restricting our attention to such vectors, we have

$$\sup_{y\in\mathrm{Im}(W_k^{1/2} D_{k-1}), y\neq 0} \frac{|y^T \Pi (Q_k - I)\Pi y|}{y^T y} \leq \epsilon.$$

However, from Lemma 2.1 we have $\Pi y = y$ for any $y \in \mathrm{Im}(W_k^{1/2} D_{k-1})$. Therefore, we have

$$\begin{aligned}
&\sup_{y\in\mathrm{Im}(W_k^{1/2} D_{k-1}), y\neq 0} \frac{|y^T \Pi (Q_k - I)\Pi y|}{y^T y} \\
&= \sup_{y\in\mathrm{Im}(W_k^{1/2} D_{k-1}), y\neq 0} \frac{|y^T (Q_k - I) y|}{y^T y} \\
&= \sup_{x\in\mathbb{R}^{n_{k-1}}, W_k^{1/2} D_{k-1} x\neq 0} \frac{|x^T D_{k-1}^T W_k^{1/2} (Q_k - I) W_k^{1/2} D_{k-1} x|}{x^T D_{k-1}^T W_k D_{k-1} x} \\
&= \sup_{x\in\mathbb{R}^{n_{k-1}}, W_k^{1/2} D_{k-1} x\neq 0} \frac{|x^T D_{k-1}^T W_k^{1/2} Q_k W_k^{1/2} D_{k-1} x - x^T D_{k-1}^T W_k D_{k-1} x|}{x^T D_{k-1}^T W_k D_{k-1} x} \\
&= \sup_{x\in\mathbb{R}^{n_{k-1}}, W_k^{1/2} D_{k-1} x\neq 0} \frac{|x^T \tilde{\mathcal{L}} x - x^T \mathcal{L} x|}{x^T \mathcal{L} x}.
\end{aligned}$$

Therefore, if $\|\Pi Q_k \Pi - \Pi\Pi\|_2 \leq \epsilon$, then

$$\sup_{x\in\mathbb{R}^{n_{k-1}}, W_k^{1/2} D_{k-1} x\neq 0} \frac{|x^T \tilde{\mathcal{L}} x - x^T \mathcal{L} x|}{x^T \mathcal{L} x} \leq \epsilon.$$

Rearranging the terms, we get for all $x \in \mathbb{R}^{n_{k-1}}$,

$$(1-\epsilon) x^T \mathcal{L} x \leq x^T \tilde{\mathcal{L}} x \leq (1+\epsilon) x^T \mathcal{L} x,$$

which is equivalent to

$$(1-\epsilon)\mathcal{L} \preceq \tilde{\mathcal{L}} \preceq (1+\epsilon)\mathcal{L}. \tag{2.3}$$

□

Now, to show that (Equation 2.2) holds, we use the following elementary lemma (whose proof is included for completeness):

Lemma 2.4. *For any symmetric positive semidefinite matrices A and B and any positive definite diagonal matrix D, we have $A \succeq B$ if and only if $DA \succeq DB$.*

Proof. First, assume $A \succeq B$. Let $C = A - B$. Then, $C \succeq 0$. Since D is a positive definite diagonal matrix, $D^{1/2}CD^{1/2} \succeq 0$. However, $D^{1/2}CD^{1/2}$ is similar to DC, because

$$D^{1/2}CD^{1/2} = D^{-1/2}(DC)D^{1/2}.$$

Therefore, DC has the same eigenvalues as $D^{1/2}CD^{1/2}$, which means $DC \succeq 0$ or equivalently, $DA \succeq DB$.

Now suppose $DC \succeq 0$. Then, $D^{1/2}CD^{1/2} \succeq 0$, due to similarity. However,

$$C = D^{-1/2}(D^{1/2}CD^{1/2})D^{-1/2},$$

and therefore $C \succeq 0$ or equivalently, $A \succeq B$. □

We can write the up-Laplacians as $\mathcal{L}_K = W_{k-1}^{-1}\mathcal{L}$ and $\mathcal{L}_J = W_{k-1}^{-1}\tilde{\mathcal{L}}$. Since W_{k-1} is a diagonal matrix of positive weights, W_{k-1}^{-1} is a positive definite diagonal matrix. Therefore, according to Lemma 2.4, if the inequality in Equation 2.3 holds, then the inequality in Equation 2.2 must also hold, i.e.,

$$(1-\epsilon)\mathcal{L}_K \preceq \mathcal{L}_J \preceq (1+\epsilon)\mathcal{L}_K.$$

Corollary 2.1. *Suppose Z_f are numbers satisfying $Z_f \geq R_k(f,f)/\alpha$, and for some $\alpha \geq 1$, $\sum_f w_f Z_f \leq \alpha \sum_f w_f R_k(f,f)$. If we sample as in* **Sparsify** *(Algorithm 4) but take each k-simplex f with probability $p'_f = w_f Z_f / \sum_f w_f Z_f$ instead of $p_f = w_f R_k(f,f)/\sum_f w_f R_k(f,f)$, then the resulting sparse complex J satisfies*

$$(1-\epsilon\alpha)\mathcal{L} \preceq \tilde{\mathcal{L}} \preceq (1+\epsilon\alpha)\mathcal{L}.$$

Proof. Note that:

$$p'_f = \frac{w_f Z_f}{\sum_f w_f Z_f} \geq \frac{w_f R_k(f,f)\alpha}{\alpha \sum_f w_f R_k(f,f)} = \frac{p_f}{\alpha^2}.$$

Proceeding as in the proof for Theorem 2.1, the new bound on the norm of random vector y is given by

$$\frac{1}{\sqrt{p'_f}} \|\Pi(\cdot, f)\|_2 \leq \frac{\alpha}{\sqrt{p_f}} \sqrt{\Pi(f,f)} = \alpha\sqrt{\mathrm{Tr}(\Pi)}.$$

Thus, constant factor approximation of generalized effective resistances introduces the same constant factor, α, in the bound on expectation in Lemma 2.2, and consequently in the final inequality, but does not change anything else. □

2.4 Generalized Cheeger Inequalities for Simplicial Complexes

In Section 2.4.1, we show that the Cheeger constant of the sparsified simplicial complex is bounded below by a multiplicative factor of the first nontrivial eigenvalue of the up-Laplacian for the original complex (via Theorem 2.3 and Corollary 2.2). We give the proof of Theorem 2.3 in Section 2.4.2.

2.4.1 Cheeger Constant of Sparsified Simplicial Complexes

We begin with the following definition of the Cheeger constant for an unweighted graph $G = (V, E)$, which was later used by Gundert and Szedlák [56] to define a Cheeger inequality for unweighted simplicial complex. We will then generalize this definition to the case of weighted simplicial complexes.

2.4.1.1 Cheeger constant and inequality for graphs.

$$h(G) := \min_{\varnothing \subsetneq A \subsetneq V} \frac{|V|\ |E(A, V \setminus A)|}{|A|\ |V \setminus A|}, \tag{2.4}$$

where $E(A, V \setminus A)$ is the set of edges that connect $A \subset V$ to $(V \setminus A) \subset V$. For a weighted graph, $G = (V, E, w)$, this definition is typically generalized to

$$h(G) := \min_{\varnothing \subsetneq A \subsetneq V} \frac{|V|}{|A|\ |V \setminus A|} \sum_{(i,j) \in E(A, V \setminus A)} w_{ij}. \tag{2.5}$$

The Cheeger inequality for graphs takes the form $c \cdot \lambda_1(L_G) \leq h(G) \leq C \cdot \sqrt{\lambda_1(L_G)}$, where λ_1 is the first nontrivial eigenvalue of a graph Laplacian. The constants c and C depend on the choice of definition for the Cheeger constant and the graph Laplaican; see, e.g., [31, Chapter 2]. Using the variational formulation for eigenvalues and a suitable test function, it is not difficult to prove that for the weighted (un-normlized) graph Laplacian, the lower bound for the Cheeger constant defined in Equation 2.5 is given by $\frac{1}{2} \cdot \lambda_1(L_G) \leq h(G)$. Here, we prove an analogous inequality for weighted simplicial complexes, which we refer to as the generalized Cheeger inequality. This inequality gives a lower bound on the Cheeger

constant; an upper bound is not possible for weighted simplicial complexes by the argument of Gundert and Szedlák [56, p.5].

2.4.1.2 Generalized Cheeger inequality for simplicial complexes of Gundert and Szedlák. We first recall the generalized Cheeger inequality for simplicial complexes of Gundert and Szedlák [56]. For a k-dimensional simplicial complex K, its *k-dimensional completion* is defined to be

$$\bar{K} := K \bigcup \left\{ \tau^* \in \binom{V}{k+1} | (\tau^* \backslash \{v\}) \in X, \forall v \in \tau^* \right\}.$$

$\bar{K}$ is the complete k-dimensional complex when K has a complete $(k-1)$-skeleton. The generalized Cheeger constant for *unweighted* simplicial complexes is defined to be

$$h(K) := \min_{\substack{V = \bigsqcup_{i=0}^{k} A_i \\ A_i \neq \varnothing}} \frac{|V||F(A_0, A_1, \ldots, A_k)|}{|F^*(A_0, A_1, \ldots, A_k)|}, \tag{2.6}$$

where $F(A_0, A_1, \ldots, A_k)$ and $F^*(A_0, A_1, \ldots, A_k)$ are the sets of all k-simplices of K and $\overline{K}$, respectively, with one vertex in A_i for all $0 \leq i \leq k$.

Theorem 2.2 ([56, Theorem 2])**.** *If $\lambda_1(\mathcal{L}_K)$ is the first nontrivial eigenvalue of the k-th up-Laplacian and if every $(k-1)$-face is contained in at most C^* k-face of K, then*

$$\frac{|V|}{(k+1)\ C^*} \cdot \lambda_1(\mathcal{L}_K) \leq h(K).$$

Remark 2.1. *Recall that the Cheeger inequality for graphs includes an upper bound of the Cheeger constant $h(G)$ in terms of $\lambda_1(L_G)$. However, as pointed out by Gundert and Szedlák, $\lambda_1(\mathcal{L}_K) = 0$ does not imply $h(K) = 0$ [56]. Therefore, a higher dimensional analogue of this upper bound of the form $h(K) \leq C \cdot \lambda_1(\mathcal{L}_K)^{\frac{1}{m}}$ is not possible. We also remark that an alternative Cheeger inequality is given in [98].*

2.4.1.3 Generalized Cheeger constant for weighted simplicial complexes. Analogous to the generalization of the unweighted Cheeger constant in Equation 2.4 to the weighted Cheeger constant in Equation 2.5, we define the generalized Cheeger constant for weighted simplicial complexes by

$$h(K) := \min_{\substack{V = \bigsqcup_{i=0}^{k} A_i \\ A_i \neq \varnothing}} \frac{|V|}{|F^*(A_0, A_1, \ldots, A_k)|} \sum_{X \in F(A_0, A_1, \ldots, A_k)} w_k(X). \tag{2.7}$$

Observe that Equation 2.7 agrees with Equation 2.6 in the case when all weights are unity. The following result can be proved analogously to Theorem 2.2:

Theorem 2.3. *Let $\lambda_1(\mathcal{L}_K)$ be the first nontrivial eigenvalue of the $(k-1)$-th weighted up-Laplacian $\mathcal{L}_K$. If every $(k-1)$-face σ is contained in at most C^* k-faces of $\bar{K}$ and $w_{k-1}(\sigma) \geq W^* > 0$, then*

$$\frac{|V|\ W^*}{(k+1)\ C^*} \cdot \lambda_1(\mathcal{L}_K) \leq h(K).$$

A proof of Theorem 2.3 is given in Section 2.4.2. Combining Theorem 2.1 and Theorem 2.3 leads to the following result:

Corollary 2.2. *In the setting as Theorem 2.1 and Theorem 2.3, we have with probability* $\frac{1}{2}$

$$\frac{|V|\ W^*}{(k+1)\ C^*}(1-\epsilon) \cdot \lambda_1(\mathcal{L}_K) \leq \frac{|V|\ W^*}{(k+1)\ C^*} \cdot \lambda_1(\mathcal{L}_J) \leq h(J).$$

Thus, the Cheeger constant of the sparsified simplicial complex, J, is bounded below by a multiplicative factor of the first nontrivial eigenvalue of the up-Laplacian for the original complex, K.

2.4.2 Proof of the Cheeger Inequality

Note that $\mathrm{Ker}(\mathcal{L}_K) = \mathrm{Ker}(\delta_{k-1}) = Z^{k-1}$. Since $B^{k-1} \subseteq Z^{k-1}$, the eigenvectors corresponding to the nonzero eigenvalues of $\mathcal{L}_K$ are contained in $(B^{k-1})^{\perp}$. By the Hodge decomposition, we know that $(B^{k-1})^{\perp} = Z_{k-1}$. Therefore, $\lambda_1(\mathcal{L}_K)$ can be formulated as [56, Sec. 4, Eq. (2)],

$$\lambda_1(\mathcal{L}_K) = \min_{f \in Z_{k-1}} \frac{(\mathcal{L}_K f, f)_{C^{k-1}}}{(f,f)_{C^{k-1}}}, \tag{2.8}$$

where $(a,b)_{C^k} = \sum_{\sigma \in S_k} w_k(\sigma) a(\sigma) b(\sigma)$ for all $a, b \in C^k$ is the inner product defined over C^k, the space of all real valued functions on S_k. We will omit the subscript C^k from here on. The key idea in the proof is to find a function $f \in Z_{k-1}$ such that

$$\frac{(\mathcal{L}_K f, f)}{(f,f)} = h(K).$$

In order to define such a function, we fix a partition $A_0, \ldots, A_k$ of the vertex set V of K, which realizes the minimum in Equation 2.7. We will refer to A_i's as blocks. For simplicity, we choose a linear ordering on V such that for all $w \in A_i$ and $v \in A_j$ we have $w < v$ if $i < j$. To keep the notation simple, we will simply write F and F^* instead of $F(A_0, \ldots, A_k)$ and $F^*(A_0, \ldots, A_k)$. Note that $\lambda_1(\mathcal{L}_K)$ does not depend on the choice of orientation.

Let $\sigma = [v_0, \ldots, v_{k-1}] \in S_{k-1}$. Then $f \in C^{k-1}$ is defined as

$$f(\sigma) = \begin{cases} (-1)^l |A_l| & \text{if } A_l \text{ is the } \textit{unique} \text{ block not containing any } v_i, \\ 0 & \text{otherwise.} \end{cases}$$

Lemma 2.5. *Let $\mathcal{L}_K$ be the $(k-1)$-th weighted up-Laplacian of K and let f be defined as above. Then,*

$$(\mathcal{L}_K f, f) = (\delta_{k-1} f, \delta_{k-1} f) = |V|^2 \sum_{\tau \in F} w_k(\tau).$$

Proof. Consider $\tau = [v_0, \ldots, v_k] \in S_k$. If $\tau \in F$, i.e., if $v_i \in A_i$ for all $i = 0, \ldots, k$, then

$$(\delta_{k-1}) f(\tau) = \sum_{i=0}^{k} [\tau \colon \tau \backslash \{v_i\}] f(\tau \backslash \{v_i\}) = \sum_{i=0}^{k} (-1)^i (-1)^i |A_i| = |V|.$$

Now suppose $\tau \notin F$, but v_i, v_j is the only pair of vertices in the same block. Let $v_i < v_j$. Then by our chosen linear ordering, $i{+}1 = j$. If l is not equal to i or $i{+}1$, then $f(\tau \backslash \{v_l\}) = 0$. The only nonzero terms are $f(\tau \backslash \{v_i\}) = f(\tau \backslash \{v_j\})$. However, these terms cancel out because $[\tau \colon \tau \backslash \{v_i\}] = -[\tau \colon \tau \backslash \{v_{i+1}\}]$.

If three vertices belong to the same block or if there are two pairs of vertices belonging to the same blocks, or indeed in any other arrangement not covered before, there are at least two empty blocks and f is zero. Therefore,

$$(\delta_{k-1} f)(\tau) = \begin{cases} |V| & \text{if } \tau \in F, \\ 0 & \text{otherwise,} \end{cases}$$

and,

$$(\delta_{k-1} f, \delta_{k-1} f) = \sum_{\tau \in F} w_k(\tau) ((\delta_{k-1} f)(\tau))^2 = |V|^2 \sum_{\tau \in F} w_k(\tau).$$

□

Lemma 2.6. *Let $f \in C^{k-1}$ be as previously defined. Then unique $z \in Z_{k-1}$, $b \in B^{k-1}$ exist such that $f = z + b$. Furthermore,*

$$\lambda_1(\mathcal{L}_K) \leq \frac{|V|^2}{(z, z)} \sum_{\tau \in F} w_k(\tau).$$

Proof. Since $Z_{k-1} = (B^{k-1})^{\perp}$, unique cochains $z \in Z_{k-1}$ and $b \in B^{k-1}$ exist such that $f = z + b$. Also, $(\mathcal{L}_K z, z) = (\mathcal{L}_K f, f)$, because $b \in B^{k-1} \subseteq \mathrm{Ker}(\mathcal{L}_K)$. The claim now follows from this fact and using Equation 2.8 and Lemma 2.5. □

Lemma 2.7. *Let $f \in C^{k-1}$ be as previously defined and let $g \in C^{k-2}$ be arbitrary. For $\tau^* \in F^*$, define*

$$q(\tau^*, g) := \sum_{\sigma \subseteq \tau^*, \sigma \in S_{k-1}} \frac{w_{k-1}(\sigma)}{d(\sigma)} (f(\sigma) - \delta_{k-2} g(\sigma))^2,$$

where for all $\sigma \in S_{k-1}$, $d(\sigma) = |\{\tau^* \supseteq \sigma | \tau^* \in F^*\}|$. Then,

1. $(f - \delta_{k-2} g, f - \delta_{k-2} g) \geq \sum_{\tau^* \in F^*} q(\tau^*, g)$.

2. *For $\tau^* = \{v_0, v_1, \ldots, v_k\} \in F^*$ with $v_0 < v_1 < \cdots < v_k$,*

$$q(\tau^*, g) \geq \frac{|V|^2}{\sum_{i=0}^{k} \frac{d(\tau \backslash \{v_i\})}{w_{k-1}(\tau \backslash \{v_i\})}}.$$

Proof. 1. By definition of the inner product,

$$(f - \delta_{k-2} g, f - \delta_{k-2} g) = \sum_{\sigma \in S_{k-1}} w_{k-1}(\sigma)(f(\sigma) - \delta_{k-2} g(\sigma))^2.$$

Now, consider the sum on the right-hand side

$$\sum_{\tau^* \in F^*} q(\tau^*, g) = \sum_{\tau^* \in F^*} \sum_{\sigma \subseteq \tau^*, \sigma \in S_{k-1}} \frac{w_{k-1}(\sigma)}{d(\sigma)} (f(\sigma) - \delta_{k-2} g(\sigma))^2.$$

Note that for any $\sigma \in S_{k-1}$ such that $\sigma \subseteq \tau^*$, the corresponding term in the summation appears exactly $d(\sigma)$ times. If $\sigma \not\subseteq \tau^*$, then the corresponding term does not appear at all. Therefore,

$$\sum_{\tau^* \in F^*} q(\tau^*, g) \leq \sum_{\sigma \in S_{k-1}} w_{k-1}(\sigma)(f(\sigma) - \delta_{k-2} g(\sigma))^2.$$

2. Let $\tau^* = [v_0, \ldots, v_k] \in F^*$, such that $v_i \in A_i$ for $i = 1, \ldots, k$. Then,

$$\begin{aligned} q(\tau^*, g) &= \sum_{i=0}^{k} \frac{w_{k-1}(\tau^* \backslash \{v_i\})}{d(\tau^* \backslash \{v_i\})} ((-1)^i |A_i| - \delta_{k-2} g(\tau^* \backslash \{v_i\}))^2 \\ &= \sum_{i=0}^{k} \frac{w_{k-1}(\tau^* \backslash \{v_i\})}{d(\tau^* \backslash \{v_i\})} (|A_i| - [\tau^* \colon \tau^* \backslash \{v_i\}] \delta_{k-2} g(\tau^* \backslash \{v_i\}))^2. \end{aligned}$$

Note that the *oriented incidence number* $[\tau^* \colon \sigma]$ is $(-1)^i$ if $\sigma = \tau^* \backslash \{v_i\}$ for $i = 1, \ldots, k$ and 0 if $\sigma \not\subseteq \tau^*$. We also observe that $\sum_{i=0}^{k} [\tau^* \colon \tau^* \backslash \{v_i\}] \delta_{k-2} g(\tau^* \backslash \{v_i\}) = \delta_{k-1} \delta_{k-2} g(\tau^* \backslash \{v_i\}) = 0$. Therefore,

$$q(\tau^*, g) = \sum_{i=0}^{k} \frac{w_{k-1}(\tau^* \backslash \{v_i\})}{d(\tau^* \backslash \{v_i\})} |A_i|^2.$$

Now, using the following version of Cauchy-Schwarz inequality (Titu's lemma / Engel's form) for positive real numbers,

$$\sum_{i=0}^{k} \frac{a_i^2}{b_i} \geq \frac{(\sum_{i=0}^{k} a_i)^2}{\sum_{i=0}^{k} b_i},$$

we obtain

$$q(\tau^*, g) \geq \frac{(\sum_{i=0}^{k} |A_i|)^2}{\sum_{i=0}^{k} \frac{d(\tau \backslash \{v_i\})}{w_{k-1}(\tau \backslash \{v_i\})}} = \frac{|V|^2}{\sum_{i=0}^{k} \frac{d(\tau \backslash \{v_i\})}{w_{k-1}(\tau \backslash \{v_i\})}}.$$

□

Finally, from Lemma 2.6, recall that $f = z + b$ where $z \in Z_{k-1}$ and $b \in B^{k-1}$. Therefore, some $g \in C^{k-2}$ exists such that $f - z = b = \delta_{k-2} g$. By Lemma 2.7,

$$(z, z) = (f - \delta_{k-2} g, f - \delta_{k-2} g) \geq \sum_{\tau^* \in F^*} \frac{|V|^2}{\sum_{i=0}^{k} \frac{d(\tau \backslash \{v_i\})}{w_{k-1}(\tau \backslash \{v_i\})}}.$$

Now, if every $(k-1)$-face σ of K is contained in C^* k-faces of $\bar{K}$ and $w_{k-1}(\sigma) \geq W^*$, then

$$\sum_{i=0}^{k} \frac{d(\tau \backslash \{v_i\})}{w_{k-1}(\tau \backslash \{v_i\})} \leq (k+1) \frac{C^*}{W^*},$$

and

$$(z, z) \geq \frac{|V|^2 \ |F^*| \ W^*}{(k+1) \ C^*}.$$

Using this inequality along with Lemma 2.6, we can write

$$\lambda_1(\mathcal{L}_K) \leq \frac{|V|^2 \ (k+1) \ C^*}{|V|^2 \ W^* \ |F^*|} \sum_{\tau \in F} w_k(\tau).$$

Recall that we defined the function f by fixing a partition $A_0, \ldots, A_k$ that realizes the minimum from Equation 2.7, which means

$$h(K) = \frac{|V|}{|F^*|} \sum_{\tau \in F} w_k(\tau),$$

and we get the stated lower bound on $h(K)$:

$$\frac{|V| \ W^*}{(k+1) \ C^*} \lambda_1(\mathcal{L}_K) \leq h(K).$$

2.5 Spectral Clustering and Label Propagation for Simplicial Complexes

We want to extend spectral clustering (Algorithm 1) and label propagation (Algorithm 2) to simplicial complexes. Such extensions have rarely been studied. We seek the simplest generalization by replacing the vertex-vertex affinity matrix A with a general simplex-simplex affinity matrix A_d.

2.5.1 Algorithms on the Dual Graph

For a fixed dimension k, we can define a *dual graph* G_k of the simplicial complex K, where each k-simplex of K becomes a vertex of G_k, and an edge between two vertices of G_k exists if the corresponding k-simplices are faces of a $(k+1)$-simplex of K. The weight of this edge is same as the weight of the $(k+1)$-simplex. The $n_k \times n_k$ weighted adjacency matrix A_k^{dual} of G_k is defined as

$$A_k^{\text{dual}}(i,j) = \begin{cases} w_f & \sigma_i \text{ and } \sigma_j \text{ are both faces of the same } (k+1)\text{-simplex } f \in S_{k+1}(K) \\ 0 & \text{otherwise} \end{cases}.$$

Recall that the degree of a k-simplex is defined as the sum of weights of its $(k+1)$-dimensional co-faces. Also notice that for every $(k+1)$-simplex incident on a k-simplex $\sigma \in K$, we have $(k+1)$ adjacent k-simplices. Therefore, if Δ_k is the diagonal weighted degree matrix of the simplicial complex K at dimension k, then the weighted degree matrix of the dual graph G_k is given by $(k+1)\Delta_k$, and the graph Laplacian of the dual graph is given by

$$L_k^{\text{dual}} = (k+1)\Delta_k - A_k^{\text{dual}}.$$

The matrix A_k^{dual} consists of non-negative elements by definition, whereas A_k^{up} can have both positive and negative elements depending on the orientations of the simplices. Although A_k^{dual} and A_k^{up} have identical sparsity structures, we can easily verify that they need not have identical spectra. For spectral clustering and label propagation at dimension k, we modify Algorithm 1 and Algorithm 2 by replacing the matrices A and Δ with the matrices A_k^{dual} and $(k+1)\Delta_k$, respectively. The modified spectral clustering and label propagation algorithms are given in Algorithm 5 and Algorithm 6, respectively.

Let $M_{k,+}$ and $M_{k,-}$ be $n_k \times n_k$ non-negative matrices defined as follows:

$$M_{k,+}(i,j) = \max(A_k^{\text{up}}(i,j), 0),$$
$$M_{k,-}(i,j) = \max(-A_k^{\text{up}}(i,j), 0).$$

We can write $A_k^{\text{up}} = M_{k,+} - M_{k,-}$ and $A_k^{\text{dual}} = M_{k,+} + M_{k,-}$. In the case of graphs ($k = 0$), the weighted adjacency between any two vertices is always non-negative so that $M_{k,-} = 0$, and $A_0^{\text{up}} = A_0^{\text{dual}}$. We also have $(0+1)\Delta_0 = \Delta$. Therefore, the earlier definitions reduce to the usual definitions of adjacency and degree matrices for graphs.

Algorithm 5: $\mathbf{y} = \mathbf{Cluster}(K, k, d)$

Data: A weighted, oriented simplicial complex K, a dimension k (where $1 \leq k \leq \dim K$), the number of clusters d.

Result: A vector $\mathbf{y}$ of cluster assignments $l \in \{1, 2, \ldots, d\}$ for the k-simplices of K.

Construct matrices A_k^{dual} and Δ_k.
Compute matrix M, where

$$M = \frac{1}{k+1} \Delta_k^{-1/2} A_k^{\text{dual}} \Delta_k^{-1/2}.$$

Construct matrix $X = [u_1 u_2 \cdots u_d] \in \mathbb{R}^{n \times d}$ where u_i's are the eigenvectors corresponding to the d largest eigenvalues of M (chosen to be orthogonal to each other in the case of repeated eigenvalues).
$Y_{ij} = X_{ij} / \left(\sum_j X_{ij}^2 \right)^{1/2}$ (normalize rows of X to have unit length).
$\mathbf{y} = \mathbf{kMeans}(Y, d)$.
Return $\mathbf{y}$ as cluster assignments for k-simplices of K.

Algorithm 6: $\mathbf{y} = \mathbf{PropagateLabels}(K, k, \mathbf{y_l})$

Data: A weighted, oriented simplicial complex K, a dimension k (where $0 \leq k \leq \dim K$), a vector $\mathbf{y_l}$ containing labels $\in \{+1, -1\}$ of first l simplices of K of dimension k.

Result: A vector $\mathbf{y}$ of label assignments $l \in \{+1, -1\}$ for all k-dimensional simplices of K.

Order the k-simplices of K so that labels $\mathbf{y_l}$ correspond to the first l simplices.
Construct matrices A_k^{dual} and Δ_k.
Compute random walk matrix P, where

$$P = \frac{1}{k+1} A_k^{\text{dual}} \Delta_k^{-1}.$$

Initialize $\mathbf{y}^{(0)} = (\mathbf{y}_l, \mathbf{0})$, $t = 0$.
Repeat until convergence:

$$\mathbf{y}^{(t+1)} = P \mathbf{y}^{(t)},$$
$$\mathbf{y}_l^{(t+1)} = \mathbf{y}_l^{(t)}.$$

Return $\text{sgn}(\mathbf{y^{(t)}})$ as label assignments for k-simplices of K.

For $k > 0$, if we can assume that the k-simplices and the $(k+1)$-simplices are oriented in such a way that the oriented adjacency between any two k-simplices is always non-negative, then we get $A_k^{\text{up}} = A_k^{\text{dual}}$. However, we cannot always assume that the simplices are suitably oriented. When $M_{k,-}$ is not all zeros, A_k^{up} and A_k^{dual} have distinct spectra. The relation between these spectra, and consequently, between the spectra of the k-th *up-Laplacian* of K and the graph Laplacian of its dual graph G_k is unclear and left for future work.

2.5.2 Relation to Other Random Walks on Simplicial Complexes

Suppose we normalize the oriented adjacency matrix A_k^{up} to obtain $\tilde{A}_k^{\text{up}}$ as follows:

$$\tilde{A}_k^{\text{up}} = \frac{1}{k+1} A_k^{\text{up}} \Delta_k^{-1}.$$

Let $\tilde{M}_{k,+}$ and $\tilde{M}_{k,-}$ be $n_k \times n_k$ non-negative matrices such that $\tilde{M}_{k,+}(i,j) = \max(\tilde{A}_k^{\text{up}}(i,j), 0)$ and $\tilde{M}_{k,-}(i,j) = \max(-\tilde{A}_k^{\text{up}}(i,j), 0)$, respectively, and $\tilde{M}_{k,+} - \tilde{M}_{k,-} = \tilde{A}_k^{\text{up}}$. The random walk matrix $\tilde{A}_k^{\text{dual}}$ for the dual graph G_k is defined as

$$\tilde{A}_k^{\text{dual}} = \frac{1}{k+1} A_k^{\text{dual}} \Delta_k^{-1}.$$

Also observe that $\tilde{A}_k^{\text{dual}} = \tilde{M}_{k,+} + \tilde{M}_{k,-}$.

Let us fix k and simplify the notation as follows: Let $A_u = \tilde{A}_k^{\text{up}}$, $A_d = \tilde{A}_k^{\text{dual}}$, $M_+ = \tilde{M}_{k,+}$, and $M_- = \tilde{M}_{k,-}$. Therefore, we have

$$A_u = M_+ - M_-,$$

$$A_d = M_+ + M_-.$$

A_d is same as the random walk matrix used in our experiments for label propagation. A different normalization is used in the spectral clustering algorithm to obtain the matrix M, which is different but closely related to A_d. Some authors [90, 97] have proposed using the following random walk matrix P to define a random walk on simplicial complexes:

$$P = \begin{pmatrix} M_+ & M_- \\ M_- & M_+ \end{pmatrix}.$$

The results of these authors rely on the equality of the form: $A_u Q = QP$, where $Q = [I, -I]$ (I is the identity matrix). In the following lemma, we show the relationship between the eigenvectors of A_u, A_d, and P. In particular, we observe that the stationary distributions of the two random walks are closely related.

Lemma 2.8. *Let* $v = \begin{pmatrix} x \\ y \end{pmatrix}$ *be an eigenvector of* $P \in \mathbb{R}^{2n_k \times 2n_k}$ *with eigenvalue* λ*, where* $x, y \in \mathbb{R}^{n_k}$*. Then, either* $x + y$ *or* $x - y$ *is an eigenvector of* $A_d \in \mathbb{R}^{n_k \times n_k}$ *or* $A_u \in \mathbb{R}^{n_k \times n_k}$*, respectively, with the same eigenvalue,* λ*. Conversely, if* u *is an eigenvector of* A_u *with eigenvalue* λ*, then* $\frac{1}{2}\begin{pmatrix} u \\ -u \end{pmatrix}$ *is an eigenvector of* P *with eigenvalue* λ*. Similarly, if* v *is an eigenvector of* A_d *with eigenvalue* μ*, then* $\frac{1}{2}\begin{pmatrix} v \\ v \end{pmatrix}$ *is an eigenvector of* P *with eigenvalue* μ*.*

Proof. To prove the first part, let $v = \begin{pmatrix} x \\ y \end{pmatrix}$ be an eigenvector of $P \in \mathbb{R}^{2n_k \times 2n_k}$ with eigenvalue λ and let $Q = [I, -I]$ where $I \in \mathbb{R}^{n_k \times n_k}$. Then, $A_u Q = QP$ and we get

$$A_u Q \begin{pmatrix} x \\ y \end{pmatrix} = A_u(x - y) = QP \begin{pmatrix} x \\ y \end{pmatrix} = \lambda Q \begin{pmatrix} x \\ y \end{pmatrix} = \lambda(x - y).$$

Therefore, either $(x - y)$ is a zero vector or it is an eigenvector of A_u with eigenvalue λ. Similarly, let $T = [I, I]$. Then $TP = [(M_+ + M_-)\ (M_- + M_+)] = [A_d\ A_d] = A_d T$ and we get

$$A_d T \begin{pmatrix} x \\ y \end{pmatrix} = A_d(x + y) = TP \begin{pmatrix} x \\ y \end{pmatrix} = \lambda T \begin{pmatrix} x \\ y \end{pmatrix} = \lambda(x + y).$$

Therefore, either $(x + y)$ is a zero vector or $(x + y)$ is an eigenvector of A_d with eigenvalue λ. Note that $(x - y)$ and $(x + y)$ cannot both be zero simultaneously.

To prove the second part of the lemma, let u be an eigenvector of A_u with eigenvalue λ, i.e., $A_u u = \lambda u$. Then,

$$P\frac{1}{2}\begin{pmatrix} u \\ -u \end{pmatrix} = \begin{pmatrix} M_+ & M_- \\ M_- & M_+ \end{pmatrix} \frac{1}{2}\begin{pmatrix} u \\ -u \end{pmatrix} = \frac{1}{2}\begin{pmatrix} A_u u \\ -A_u u \end{pmatrix} = \lambda \frac{1}{2}\begin{pmatrix} u \\ -u \end{pmatrix}.$$

Therefore, $\frac{1}{2}\begin{pmatrix} u \\ -u \end{pmatrix}$ is an eigenvector of P with eigenvalue λ. A similar argument shows that if v is an eigenvector of A_d with eigenvalue μ, i.e., $A_d v = \mu v$, then $\frac{1}{2}\begin{pmatrix} v \\ v \end{pmatrix}$ is an eigenvector of P with eigenvalue μ. □

Remark 2.2. *Both* P *and* A_d *are left stochastic matrices. Therefore, all of their eigenvalues belong to the interval* $[-1, 1]$*. Let* v *be a stationary distribution of* A_d*, i.e.,* $v = A_d v$*. By lemma 2.8,* $\frac{1}{2}\begin{pmatrix} v \\ v \end{pmatrix}$ *is a stationary distribution of the random walk defined by* P*.*

Remark 2.3. *Eigenvectors of the form* $\begin{pmatrix} u \\ -u \end{pmatrix}$ *are orthogonal to the eigenvectors of the form* $\begin{pmatrix} v \\ v \end{pmatrix}$*, which would mean that* A_u *and* A_d *are related to subspaces of* P *that are orthogonal*

to each other. However, P can have an eigenvector of the form $v = \begin{pmatrix} x \\ 0 \end{pmatrix}$ with eigenvalue λ. In this case, x must satisfy $M_+x = \lambda x$ and $M_-x = 0$, i.e., x must be an eigenvector of M_+ and x must also belong to the kernel space of M_-. If such a vector x exists, then it is an eigenvector of both A_u and A_d with the same eigenvalue λ.

2.6 Experimental Validation

In Section 2.6.1, we conduct numerical experiments to illustrate the inequalities bounding the spectrum of the up-Laplacian of the sparsified simplicial complex, proven in Theorem 2.1. In Section 2.6.2, we show the results for spectral clustering and label propagation on simplicial complexes before and after sparsification. We show that the results obtained for sparsified simplicial complexes are similar to those of the original simplicial complex. In both cases, we also include the analogous results for graphs to help illustrate our results on simplicial complexes in a more familiar context.

2.6.1 Preservation of the Spectrum of the Up-Laplacian

Our sparsification algorithm for simplicial complexes is an extension of the sparsification algorithm for graphs. Therefore, we begin with a review of graph sparsification [117]. Recall that if a graph H is an ε-approximation of a graph G and n is the number of vertices in H and G, then we have the following inequality:

$$(1-\varepsilon)x^T L_G x \leq x^T L_H x \leq (1+\varepsilon)x^T L_G x, \qquad \forall x \in \mathbb{R}^n. \tag{2.9}$$

Subtracting $x^T L_G x$ from all terms in this inequality, we obtain

$$-\varepsilon x^T L_G x \leq x^T (L_H - L_G) x \leq \varepsilon x^T (L_G) x, \qquad \forall x \in \mathbb{R}^n. \tag{2.10}$$

Let $\lambda_{max}(L_G)$, $\lambda_{max}(L_H)$ and $\lambda_{max}(L_H - L_G)$ be the maximum eigenvalues of L_G and L_H and $L_H - L_G$, respectively. Also, let $\lambda_{min}(L_G)$ be the minimum eigenvalue of L_G. Looking at the inequality on the right-hand side of (Equation 2.10), after some algebraic manipulations, we obtain

$$\lambda_{max}(L_H - L_G) = \max_{||x||=1} x^T (L_H - L_G) x \leq \varepsilon \max_{||x||=1} x^T (L_G) x = \varepsilon \lambda_{max}(L_G).$$

Similarly, for the inequality on the left-hand side of (Equation 2.10), we obtain

$$0 = -\varepsilon\lambda_{min}(L_G) = -\varepsilon \min_{||x||=1} x^T L_G x = \max_{||x||=1} -\varepsilon x^T L_G x$$
$$\leq \max_{||x||=1} x^T(L_H - L_G)x = \lambda_{max}(L_H - L_G).$$

Together, we have the inequality

$$0 \ \leq \ \lambda_{max}(L_H - L_G) \ \leq \ \varepsilon\lambda_{max}(L_G). \tag{2.11}$$

Moving from graphs to simplicial complexes, we can obtain the analogous inequality in the setting of simplicial complex sparsification. Let J be a sparsified version of K following the setting of Theorem 2.1. Suppose for a fixed dimension k (where $1 \leq i \leq \dim K$), K and J have $(k-1)$-th up-Laplacians $\mathcal{L}_K := \mathcal{L}_{K,k-1}$ and $\mathcal{L}_J := \mathcal{L}_{J,k-1}$, respectively, we have

$$(1-\varepsilon)x^T\mathcal{L}_K x \leq x^T\mathcal{L}_J x \leq (1+\varepsilon)x^T\mathcal{L}_K x, \qquad \forall x \in \mathbb{R}^{n_{k-1}}. \tag{2.12}$$

A similar argument leads to the following inequality:

$$0 \ \leq \ \lambda_{max}(\mathcal{L}_J - \mathcal{L}_K) \ \leq \ \varepsilon\lambda_{max}(\mathcal{L}_K). \tag{2.13}$$

Notice that inequality Equation 2.11 is a special case of the inequality Equation 2.13.

2.6.1.1 Preservation of the spectrum of the sparsified graph Laplacian. To demonstrate how the spectrum of the graph Laplacian is preserved during graph sparsification, we set up the following experiments. Note that graph sparsification of large graphs is well known; the results described here are used only for comparative purposes. In particular, we would like to give a simple example to compare similar behaviors in preserving the spectrum of up-Laplacian for both graphs and simplicial complexes.

Consider a complete graph G with $n_0 = 40$ vertices and $n_1 = 780$ edges. We run multiple sparsification processes on this graph G and study the convergence behavior based on the inequality in Equation 2.9. For each sparsification process, we use a sequence of sample sizes, ranging between 10 and $2n_1$. For each sample size q, we set $\varepsilon = \sqrt{n_0 \log n_0/q}$ by assuming that $9C^2 = 1$ in the hypothesis of Theorem 2.1. As q varies, we correspondingly obtain a sequence of varying ε values.

In particular, we run 25 simulations on G. For each simulation, we fix a unit vector x uniformly randomly sampled from $\mathbb{S}^{n_0}$ and perform 25 instances of experiments. For each

instance, we apply our sparsification procedure to generate the convergence plot using the list of fixed sample sizes q and their corresponding ε's. Specifically, for each sample size, we obtain a sparse graph H and compute $x^T L_H x$ and $\lambda_{max}(L_H - L_G)$, and we observe the convergence behavior of these quantities as the sample size increases.

In Figure 2.2, we show the convergence behavior based on the inequality in Equation 2.9. For a single simulation, we compute the pointwise average of $x^T L_H x$ across the 25 instances, and we plot these values as a function of the sample size q, which gives rise to a single convergence curve in aqua. Then, we compute the pointwise average of the aqua curves across all simulations, producing the red curve. Since each simulation (for a fixed x) has a different upper bound curve $(1-\varepsilon)x^T L_G x$ and lower bound curve $(1+\varepsilon)x^T L_G x$, respectively (not shown here), the pointwise average of the upper and lower bound curves across all simulations is plotted in blue. We observe that, on average, these curves respect the inequality Equation 2.9, that is, the red curve is nested within its approximated theoretical upper and lower bounds in blue.

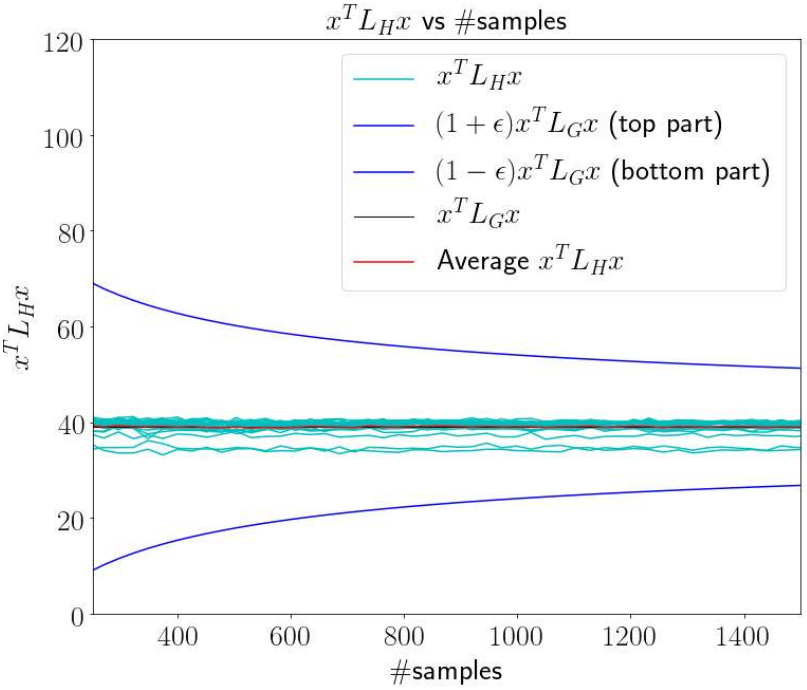

FIGURE. 2.2. The result of a numerical experiment illustrating inequalities that control the spectrum of sparsified graph Laplacians. For an ensemble of vectors, $x \in \mathbb{S}^{n_0}$, and sparsified graphs, H, we plot the terms in inequality Equation 2.9.

In Figure 2.3, we illustrate the theoretical upper and lower bounds for $\lambda_{max}(L_H - L_G)$ given in Equation 2.11 as the sample size q increases. In particular, we run a single simulation with 25 instances, computing $\lambda_{max}(L_H - L_G)$. Each instance gives us a convergence curve shown in aqua. We compare the pointwise average of $\lambda_{max}(L_H - L_G)$ (in red) with its (approximated) theoretical upper bound in blue and lower bound (i.e., 0, the x-axis). On average, the experimental results respect the inequality Equation 2.11. Figure 2.4 illustrates how the number of edges increases with the number of samples across all instances.

2.6.1.2 Preservation of the spectrum of the up-Laplacian for a sparsified simplicial complex. To demonstrate that the spectrum of the up-Laplacian is preserved during the sparsification of a simplicial complex, we set up a similar experiment. We start with a two-dimensional simplicial complex, K, that contains all edges and triangles on $n_0 = 40$ vertices (with $n_1 = 780$ edges and $n_2 = 9880$ faces) and a sequence of fixed sample sizes q. For each sample size q, we solve for $\varepsilon = \sqrt{n_1 \log n_1 / q}$ assuming that $9C^2 = 1$ in the hypothesis of Theorem 2.1, to get the corresponding sequence of ε values. With the simplicial complex K and the sequence of sample sizes fixed, we run 25 simulations, each

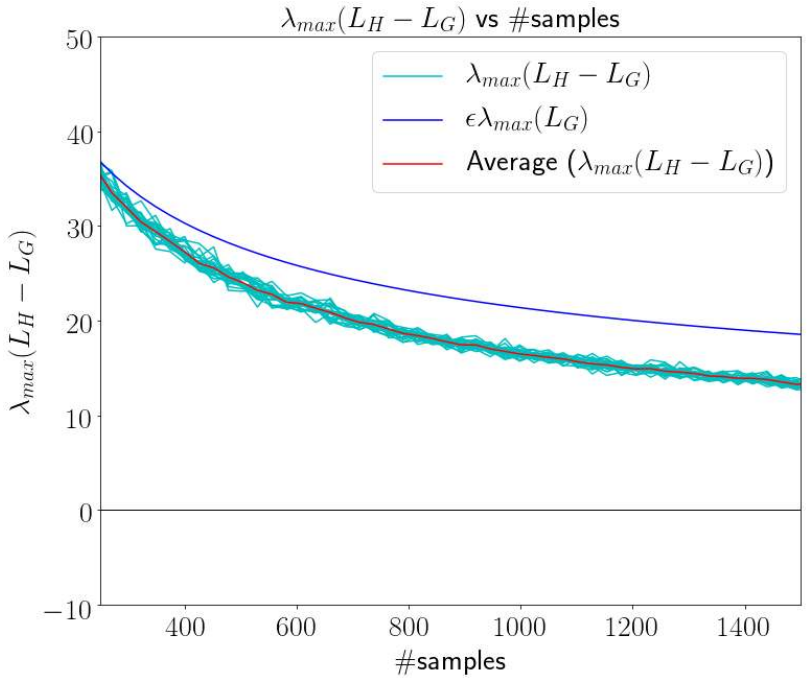

FIGURE. 2.3. The result of a numerical experiment illustrating inequalities that control the spectrum of sparsified graph Laplacians. For an ensemble of sparsified graphs, H, we plot the terms in the inequality Equation 2.11.

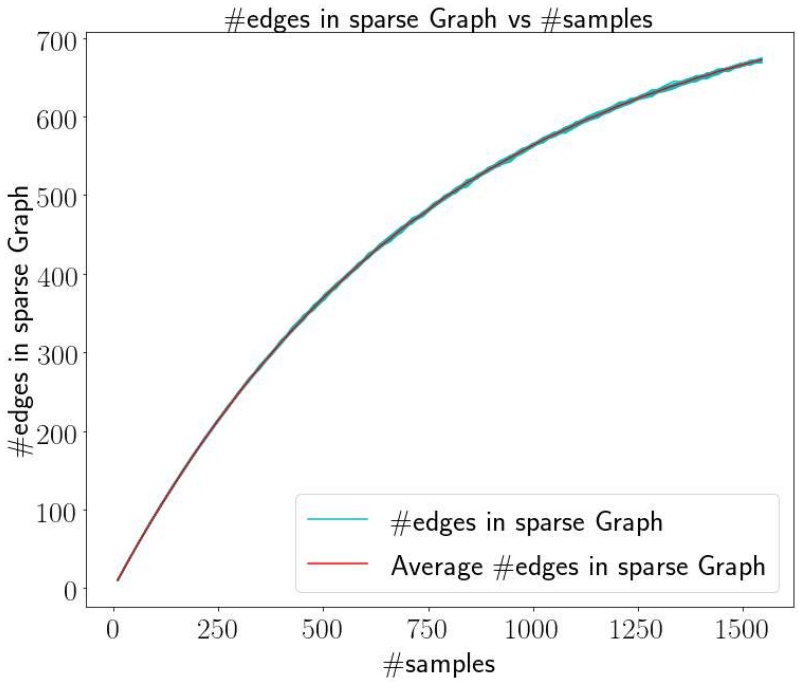

FIGURE. 2.4. Number of edges in the sparse graph as a function of sample size in the case of graph sparsification.

simulation consisting 25 instances and a fixed randomly sampled unit vector x as described previously. This time, however, we sparsify the faces of the simplicial complex by applying Algorithm 4 with $k = 2$. In Figure 2.5 and Figure 2.6, we plot the terms in inequalities describing the spectrum for these sparsified simplicial complexes. In Figure 2.5, following the same procedure as for graph sparsification, we obtain a plot that respects the Equation 2.12. The curves in aqua show the pointwise average of $x^T \mathcal{L}_J x$ across all instances in a single simulation, whereas the red curve represents pointwise average across all instances and all simulations. Since the random vector x is resampled for each simulation, the upper and lower bound curves are different for every simulation. In Figure 2.5, we plot their pointwise average across all simulations as the upper and lower bound curves in blue.

In Figure 2.6, to illustrate Equation 2.13, we run a single simulation with 25 instances. Each instance gives us a sequence of $\lambda_{max}(\mathcal{L}_J - \mathcal{L}_K)$ values as a function of sample size. We plot them as curves in aqua. We compare the pointwise averages of $\lambda_{max}(\mathcal{L}_J - \mathcal{L}_K)$ (in red) with its (approximated) theoretical upper and lower bounds in blue. Figure 2.7 shows how the number of faces scales with the increasing number of samples across all instances.

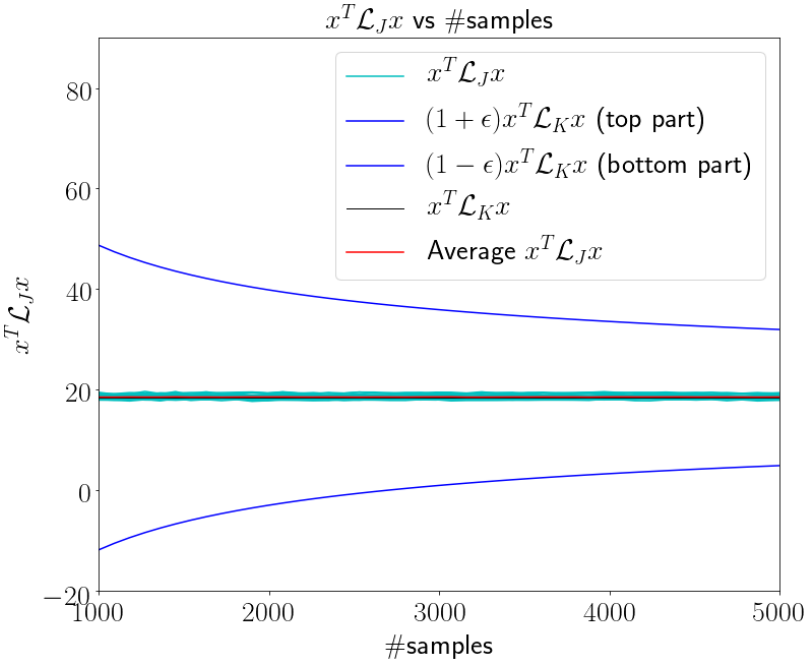

FIGURE. 2.5. The result of a numerical experiment illustrating inequalities that control the spectrum of the up-Laplacian for sparsified simplicial complexes. For an ensemble of vectors, $x \in \mathbb{S}^{n_1}$, and sparsified simplicial complexes, J, we plot the terms in inequality Equation 2.12.

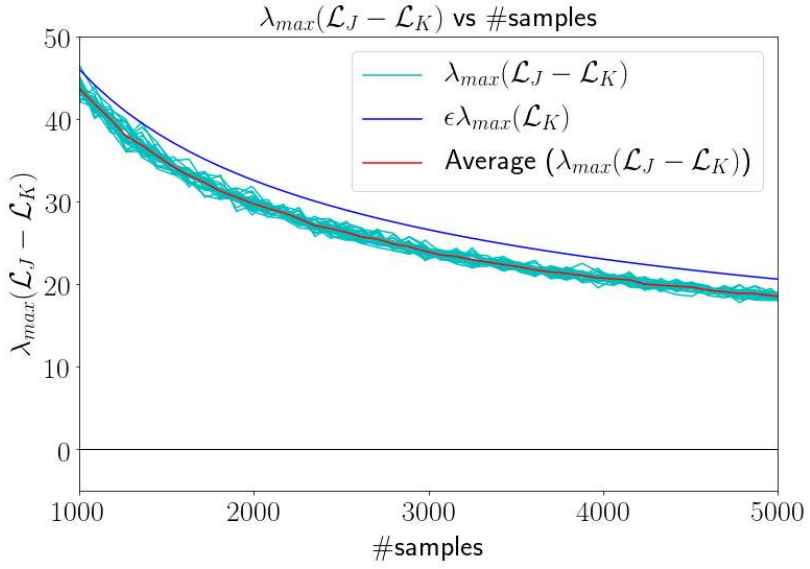

FIGURE. 2.6. The result of a numerical experiment illustrating inequalities that control the spectrum of the up-Laplacian for sparsified simplicial complexes. For an ensemble of sparsified simplicial complexes, J, we plot the terms in the inequality Equation 2.13.

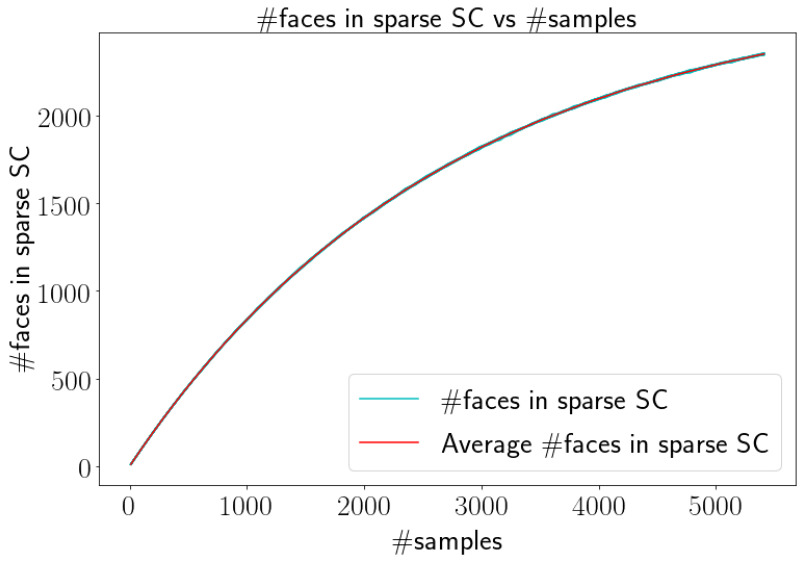

FIGURE. 2.7. Number of faces/triangles in the sparse simplicial complex as a function of sample size in the case of simplicial complex sparsification.

2.6.2 Spectral Clustering and Label Propagation for Simplicial Complexes

Here, we apply spectral clustering and label propagation to simplicial complexes before and after sparsification. We demonstrate, via numerical experiments, that preserving the structure of the up-Laplacian via sparsification also preserves the results of these two spectral algorithms on simplicial complexes.

2.6.2.1 Datasets. For comparative purposes, we consider a graph that contains two complete subgraphs, with 20 vertices (and 190 edges) each, that are connected by $64 = 8 \times 8$ edges spanning across the two subgraphs. We refer to this graph, G, as the *dumbbell graph*; it has $n_0 = 40$ vertices and $n_1 = 444$ edges. All edge weights are set to be 1. To compute the sparsified graph, the number of samples, q, is set to be $0.5n_1$.

Similarly, we consider a simplicial complex that contains two complete subcomplexes with 10 vertices, 45 edges, and 120 triangles each. The two subcomplexes are connected by 16 cross edges and 48 cross triangles so that the simplicial complex is made up of $n_0 = 20$ vertices, $n_1 = 106$ edges, and $n_2 = 288$ triangles. We refer to this simplicial complex, K, as the *dumbbell complex*. The weights on all edges and triangles are set to be 1. To compute the sparsified simplicial complex, the number of samples, q, is set to be $0.75n_2$. We compare

the result of spectral clustering on the dumbbell graph to the result on a dumbbell complex.

2.6.2.2 Spectral clustering for simplicial complexes. We first illustrate the spectral clustering results before and after graph sparsification in Figure 2.8 (a)-(b). Since graph sparsification preserves the spectral properties of graph Laplacian, we expect it to also preserve (to some extent) the results of spectral methods, such as spectral clustering.

To apply spectral clustering to higher order simplices, we follow the dual graph approach proposed in Section 2.5.1. Specifically, we modify the spectral clustering algorithm for graphs (Algorithm 1) by replacing the vertex-vertex affinity matrix A and the degree matrix D with an edge-edge affinity matrix A_1^{dual} and the degree matrix $2\Delta_1$, respectively. To illustrate our edge clustering results, we visualize the resulting clusters in the dual graph. The results are plotted in Figure 2.8 (c)-(d) for two clusters and Figure 2.9 for three clusters.

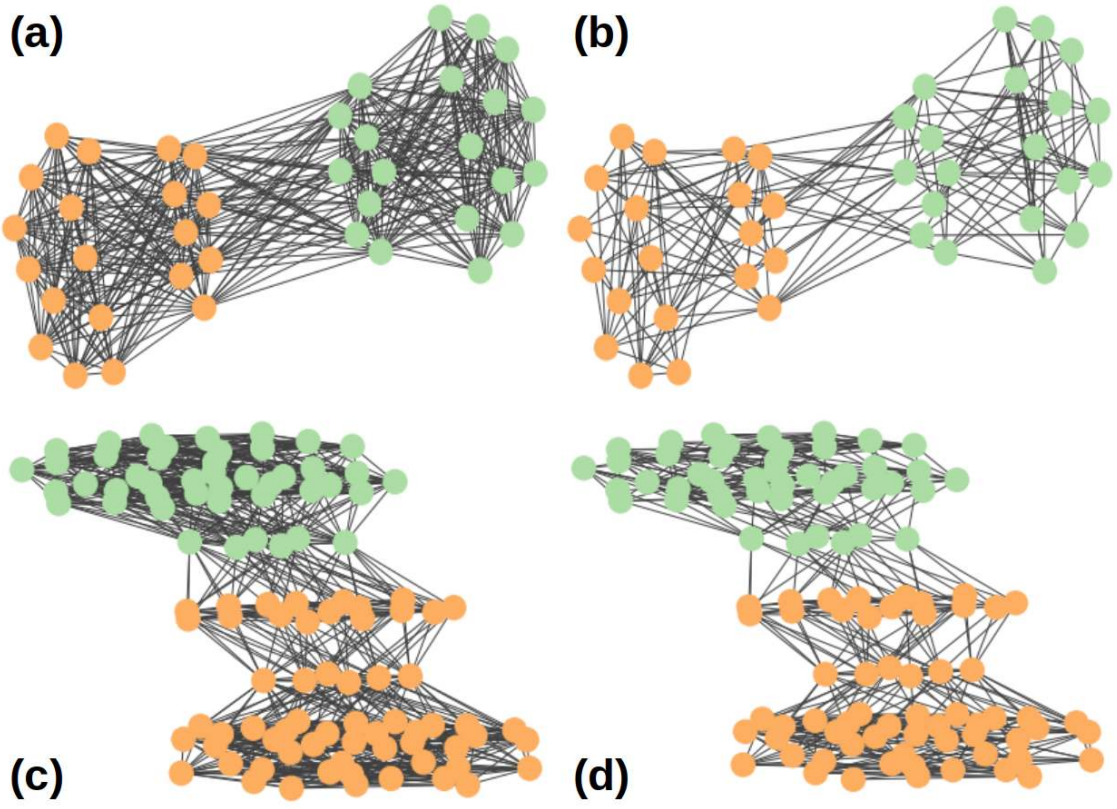

FIGURE. 2.8. Spectral clustering of graphs and simplicial complexes. **(a)-(b)**: Spectral clustering of graphs before and after sparsification. **(c)-(d)**: Spectral clustering of simplicial complexes into two clusters. In both cases, we observe that the clusters are similar before and after sparsification.

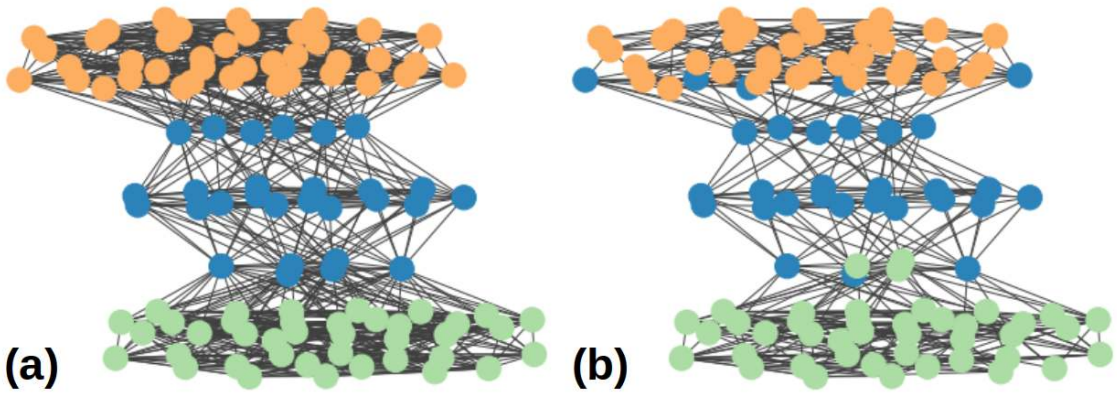

FIGURE. 2.9. Spectral clustering of simplicial complexes into three clusters before **(a)** and after **(b)** sparsification.

We observe that our modified spectral clustering algorithm results in clusters that agree reasonably well before and after sparsification.

2.6.2.3 Label propagation on simplicial complexes. We envision label propagation to be generalized to random walks on higher dimensional simplices, such as edges and triangles. A direct application of our work is to sparsify the top-dimensional simplices (e.g., triangles in a two-dimensional simplicial complex) and examine how label propagation behaves on these top-dimensional simplices of the sparsified representation.

First, we apply label propagation (Algorithm 2) to the dumbbell graph dataset to demonstrate (Figure 2.10) that preserving the structure of graph Laplacian via sparsification also preserves the results of label propagation on graphs. Then, we apply the modifed version (Algorithm 6) to the dumbell complex and show (Figure 2.11) via the dual graph representation that the results obtained from sparsified simplicial complexes are similar to those of the original simplicial complex. Figure 2.12 shows a few more instances of label propagation on the dumbbell complex with different initial labels.

2.7 Discussion

We have presented a spectral sparsification algorithm for simplicial complexes that preserves the spectral properties of the up-Laplacian. Our work is strongly motivated by the study of an emerging class of learning algorithms based on simplicial complexes and those spectral algorithms that operate with higher order Laplacians. We would like to

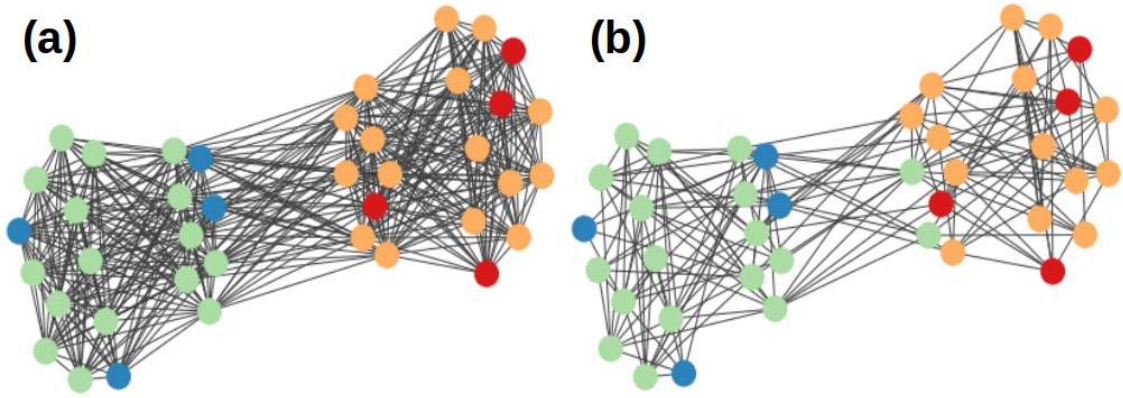

FIGURE. 2.10. The results of label propagation on the dumbbell graph: **(a)** before sparsification and **(b)** after sparsification. The red (+1) and blue (-1) represent the given vertex labels, and the orange (+1) and green (-1) correspond to the propagated vertex labels.

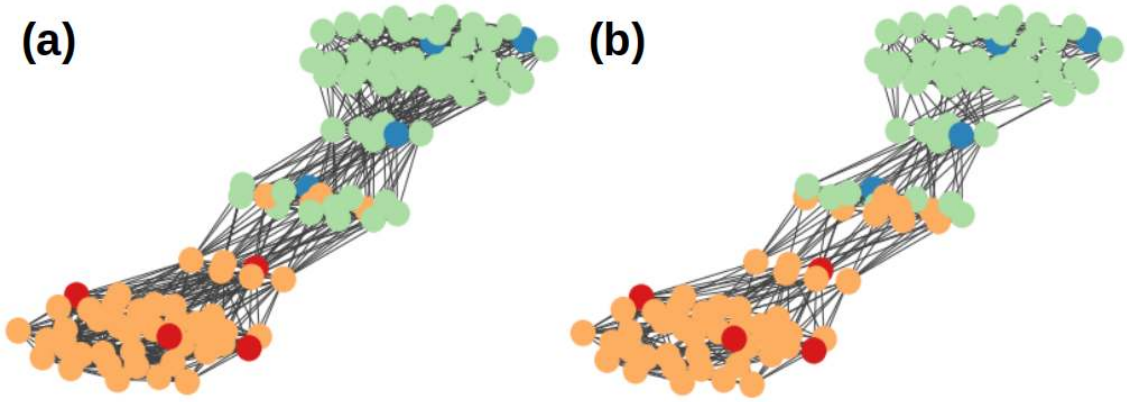

FIGURE. 2.11. The results of label propagation on simplicial complexes: **(a)** before sparsification and **(b)** after sparsification. The red (+1) and blue (-1) colored vertices correspond to given edge labels, and the orange (+1) and green (-1) colored vertices correspond to propagated edge labels.

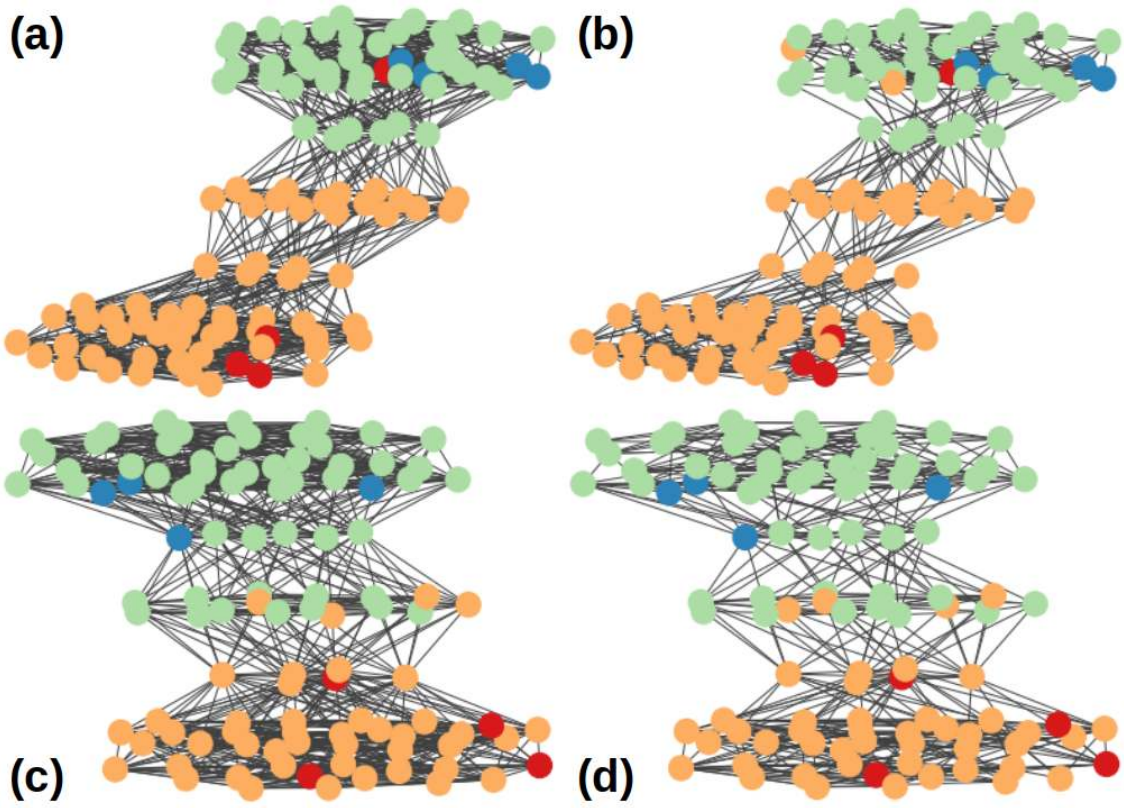

FIGURE. 2.12. More instances of label propagation on the edges of the dumbbell shaped simplicial complex: **(a), (c)** before sparsification and **(b), (d)** after sparsification.

understand the benefits and incurred error when such learning algorithms are applied to sketches of the data. In this section, we discuss some of the crucial challenges and future research directions, and then discuss possible real-world applications.

2.7.1 Open Problems

First, we discuss two open problems with the work presented in this chapter. We generalize the notion of effective resistances to higher dimensional simplices. However, the physical interpretation of the generalized effective resistances remains an open question. We have proposed an algorithm to sparsify simplicial complexes by their generalized effective resistances, but we do not yet have an efficient implementation.

2.7.1.1 Challenges for efficient implementation. To compute generalized effective resistances of i-simplices of K exactly, we need to solve linear systems involving the up-Laplacian. The best solution is to compute a QR or SVD decomposition of the scaled

incidence matrix $W_i^{1/2} D_{i-1}$, which could be done in $O(n_i \cdot n_{i-1}^2)$ time, where n_i is the number of i-simplices and n_{i-1} is the number of $(i-1)$-simplices of K.

Spielman and Srivastava [117] gave an algorithm that can approximate effective resistances and produce the sparse graph in $O(m \log(r)/\epsilon^2)$ time, where m is the number of edges, and r is the ratio of the largest to the smallest edge weights. The key to their algorithm was an efficient symmetric diagonally dominant (SDD) linear system solver [120] that approximately solves the linear systems involving the graph Laplacian in $\tilde{O}(m \log(1/\delta))$, where δ is an error parameter. Recent SDD solvers, using graph-based preconditioners such as low-stretch spanning trees, have improved the running time even further [64, 35].

However, the up-Laplacians $\mathcal{L}_{K,i}$ for $i \geq 1$ are not diagonally dominant matrices. Therefore, these fast SDD solvers may not be applied directly to approximate generalized effective resistance. Although solving linear systems of higher order Laplacians has been studied for limited classes of complexes [34, 38, 69], no such solvers exist for up-Laplacians of arbitrary (nongeometric) simplicial complexes.

Generalizations of spanning trees to higher dimensions may be useful in constructing fast solvers for up-Laplacians of arbitrary simplicial complexes. Alternatively, we can think of sparsification using generalized effective resistance as a form of leverage score sampling. Computation of exact leverage scores has the same time complexity as the computation of exact generalized effective resistance. However, fast algorithms to compute constant factor approximations to leverage scores [43, 32] exist, which in theory, can run in $o(n_i \cdot n_{i-1}^2)$ time. However, further analysis is required before we can apply any of these approaches to approximate generalized effective resistances and make claims about the runtime of such implementations.

2.7.1.2 Physical interpretation of generalized effective resistance. We believe the generalization of effective resistance to simplicial complexes, introduced Section 2.3, may find other applications in analyzing simplicial complexes. Although the generalization is algebraically straightforward, its interpretation and properties pose many natural and exciting questions. For example, does it have an interpretation in terms of a random process, such as effective commute time analogous to the case of a graph (see, e.g., [49])? Is it related to the minimum spanning objects in the simplicial complex? Does it play a further role in the spectral clustering of simplicial complexes?

2.7.2 Future Work

We have proposed spectral algorithms for an arbitrary but fixed, dimension k of a simplicial complex in this work. A natural extension would be to develop algorithms that operate on simplices of several dimensions simultaneously.

2.7.2.1 Multidimensional random walks. To propagate labels on k-simplices of a simplicial complex K, our label propagation algorithm (Algorithm 6) utilizes a random walk on the dual graph G_k. The k-simplices correspond to the nodes of G_k, and their adjacency is determined by the shared $(k+1)$-dimensional co-faces. We can think of this random walk as a composition of two transitions: first from a k-simplex to one of its $(k+1)$-dimensional co-faces, and then from the $(k+1)$-simplex to one of its k-dimensional faces.

However, given a simplicial complex K of dimension $\dim K$, we can define a graph $G^*(V^*, E^*)$, where we represent each simplex of K as a vertex in G^*. The vertex set V^* is partitioned into $\dim K + 1$ partitions such that all k-simplices of K are mapped to vertices in the k^{th} partition of V^*. The edges of G^* represent incidence relations, i.e., a vertex representing a k-simplex σ_i is only connected to vertices representing its $(k-1)$-dimensional faces and $(k+1)$-dimensional co-faces. We illustrate this idea in Figure 2.13, where a two-dimensional simplicial complex is embedded into a three-partite graph. Now, we can define a random walks on elements of k^{th} partition of G^* as excursions from the k^{th} partition to the $(k+j)^{th}$ (or the $(k-j)^{th}$) partition and back. These random walks spanning multiple dimensions can represent more complex stochastic processes. We are interested in studying the properties of such random walks, their physical interpretations, and applications in real-world problems such as information propagation.

The random walks described so far in this chapter are defined for a fixed dimension k. Although the transition probabilities may be defined in terms of shared $(k-j)$-dimensional faces or $(k+j)$-dimensional cofaces, the random walker can transition from one k-simplex only to another k-simplex. However, we can imagine a random walk on all the vertices of the graph embedding of K, for example, the vertices of the three-partite graph in Figure 2.13. Although defining such a random walk is computationally straightforward, its physical interpretation is difficult since the vertices in different partitions represent simplices of different dimensions. Therefore, it cannot be treated uniformly.

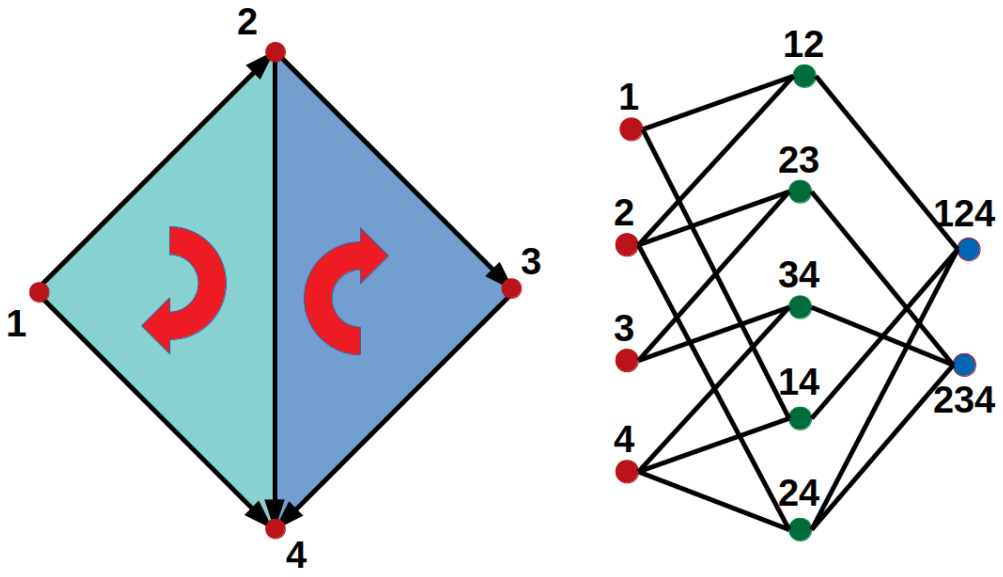

FIGURE. 2.13. A two-dimensional simplicial complex (left) and its embedding into a three-partite graph (right).

2.7.2.2 Multilevel and Hodge sparsification. For sparsification at a fixed dimension k, our sparsification algorithm guarantees that the spectrum of $\mathcal{L}^{\text{up}}_{k-1}(K)$ will be approximately preserved. However, when we remove a k-simplex from K, we also implicitly remove all its higher dimensional cofaces. Consequently, all the up-Laplacians for dimensions $\geq k$ will also change after sparsification. We do not yet have a way to analyze the extent of this change. Ideally, we would like to show that the differences between up-Laplacians of higher dimensions are also bounded.

We are also interested in performing multilevel sparsification of simplicial complexes; for example, we would like to sparsify triangles and edges simultaneously while preserving spectral properties of the dimension-0 and dimension-1 up-Laplacians. Multilevel sparsification is also related to preserving the spectral properties of the Hodge Laplacian. However, such sparsification is challenging because we would like to maintain the structure of simplicial complexes simultaneously. Multidimensional random walks mentioned earlier may also prove useful in this context, in which we use them to sample simplices. We may also be able to relax our structural constraints to work with hypergraphs instead. Finally, we are also interested in deriving formal connections between homological sparsification and

spectral sparsification of simplicial complexes.

2.7.3 Applications

We can use a simplicial complex to model higher order interactions between three or more objects, which graphs fail to capture. The spectra of the corresponding higher order Laplacians capture the topology and geometry of the interaction patterns, which are essential in many applications.

For instance, in sensor networks, three sensors with overlapping coverage areas are modeled as a triangle; four sensors form a tetrahedron, and so on, thereby forming a simplicial complex over the set of sensors. Higher order Laplacians of this simplicial complex can be used to localize holes in coverage and determine redundancies and sparsest coverages [124].

In the case of social networks, users interact with other users individually as well as in groups. Pairwise interactions are not sufficient to capture the different interaction patterns arising from the group settings. We can model such interactions as simplicial complexes. Researchers have now begun to explore the utility of higher order simplices in analysis tasks such as community detection [66] and clustering [10].

Random walks on graphs have been used to significant effect in many applications, for instance, to study human mobility patterns in a transportation network [100] or to study surfing behavior in the world wide web [86, 29, 12]. In random walks on a graph, the walker's next step depends only on the current state. However, modeling the random walker's past states in addition to the current and the next state can be beneficial in the aforementioned applications. For example, consider studying human mobility patterns in air travel [107]. A journey from A to B may consist of multiple layovers. Instead of encoding each leg of the journey independently, we may want to encode the entire journey as one. In [107], the authors modeled such journeys using *memory nodes*. However, we can model such travel patterns as simplices of a simplicial complex. Such encoding also allows us to differentiate between a layover and an actual visit. Higher order random walks on such a simplicial complex can be used to understand the complex dynamics of air travel. In the case of the world wide web, we can represent the sequence of web pages visited by the user as a simplex. Then, we can rank the web pages based on the higher order topological structures in the

resulting simplicial complex [15, 54].

Simplicial complexes allow us to encode higher order patterns in transportation networks or the world wide web, and analyze human mobility or user's web surfing behavior in a more comprehensive way. However, the simplicial complexes arising from these applications can be large and dense. The spectral sparsification algorithm proposed in Section 2.3.1 can be applied to reduce the size of these complexes and make learning on them more feasible. In a more direct application, the sparsification algorithm can be used to determine the sparsest coverage in a sensor coverage network.

2.7.4 Conclusion

The applications described above illustrate the advantages of using simplicial complexes to model intricate interaction patterns in the data. There is a growing interest in developing methods that operate directly on simplicial complexes, leveraging the structural information they encode. In this chapter, we present spectral algorithms to sparsify simplicial complexes and perform unsupervised and semisupervised learning tasks. We also provide methods to study random processes on simplicial complexes. We hope to extend these methods to hypergraphs and develop a unified framework for spectral methods on data modeled as graphs (undirected and directed), simplicial complexes, and hypergraphs.

CHAPTER 3

LEARNING WITH TOPOLOGICAL FEATURES OF NETWORKS

In this chapter, we work with datasets where individual samples are modeled as networks. We apply techniques from TDA to extract topological features from these networks and use them to perform various learning tasks. Our goal is to study structural and functional brain abnormalities in autism spectrum disorders (ASD). We model structural or functional relationships between brain regions as networks, use persistent homology to extract their topological features, and use these features to perform tasks like statistical inference, regression, and classification.

In Section 3.1, we give an introduction and describe our motivation. In Section 3.2, we describe the construction of different brain networks. We also give a brief description of persistent homology, and describe different ways to represent topological features. Persistent homology is a core technique from topological data analysis (TDA) that allows us to summarize the topological features of the brain networks. In Section 3.3, we describe how to interface these topological features with various learning tasks. In Section 3.4, Section 3.5, and Section 3.6, we present applications of these techniques to study the structural and functional brain network abnormalities in autism. Lastly, in Section 3.7, we discuss the implications of the presented methods and possible directions for future research.

3.1 Introduction

Autism is a complex developmental disorder characterized by impairment in social interactions, difficulty in verbal and nonverbal communication, and repetitive behaviors. Although the exact mechanism of its development remains unclear, there is evidence relating autism to abnormal functional and structural connectivity between brain regions. In this chapter, we apply persistent homology, a core TDA technique, to extract topological features of structural and functional brain connectivity networks and explore ways to utilize them in

statistical inference, autism classification, and prediction of cognitive and behavioral scores.

3.1.1 Structural and Functional Brain Connectivity

We can identify the structural abnormalities in the brain using voxel-based morphometry by comparing gray matter and white matter volumes or densities and using cortical thickness and their respective growth trajectories [112, 135] across diagnostic groups. Although the gross brain differences have been well documented [37], investigations into specific regional abnormalities in brain structure have reported conflicting results [122].

Structural covariance MRI (scMRI) maps regions of gray matter that have a statistically significant correlation with a specific seed region of interest (ROI) across subjects, capturing the shared developmental or genetic influences between the gray matter region and the seed ROI [134]. Seeley et al. [113] have used scMRI to demonstrate that specific adult dementias affect distinct *intrinsic connectivity networks* (ICNs) and the corresponding gray matter regions. Using a similar technique, Zielinski et al. [133] have shown that network-specific structural differences between autism and control groups are consistent with clinical aspects of the disease and that reported functional abnormalities in autism may have a structural basis. Several recent studies have applied the scMRI technique to find evidence of network-specific structural abnormalities in other diseases such as Alzheimer's [88] and Huntington's [87].

Functional MRI scans produce a time series of signals, called BOLD (blood oxygenation level-dependent) signals, representing the level of activity in gray matter regions of the brain. Functional connectivity networks constructed from the fMRI time series capture the level of synchronicity between distinct spatial ROIs of the brain across time. Such networks have shown promising ability to explain impairments in functional connectivity of the brain in neuropsychiatric disorders such as autism [1, 11]. Several models that attempt to classify subjects with autism from control subjects, based on functional connectivity networks, have been proposed in the literature [2, 94]. Advances in artificial neural networks (NN) have opened up a new line of research in autism. Various NN models [4, 57, 59] have been proposed for classification of autistic subjects in recent years. To our knowledge, the classifier proposed by Heinsfeld et al. [59] has the best classification accuracy at 70.2% on the *Autism Brain Imaging Data Exchange* (ABIDE) dataset [20, 39].

3.1.2 Network Comparison and TDA

Network comparison and machine learning with networks as input samples are not easy problems to solve. Graph-theoretic measures have been proposed previously to compare networks [19, 108]. However, a significant drawback of these measures is their reliance on a fixed topology. These measures are typically based on a binary graph obtained by thresholding the connectivity matrix. The choice of a threshold is crucial in such analyses. Different heuristics have been suggested to determine the threshold depending on which properties of the network are of interest. However, it is often not possible to determine a single optimal threshold.

TDA [21, 50] of networks goes beyond graph-theoretic analysis by utilizing tools from computational topology to describe the architecture of networks in more flexible ways. In particular, it encodes higher order (not just pairwise) interactions in the system and studies the brain network's topological features across all possible thresholds. Persistent homology, a core technique in TDA, is often used to study complex networks such as collaboration [24] and brain networks [72]. Topological methods have shown promise in modeling transitions between brain states in functional imaging data using combined information in space and time [111].

3.1.3 Contributions

In this chapter, we describe applications of persistent homology to study abnormalities in structural and functional brain connectivity in autism. In particular,

- We propose a novel statistical inference procedure based on topological signatures derived from the structural brain networks. We apply this procedure to investigate topological differences in the gray matter structure of the brain. Our results provide evidence of statistically significant structural abnormalities in autism.

- We propose kernel partial least squares (kPLS) regression to statistically quantify the relationship between topological features of functional brain networks and behavioral phenotypes in autism. We show that combining correlations with topological features gives a better prediction of autism severity than using correlations alone.

- We explore the utility of topological features in the classification of ASD versus

typically developing control subjects. Once again, we show that combining topological features with functional correlations typically leads to better classification accuracy. However, our experiments also show that the improvement in classification accuracy due to topological features is not always statistically significant. Therefore, we offer a cautionary tale regarding the limited discriminative power of topological features derived from fMRI data for the classification of autism.

3.2 Background

Brain networks are the primary objects of interest in this chapter. We utilize the topological features of the brain networks to perform tasks such as statistical inference, classification, and regression on a given set of brain networks. In this section, we will describe how to construct different forms of brain networks and how to compute their topological features.

3.2.1 Brain Networks

Networks are used to encode pairwise relationships among points in a dataset. The brain networks we consider in this chapter are modeled as fully connected, weighted, undirected graphs. We consider two types of graphs, structural correlation graphs (SCGs) and functional correlation graphs (FCGs). In both cases, the nodes represent different regions of interest (ROIs) in the brain with an associated vector or a signal. The edge weights are given by the Pearson correlation coefficients between signals from pairs of nodes. Alternatively, these graphs can also be represented as correlation matrices. We give a short description of how to construct each graph below.

3.2.1.1 Structural correlation graphs. We construct structural correlations graphs (SCGs) from standard clinical MRIs of a group of subjects. These graphs capture the shared structural influences or structural similarities between pairs of ROIs across the group. The nodes of an SCG represent gray matter ROIs in the brain. With each region, we associate a signal formed by the gray matter densities at that location in the MRI scan across all subjects in the group. We construct the SCG by obtaining signals for all ROIs and computing Pearson correlation coefficients between signals from pairs of ROI. Figure 3.1 illustrates the construction of SCGs.

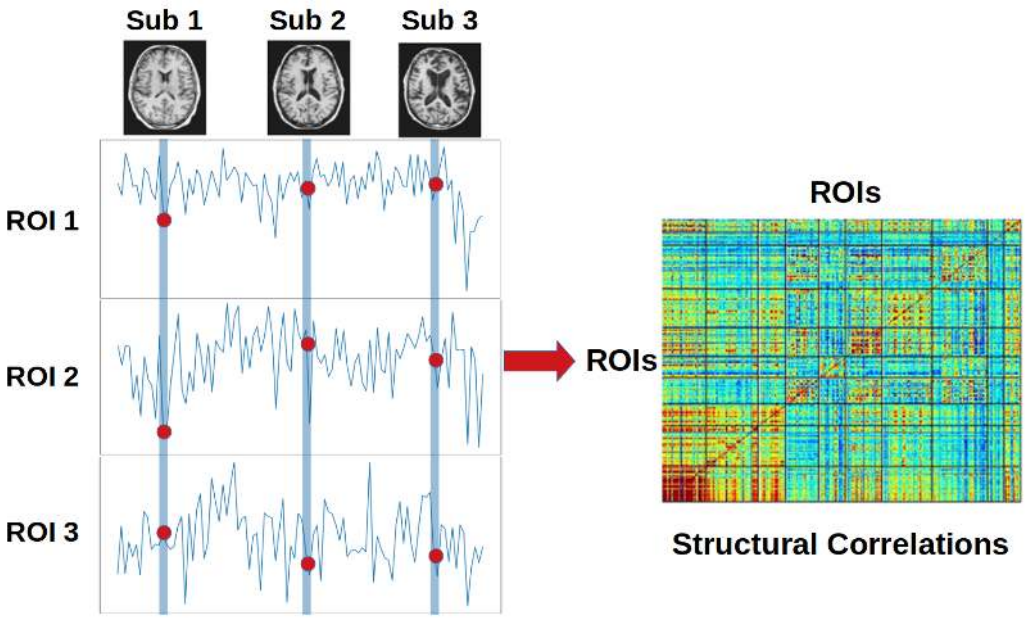

FIGURE. 3.1. Structural correlation graphs (SCGs) encode pairwise correlations between gray matter ROI intensities across a group of subjects.

We use structural covariance MRI (scMRI) to identify a set of ROIs. scMRI is a technique that maps gray matter ROIs that have a statistically significant correlation with a specific seed region across subjects, suggesting shared developmental or genetic influences between the gray matter ROIs and the seed region [134]. We use this technique to identify ROIs underlying a specific *intrinsic connectivity network* (ICN), according to methods published previously [134]. First, we determine a seed ROI known to anchor a specific ICN, and then we implement a generalized linear model (GLM) to identify regions that have covarying gray matter densities with that seed ROI across subjects.

Specifically, for a given seed ROI, we perform separate condition-by-covariate analysis for each voxel with the mean seed gray matter density as the covariate of interest and the disease status as the grouping variable. We include total brain volume (TBV) and age as covariates of no interest. This design enables us to determine the whole-brain patterns of seed-based structural covariance in a group of subjects. We perform one-sample t-tests with family-wise error correction to identify regions with significant gray matter density covariance with the seed ROI across subjects in a diagnostic group. Once the ROIs are identified based on their structural relationship with a specific seed region, we model the

structural relationships between all pairs of ROIs as the SCG. The vertices of the SCG represent gray matter ROIs, and the magnitudes of the Pearson correlation coefficients give the edge weights.

3.2.1.2 Functional correlation graphs. We construct the functional correlation graphs from resting-state functional magnetic resonance imaging (rs-fMRI) scans. Throughout the scan, the rs-fMRI measures the fluctuations in the blood oxygenation level in the brain's gray matter regions. These fluctuations, which represent the level of activity in the brain region, are called the blood-oxygenation-level dependent (BOLD) signals. After processing the raw scans to remove noise, motion, alignment, and other artifacts, the brains are mapped to a standard template on which the ROIs are defined. Given a predetermined set of ROIs, we obtain the corresponding BOLD signals in the form of a time series and arrange them in a data matrix where rows are indexed by ROIs, and columns are indexed by time points. FCGs capture the level of synchronicity between distinct spatial locations across time. To construct FCGs, we compute the Pearson correlation coefficients for all pairs of rows in the data matrix. Figure 3.2 illustrates the construction of FCGs.

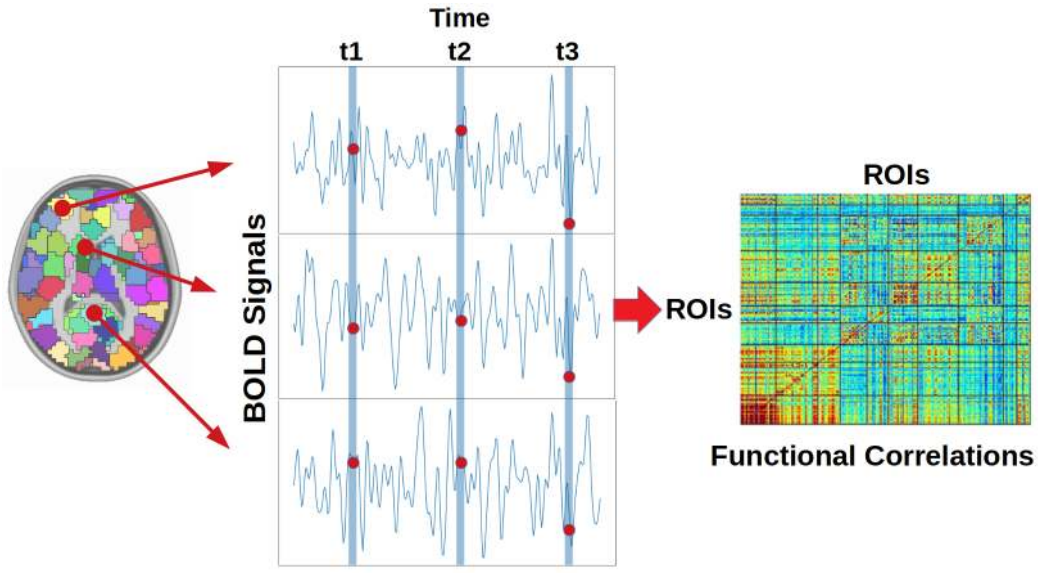

FIGURE. 3.2. Functional correlation graphs (FCGs) encode correlations across time between BOLD signals from pairs of ROIs.

3.2.2 Persistent Homology

Persistent homology [44] measures how the topological features of data evolve across a varying scale parameter α, where connected components, tunnels, and voids are considered as 0-, 1-, and 2-dimensional features. Here we will illustrate the main idea behind persistent homology and its different representations through a simple example. For a detailed description, we refer readers to the seminal work on the topic by Edelsbrunner et al. [45], and to excellent surveys by Carlsson [21] and Ghrist [50].

3.2.2.1 Persistence barcodes and persistence diagrams. In a typical setting, we begin with a point cloud $\mathbb{X} \in \mathbb{R}^d$ in a metric space, denoted as $(\mathbb{X}, d_{\mathbb{X}})$. For some $\alpha \geq 0$, the union of balls of radius α centered at $x \in \mathbb{X}$ under the metric $d_{\mathbb{X}}$ forms a topological space. As α increases, the union of balls undergoes topological changes, where topological features appear and disappear. The radius α is often viewed as a time parameter. Persistent homology associates a life span (i.e., death time minus birth time), the *persistence*, to the topological features.

Figure 3.3 shows an example, where $\mathbb{X}$ is a point cloud in $\mathbb{R}^2$ and $d_{\mathbb{X}}$ is the Euclidean distance metric. As the radius α increases, we keep track of the topological changes in the union of balls. At time $\alpha = 0$, each colored point is *born* (appears) as an independent connected component. At $\alpha = 2.5$, the green component merges into the red component and *dies* (disappears). Therefore, the green component has a persistence of 2.5. At $\alpha = 3$, the orange component merges into the pink component and dies. Hence, it has a persistence of 3. Similarly, the blue component dies at $\alpha = 3.2$ and the pink component dies at $\alpha = 3.7$. At time $\alpha = 4.2$, the collection of components forms a tunnel that has a persistence of 1.4. This tunnel disappears at $\alpha = 5.6$. The red component born at time 0 never dies, and thus it has a persistence of ∞.

The topological information of this process can be computed using simplicial complexes (Figure 3.3(b)) and summarized either as a persistence barcode (Figure 3.3(c)) or a *persistence diagram* (PD, Figure 3.3(d)). A persistence diagram is a finite multiset of points in the plane. Each point of the diagram corresponds to a topological feature, and its coordinates (b, d) specify at which scales the feature appears (birth time b) and disappears (death time d). For example, the green component (a 0-dimensional feature) is mapped to a green point $(0, 2.5)$ in the diagram given its birth time $b = 0$ and death time $d = 2.5$;

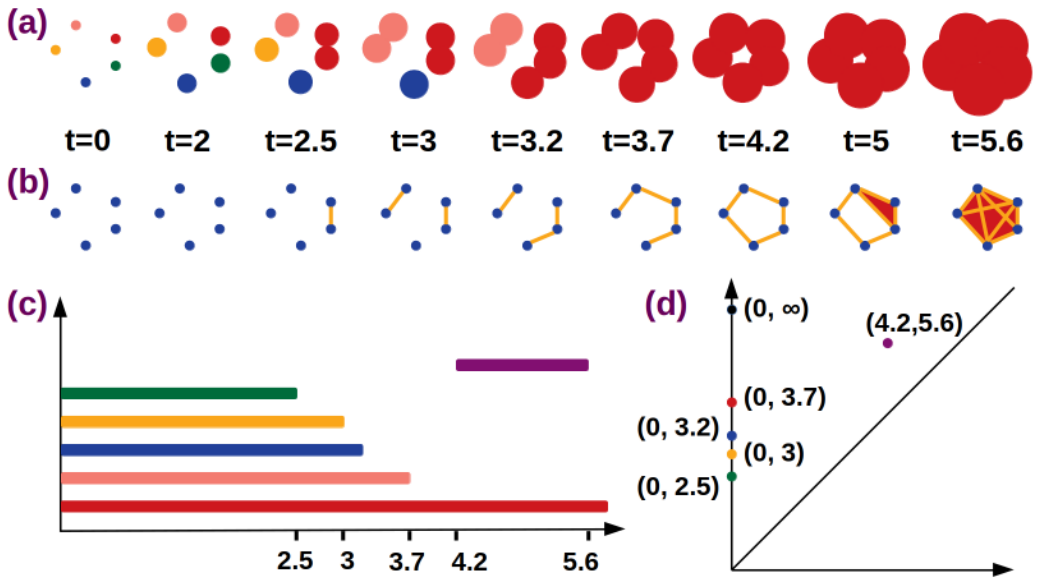

FIGURE. 3.3. Computing persistent homology of a point cloud in $\mathbb{R}^2$. **(a)** A nested sequence of topological spaces formed by unions of balls at increasing parameter values. **(b)** A sequence of simplicial complexes that captures the same topological information as in **(a)**. **(c)** Persistence barcode. **(d)** 0- and 1-dimensional persistence diagrams (the purple point) combined.

and the tunnel (a 1-dimensional feature) is represented by a purple point $(4.2, 5.6)$ in the diagram. A persistence barcode is an equivalent representation in which each topological feature appears as a horizontal bar that starts at the scale at which the feature appears (b) and ends at the scale at which the feature disappears (d).

3.2.2.2 Persistence landscapes and persistence images. Persistence landscapes (PL) [18] transform a persistence diagram into a sequence of piecewise linear functions. Informally, these functions are obtained by first changing the coordinates of points (b, d) to $((b+d)/2, (d-b)/2)$, and then placing tent functions (isosceles triangles) anchored at each point in the transformed space such that their base is on the x axis and the length of the base equals the persistence of the point. The envelopes of the union of the tent functions form the piecewise linear functions that are then sampled uniformly to obtain a discrete vector representation. See Figure 3.4(b) for an example, where functions λ_1 and λ_2 are the piecewise linear functions that form the PL.

A persistence image (PI) [3] is another way of transforming a persistence diagram into a vector representation. Informally, a PI is a *heatmap*. First, the coordinates of the points

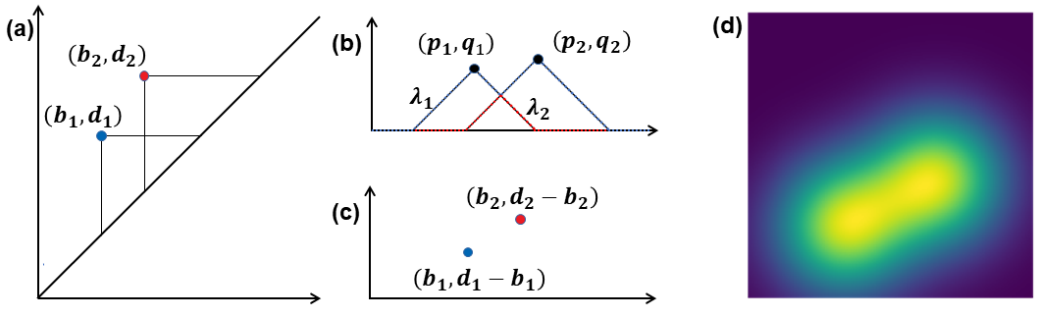

FIGURE. 3.4. Different representations of topological features: **(a)**: 1-dimensional persistence diagram. **(a)**: The persistence landscapes of 1-dimensional features of order 1 (λ_1, in blue) and order 2 (λ_2, in red). **(c)-(d)**: Transformation of the persistence diagram and the persistence image of the 1-dimensional features.

in the persistence diagram are changed to (b, p), where b is birth and $p = d - b$ is the persistence of the feature. Then the PI is constructed as a square grid of pixels, where the pixel's intensity is given by the weighted sum of Gaussians centered at the points in the transformed space. Figure 3.4(c-d) illustrates the transformation of the 1-dimensional persistence diagram in Figure 3.4(a) and the corresponding persistence image, respectively.

3.2.2.3 Persistent homology of correlation graphs. To apply these persistent homology computations to a structural or functional correlation graph, we first need to transform the graph into a point cloud in a metric space. Recall that these graphs have an equivalent representation as correlation matrices. We use the distance

$$d(x, y) = \sqrt{1 - \mathrm{Cor}(x, y)}$$

to define distances between the vertices x, y of the graph, where $\mathrm{Cor}(x, y)$ is the correlation, i.e., the weight of the edge between x and y. Using this distance as the filtration parameter, we can compute the persistent homology of the correlation graph. Note that we need only the pairwise distances between the vertices for persistent homology computation. We do not need to compute embedding of the vertices into a Euclidean space explicitly.

3.2.3 Graph Filtration

Another way to extract topological features at multiple scales from a correlation graph G is to construct a nested sequence of graphs from G, referred to as the graph filtration. Let $V = \{v_i \mid i = 1, \ldots, n\}$ be the vertex set of G with n vertices. Let E denote the edge

set and the edge weights be given by $w : E \to \mathbb{R}^+$. The edge between vertices v_i, v_j is denoted by e_{ij} and its weight is denoted by w_{ij}. $|E|$ denotes the number of edges. For a given threshold λ, we obtain a binary graph G_λ by removing edges with weight $w_{ij} \leq \lambda$. The adjacency matrix $A_\lambda = (a_{ij}(\lambda))$ is given by

$$a_{ij}(\lambda) = \begin{cases} 0 & w_{ij} \leq \lambda, \\ 1 & otherwise. \end{cases}$$

As λ increases, more and more edges are removed from the graph. We can generate a sequence of thresholds in ascending order, $\lambda_0 = 0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_q$, where $q \leq |E|$. Corresponding to the sequence of thresholds, we get a nested sequence of binary graphs, referred to as a *graph filtration* $\mathbf{G}$ [73]:

$$G_{\lambda_0} \supseteq G_{\lambda_1} \supseteq G_{\lambda_2} \supseteq \cdots \supseteq G_{\lambda_q}.$$

We can measure the connectivity of a graph by its 0^{th} Betti number, β_0, which is the number of connected components in the graph. As the threshold λ increases, $\beta_0(G_\lambda)$ of the corresponding graph also increases. The number of connected components of the graphs in filtration $\mathbf{G}$ form a monotonic sequence of integers,

$$\beta_0(G_{\lambda_0}) \leq \beta_0(G_{\lambda_1}) \leq \beta_0(G_{\lambda_2}) \leq \cdots \leq \beta_0(G_{\lambda_q}).$$

Suppose we start with a connected graph $G = G_{\lambda_0}$. We have $\beta_0(G_{\lambda_0}) = 1$ and $\beta_0(G_{\lambda_q}) = |V| = n$ by construction. The plot of $\beta_0(G_\lambda)$ as a function of threshold λ is called the β_0 curve. We illustrate this idea with a simple example in Figure 3.5.

A finite graph with n vertices can have at most $\binom{n}{2}$ distinct edge weights. If we choose the set of all the unique edge weights, sorted in ascending order, to be the thresholds, then with finitely many threshold values, we can estimate the β_0 curve for all λ. Computing the β_0 curve for a given graph could follow the standard algorithm for persistent homology [45]. In practice, we use a simpler algorithm to capture the λ values since we are concerned only with tracking the number of components (clusters) during the filtration. This algorithm relies on the fact that the thresholds λ_i at which the number of connected components $\beta_0(G_{\lambda_i})$ changes are precisely the weights of the edges in the maximal spanning tree of the graph. Therefore, to derive the β_0 curve, we need to compute only the maximal spanning tree of G.

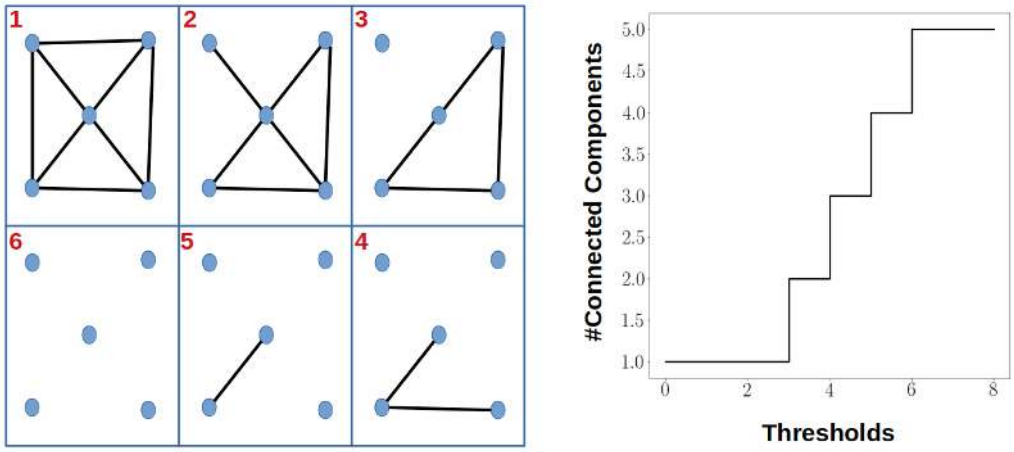

FIGURE. 3.5. An example of graph filtration. On the left is a sequence of graphs (clockwise) corresponding to an increasing sequence of thresholds. On the right, we plot the β_0 curve: the number of connected components as a function of thresholds.

3.3 Methods

Topological features such as β_0 curves and persistence diagrams are novel ways to summarize the shape of the data. In this section, we will describe ways to utilize these features in various learning tasks.

3.3.1 Statistical Inference with Structural Correlation Graphs

Suppose our data consist of clinical MRIs of subjects divided into two diagnostic groups (samples), autism (ASD) and typically developing controls (TDC). We want to test whether these two samples come from the same underlying distribution. Specifically, we want to test whether there are any statistically meaningful differences in the 0-dimensional topology (connectivity) of the SCGs derived from the two samples. We do this by examining the equivalence of the corresponding β_0 curves.

Let G and H represent the SCGs obtained from autism and control samples, respectively, with corresponding graph filtrations $\mathbf{G}$ and $\mathbf{H}$. We want to test the null hypothesis,

$$H_0 : \beta_0(G_\lambda) = \beta_0(H_\lambda) \quad \text{for all } \lambda,$$

against the alternative,

$$H_1 : \beta_0(G_\lambda) \neq \beta_0(H_\lambda) \quad \text{for some } \lambda.$$

Since we are dealing with finite graphs, a discrete version of the null hypothesis is stated as

$$H_0 : \beta_0(G_{\lambda_i}) = \beta_0(H_{\lambda_i}) \quad \text{for all } \lambda_i,\ i = 1, 2, \ldots, q,$$

with the alternative being

$$H_1 : \beta_0(G_{\lambda_i}) \neq \beta_0(H_{\lambda_i}) \quad \text{for some } \lambda_i,\ i = 1, 2, \ldots, q.$$

The distance between two graph filtrations $\mathbf{G}$ and $\mathbf{H}$ with respect to the corresponding β_0 curves can be defined as

$$D_q(\mathbf{G}, \mathbf{H}) = \sup_{0 \leq i \leq q} |\beta_0(G_{\lambda_i}) - \beta_0(H_{\lambda_i})|. \tag{3.1}$$

Intuitively, D_q measures the largest gap between the two β_0 curves. The p-value is the probability that D_q will take a value equal to or greater than the observed value under the null hypothesis. In order to determine this p-value, we need the distribution of D_q under the null hypothesis.

3.3.1.1 Permutation test. A permutation test provides a simple way to estimate the distribution of D_q under the null hypothesis. Let D_q^* denote the value computed from the two original samples. To estimate the sampling distribution of D_q, in each iteration, we pool subjects from both samples, randomly permute subject group labels in the pooled data, and form two new samples. We construct SCGs for these two samples separately and apply graph filtration to both SCGs to obtain the corresponding β_0 curves. Finally, we compute the distance D_q between the two curves. Thus, each permutation gives us a new value of D_q. The p-value is given by the fraction of D_q values greater than or equal to D_q^*.

An exact permutation test would require computing D_q for all possible permutations of the samples, which is computationally infeasible. Instead, we perform the test on a random subset of all possible permutations. This random permutation test is not exact. However, with enough permutations, it can closely approximate the exact test. The p-value $\hat{p}$ obtained is an estimate of the true underlying p-value p. For each permutation, the test $(D_q \geq D_q^*)$ is a Bernoulli trial with probability of success p. When permutations are independently sampled from a uniform distribution, $\hat{p}$ is an unbiased estimator of p. The standard error of $\hat{p}$ can be approximated by $\sqrt{\hat{p}(1-\hat{p})/N}$, where N is the number of permutations performed.

3.3.1.2 Bootstrap. Permutation tests are widely used to test hypotheses when the underlying distribution is unknown. The only underlying assumption is that the samples are exchangeable under the null distribution. However, in complex cases like ours, even that assumption may be too strong or too difficult to verify. Bootstrap tests are commonly used to obtain confidence intervals, and they generally have low power compared to permutation tests. However, they also make much weaker assumptions about the underlying distribution. Here, we will implement a version of bootstrap to estimate the sampling distribution of D_q. We once again combine both samples to create one pool of subjects. Then, we generate two new samples (same size as the original) by sampling subjects *with replacement* from this pool and ignoring their original group labels. With these new bootstrap samples, we proceed to compute the SCGs, the β_0 curves, and the distance D_q in the same way as the permutation test. We compute the p-value as the fraction of D_q values obtained from the bootstrap samples that are greater than or equal to D_q^* obtained from the original sample. Note that the only difference between the two tests implemented here is whether we re-sample from the pooled data with (bootstrap) or without (permutation) replacement.

3.3.2 Kernels for Persistence Diagrams

Classical machine learning algorithms are often designed to work with data in vector form. Persistence diagrams, however, are multisets of points that do not have a natural vector representation, i.e., the space of persistence diagrams is not an inner product space (Hilbert space). Therefore, persistence diagrams cannot be used directly in classical machine learning. There are two possible solutions to this problem: to apply transformations such as persistence landscapes or persistence images that admit vector representations, or to define an inner product (kernel) on the space of persistence diagrams and use the kernelized versions of learning algorithms. Here, we will describe four kernels for persistence diagrams proposed in the recent literature.

In what follows, we will denote persistence diagrams by capital letters (A, B, D, etc.). A persistence diagram A is the disjoint union of a multiset $\{A_i\}_{i=0}^{N_{max}}$ and the diagonal set $\Delta = \{(a, a) \mid a \in \mathbb{R}_+\}$ counted with infinite multiplicity. N_{max} is the largest dimension of the topological features included the persistence diagram. All points in the diagonal set have 0 persistence. All points in $\{A_i\}_{i=0}^{N_{max}}$ have finite persistence except one point in

A_0, which is not included in any kernel computations. Each set A_i, which is the collection of points corresponding to i-dimensional features, is also referred to as the i-dimensional persistence diagram.

3.3.2.1 Persistence scale-space kernel. For two i-dimensional persistence diagrams A_i and B_i, the persistence scale-space kernel [105] K_S is defined as

$$K_S(A_i, B_i, \sigma) = \frac{1}{8\pi\sigma} \sum_{p \in A_i, q \in B_i} e^{\frac{\|p-q\|}{8\sigma}} - e^{\frac{\|p-\overline{q}\|}{8\sigma}}, \tag{3.2}$$

where $\forall\ q = (a, b) \in B_i$, $\overline{q} = (b, a)$. Here, σ is a tuneable kernel parameter.

3.3.2.2 Persistence-weighted Gaussian kernel. For two i-dimensional persistence diagrams A_i and B_i, the persistence-weighted Gaussian kernel [67] K_G is defined as

$$K_G(A_i, B_i, \sigma) = \sum_{p \in A_i, q \in B_i} w(p)w(q)e^{-\frac{\|p-q\|^2}{2\sigma^2}}, \tag{3.3}$$

where $w(p)$ is the weight assigned to the point p. Kusano et al. [67] suggested using $w(p) = w_{arc}(p) = \arctan(C(d-b)^t)$ as the weight for $p = (b, d)$, where C is a positive constant for practical purposes, and t is assumed to be greater than the dimension of underlying space. Here, σ is the tuneable kernel parameter.

3.3.2.3 Sliced Wasserstein kernel. Let A_i and B_i be two i-dimensional persistence diagrams. Given a unit vector θ in $\mathbb{R}^2$, let $L(\theta) = \{\lambda\theta \mid \lambda \in \mathbb{R}\}$ denote the line and $\pi(\theta, p)$ denote the orthogonal projection of point p on the line $L(\theta)$. To compute the sliced Wasserstein kernel [23], we first augment the persistence diagram A_i with the orthogonal projection π_Δ of points in B_i and vice versa to obtain two new sets A_i^* and B_i^*. The sliced Wasserstein distance between these two sets is approximated as

$$SW(A_i^*, B_i^*, M) = \frac{1}{\pi} \sum_{j=1}^{M} \|V(A_i^*, \theta_j) - V(B_i^*, \theta_j)\|_1. \tag{3.4}$$

Here, $\theta_j = j\pi/M - \pi/2$ are the sampled directions, M is the user-defined number of directions, and $V(A_i^*, \theta_j)$ is the vector of dot products $< p, \theta_j >$ of all points $p \in A_i^*$. The sliced Wasserstein kernel [23] is then computed as

$$K_W(A_i^*, B_i^*, M) = e^{\frac{-SW(A_i^*, B_i^*, M)}{2\sigma^2}}. \tag{3.5}$$

Here, σ is the tuneable kernel parameter.

3.3.2.4 Persistence Fisher kernel. Let A_i and B_i be two i-dimensional persistence diagrams with A_i^* and B_i^* defined as in the case the sliced Wasserstein kernel. Given a bandwidth $\sigma > 0$, and a set of points Θ, we can define a smooth and normalized measure on A_i as [71]

$$\rho_{A_i} := \left[\frac{1}{Z} \sum_{u \in A_i} \mathrm{N}(x; u, \sigma I) \right]_{x \in \Theta},$$

where $Z = \int_\Theta \sum_{u \in A_i} \mathrm{N}(x; u, \sigma I)$ is a normalizing constant, N is a Gaussian function, and I is the identity matrix. Therefore, each persistence diagram can be regarded as a point in a probability simplex $\mathbb{P} := \{\rho | \int \rho(x) \mathrm{d}x = 1, \rho(x) \geq 0\}$. The Fisher information metric between two points, $\rho_i, \rho_j \in \mathbb{P}$, is defined as

$$d_{\mathbb{P}}(\rho_i, \rho_j) = \arccos \int \sqrt{\rho_i(x)\rho_j(x)} \mathrm{d}x.$$

Le and Yamada [71] defined the Fisher information metric between two persistence diagrams A_i and B_j as

$$d_{FIM}(A_i, B_j) = d_{\mathbb{P}}(\rho_{A_i^*}, \rho_{B_i^*}),$$

and the persistence Fisher kernel is defined as

$$K_F(A_i, B_i) = e^{-t d_{FIM}(A_i, B_i)},$$

where t is a positive scalar and a tuneable kernel parameter.

3.3.3 Regression with Persistence Diagrams

Partial least squares regression (PLS) [84] is a dimensionality reduction technique closely related to principal component analysis (PCA). It reduces the data X and the target Y simultaneously in such a way that the covariance between their projections is maximized. This simultaneous reduction facilitates a better regression fit between X and Y compared to principal component regression. In our case, the data X are the functional connectivity features and the targets Y are the behavioral measures. For PLS regression using persistence diagrams as features, we use the kernelized version of the PLS algorithm (kPLS) proposed by Rosipal and Trejo [106].

We assume that data X are mapped to an inner product space $\mathcal{X}$ by a mapping Φ. Let K be the inner product matrix, such that $K(i, j) = \langle \Phi(x_i), \Phi(x_j) \rangle_{\mathcal{X}}$ for features $x_i, x_j \in X$.

The kPLS algorithm initializes a random unit vector v and repeats the following steps until convergence: (1) $u = Kv$, (2) $u \leftarrow u/\|u\|$, (3) $v = YY^T u$, and finally, (4) $v \leftarrow v/\|v\|$. To find additional latent vectors, we first set $K \leftarrow (I - uu^T)K(I - uu^T)$, removing the variation in the data in direction of u. The regression equation for kPLS is given by $\hat{Y} = KV(U^T KV)^{-1}U^T Y$, where U and V have vectors u and v as columns.

3.3.4 Neural Networks with Persistence Diagrams

To train a neural network with persistence diagrams, we can use the approach proposed by Hofer et al. [60], where a projection layer is defined as a set of nodes of the neural network that takes a persistence diagram as input and outputs an n-dimensional vector. We denote the i-dimensional persistence diagram as D_i and a point in the diagram $(b, d) \in D_i$. Let $\boldsymbol{\mu} = [\mu_x, \mu_y]$ and $\sigma = [\sigma_x, \sigma_y]$ denote the location (mean) and scale (bandwidth) of a Gaussian distribution in $\mathbb{R}^2$. We define $s_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu}$ as follows:

$$s_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu}(x, y) = \begin{cases} e^{-\sigma_x^2(x-\mu_x)^2 - \sigma_y^2(y-\mu_y)^2} & y \in [\nu, \infty) \\ e^{-\sigma_x^2(x-\mu_x)^2 - \sigma_y^2(\ln \frac{y}{\nu} + \nu - \mu_y)^2} & y \in (0, \nu) \\ 0, & y = 0. \end{cases}$$

Here, ν is a cutoff parameter for handling points with persistence (the difference between birth and death) close to zero. The projection of the i-dimensional persistence diagram D_i w.r.t to $s_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu}$ is

$$S_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu} = \sum_{(x,y)\in\rho(D_i)} s_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu}(x, y),$$

where $\rho(D_i)$ denotes the set of points obtained from D_i after a change of coordinates such that each point $(b, d) \in D_i$ is mapped to a point $(x, y) = (b + d, d - b) \in \rho(D_i)$. Note that $S_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu}$ maps D_i to a scalar value. Now, suppose $\theta = \{(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)\}_{i=1}^n$ is a set of location and scale parameters for n structure elements. The projection layer $\mathcal{S}_{\theta,\nu}$ for D_i is composed of n nodes, where each node corresponds to one structure element $(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i)$ and outputs the projection $S_{\boldsymbol{\mu}_i,\boldsymbol{\sigma}_i,\nu}$ of D_i. Each of the n nodes outputs one scalar value, and these values are concatenated to form the output vector. Note that projection layers are defined independently for each dimension of the persistence diagram. Hofer et al. [60] showed that the function $s_{\boldsymbol{\mu},\boldsymbol{\sigma},\nu}(x, y)$ is stable with respect to the Wasserstein and the bottleneck distance, and differentiable with respect to the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, so that gradient descent can be applied to train the neural network.

3.4 Statistical Inference with Structural Networks

In this section, we apply the statistical inference procedure for structural correlation graphs (SCGs), described in Section 3.3.1, to study abnormalities in the brain network architecture. Specifically, we investigate topological differences in the gray matter structure captured by SCGs derived from three intrinsic connectivity networks strongly implicated in autism, namely, the salience network (SN), the default mode network (DMN), and the executive control network (ECN).

3.4.1 Contributions

Using structural covariance maps, Zielinski et al. [133] have shown the existence of multidimensional, structure-function relationships. The analysis described in this section helps summarize these relationships using a robust topological data analysis model. The SCGs encode all pairwise associations among the ROIs, where the extent of an association is measured by the magnitude of correlations across subjects. Our results confirm the significant differences in structural covariance in autism, which is consistent with the findings of [133].

In particular, our experiments provide evidence of statistically significant differences in the 0-dimensional topological features of SCGs derived from SN (SN-SCGs), and show decreased structural covariance among individuals with autism in the integration of frontal lobe regions with SN hubs in the frontoinsula. Our findings of decreased integration of salience and executive networks, with increased integration of default regions within the frontal lobe, align with results investigating functionally defined intrinsic connectivity networks [1] and suggest that shared developmental influences may underlie the particular specificity of SN connectivity abnormalities in autism [125].

3.4.2 Methods

We compare 49 male subjects with autism (ASD), aged 3-22 years, to 49 age-, gender-, and IQ-matched typically developing control subjects (TDC). The groupwise mean age (standard deviation) is 13.27 (5.07) for ASD subjects and 13.67 (5.53) for TDC subjects. Images are acquired using a Siemens 3.0 Tesla MRI scanner. Whole-brain isotropic MPRAGE image volumes are acquired in the sagittal plane using an 8-channel receive-only RF head

coil, employing standard techniques (TR = 2300 ms, TE median = 3 ms, matrix median = $256 \times 256 \times 160$, flip angle = 12°, voxel resolution = 1 mm^3, acquisition time = 9 min 12 sec).

3.4.2.1 Data preprocessing. We derive structural correlation graphs (SCGs) from the intrinsic connectivity networks (ICNs) previously reported by Zielinski et al. [133, 134]. Customized image analysis templates are created by normalizing, segmenting, and averaging T1 images using the *Statistical Parametric Mapping* (SPM8) software according to the processing pipeline proposed in [5, 129]. First, the images are transformed into standard space using a 12-parameter affine-only linear transformation and segmented into three tissue classes representing gray matter, white matter, and cerebrospinal fluid. Then, smoothly varying intensity changes, as well as artifactual intensity alterations resulting from the normalization step, are corrected for using a standard modulation algorithm within SPM8. Finally, the resulting segmented maps are smoothed using a 12-mm full-width at half-maximum Gaussian kernel.

In the scMRI analysis, a two-pass procedure is utilized, wherein study-specific templates are first created by segmenting our sample using a canonical pediatric template. Then, tissue-specific prior probability maps are created from our sample. The tissue compartments are then resegmented using this sample-specific template so that the age range of our sample precisely matches that of the template(s) upon which the ultimate segmentations are based.

3.4.2.2 Structural correlation graphs and statistical inference. We want to construct SCGs that capture structural relationships, across subjects, between all pairs of gray matter regions, from a predefined set of regions. We begin by constructing a *whole brain* SCG as follows: 1-mm spheres are placed at grid points of a uniform grid on the entire preprocessed image volume. After applying the gray matter mask, we obtain a set of 7266 regions. The *whole-brain* SCG (denoted Global-SCG) is constructed by computing correlations, across subjects, between all pairs of these regions.

To study network-specific structural relationships, 4-mm-radius spherical seed ROIs are selected within the right frontoinsular cortex (R FI) [113], the right dorsolateral prefrontal cortex (R DLPC) [114], and the right posterior cingulate cortex (R PCC) [46]. These regions anchor the salience network (SN), the executive control network (ECN), and the default mode network (DMN), respectively [113, 46].

For each diagnostic group and each seed ROI, we obtain the set of regions covarying with the seed ROI, following the process described in Section 3.2.1. The structural covariance maps corresponding to the seed ROI are shown in Figure 3.6(a)-(c). Further comparisons in Figure 3.7 show that the maps for two diagnostic groups do not entirely overlap. Some regions present on the map for the control group are absent on the map for the autism group. Conversely, some regions are present only on the map for the autism group but not on the map for the control group. Table 3.1 lists the number of regions present in controls but not in autism, in autism but not in controls, and in both as well as in either autism or control. A network-specific set of ROIs is given by the union of all regions covarying with the corresponding seed ROI, in either the autism group map or the control group map.

Thus, we have one *Global* set of ROIs and three network-specific sets of regions. For each set of ROIs, we compute the group level SCGs and perform the random permutation test as described in Section 3.3.1. We perform one million permutations in the case of network-specific SCGs and $10,000$ permutations in the case of the Global-SCG (due to computational constraints). We also perform the bootstrap test as described in Section 3.3.1 with one million bootstrap samples in the case of network SCGs and $10,000$ in the case of the Global-SCG.

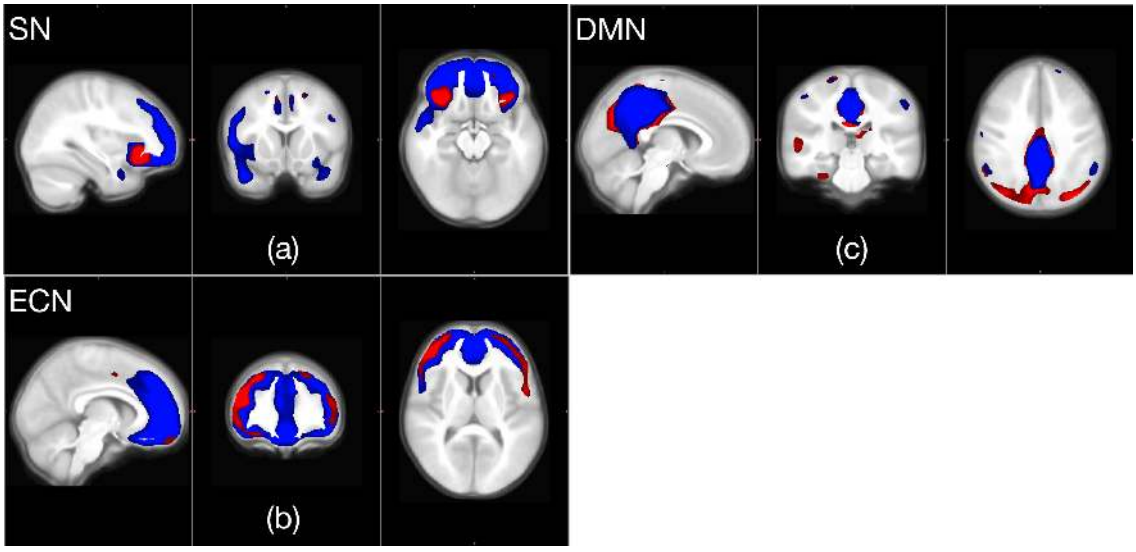

FIGURE. 3.6. Structural covariance maps with seed in R FI, R DLPC, and R PCC, anchoring SN (**(a)**), ECN (**(b)**), and DMN (**(c)**), respectively. Red represents the autism group map; blue represents the control group map.

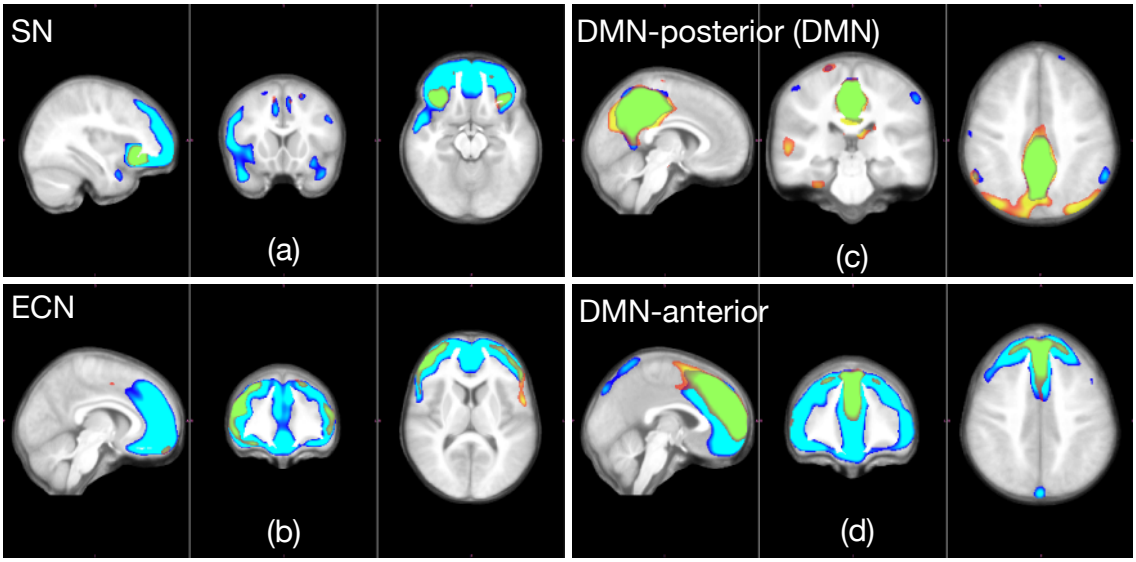

FIGURE. 3.7. scMRI maps are further illustrated here with red to yellow (autism) and dark blue to light blue (control) color maps. The color gradation indicates increasing statistical significance. The overlapping regions among the autism and control groups are highlighted in green. Note for **(c)** and **(d)**: Our data consist of subjects with an average age of about 13 years. The underlying structure of the DMN is not fully developed at this age. We include two DMN maps with different seeds to show that the posterior part **(c)** is not yet integrated with the anterior part **(d)**. In our analysis, we use the posterior covariance map **(c)**, which corresponds to the most common seed for DMN in adults (R PCC).

TABLE. 3.1. The number of ROIs identified from scMRI map for a given seed region. The last column shows the number of nodes in the corresponding SCG.

	Controls only	Autism only	Both	Either
R PCC (DMN-SCG)	9	21	9	39
R FI (SN-SCG)	21	1	10	32
R DLPC (ECN-SCG)	22	5	12	39

3.4.2.3 Type I error rate for permutation test. The number of permutations in an exact permutation test depends on the number of subjects in the sample. Our random permutations test closely approximates this exact test. However, note that the test does not take into account the number of ROIs. To check whether our test is robust to an increasing number of ROIs, we have implemented the following simulation.

We generate two $m \times n$ data matrices (m is the number of subjects, and n is the number of ROIs), where each element is randomly generated from a standard normal distribution

so that both the samples (data matrices) come from the same underlying distribution. We compute the SCGs from these data matrices and perform the random permutations test as described in section Section 3.3.1. Since the samples' underlying distribution is the same, rejecting the null hypothesis would be a type I error.

We keep the number of subjects $m = 100$ fixed through the whole simulation. Then we set the number of ROIs $n = 10, 50, 100, 500, 1000$, respectively. For each value of n, we perform 100 random permutation tests as described in the previous paragraph to find the proportion of tests resulting in type I error. Table 3.2 shows the findings. At 0.05 level of significance, our test is more conservative, meaning it is much less likely to get false positives.

3.4.3 Results

We apply the random permutation test and the bootstrap test to compare SCGs across groups of subjects with autism (ASD) and typically developing control subjects (TDC). We begin by comparing the global SCGs composed of 7266 gray matter regions in the preprocessed images. Then, for closer analysis, we compare the SCGs generated with seed ROIs anchoring the three ICNs (SN, ECN, and DMN), referred to as SN-SCG, ECN-SCG, and DMN-SCG. These are much smaller SCGs, composed of 32, 39, 39 ROIs, respectively (see Table 3.1). Recall that the structural covariance maps for the autism and the control groups overlap in very few regions. We construct and compare SCGs derived from sets of regions present in either control or autism. The β_0 curves corresponding to the global SCGs and the seed-specific SCGs are shown in Figure 3.8.

Table 3.3 lists the estimated p-values obtained from the random permutation tests along with the standard errors in the estimation. Also note that the number of permutations is sufficiently large in each case so that the estimation standard errors are small and not likely to affect the outcomes of the permutation tests.

TABLE. 3.2. Type I error rate for increasing number of ROIs.

number of ROIs	10	50	100	500	1000
Error rate	0.01	0.02	0.03	0.05	0.03

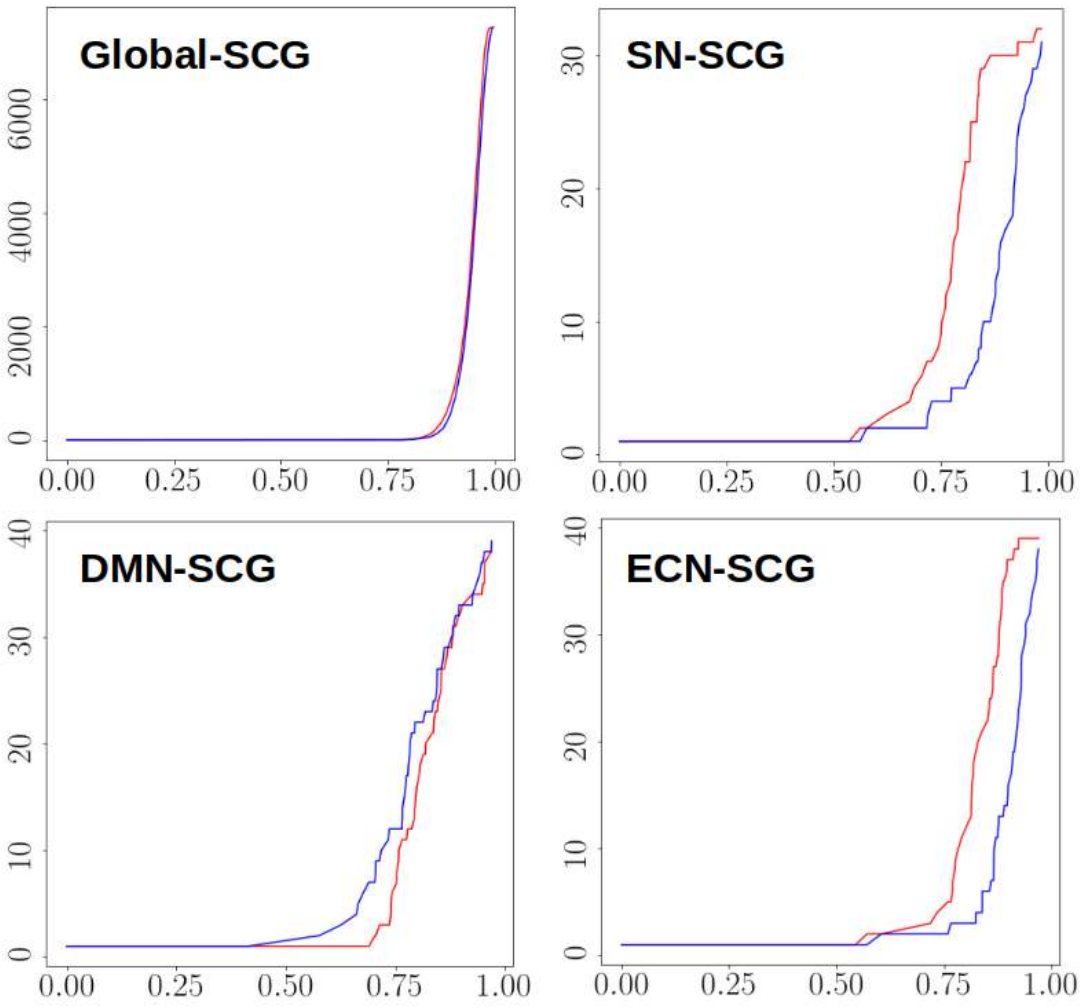

FIGURE. 3.8. β_0 curves from Global SCGs as well as SN-SCGs, ECN-SCGs, and DMN-SCGs, generated from regions present in either autism (red) or controls (blue), respectively.

TABLE. 3.3. Estimated p-values for random permutation test on SCGs and corresponding standard errors.

	Global-SCG	DMN-SCG	SN-SCG	ECN-SCG
p-value estimate	0.3985	0.3658	**0.00614**	0.1118
standard error	0.004895	0.000481	$\mathbf{7.81172 \times 10^{-5}}$	0.000315

Table 3.4 lists the p-values obtained from the bootstrap test. Note that the conclusion from both tests is comparable at the 0.05 level of significance. By combining topological data analysis with statistical inference, our results provide evidence of statistically significant network-specific structural abnormalities in autism SN-SCGs.

Figure 3.9 shows a comparative visualization of the two SN-SCGs that correspond to ASD and TDC groups, respectively, at the threshold $\lambda = 0.8174$. This is the threshold at which the gap between the corresponding β_0 curves is the largest ($D_q = 21$, with β_0 being 6 and 27, for ASD and TDC, respectively).

TABLE. 3.4. Bootstrap test on SCGs: p-values.

	Global-SCG	DMN-SCG	SN-SCG	ECN-SCG
p-value	0.3685	0.3970	**0.0073**	0.0890

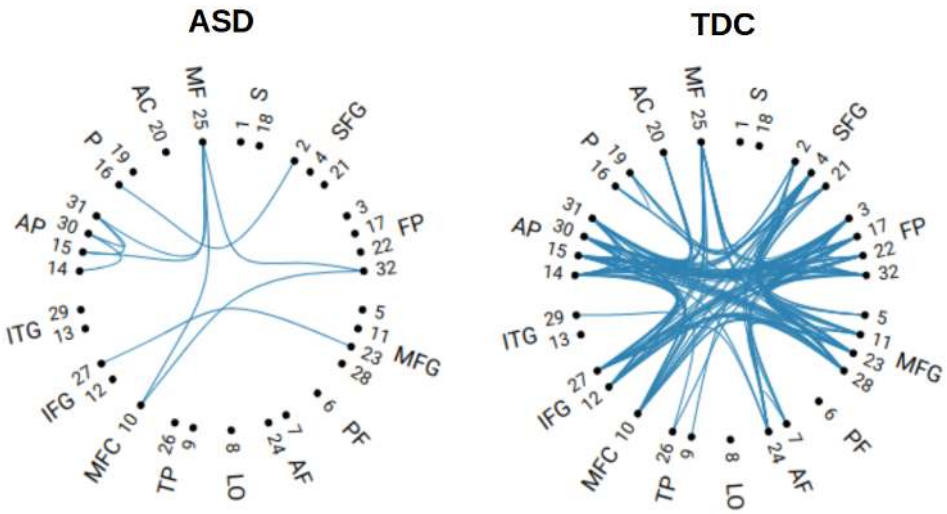

FIGURE. 3.9. SN-SCG at threshold $\lambda = 0.8174$ that corresponds to D_q^* for both ASD (left) and TDC (right) groups. The ROIs (4mm spheres) are grouped by anatomical regions in which they are placed as follows : S - SMA, SFG - sup frontal gyrus, FP - frontal pole, MFG - middle frontal gyrus, PF - post fusiform, AF - anterior Fronto-Insular, LO - Lateral Occipital, TP - Temporal pole, MFC - medial frontal cortex, IFG - inferior frontal gyrus, ITG - inferior temporal gyrus, AP - anterior paracingulate, P - paracingulate, AC - anterior cingulate, MF - medial frontal (ventromedial prefrontal cortex). Image courtesy of Yiliang Shi.

The visualization shows that the gray matter densities are much less correlated across the ASD subjects compared to the TDC subjects. The most likely reason behind these differences is that there is more idiosyncrasy in the autism cohort in which regions show greater or lesser cortical thickness or subtle differences in gyri or sulci positions. This idiosyncrasy might be the most reliable difference in autism, meaning that each ASD case is unique, different from the group mean values in some way.

3.5 Regression with Functional Networks

In this section, we utilize topological features in the form of persistence diagrams to analyze the relationship between functional brain connectivity and behavioral phenotypes.

3.5.1 Contributions

We propose a novel method to analyze the relationship between functional brain connectivity and behavioral phenotypes. First, we extract persistence diagrams from the functional correlation graphs and compute the scale-space kernel defined in Section 3.3.2. Then, we use kernel partial least squares (kPLS) regression (described in Section 3.3.3) to statistically quantify the relationship between persistence diagrams and behavior measures. The kPLS regression model also provides a simple way to combine topological features with other types of features through linear combinations of kernels. We test the ability of our proposed model to predict autism severity (ADOS scores) and show that combining correlations with topological features gives a better prediction of autism severity than using correlations alone.

3.5.1.1 My contribution. This is a collaborative work. My main contributions include computing topological features and utilizing the scale-space kernel in the regression model.

3.5.2 Methods

We apply kPLS regression to predict autism severity using topological features derived from FCGs. Autism severity is measured by the Autism Diagnostic Observation Schedule (ADOS) score, which is an evaluation for autism based on observed social and communication behaviors. Typically, subjects with an ADOS score > 8 are diagnosed with ASD. As a baseline, we also use the pairwise correlations from the associated correlation matrix as

predictors in the kPLS regression. Our results show that combining the correlation features with the topological features improves the predictive power of the kPLS.

3.5.2.1 Data and preprocessing. We use data from the Autism Brain Imaging Data Exchange (ABIDE) [20], which aggregates rs-fMRI scans and behavioral information such as ADOS from ASD subjects and controls from multiple international sites. However, to avoid heterogeneity across sites due to differences such as different rs-fMRI scanners and imaging protocols, we limit our analysis to a single site. Our data consist of 87 subjects, with 57 ASD and 30 typically-developing control (TDC) subjects.

We preprocess all rs-fMRI scans using the Functional Connectomes-1000 scripts. The preprocessing steps include skull stripping, motion correction, registration, segmentation, and spatial smoothing. Then, we obtain the BOLD signals corresponding to 264 regions based on Power's regions of interest [102], and construct the FCGs. Flattening the strictly upper triangular part of the corresponding 264×264 correlation matrix results in a $34,716$-dimensional vector of *correlation features* for each subject. We also compute *topological features* from each subject's FCG in the form of 0- and 1-dimensional persistence diagrams using the procedure described in Section 3.2.2.

3.5.2.2 kPLS regression. We use kPLS regression to relate the ADOS scores with correlation features, topological features or a combination of both. For regression with correlation features, we use a linear kernel K^{cor} defined as the Euclidean dot product between correlation vectors. For regression with topological features, we use scale space kernels K^{TDA_0} and K^{TDA_1} for 0- and 1-dimensional persistence diagrams, as defined in Section 3.3.2. The kernels are normalized by the median of the absolute values of their entries. To combine topological and correlation features, we use a linear combination of the kernels to form

$$K^{\text{TDA+cor}} = w_0 K^{\text{TDA}_0} + w_1 K^{\text{TDA}_1} + (1 - w_0 - w_1) K^{\text{cor}}.$$

The kPLS regression with this kernel has four free parameters: the bandwidth parameters σ_0 and σ_1 for the scale space kernels, and the weights w_0, w_1 in the linear combination.

We train all prediction models using leave-one-out cross-validation (LOOCV), i.e., for each subject, we train the model on the other $n-1$ subjects and predict the left-out subject's ADOS score. We evaluate the accuracy of each model using the root mean squared error

(RMSE) between predicted and true ADOS scores. The free parameters are tuned by grid search, with weights w_0 and w_1 uniformly spaced in the range $[0, 1]$ with a step size of 0.05, and the bandwidths on a log scale ($\log_{10}(\sigma_0)$ and $\log_{10}(\sigma_1)$) in the range $[-8, 6]$ with a step size of 0.2. Setting $(1 - w_0 - w_1) = 0$ results in a model using only the topological features.

3.5.3 Results

In addition to the kPLS models using K^{cor}, $K^{\text{TDA+cor}}$, and K^{TDA}, we also consider a simple model ($ADOS_{mean}$) that uses the mean ADOS value of $n - 1$ subjects as the prediction for the remaining subject. Both the correlation features and the topological features show promising predictive power over this *mean predictor* (see Table 3.5). We also generate n random correlation matrices from independent and identically distributed ($N(0, 1)$) time series of the same size as the real fMRI data and compute their linear kernel. The regression model with random signals has a root mean squared error of 6.47359, which is worse than the RMSE of the $ADOS_{mean}$ predictor. This test also ensures that using correlation and topological features is better than using random signals. To determine the statistical significance of our results, we perform permutation tests using $\text{RMSE}_{\text{method}_2}$ - $\text{RMSE}_{\text{method}_1}$ as the test statistic. We compare RMSEs of all pairs of models, including the kPLS models with three kernels and the $ADOS_{mean}$ model. Each test includes $100,000$ permutations, where we randomly swap the $method_1$ and $method_2$ predictions for subjects and compute the new RMSEs and the test statistic. The p-value is the percentage of permuted RMSE differences that were greater than the unpermuted difference in the RMSE. From our results, both K^{TDA} (parameters: $\sigma_0 = -6.6$, $\sigma_1 = 1.8$, $w_0 = 0.05$, $w_1 = 0.95$) and K^{cor} show evidence of improvement over the baseline. $K^{\text{TDA+cor}}$, which combines the correlation features with the topological features, has the best predictive power. It is also

TABLE. 3.5. ADOS prediction results. Columns 2–4 are p-values for the permutation test of improvement of row method over column method.

	RMSE	$ADOS_{mean}$	K^{TDA}	K^{cor}
$ADOS_{mean}$	6.4302	-	-	-
K^{TDA}	6.3553	0.316	-	-
K^{cor}	6.0371	0.055	0.095	-
$K^{\text{TDA+cor}}$	**6.0156**	**0.048**	0.075	0.288

the only method that is statistically significantly better than baseline. The best prediction result is obtained with parameters $\sigma_0 = -7.8$, $\sigma_1 = 2.8$, $w_0 = 0.10$, $w_1 = 0.40$. These results show that topological features derived from correlations of rs-fMRI have the potential to explain the connection between functional brain networks and autism severity.

3.6 Classification with Functional Networks

In this section, we explore the utility of topological features in the classification of ASD versus TDC subjects. We experiment with three representations of topological features described in Section 3.2.2, namely, persistence diagrams (PD), persistence landscapes (PL), and persistence images (PI). We train several types of classifiers using these representations and compare their performance using a paired permutation test.

3.6.1 Contributions

We train three types of ASD classifiers – random forests (RF), support vector machines (SVM), and neural networks (NN) – using the various representations of topological features. We train SVM classifiers using the four kernels for persistence diagrams described in Section 3.3.2. The neural network classifiers utilize the projection layer for the persistence diagrams described in Section 3.3.4. We also propose hybrid classifiers augmenting topological features with functional correlations, which typically outperform the classifiers that use functional correlations or topological features alone. However, our experiments also show that the improvement in the classification accuracy due to topological features is not always statistically significant. Therefore, we offer a cautionary tale to the practitioners regarding the limited discriminative power of topological features derived from fMRI data for the classification of autism.

3.6.1.1 My contributions. This is a collaborative work. My main contributions include experiment design, i.e., selecting the topological features and their representations, deciding the training and testing procedures, and determining the testing procedure to evaluate the statistical significance of the results. In addition, I also selected various explicit (persistence landscapes, persistence images) or implicit (kernels) vector representations of topological features and prototyped the SVM classifiers that use these representations exclusively or in combination with correlation features.

3.6.2 Methods

We use Craddock 200 (**CC200**) and Craddock 400 (**CC400**) datasets from ABIDE Preprocessed [20, 39], which include rs-fMRI scans from 16 imaging sites. The scans are preprocessed using the Configurable Pipeline for Analysis of Connectomes (without global signal regression) before extracting BOLD signals for $n = 200$ and $n = 400$ ROIs, for **CC200** and **CC400** ROIs, respectively. After filtering out subjects with missing or invalid data, the datasets contain 1035 subjects, of which 505 are ASD patients and 530 are TDC. For each subject, we construct the FCG and compute the 0- and 1-dimensional persistence diagrams, denoted PD_0 and PD_1, respectively.

3.6.2.1 Classification with correlation features. Similar to Section 3.5.2, we obtain correlation feature vectors by flattening the strictly upper triangular parts of the correlation matrices associated with FCGs. We train SVM classifiers with correlation features ($\mathbf{SVM}_{Corr}$), using a linear kernel (Euclidean dot product). The *cost* parameter that controls the trade-off between misclassification and margin size is estimated via a grid search. To train RF classifiers with correlation features ($\mathbf{RF}_{Corr}$), we use m trees and a maximum depth of k, where m and k are parameters estimated via a grid search.

3.6.2.2 Classification with topological features. Persistence images and persistence landscapes are easily vectorized, and we can use them to train classification models in a straightforward way. We use 200×200 persistence images with 0 spread. We compute persistence landscapes of order 1 through 5 sampled at 2000 discrete points. For persistence diagrams, we use the kernels defined in Section 3.3.2 for SVM models. In the case of the scale-space kernel, we consider only 0- and 1-dimensional topological features and compute 0- and 1-dimensional kernels separately, which we then combine using a weighted average. For the persistence-weighted Gaussian kernel, we use a default weight function that assigns unit weight to each point in the persistence diagram. For the sliced Wasserstein kernel, the number of directions is fixed to $M = 10$. The tuneable parameters such as bandwidths are determined using a grid search for all kernels.

3.6.2.3 Classification with combined features. Lastly, we propose hybrid classifiers that combine correlation and topological features. In the case of SVM classifiers, we follow the procedure proposed in Section 3.5.2, combining features through a linear combination respective kernels. For instance, in the case of persistence diagrams, the

combined kernel is given by

$$K_{PD+Corr} = w_0 K_{PD_0} + w_1 K_{PD_1} + (1 - w_0 - w_1) K_{Corr},$$

where $w_0, w_1 \geq 0$, and $w_0 + w_1 \leq 1$ are parameters estimated using grid search, and K_{PD_0} and K_{PD_1} are kernels on PD_0 and PD_1, respectively. K_{Corr} denotes the kernel for correlation features. For all vector representations, such as correlation features, persistence images, and persistence landscapes, we use the Euclidean dot product as the kernel.

3.6.2.4 NN models. In addition to the RF and SVM models, we also train NN models with three, five, and seven fully connected layers. For NN classifiers using persistence diagrams, the input layer is replaced by the projection layer described in Section 3.3.4. We also train hybrid NN models where correlation and topological features are processed in parallel branches before concatenating outputs in the last layer to form a single vector.

3.6.2.5 Permutation tests. We use paired permutation tests to determine the statistical significance of the improvement in classification accuracy of one classifier over another. For a pair of models, say $model_1$ and $model_2$, we use the difference in their average prediction accuracy (Avg. $\text{accuracy}_{model_1}$ − Avg. $\text{accuracy}_{model_2}$) as the test statistic. Due to the large sample size, it is not feasible to compute all possible permutations. Instead, we perform $100,000$ permutations to approximate the p-values. In each permutation, we randomly swap the labels predicted by $model_1$ and $model_2$ for the subjects and compute the new test statistic. The p-value is the fraction of samples where the permuted labels' test statistic is greater than the test statistic for the original labels.

3.6.3 Results

We train three types of classifiers, each using correlation features, topological features, or their combination, to classify Asd and TDC subjects from the **CC200** and **CC400** datasets. The performance of the classifiers is summarized in Table 3.6 and Table 3.7, respectively. We employ a five-fold stratified cross-validation scheme to train the classifiers and report the mean classification accuracy over all folds. Following abbreviations are used to indicate the types of features used to train a classifier: *Corr* for correlation features and *PD*, *PI*, and *PL* for persistence diagrams, persistence images, and persistence landscapes, respectively. $\mathbf{NN5}_{PD+Corr}$ denotes a five-layer hybrid neural network model, combining correlation with topological features from PD. For SVM models with PDs, we report the

TABLE. 3.6. Mean classification accuracy using various classifiers and feature combinations for the CC200 data.

Model	Accuracy	Model	Accuracy	Model	Accuracy
SVM_{Corr}	65.41				
RF_{Corr}	64.81				
$NN3_{Corr}$	68.35				
$NN5_{Corr}$	68.46				
$NN7_{Corr}$	67.1				
SVM_{PD}	53.03	SVM_{PI}	54.54	SVM_{PL}	53.03
RF_{PD}	–	RF_{PI}	52.25	RF_{PL}	52.51
$NN3_{PD}$	56.06	$NN3_{PI}$	58.56	$NN3_{PL}$	55.36
$NN5_{PD}$	56.15	$NN5_{PI}$	59.09	$NN5_{PL}$	55.18
$NN7_{PD}$	55.48	$NN7_{PI}$	56.75	$NN3_{PL}$	54.85
$SVM_{PD+Corr}$	65.86	$SVM_{PI+Corr}$	64.25	$SVM_{PL+Corr}$	65.65
$NN3_{PD+Corr}$	**69.19**	$NN3_{PI+Corr}$	**67.2**	$NN3_{PL+Corr}$	**68.5**
$NN5_{PD+Corr}$	68.2	$NN5_{PI+Corr}$	66.87	$NN5_{PL+Corr}$	67.45
$NN7_{PD+Corr}$	64.47	$NN7_{PI+Corr}$	65.1	$NN7_{PL+Corr}$	67.02

TABLE. 3.7. Mean classification accuracy using various classifiers and feature combinations for the CC400 data.

Model	Accuracy	Model	Accuracy	Model	Accuracy
SVM_{Corr}	66.33				
RF_{Corr}	63.92				
$NN3_{Corr}$	63.92				
$NN5_{Corr}$	65.58				
$NN7_{Corr}$	62.06				
SVM_{PD}	53.69	SVM_{PI}	53.76	SVM_{PL}	53.69
RF_{PD}	–	RF_{PI}	53.04	RF_{PL}	53.12
$NN3_{PD}$	55.9	$NN3_{PI}$	56.1	$NN3_{PL}$	54.24
$NN5_{PD}$	56.04	$NN5_{PI}$	57.39	$NN5_{PL}$	55.72
$NN7_{PD}$	54.33	$NN7_{PI}$	55.58	$NN3_{PL}$	53.67
$SVM_{PD+Corr}$	63.36	$SVM_{PI+Corr}$	62.68	$SVM_{PL+Corr}$	64.12
$NN3_{PD+Corr}$	**67.84**	$NN3_{PI+Corr}$	**67.02**	$NN3_{PL+Corr}$	**66.76**
$NN5_{PD+Corr}$	66.03	$NN5_{PI+Corr}$	66.23	$NN5_{PL+Corr}$	66.48
$NN7_{PD+Corr}$	61.25	$NN7_{PI+Corr}$	64.16	$NN7_{PL+Corr}$	65.26

results using only the scale-space kernel in Table 3.6 and Table 3.7. We observe that the three representations of topological features (PD, PI, and PL) lead to similar performance (Table 3.6 and Table 3.7), and the same is true for the performance of the kernel SVM models using different kernels (Table 3.8). The highest mean classification accuracy is obtained using the three-layer hybrid NN models (Table 3.6 and Table 3.7, bold and red entries). However, we note that adding more layers leads to a decrease in classification accuracy, and include only three-layer NN classifiers in the permutation tests. Table 3.9 reports the p-values for the permutation tests comparing row methods with column methods.

TABLE. 3.8. Mean accuracy for kernel SVM with different kernels for persistence diagrams.

PD Kernel	CC200	CC400
K_S	53.03	53.69
K_{S+Corr}	65.86	63.36
K_G	52.51	53.12
K_{G+Corr}	62.98	61.41
K_W	55.36	64.24
K_{W+Corr}	64.73	64.12
K_F	55.18	55.72
K_{F+Corr}	61.48	60.25

TABLE. 3.9. The statistical significance of improvements in classification accuracy for CC200, captured by p-values, comparing each row method against each column method.

	RF_{Corr}	SVM_{Corr}	$SVM_{PD+Corr}$	$NN3_{Corr}$
SVM_{Corr}	0.1502			
$SVM_{PD+Corr}$	0.1943	0.4213		
$NN3_{Corr}$	**0.0461**	**0.048**	0.0631	
$NN3_{PD+Corr}$	**0.0406**	**0.0446**	**0.0414**	0.1894
	RF_{Corr}	SVM_{Corr}	$SVM_{PI+Corr}$	$NN3_{Corr}$
$SVM_{PI+Corr}$	0.1943	0.4213		
$NN3_{Corr}$			**0.0420**	
$NN3_{PI+Corr}$	**0.0493**	0.0763	0.0734	0.7432
	RF_{Corr}	SVM_{Corr}	$SVM_{PL+Corr}$	$NN3_{Corr}$
$SVM_{PL+Corr}$	0.1623	0.3513		
$NN3_{Corr}$			0.0581	
$NN3_{PL+Corr}$	**0.0467**	0.0683	0.0717	0.3524

The p-values in Table 3.9 indicate that the improvement in classification accuracy for NN classifiers (hybrid and those using only correlation features) over SVM and RF classifiers is statistically significant (Table 3.9). However, the test comparing the hybrid NN classifier against NN classifier using only correlation features indicates that the improvement due to topological features may not be statistically significant (Table 3.9).

3.7 Discussion

In this chapter, we describe applications of TDA to study structural and functional brain networks in autism spectrum disorders. Specifically, we use persistent homology to extract topological features from these brain networks and use these features to perform tasks like statistical inference, regression, and classification. In a related study [9] not covered in this dissertation, we also show that topological features are robust to variations in preprocessing strategies. Our results show that we may use topological features to identify differences in structural and functional connectivity caused by autism and to predict phenotypes and cognitive scores.

3.7.1 Structural Brain Abnormalities in ASD

Similar covariance of gray matter density across a population of individuals is thought to be mediated by shared genetic or developmental factors. The structural covariance maps, proposed by Zielinski et al. [133], exhibit differences in architecture, with extensive regions of the frontal lobe showing greater alignment with salience (SN) and executive control network (ECN) rather than with the default mode network (DMN) [134]. The analysis presented in Section 3.4 helps summarize these relationships using a more robust topological data analysis model. Our experiments provide evidence of statistically significant differences in the 0-dimensional topological features of SCGs derived from SN (SN-SCGs). We find decreased integration of salience and executive networks, with increased integration of default regions within the frontal lobe. These findings align with results investigating functionally defined intrinsic connectivity networks [1] and suggest that shared developmental influences may underlie the particular specificity of SN connectivity abnormalities in autism [125].

The DMN has also been associated with atypical connectivity in autism [8, 11, 28, 81]). However, our experiments fail to capture any statistically significant differences in the

topology of SCGs derived from DMN. A possible reason is the existence of more complex topological differences that are not captured by pairwise relationships between the ROIs of DMN-SCG. The β_0 curves encode only 0-dimensional topological features that correspond to the evolution of the number of connected components. Analyzing three-way or four-way interactions between ROIs, capturing higher order topological features such as tunnels and voids, and focusing on specific nodes and edges directly involved in merging components in the graph filtration may provide further insights into the differences in DMN architecture.

3.7.2 ASD Classification and Prediction of Behavioral Phenotypes

In Section 3.5 and Section 3.6, we explore the utility of topological features in the prediction of behavioral phenotypes and autism classification, respectively. In both cases, we observe that combining topological and correlation features leads to improved performance. In the case of classification, a simple three-layer hybrid neural network classifier achieves an average accuracy of 69.2%, which is close to the state of the art (70%) on the entire ABIDE dataset. However, this result, combined with the ones reported by several other researchers [2, 59, 94], leads us to conjecture that 70% might be the best classification accuracy any classifiers can achieve on the *entire* ABIDE dataset. Several possible factors may lead to this relatively low performance. One reason is the heterogeneity of the data across sites. Another reason is the relatively poor single-subject reliability due to low temporal resolution and short acquisition sequences ($\leq$ 10 min/subject) of the rs-fMRI scans. The wide developmental age range of the subjects also contributes to the challenge, and the classifiers may perform better on samples with a narrower age range. Finally, heterogeneity of the autism spectrum disorders themselves may also result in low classification accuracy. Grouping subjects with a wide range of symptoms under a single label may not be as effective as discriminating between several subsets.

3.7.3 Future Work

The work presented in this chapter is only a small set of possible applications of TDA in neuroscience. Here, we describe two new directions for future research.

3.7.3.1 Dynamic functional connectivity. The functional connectivity graphs considered in this chapter represent the functional relationship between brain regions. Alternatively, we can construct brain networks where vertices are the time points in rs-fMRI,

and the edges represent the similarity between image volumes at two different time points. These networks reflect dynamical aspects of connectivity, such as the duration and frequency of brain microstates. Although less intuitive than traditional functional connectivity, such networks may offer new insights into aspects of cognition and neuropathology. In [9], we presented preliminary results relating these time-domain networks with cognitive and personality features such as fluid intelligence. We are interested in further studying such networks to understand the temporal dynamics of brain function.

3.7.3.2 Structural and functional connectivity at finer scales. The structural and functional networks considered in this chapter are abstract networks inferred from clinical or rs-fMRI imaging data. Researchers have now begun applying TDA to study brain connectivity at a much finer scale, such as neural circuits and synaptic activity with [104] or without [52] external stimuli. In both cases, recent work has shown that the synaptic activity networks exhibit higher order functional connectivity (e.g., cliques/simplices) and topological features (e.g., cavities) [104, 52]. Categorizing different types of neurons [63] and understanding the differences in synaptic connectivity arising from morphological diversity within neuronal types [95] is an emerging area of research. TDA techniques described in this chapter as well as the ideas presented in Chapter 2 may be applicable in such settings.

3.7.4 Conclusion

In this chapter, we describe applications of TDA to study brain networks in autism. We demonstrate that TDA techniques are well suited to characterize and summarize complex patterns in brain connectivity. Using topological features of brain networks, we are able to identify differences in structural and functional connectivity in ASD vs TDC subjects, and predict phenotypes and cognitive scores. The same techniques can be applied to study the mechanisms underlying other complex brain disorders such as dementia, Huntington's, or Alzheimer's. We may further apply these techniques to track the progression of the disorder and the effectiveness of treatments.

TDA techniques can be used to advance our understanding of the organizational principles underlying brain networks. For example, recent literature has provided evidence of higher order structural and functional connectivity between neurons [104, 52, 95]. The activation of a single neuron depends on its interaction with multiple neurons. Such type

of higher order connectivity between neurons can be modeled as a simplicial complex. We may apply the learning algorithms or the notions of random walks on simplicial complexes presented in Chapter 2 to identify different types of neurons and study their communication patterns.

Our work represents a small set of possible applications of TDA in neuroscience. The interest in topological analysis of structural and functional brain networks is growing. In recent years, TDA techniques such as persistent homology and mapper [115] have been applied to study brain networks [99, 116, 111] in several different settings. TDA techniques have the potential to transform the way we study neurons and neural communications, and we hope to continue exploring their applications in neuroscience in the future.

CHAPTER 4

STRUCTURAL VARIABILITY IN GRAPH ENSEMBLES

In this chapter, we study an ensemble of graphs obtained from graph reduction algorithms. Our goal is to quantify the structural variability associated with the vertices and the edges of the reduced graphs.

In Section 4.1, we describe the motivation behind the work presented in the chapter. In Section 4.2, we give a brief description of relevant concepts such as similarity measures for clusterings. In Section 4.3 and Section 4.3, we present methods to capture the structural variability associated with the supervertices and superedges of a reduced graph from the ensemble. Lastly, in Section 4.5, we discuss the advantages and disadvantages of the proposed methods and conclude with a discussion of future directions.

4.1 Introduction

Consider a graph G with n nodes. We can use several graph reduction algorithms to reduce G to a smaller graph H with k nodes where $k \ll n$. Different algorithms result in slightly different reduced graphs. Some graph reduction algorithms employ randomization to improve efficiency, producing a different reduced graph in each run with a fixed input graph. Given a collection of reduced graphs obtained from G using multiple algorithms or multiple runs of the same (randomized) algorithm, we measure the variability of the supervertices and the superedges of a reduced graph.

4.1.1 Graph Reduction and Structural Variability

The large sizes of real-world graphs can often hamper the efficiency of their analysis and visual interpretability. The idea of graph reduction is to reduce the size of a graph while preserving its properties of interest. The two main types of graph reduction are graph sparsification and graph coarsening. Graph *sparsification* reduces the number of edges in a

graph while maintaining the number of vertices. Its theoretical approximation bounds are well studied in the literature (e.g., [130]; also see Chapter 2). Therefore, graph sparsification is more commonly used in the analysis compared to coarsening.

Graph *coarsening*, on the other hand, reduces the number of vertices, thus implicitly reducing the number of edges. Coarsening is typically performed by clustering the vertices of a graph to form supervertices. The idea of coarsening has its roots in finite element analysis, but many existing coarsening techniques are heuristic in nature and lack a firm theoretical footing compared to sparsification. Nevertheless, they appear in many applications such as visualization [58], graph partitioning [110], and dimensionality reduction [27].

Many graph reduction algorithms focus on preserving the spectrum of the adjacency or the Laplacian matrix [62] since its spectrum captures many important combinatorial properties of a graph [31]. Although graph reduction techniques are often used in analysis and visualization, the variability associated with their outputs has remained mostly unexplored. There are two main sources of variability in graph reduction:

- Some graph reduction algorithms employ randomization to improve computational efficiency. As a result, the same algorithm applied to the same input may produce different outputs across different runs. This type of variability is called *residual variability*, which arises from the process itself being inherently unpredictable or stochastic even with deterministic inputs [65].

- Different graph reduction algorithms focus on preserving slightly different properties of the original graph and / or have slightly different implementations. As a result, when applied to the same graph, these algorithms can produce different reduced graphs. We refer to this type of variability as *ensemble variability*.

In both cases, we are interested in measuring the variability of the supervertices and the superedges of the reduced graphs in an ensemble.

4.1.2 Consensus Clustering

Consensus clustering is used to represent the consensus across multiple clusterings from the same input data, either using multiple clustering algorithms or multiple runs of the same algorithm [89]. It can be used to determine the number of clusters in the data, study the

sensitivity of a clustering algorithm with respect to initial conditions and model parameters, or assess the stability of the discovered clusters [89]. The consensus approach has been used to rank and select clustering results in *Clustervision* [68].

In [47], Fiol-Gonzalez et al. provided a visual exploration tool for ensemble clustering analysis. Their main contribution was the *co-association matrix* generated by an ensemble of clustering results (e.g., from the outputs of multiple classifiers), where a (i, j) entry in the matrix captures the probability that node i and node j belong to the same cluster. Similarly, in [48], Gates et al. proposed an *element-centric* unified framework to compare clusterings. This framework also compares individual elements based on the relationships induced by the clusterings. For an ensemble of clusterings, the authors provided a way to compute the elementwise average agreement and *frustration*, which is an elementwise measure of clustering uncertainty in the ensemble. The authors also showed that the traditional *cluster-centric* comparison metrics, such as the Adjusted Rand Index, can be computed using this unified framework by appropriately weighting the cluster-induced relationships.

However, previous work [47, 48] focused on variability associated with the individual vertices of the input graph. Instead, we are interested in measuring the variability with respect to the supervertices and superedges of the reduced graph.

4.1.3 Contributions

A graph reduction algorithm reduces the size of a graph while preserving its properties of interest. However, the structural variability in graph reduction can induce uncertainty in the insights obtained from the reduced graph. For example, community structures inferred from a social network using different graph reduction algorithms, or different runs of the same (randomized) algorithm may be slightly different. Even when the different instances of reduced networks agree on the global community structure, there may be variations in the size and connectivity of individual communities. Understanding such variability can help us gain a deeper understanding of the reduced graph and the original data it is associated with.

We present a general and flexible framework to quantify and visualize the structural variability associated with graph reduction algorithms. Our contributions are as follows:

- We present a set of local similarity measures to capture the variability associated

with the supervertices and superedges of a reduced graph in an ensemble.

- Using the idea of co-clustering, we present a second, more general framework to capture the structural variability associated with a reduced graph in a given ensemble.

Subsequently, we give an interactive visual demo which provides different ways to encode the structural variability for a given reduced graph, and allows users to better understand the structural in multimodel or multirun graph reduction scenarios as detailed in [70].

4.1.3.1 My contribution. This is a collaborative work. My main contributions include developing various measures that capture the structural variability, which is the focus of this chapter.

4.2 Background

Let $G = (V, E, w)$ be a simple, weighted, undirected graph with n vertices and m edges, positive edge weights $w : E \to \mathbb{R}^+$, with $w_e := w(e)$ for an edge $e \in E$.

Given a graph G, a coarse graph $H = (V', E', w')$ is constructed using a clustering that partitions vertices of G into k partitions. Let $\mathbf{S} = \{s_1, s_2, \ldots, s_k\}$ denote the clustering. Each partition $s_i \subseteq V$ $(1 \leq i \leq k)$ represents a *supervertex* in H denoted by c_i. The supervertices c_i and c_j of H are connected via a *superedge*, whose weight represents an aggregation of weights of all edges between vertices of G in clusters s_i and s_j. We identify the clustering $\mathbf{S}$ with a *coarsening matrix* $M \in \mathbb{R}^{k \times n}$ defined as [62, Section 3.1]:

$$M_{ij} = \begin{cases} 1 & \text{if } v_j \in s_i, \\ 0 & \text{otherwise.} \end{cases}$$

Let $\mathbf{S}^1 = \{s_1^1, s_2^1, \cdots, s_k^1\}$ and $\mathbf{S}^2 = \{s_1^2, s_2^2, \cdots, s_k^2\}$ be two clusterings of vertices V in G, corresponding to the reduced graphs H^1 and H^2, respectively. Let M^1 and M^2 denote the corresponding coarsening matrices. The confusion matrix of $\mathbf{S}^1$ and $\mathbf{S}^2$ is defined as $F = M^1(M^2)^T$. $F_{ij} = |s_i^1 \cap s_j^2|$ captures the size of overlap between partitions $s_i^1 \in \mathbf{S}^1$ and $s_j^2 \in \mathbf{S}^2$.

4.3 Capturing Variability via Local Similarity Measures

The *global* similarity between reduced graphs H^1 and H^2 can be measured using the Jaccard index, adjusted Rand index, or mutual information of the corresponding clusterings

$\mathbf{S}^1$ and $\mathbf{S}^2$. However, to quantify the variability associated with individual supervertices of the reduced graphs, we need a local measure. Each supervertex c_i^1 of H^1 corresponds to a cluster s_i^1 in $\mathbf{S}^1$. We can define a *local* similarity measure for each individual cluster s_i^1 in $\mathbf{S}^1$, with respect to clustering $\mathbf{S}^2$ as the contribution of s_i^1 to the global similarity of $\mathbf{S}^1$ and $\mathbf{S}^2$.

4.3.1 Jaccard Index and Local Jaccard Index

We measures the similarity between H^1 and H^2 by considering whether a pair of vertices of the input graph G belongs to the same cluster or different clusters in clusterings $\mathbf{S}^1$ and $\mathbf{S}^2$. The Jaccard index is defined as

$$J(\mathbf{S}^1, \mathbf{S}^2) = \frac{n_{11}}{n_{11} + n_{10} + n_{01}}, \tag{4.1}$$

where n_{11} is the number of pairs in the same cluster under $\mathbf{S}^1$ and $\mathbf{S}^2$, n_{10} the number of pairs in the same cluster under $\mathbf{S}^1$ but different clusters under $\mathbf{S}^2$, and n_{01} the number of pairs in different clusters under $\mathbf{S}^1$ but the same cluster under $\mathbf{S}^2$. Then, we have [85, Section 2.1, page 3]

$$n_{11} = \frac{1}{2}\left(\left(\sum_{i=1}^{k}\sum_{j=1}^{k} F_{ij}^2\right) - n\right). \tag{4.2}$$

Using the relation between the Mirkin metric and the Rand index, we have [85, Section 2.1 Equation 9]

$$n_{01} + n_{10} = \frac{1}{2}\left(\sum_{i=1}^{k} |s_i^1|^2 + \sum_{j=1}^{k} |s_j^2|^2 - 2\sum_{i=1}^{k}\sum_{j=1}^{k} F_{ij}^2\right). \tag{4.3}$$

Adding n_{11} to both sides, we get

$$n_{11} + n_{01} + n_{10} = \frac{1}{2}\left(\sum_{i=1}^{k} |s_i^1|^2 + \sum_{j=1}^{k} |s_j^2|^2 - \left(\sum_{i=1}^{k}\sum_{j=1}^{k} F_{ij}^2\right) - n\right). \tag{4.4}$$

Based on $J(\mathbf{S}^1, \mathbf{S}^2)$, and using the expressions in Equation 4.2, and Equation 4.4, we define the *local Jaccard index* of a cluster s_i^1 in $\mathbf{S}^1$ with respect to $\mathbf{S}^2$ as

$$LJ(s_i^1, \mathbf{S}^2) := \frac{\frac{1}{2}\left(\left(\sum_{j=1}^{k} F_{ij}^2\right) - \frac{n}{k}\right)}{n_{11} + n_{10} + n_{01}}. \tag{4.5}$$

By definition, $\sum_{i=1}^{k} LJ(s_i^1, \mathbf{S}^2) = J(\mathbf{S}^1, \mathbf{S}^2)$.

4.3.2 Adjusted Rand Index and Local Adjusted Rand Index

Another way to measure similarity between H^1 and H^2 is to compute the adjusted Rand index of the corresponding clusterings $\mathbf{S}^1$ and $\mathbf{S}^2$. The adjusted Rand index has an expected value of 0 for independent clusterings and a value of 1 for identical clusterings [128]. For $\mathbf{S}^1$ and $\mathbf{S}^2$, the Rand index is defined as

$$R(\mathbf{S}^1, \mathbf{S}^2) = \frac{n_{11} + n_{00}}{\binom{n}{2}}, \tag{4.6}$$

where n_{11} and n_{00} are as defined earlier. Combined with Equation 4.1, $n_{11}+n_{10}+n_{01}+n_{00} = \binom{n}{2}$. The adjusted Rand index is defined as the normalized difference between the Rand index and its expected value [128, 85]

$$\begin{aligned} AR(\mathbf{S}^1, \mathbf{S}^2) &= \frac{R(\mathbf{S}^1, \mathbf{S}^2) - \mathbb{E}[R(\mathbf{S}^1, \mathbf{S}^2)]}{1 - \mathbb{E}[R(\mathbf{S}^1, \mathbf{S}^2)]} \\ &= \frac{\sum_{i=1}^{k}\sum_{j=1}^{k}\binom{F_{ij}}{2} - r_3}{\frac{1}{2}(r_1 + r_2) - r_3}, \end{aligned} \tag{4.7}$$

where $r_1 = \sum_{i=1}^{k}\binom{|s_i^1|}{2}$, $r_2 = \sum_{j=1}^{k}\binom{|s_j^2|}{2}$, $r_3 = \frac{r_1 r_2}{\binom{n}{2}}$, and $F_{ij} = |s_1^1 \cap s_j^2|$.

Based on $AR(\mathbf{S}^1, \mathbf{S}^2)$, we define *local Adjusted Rand index* of a cluster s_i^1 in $\mathbf{S}^1$ with respect to $\mathbf{S}^2$ as

$$LAR(s_i^1, \mathbf{S}^2) = \frac{\sum_{j=1}^{k}\binom{F_{ij}}{2} - r_3'}{\frac{1}{2}(r_1 + r_2) - r_3}, \tag{4.8}$$

where r_1, r_2, r_3 are the same as in Equation 4.7, and $r_3' = \binom{|s_i^1|}{2} r_2 / \binom{n}{2}$. By definition, we have $\sum_{i=1}^{k} LAR(s_i^1, \mathbf{S}^2) = AR(\mathbf{S}^1, \mathbf{S}^2)$.

4.3.3 Mutual Information and Local Mutual Information

Mutual information measures the mutual dependence between random variables. Similar to the Jaccard index and adjusted Rand index, we can use the mutual information of $\mathbf{S}^1$ and $\mathbf{S}^2$ to measure the similarity between reduced graphs H^1 and H^2. Given a clustering $\mathbf{S}$, under uniform distribution, the probability of any vertex v being in a cluster s_i is $Pr(v \in s_i) = P_i = \frac{|s_i|}{n}$. Given a pair of clusterings $\mathbf{S}^1$ and $\mathbf{S}^2$, the joint distribution of the probability of a vertex being in cluster $s_i^1 \in \mathbf{S}^1$ and $s_j^2 \in \mathbf{S}^2$ is

$$Pr(v \in s_i^1 \text{ and } v \in s_j^2) = P_{ij} = \frac{|s_i^1 \cap s_j^2|}{n} = \frac{F_{ij}}{n}. \tag{4.9}$$

The mutual information $MI(\mathbf{S}^1, \mathbf{S}^2)$ of $\mathbf{S}^1$, and $\mathbf{S}^2$ is defined [85] as

$$MI(\mathbf{S}^1, \mathbf{S}^2) = \sum_{i=1}^{k} \sum_{j=1}^{k} P_{ij} \log \frac{P_{ij}}{P_i P_j}.$$

Based on $MI(\mathbf{S}^1, \mathbf{S}^2)$, we define *local mutual information* of a cluster s_i^1 in $\mathbf{S}^1$ with respect to $\mathbf{S}^2$ as

$$LMI(s_i^1, \mathbf{S}^2) = \sum_{j=1}^{k} P_{ij} \log \frac{P_{ij}}{P_i P_j} \tag{4.10}$$

4.3.4 Vertex Variability for Reduced Graphs

Consider an ensemble $\mathcal{H} = \{H^1, H^2, \ldots, H^l\}$ of l reduced graphs, obtained from input graph G. These reduced graphs may be the result of multiple runs of a randomized graph reduction algorithm or the results of multiple graph reduction algorithms. We identify each reduced graph in the ensemble with a clustering of the vertices of the input graph G. Let $\mathcal{S} = \{\mathbf{S}^1, \mathbf{S}^2, \ldots, \mathbf{S}^l\}$ denote the ensemble of the clusterings corresponding to the reduced graphs in $\mathcal{H}$. Also associated with each clustering $\mathbf{S}^t \in \mathcal{S}$ is a coarsening matrix M^t. The structural variability of the supervertices of a reduced graph is the same as variability associated with the individual clusters of the associated clustering.

Without loss of generality, suppose we want to capture the variability associated with the supervertices of the reduced graph H^1 (with corresponding clustering $\mathbf{S}^1$). For each cluster s_i^1 in $\mathbf{S}^1$, we compute its local similarity with each of the remaining clusterings $\{\mathbf{S}^2, \ldots, \mathbf{S}^l\} \in \mathcal{S}$. Formally, let α_i^t denote the local similarity of $s_i^1 \in \mathbf{S}^1$ with $\mathbf{S}^t \in \mathcal{S}$ for $t = 2, \ldots, l$, i.e., α_i^t is equal to $LJ(s_i^1, \mathbf{S}^t)$, $LAR(s_i^1, \mathbf{S}^t)$, or $LMI(s_i^1, \mathbf{S}^t)$.

If H^1 has k supervertices, we can arrange their local similarities with respect to $\{\mathbf{S}^2, \ldots, \mathbf{S}^l\}$ in form of a $k \times l$ matrix. For example, if we measure local similarities with local Jaccard index, we can define Q_{LJ}^1 where $Q_{LJ}^1(i,t) = LJ(s_i^1, \mathbf{S}^t)$. Similarly, we can define matrices Q_{LAR}^1 and Q_{LMI}^1 using the local adjusted Rand index and local mutual information measures. We conjecture that the variability associated with a supervertex c_i^1 of H^1 is described by the standard deviation of the distribution of α_i^t for $t = 2, \ldots, l$, which is given by the i^{th} row of Q_{LJ}^1 (or Q_{LAR}^1 or Q_{LMI}^1).

4.3.5 Edge Variability in Graph Sparsification

Consider an ensemble $\mathcal{H} = \{H^1, H^2, \ldots, H^l\}$ of l sparsified graphs obtained from the same input graph $G(V, E, w)$. Graph sparsification algorithms reduce the size of the graph by deleting and reweighting its edges, while the vertices are unaffected. Therefore, all sparsified graphs in the ensemble have the same vertex set V. Let $w^t(i, j)$ denote the weight of the edge between vertices v_i and v_j in H^t. The empirical variability of an edge between $v_i, v_j \in V$ can be captured by the mean and the standard deviation of the edge weights $w^t(i, j)$ for $1 \leq t \leq l$. For some sparsification algorithms, we may be able to compute the edge variability analytically. Here, we give the analytical expressions for the expected edge weight and its standard deviation for graph sparsification based on effective resistances.

Given a graph G, the edge sampling process of Algorithm 3 in Chapter 2 can be viewed as a sequence of independent Bernoulli trials. For an edge $e \in E$, with a sampling probability p_e, a Bernoulli trial is a success if e is sampled; otherwise, it is a failure. Every time the edge e is sampled, it is added to the sparse graph K with weight w_e/qp_e. After q trials, the final weight of e in K is $X_e w_e/qp_e$, where X_e is the number of times e is sampled out of q trials. Since X_e captures the number of successes in q independent trials, it follows a binomial distribution with parameters q (i.e., the total number of trials) and p_e (i.e., the probability of success), denoted as $B(q, p_e)$. Therefore, $\mathbb{E}(X_e) = qp_e$, and $\mathrm{Var}(X_e) = qp_e(1 - p_e)$.

Therefore, the expected weight of an edge e in a reduced graph $K = (V, E', w')$ is

$$\mathbb{E}(w'_e) = \mathbb{E}(X_e w_e/qp_e) = (w_e/qp_e)\mathbb{E}(X_e) = w_e, \tag{4.11}$$

and its variance is given by

$$\begin{aligned} \mathrm{Var}(w'_e) &= \mathrm{Var}(X_e w_e/qp_e) = w_e^2/(q^2 p_e^2)\mathrm{Var}(X_e) \\ &= w_e^2(1 - p_e)/qp_e. \end{aligned} \tag{4.12}$$

The actual distribution of weights across edges in K may be unimodal or bimodal, depending on the parameter values.

4.3.6 Edge Variability for Graph Reduction

Now, suppose $\mathcal{H} = \{H^1, H^2, \ldots, H^l\}$ is an ensemble of l reduced graphs obtained using graph coarsening. Without loss of generality, suppose we want to capture the variability

associated with the superedges of the reduced graph H^1. To do that, first we need to establish correspondences between supervertices of H^1 and each of the remaining reduced graphs $\{H^2, \ldots, H^l\}$ in the ensemble $\mathcal{H}$. Consider reduced graphs H^1 and H^2 with corresponding clusterings $\mathbf{S}^1$ and $\mathbf{S}^2$. The problem of establishing correspondences between vertices of H^1 and H^2 is the same as the problem of a perfect matching in a complete bipartite graph. It can be solved in several different ways. We give two examples of matching strategies below.

For simplicity, we assume that H^1 and H^2 have the same number of supervertices. Recall that the clusters in $\mathbf{S}^1$ and $\mathbf{S}^2$ represent the supervertices of H^1 and H^2, respectively. Therefore, a matching between the supervertices of H^1 and H^2, is equivalent to a matching between the clusters of $\mathbf{S}^1$ and $\mathbf{S}^2$.

If the clusterings $\mathbf{S}^1$ and $\mathbf{S}^2$ are computed using spectral embedding (such as in spectral clustering), we can associate each cluster in $\mathbf{S}^1$ and $\mathbf{S}^2$ with its centroid in the embedded space. Then, we can assign correspondences between clusters of $\mathbf{S}^1$ and $\mathbf{S}^2$ based on the Euclidean distances between the centroids. Formally, let $x_i^1 \in \mathbb{R}^k$ be the centroid of cluster $s_i^1 \in \mathbf{S}^1$ in the k-dimensional embedding space. Similarly, let $x_j^2 \in \mathbb{R}^k$ be the centroid of cluster $s_j^2 \in \mathbf{S}^2$ in the k-dimensional embedding space. Then, for each cluster $s_i^1 \in \mathbf{S}^1$, we find a cluster $s_j^2 \in \mathbf{S}^2$ that minimizes the Euclidean distance $\|x_i^1 - x_j^2\|_2$.

When a spectral embedding is not available, we can assign correspondences based on the size of the overlap $s_i^1 \cap s_j^2$ between clusters $s_i^1 \in \mathbf{S}^1$ and $s_j^2 \in \mathbf{S}^2$. In this case, the best matching is given by a permutation that maximizes the trace of the confusion matrix F of $\mathbf{S}^1$ and $\mathbf{S}^2$. We can approximate this maximal matching using a simple greedy strategy. We begin with two sets, $A = \mathbf{S}^1 = \{s_1^1, s_2^1, \ldots, s_k^1\}$ and $B = \mathbf{S}^2 = \{s_1^2, s_2^2, \ldots, s_k^2\}$, each containing k elements. We find clusters $s_i^1 \in A$ and $s_j^2 \in B$, such that their intersection $s_i^1 \cap s_j^2$ is the largest for all $1 \leq i, j \leq k$, and assign a correspondence between s_i^1 and s_j^2. Then, we update sets A and B by removing s_i^1 and s_j^2, i.e., we set $A \leftarrow A \setminus \{s_i^1\}$ and $B \leftarrow B \setminus \{s_j^2\}$ so that both sets now contain $k - 1$ elements. We repeat these two steps until each cluster of $\mathbf{S}^2$ is matched to a cluster of $\mathbf{S}^1$.

Let π^t denote the correspondence between supervertices of H^1 and H^t for $t = 2, \ldots, l$. The structural variability associated with an edge (c_i^1, c_j^1) in G^1 is given by the mean and standard derivation of the weights associated with corresponding edges $\{(\pi^t(c_i^1), \pi^t(c_j^1)) \in H^t\}$ for $2 \leq t \leq l$.

4.4 Capturing Variability via Co-Clustering Probabilities

Here, we propose an alternative method to capture the structural variability associated with the vertices and edges of reduced graphs in an ensemble. This method builds on the ideas proposed by Fiol-Gonzalez et al. [47] and Gates et al. [48].

Consider a weighted undirected graph G, with vertex set $V = \{v_1, v_2, \ldots, v_n\}$ and number of vertices $|V| = n$. Let $\mathcal{H}$ and $\mathcal{S}$ denote the ensembles of reduced graphs and the corresponding clusterings. For each clustering $\mathbf{S}^t \in \mathcal{S}$, we define a cluster-induced graph whose $n \times n$ adjacency matrix A^t is defined as

$$A^t_{jk} = \begin{cases} 1 & \text{if } v_j, v_k \in s^t_i \text{ for some } s^t_i \in \mathbf{S}^t, \\ 0 & \text{otherwise.} \end{cases}$$

A^t indicates the co-clustering of vertex pairs in the clusters of clustering $\mathbf{S}^t$. Figure 4.1 illustrates an example. Let A^* denote the elementwise average of matrices A^t for $t = 1, \ldots, l$, i.e., $A^* = \frac{1}{l}\sum_{t=1}^{l} A^t$. A^*_{jk} is the empirically estimated co-clustering probability of vertices v_j and v_k across all clusterings in the ensemble.

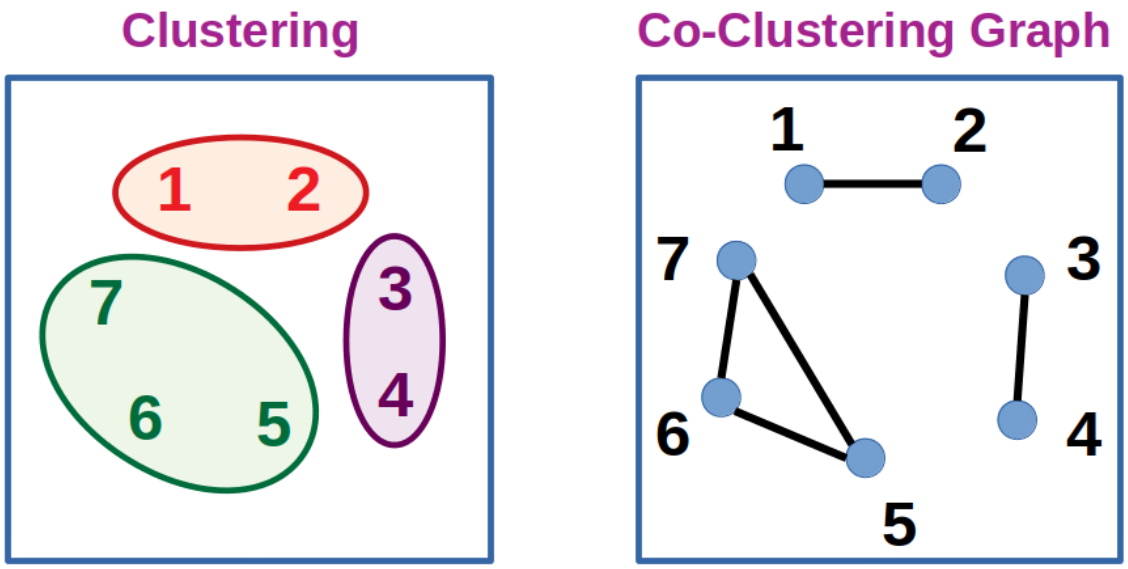

FIGURE. 4.1. Example of a co-clustering graph. On the left is a clustering and on the right is the corresponding co-clustering graph.

Now, for any given reduced graph H^t, with the corresponding clustering $\mathbf{S}^t = \{s_1^t, \ldots, s_k^t\}$, and the coarsening matrix M^t, we define a matrix Q^t

$$Q^t = (M^t)^T A^* M^t.$$

Q^t aggregates the pairwise co-clustering probabilities of vertices and edges of G contained within the supervertices and superedges of the reduced graph H^t. We can also normalize Q^t to obtain $\tilde{Q}^t$ as

$$\tilde{Q}^t = (D^t)^{-1/2} Q^t (D^t)^{-1/2}.$$

Here, D^t is a $k \times k$ diagonal matrix where $D^t(i,i)$ is the size of cluster $s_k^t \in \mathbf{S}^t$.

By construction, if all clusterings in $\mathcal{S}$ are identical, then Q^t and $\tilde{Q}^t$ are diagonal matrices for all $\mathbf{S}^t \in \mathcal{S}$. When the clusterings in $\mathcal{S}$ are not identical, we conjecture that the closer $\tilde{Q}^t$ is to a diagonal matrix, the more "stable" is the clustering $\mathbf{S}^t$. We can judge the local stability of the supervertices and superedges of the corresponding reduced graph by examining the diagonal and the off-diagonal entries of $\tilde{Q}^t$.

Consider a cluster $s_i^t \in \mathbf{S}^t$ such that every other clustering in the ensemble has a cluster with exactly the same members as s_i^t. We call such a cluster a *stable* cluster. Any given pair of vertices in a stable cluster is clustered together across all clusterings in $\mathcal{S}$, and therefore, has a co-clustering probability of 1. Therefore, by construction, the diagonal entry $\tilde{Q}^t(i,i)$ corresponding to s_i^t is $|s_i^t| - 1$, where $|s_i^t|$ is the size of the cluster. When a cluster s_i^t is not stable, we can obtain a measure of its variability by examining the difference between $\tilde{Q}^t(i,i)$ and $|s_i^t| - 1$ (see Section 4.5.2 for an example).

The off-diagonal entries of $\tilde{Q}^t$ are the aggregated co-clustering probabilities of the pairs of vertices that are split across different clusters by clustering $\mathbf{S}^t$, normalized by the cluster sizes. If s_i^t is a stable cluster, a vertex $v_p \in s_i^t$ is never co-clustered with a vertex $v_q \in V \setminus s_i^t$ in any clustering of $\mathcal{S}$. Therefore, by construction, $\tilde{Q}^t(i,j) = 0$ for all $j \neq i$, indicating an isolated supervertex in the reduced graph. When s_i^t is not a stable cluster, the corresponding off-diagonal entries of $\tilde{Q}^t$ are nonzero. A large off-diagonal entry $\tilde{Q}^t(i,j)$ implies that vertex pairs with high co-clustering probability are split across clusters s_i^t and s_j^t of $\mathbf{S}^t$ instead of being clustered together. Therefore, large off-diagonal entries indicate greater instability in the memberships of the corresponding clusters.

4.5 Discussion

We propose to apply the methods described in Section 4.3 and Section 4.4 to visualize the variability of supervertices and superedges of a reduced graph from an ensemble. Here, we discuss potential drawbacks of the local similarity-based variability measures and the advantages of the co-clustering-based variability.

4.5.1 Drawbacks of Local Similarity-Based Method

We define local similarities as the contributions of individual clusters to the global similarity, i.e., $Q^1_{LJ}(i,t)$ is the contribution of the cluster $s^1_i \in \mathbf{S}^1$ to the global similarity $J(\mathbf{S}^1, \mathbf{S}^t)$. We conjecture that the standard deviation of the i^{th} row is an indicator of the variability of the i^{th} vertex of the reduced graph H^1. However, this conjecture may not hold in some cases.

Consider the ensemble $\mathcal{S}$ of clusterings and suppose we want to measure the local variability for clusters of $\mathbf{S}^1$. The local similarity measures for any cluster s^1_i, as defined in Section 4.3, are not independent of variations in other clusters in $\mathbf{S}^1$ and $\mathbf{S}^t$. In the case of Equation 4.5, this dependence comes from n_{10} and n_{01} in the denominator. In the case of Equation 4.8, the dependence comes from r_1, r_2, r_3, and r_4.

To illustrate this point, consider the ensemble of four clusterings in Figure 4.2. The matrix $Q^{(a)}_{LJ}$ measures the local similarities of clusters in clustering **(a)** using the local Jaccard index.

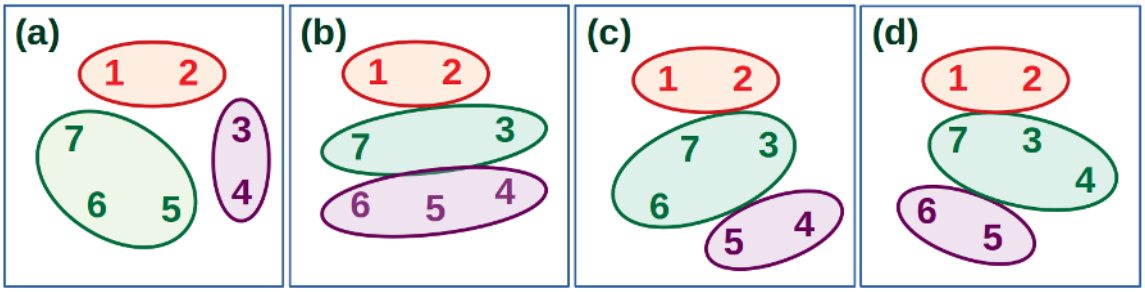

FIGURE. 4.2. Example of a clustering ensemble.

$$Q_{LJ}^{(a)} = \begin{array}{c} \\ \{1,2\} \\ \{3,4\} \\ \{5,6,7\} \end{array} \begin{array}{c} \begin{array}{ccc} (b) & (c) & (d) \end{array} \\ \begin{pmatrix} 0.1042 & \text{-}0.02084 & 0.1667 \\ 0.1042 & \text{-}0.02084 & 0.1667 \\ 0.1190 & 0.1190 & 0.1905 \end{pmatrix} \end{array}$$

The standard deviations of the rows corresponding to clusters $\{1,2\}$, $\{3,4\}$, and $\{5,6,7\}$ are 0.0779, 0.0779, and 0.0337, respectively. Cluster $\{1,2\}$ is a "stable" cluster that appears in all four clusterings. However, it has a nonzero local variability. Cluster $\{1,2\}$ also does not have the lowest variability among the three clusters of **(a)**. Even though this behavior may appear counterintuitive, it may not be entirely undesirable. We may want the variability of a supervertex to take into account the variations in its neighborhood.

To measure the variability of superedges using local similarity measures, we need to find correspondences between clusters. The variability of superedges depends on how we assign these correspondences. There are several ways to assign correspondences. In Section 4.3, we describe two simple strategies. However, in some cases, finding good mappings between clusters may not be possible. For example, consider the two clusterings given in Figure 4.3. The confusion matrix for the two clusterings is a matrix of all ones. All possible mappings between clusters of the two clusterings have the same cost. These clusterings illustrate the difficulties in finding correspondences.

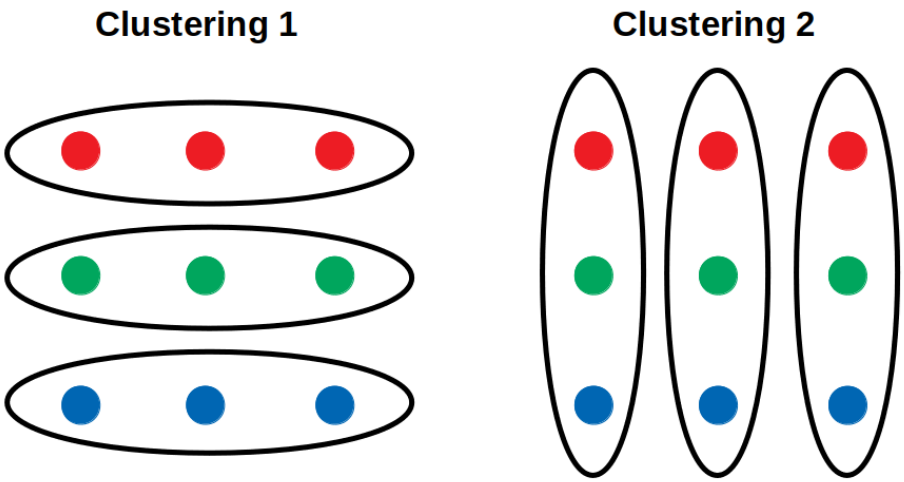

FIGURE. 4.3. Two clusterings illustrating difficulties in assigning correspondences.

4.5.2 Advantages of Co-Clustering-Based Method

We believe that the co-clustering-based method to capture structural variability described in Section 4.4 can overcome some of the drawbacks of the local similarity-based methods. For instance, consider again the clusterings given in Figure 4.2. To capture the variability of the clusters of clustering **(a)** using the co-clustering based method, we compute the matrices $Q^{(a)}$ and $\tilde{Q}^{(a)}$ as described in Section 4.4.

$$Q^{(a)} = \begin{array}{c} \\ \{1,2\} \\ \{3,4\} \\ \{5,6,7\} \end{array} \begin{array}{c} \begin{array}{ccc} \{1,2\} & \{3,4\} & \{5,6,7\} \end{array} \\ \begin{pmatrix} 2.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 2.0 \\ 0.0 & 2.0 & 2.5 \end{pmatrix} \end{array}$$

$$\tilde{Q}^{(a)} = \begin{array}{c} \\ \{1,2\} \\ \{3,4\} \\ \{5,6,7\} \end{array} \begin{array}{c} \begin{array}{ccc} \{1,2\} & \{3,4\} & \{5,6,7\} \end{array} \\ \begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.5 & 0.8165 \\ 0.0 & 0.8165 & 0.8334 \end{pmatrix} \end{array}$$

For cluster $\{1,2\}$, the corresponding diagonal value of $\tilde{Q}^{(a)}$ is 1 and the off-diagonal values are all 0, identifying it as a stable cluster. Cluster $\{3,4\}$, on the other hand, is identified as an unstable cluster by the low diagonal value of 0.5 and the high off-diagonal value of 0.8165. The high off-diagonal value of 0.8165 in $\tilde{Q}^{(a)}$ indicates that a pair of vertices with high co-clustering probability is split across clusters $\{3,4\}$ and $\{5,6,7\}$. Looking at the clusterings in Figure 4.2, we see that the pair of vertices is 3 and 7, which is co-clustered in three of the four clusterings.

4.5.3 Conclusion

The methods presented in this chapter help us understand the structural variability in an ensemble of reduced graphs. This work has direct applications in visualization. We may also be able to use the structural variability measures, particularly the co-clustering based measures, to compare different graph reduction and graph clustering algorithms.

For many graph reduction and graph clustering algorithms, it may be difficult to derive an analytical measure that quantifies uncertainty. The work presented in this chapter is the first step toward developing a more general framework for uncertainty quantification for such algorithms.

CHAPTER 5

CONCLUSION AND A VISION OF FUTURE RESEARCH

The past few decades have seen an explosion of complex forms of data. The work presented in this dissertation demonstrates that TDA can be particularly useful in studying complex forms of data such as trees, graphs, simplicial complexes and hypergraphs. Our vision is to develop new methodologies for data science with an end-to-end integration of TDA with modern machine learning pipelines. Figure 5.1 shows how our existing work fits into a machine learning pipeline.

5.1 Using TDA to Handle Complex Input Data

In Chapter 3 and Chapter 4 of this dissertation, we work with ensembles of graphs which are becoming increasingly common. However, machine learning models are typically

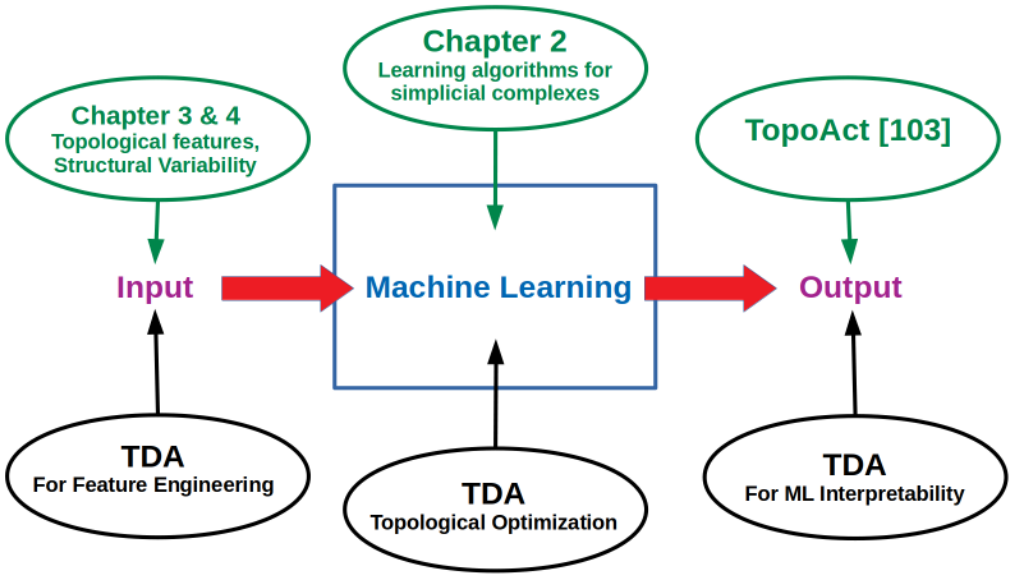

FIGURE. 5.1. Research vision and our contributions.

designed for data in vector forms. Therefore, we focus on extracting topological features with vector representations from graphs.

In Chapter 3, we show how persistent homology, a core TDA technique, can be used to extract topological features from brain networks and utilize them in various machine learning applications. Although persistence diagrams are non-Euclidean objects themselves, we describe ways to *linearize* them, either explicitly via vector embedding, or implicitly via kernels. This approach also allows us to combine topological features with traditional features.

The persistence diagrams considered in Chapter 3 are obtained using a single scale parameter. Although persistence diagrams for two or more parameters do not exist [22], researchers have begun exploring the theoretical and algorithmic aspects of multiparameter persistent homology [22, 74, 75]. For instance, in [36], Corbet et al. define a kernel for multiparameter persistence, and show that this kernel can be approximated efficiently. This is an active area of research with potential applications in multivariate data analysis. Apart from persistence diagrams, topological summaries such as Euler characteristic curves [76, 77, 83] and mapper graphs [93, 80, 30] have also been used in various machine learning applications.

TDA provides several tools to summarize and characterize complex forms of data. We are interested in finding new application domains and new ways to utilize different topological summaries for machine learning. We believe that topological summaries may lead to a unifying framework that allows modern machine learning pipelines to handle complex data types such as trees, graphs, simplicial complexes, and hypergraphs.

5.2 TDA to Improve the Learning Process

Objects like simplicial complexes or hypergraphs allow us to model higher order interactions among three or more data points. These interactions are essential in real-world situations, such as social relationships, coauthorships, or protein interactions. However, algorithms that can exploit the structural information encoded in these objects have rarely been explored in the machine learning literature.

In Chapter 2, we present spectral algorithms for simplicial complexes, such as sparsification, clustering, and label propagation. We are interested in extending our work

in the setting of deep learning, for example, to extend graph convolution networks to simplicial complexes and hypergraphs. Our work in Chapter 2 provides a basis for a unifying framework of learning algorithms that can leverage the complex relationships between data points encoded in graphs, simplicial complexes, and hypergraphs.

Researchers have started exploring ways to incorporate topological information into deep learning process itself. For example, Liu et al. [78] proposed a persistent convolutional neural network for audio signals to perform tasks such as classification and music tagging, whereas Clough et al. [33] used topological priors based on desired sequences of Betti numbers for lung segmentation in cardiac MRI. Bruel-Gabrielsson et al. [16] incorporated topological constraints based on persistence diagrams in the optimization algorithm to reconstruct surfaces from point cloud data. Poulenard et al. [101] described how persistence diagrams can be used in optimization over real-valued functions for shape matching.

Many of these methods were tailored for specific applications. However, the key idea behind all these methods is similar: to design functions that are differentiable with respect to topological measurements. For example, Bruel-Gabrielsson et al. [17] proposed a general framework for incorporating topological information in deep learning in the form of a differentiable topology layer. The utility and flexibility of this framework is demonstrated by applying it (a) to regularize the weights of machine learning models, (b) to construct a loss function for deep generative models that incorporates topological priors, and (c) to perform adversarial attacks on models trained with persistence features.

The existing methods of incorporating topological information in the learning process have mostly focused on topological features of individual instances. Instead, Chen et al. [26] proposed a method of regularization over topological complexity of the classifier itself by incorporating the importance of topological features such as connected components and loops. Building on this idea, we are interested in designing topological constraints for the latent spaces of network layers. Such constraints may be useful, for example, in promoting latent spaces with specific topological characteristics. This line of research presents many exciting opportunities as well as theoretical and algorithmic challenges. We will address these challenges in incorporating topological optimizations into the learning process and explore applications of such optimizations in new domains.

5.3 Using TDA to Explain Complex Learning Models

Understanding and explaining model behavior has become crucial with the emergence of deep learning. In [103], we described a way to utilize topological summaries to understand the organization of the activation space for trained deep learning models. Topological summaries can also be useful in characterizing the shape and the structure of the model's parameter spaces. We can further extend these techniques to understand how the activation spaces and the parameter spaces change across layers, and across training epochs, and utilize this knowledge to guide network design and the learning process. We can use topological summaries to understand the model's response to adversarial inputs and improve the model's robustness.

5.4 Final Remarks

Our ultimate goal is to bring together the fields of TDA and machine learning. Achieving this goal will require fundamental research in areas of computational topology, computational geometry, algorithms, graph theory, and machine learning. By integrating ideas from TDA with modern machine learning, we hope to provide

- A unified learning framework to handle complex (e.g., non-Euclidean) data types,
- Topology-aware optimizations for machine learning, and
- Valuable methods to understand model behaviors.

The work described in this dissertation represents the initial steps in each of these directions. By combining TDA with machine learning, our goal is to transform the data science landscape, with applications beyond medical imaging and neuroscience.

REFERENCES

[1] A. E. Abbott, A. Nair, C. L. Keown, M. Datko, A. Jahedi, I. Fishman, and R. A. Müller, *Patterns of atypical functional connectivity and behavioral links in autism differ between default, salience, and executive networks*, Cerebral Cortex, 26 (2016), pp. 4034–4045.

[2] A. Abraham, M. P. Milham, A. Di Martino, R. C. Craddock, D. Samaras, B. Thirion, and G. Varoquaux, *Deriving reproducible biomarkers from multi-site resting-state data: An autism-based example*, NeuroImage, 147 (2017), pp. 736–745.

[3] H. Adams, T. Emerson, M. Kirby, R. Neville, C. Peterson, P. Shipman, S. Chepushtanova, E. Hanson, F. Motta, and L. Ziegelmeier, *Persistence images: A stable vector representation of persistent homology*, Journal of Machine Learning Research, 18 (2017), pp. 218–252.

[4] M. A. Aghdam, A. Sharifi, and M. M. Pedram, *Combination of rs-fMRI and sMRI data to discriminate autism spectrum disorders in young children using deep belief network*, Journal of Digital Imaging, 31 (2018), pp. 895–903.

[5] M. Altaye, S. K. Holland, M. Wilke, and C. Gaser, *Infant brain probability templates for MRI segmentation and normalization*, NeuroImage, 43 (2008), pp. 721–730.

[6] R. Andersen, F. Chung, and K. Lang, *Local graph partitioning using PageRank vectors*, in 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), IEEE, 2006, pp. 475–486.

[7] R. Andersen and K. J. Lang, *Communities from seed sets*, International Conference on the World Wide Web, (2006), pp. 223–232.

[8] J. S. Anderson, J. A. Nielsen, A. L. Froehlich, M. B. DuBray, T. J. Druzgal, A. N. Cariello, J. R. Cooperrider, B. A. Zielinski, C. Ravichandran, P. T. Fletcher, A. L. Alexander, E. D. Bigler, N. Lange, and J. E. Lainhart, *Functional connectivity magnetic resonance imaging classification of autism*, Brain, 134 (2011), pp. 3742–3754.

[9] K. L. Anderson, J. S. Anderson, S. Palande, and B. Wang, *Topological data analysis of functional MRI connectivity in time and space domains*, in Connectomics in NeuroImaging, Springer International Publishing, 2018, pp. 67–77.

[10] F. Arrigo, D. J. Higham, and F. Tudisco, *A framework for second-order eigenvector centralities and clustering coefficients*, Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, 476 (2020). doi:10.1098/rspa.2019.0724.

[11] M. Assaf, K. Jagannathan, V. D. Calhoun, L. Miller, M. C. Stevens, R. Sahl, J. G. O'Boyle, R. T. Schultz, and G. D. Pearlson, *Abnormal functional connectivity of default mode sub-networks in autism spectrum disorder patients*, Neuroimage, 53 (2010), pp. 247–256.

[12] L. Backstrom and J. Leskovec, *Supervised random walks: Predicting and recommending links in social networks*, in Proceedings of the fourth ACM international conference on Web search and data mining - WSDM'11, Association for Computing Machinery, 2011, pp. 635–644.

[13] J. Batson, D. A. Spielman, N. Srivastava, and S.-H. Teng, *Spectral sparsification of graphs: Theory and algorithms*, Communications of the ACM, 56 (2013), pp. 87–94.

[14] A. A. Benczúr and D. R. Karger, *Approximating s-t minimum cuts in õ(n2) time*, in Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing - STOC'96, ACM Press, 1996, pp. 47–55.

[15] A. R. Benson, D. F. Gleich, and J. Leskovec, *Tensor spectral clustering for partitioning higher-order network structures*, in Proceedings of the 2015 SIAM International Conference on Data Mining, Society for Industrial and Applied Mathematics, June 2015, pp. 118–126.

[16] R. Brüel-Gabrielsson, V. Ganapathi-Subramanian, P. Skraba, and L. J. Guibas, *Topology-aware surface reconstruction for point clouds.* Preprint, arXiv:1811.12543 [cs.CG], 2018.

[17] R. Brüel-Gabrielsson, B. J. Nelson, A. Dwaraknath, P. Skraba, L. J. Guibas, and G. Carlsson, *A topology layer for machine learning.* Preprint, arXiv:1905.12200 [cs.LG], 2019.

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

[19] E. Bullmore and O. Sporns, *Complex brain networks: Graph theoretical analysis of structural and functional systems*, Nature Reviews Neuroscience, 10 (2009), pp. 186–198.

[20] C. Cameron, B. Yassine, C. Carlton, C. Francois, E. Alan, J. András, K. Budhachandra, L. John, L. Qingyang, M. Michael, Y. Chaogan, and B. Pierre, *The Neuro Bureau Preprocessing Initiative: Open sharing of preprocessed neuroimaging data and derivatives*, Frontiers in Neuroinformatics, 7 (2013). doi:10.3389/conf.fninf.2013.09.00041.

[21] G. Carlsson, *Topology and data*, Bulletin of the American Mathematical Society, 46 (2009), pp. 255–308.

[22] G. Carlsson and A. Zomorodian, *The theory of multidimensional persistence*, Discrete & Computational Geometry, 42 (2009), pp. 71–93.

[23] M. Carrière, M. Cuturi, and S. Oudot, *Sliced Wasserstein kernel for persistence diagrams*, Proceedings of the 34th International Conference on Machine Learning, 70 (2017), pp. 664–673.

[24] C. J. Carstens and K. J. Horadam, *Persistent homology of collaboration networks*, Mathematical Problems in Engineering, 2013 (2013). doi:10.1155/2013/815035.

[25] A. K. Chandra, P. Raghavan, W. L. Ruzzo, R. Smolensky, and P. Tiwari, *The electrical resistance of a graph captures its commute and cover times*, Computational Complexity, 6 (1996), pp. 312–340.

[26] C. Chen, X. Ni, Q. Bai, and Y. Wang, *A topological regularizer for classifiers via persistent homology*, in Proceedings of Machine Learning Research, vol. 89, 2019, pp. 2573–2582.

[27] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, *Harp: Hierarchical representation learning for networks*, 32nd AAAI Conference on Artificial Intelligence, Web (2018). https://aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16273 (10 Dec 2020).

[28] J. E. Chen, C. Chang, M. D. Greicius, and G. H. Glover, *Introducing co-activation pattern metrics to quantify spontaneous brain network dynamics*, NeuroImage, 111 (2015), pp. 476–488.

[29] F. Chierichetti, R. Kumar, P. Raghavan, and T. Sarlos, *Are web users really markovian?*, in Proceedings of the 21st International Conference on World Wide Web - WWW'12, ACM Press, 2012, pp. 609–618.

[30] D. H. Chitwood, M. Eithun, E. Munch, and T. Ophelders, *Topological mapper for 3d volumetric images*, in Lecture Notes in Computer Science, Springer International Publishing, 2019, pp. 84–95.

[31] F. Chung, *Spectral Graph Theory*, American Mathematical Society, 1996.

[32] K. L. Clarkson and D. P. Woodruff, *Low rank approximation and regression in input sparsity time*, Journal of the ACM, 63 (2012), pp. 1–45.

[33] J. R. Clough, I. Oksuz, N. Byrne, J. A. Schnabel, and A. P. King, *Explicit topological priors for deep-learning based image segmentation using persistent homology*, in Lecture Notes in Computer Science, Springer International Publishing, 2019, pp. 16–28.

[34] M. B. Cohen, B. T. Fasy, G. L. Miller, A. Nayyeri, R. Peng, and N. Walkington, *Solving 1-Laplacians in nearly linear time: Collapsing and expanding a topological ball*, in Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms - SODA'14, Society for Industrial and Applied Mathematics, 2014, pp. 204–216.

[35] M. B. Cohen, R. Kyng, G. L. Miller, J. W. Pachocki, R. Peng, A. B. Rao, and S. C. Xu, *Solving SDD linear systems in nearly mlog1/2n time*, in Proceedings of the 46th Annual ACM Symposium on Theory of Computing - STOC'14, ACM Press, 2014, pp. 343–352.

[36] R. Corbet, U. Fugacci, M. Kerber, C. Landi, and B. Wang, *A kernel for multi-parameter persistent homology.* Preprint, arXiv:1809.10231 [cs.LG], 2018.

[37] E. Courchesne, K. Pierce, C. M. Schumann, E. Redcay, J. A. Buckwalter, D. P. Kennedy, and J. Morgan, *Mapping early brain development in autism*, Neuron, 56 (2007), pp. 399–413.

[38] S. I. Daitch and D. A. Spielman, *Support-graph preconditioners for 2-dimensional trusses*. Preprint, arXiv:0703119 [cs.NA], 2007.

[39] A. Di Martino, C. G. Yan, Q. Li, E. Denio, F. X. Castellanos, K. Alaerts, J. S. Anderson, M. Assaf, S. Y. Bookheimer, M. Dapretto, B. Deen, S. Delmonte, I. Dinstein, B. Ertl-Wagner, D. A. Fair, L. Gallagher, D. P. Kennedy, C. L. Keown, C. Keysers, J. E. Lainhart, C. Lord, B. Luna, V. Menon, N. J. Minshew, C. S. Monk, S. Mueller, R. A. Müller, M. B. Nebel, J. T. Nigg, K. O'Hearn, K. A. Pelphrey, S. J. Peltier, J. D. Rudie, S. Sunaert, M. Thioux, J. M. Tyszka, L. Q. Uddin, J. S. Verhoeven, N. Wenderoth, J. L. Wiggins, S. H. Mostofsky, and M. P. Milham, *The autism brain imaging data exchange: Towards a large-scale evaluation of the intrinsic brain architecture in autism*, Molecular Psychiatry, 19 (2014), pp. 659–667.

[40] R. Diestel, *Graph Theory*, Springer Graduate Texts in Mathematics, 2000.

[41] D. Dotterrer and M. Kahle, *Coboundary expanders*, Journal of Topology and Analysis, 4 (2012), pp. 499–514.

[42] P. G. Doyle and J. L. Snell, *Random Walks and Electric Networks*, The Mathematical Association of America, 1984.

[43] P. Drineas, M. Magdon-Ismail, M. W. Mahoney, and D. P. Woodruff, *Fast approximation of matrix coherence and statistical leverage*, Journal of Machine Learning Research, 13 (2012), pp. 3475–3506.

[44] H. Edelsbrunner and J. Harer, *Persistent homology – A survey*, Contemporary Mathematics, 453 (2008), pp. 257–282.

[45] H. Edelsbrunner, D. Letscher, and A. Zomorodian, *Topological persistence and simplification*, Discrete & Computational Geometry, 4 (2002), pp. 511–533.

[46] D. A. Fair, A. L. Cohen, N. U. F. Dosenbach, J. A. Church, F. M. Miezin, D. M. Barch, M. E. Raichle, S. E. Petersen, and B. L. Schlaggar, *The maturing architecture of the brain's default network*, Proceedings of the National Academy of Sciences, 105 (2008), pp. 4028–4032.

[47] S. Fiol-González, C. de Almeida, A. Rodrigues, S. Barbosa, and H. Lopes, *Visual exploration tools for ensemble clustering analysis*, Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 3 (2019), pp. 259–266.

[48] A. J. Gates, I. B. Wood, W. P. Hetrick, and Y.-Y. Ahn, *Element-centric clustering comparison unifies overlaps and hierarchy*, Scientific Reports, 9 (2019), p. 8574. doi:10.1038/s41598-019-44892-y.

[49] A. Ghosh, S. Boyd, and A. Saberi, *Minimizing effective resistance of a graph*, SIAM Review, 50 (2008), pp. 37–66.

[50] R. Ghrist, *Barcodes: The persistent topology of data*, Bulletin of the American Mathematical Society, 45 (2008), pp. 61–75.

[51] ——, *Elementary Applied Topology*, CreateSpace, United States, 2014.

[52] C. Giusti, E. Pastalkova, C. Curto, and V. Itskov, *Clique topology reveals intrinsic geometric structure in neural correlations*, Proceedings of the National Academy of Sciences, 112 (2015), pp. 13455–13460.

[53] D. F. Gleich, *PageRank beyond the web*, SIAM Review, 57 (2015), pp. 321–363.

[54] D. F. Gleich, L.-H. Lim, and Y. Yu, *Multilinear PageRank*, SIAM Journal on Matrix Analysis and Applications, 36 (2015), pp. 1507–1541.

[55] L. J. Grady and J. R. Polimeni, *Discrete Calculus: Applied Analysis on Graphs for Computational Science*, Springer Verlag London, 2010.

[56] A. Gundert and M. Szedlák, *Higher dimensional discrete Cheeger inequalities*, Journal of Computational Geometry, 6 (2015), pp. 54–71.

[57] X. Guo, K. C. Dominick, A. A. Minai, H. Li, C. A. Erickson, and L. J. Lu, *Diagnosing autism spectrum disorder from brain resting-state functional connectivity patterns using a deep neural network with a novel feature selection method*, Frontiers in Neuroscience, 11 (2017), p. 460. doi:10.3389/fnins.2017.00460.

[58] D. Harel and Y. Koren, *A fast multi-scale method for drawing large graphs*, Journal of Graph Algorithms and Applications, 6 (2002), pp. 179–202.

[59] A. S. Heinsfeld, A. R. Franco, R. C. Craddock, A. Buchweitz, and F. Meneguzzi, *Identification of autism spectrum disorder using deep learning and the ABIDE dataset*, NeuroImage: Clinical, 17 (2018), pp. 16–23.

[60] C. Hofer, R. Kwitt, M. Niethammer, and A. Uhl, *Deep learning with topological signatures*, Advances in Neural Information Processing Systems, (2017), pp. 1634–1644.

[61] D. Horak and J. Jost, *Spectra of combinatorial Laplace operators on simplicial complexes*, Advances in Mathematics, 244 (2013), pp. 303–336.

[62] Y. Jin, A. Loukas, and J. F. JaJa, *Graph coarsening with preserved spectral properties*. Preprint, arXiv:1802.04447 [cs.SI], 2019.

[63] L. Kanari, P. Dłotko, M. Scolamiero, R. Levi, J. Shillcock, K. Hess, and H. Markram, *A topological representation of branching neuronal morphologies*, Neuroinformatics, 16 (2017), pp. 3–13.

[64] J. A. Kelner, L. Orecchia, A. Sidford, and Z. A. Zhu, *A simple, combinatorial algorithm for solving sdd systems in nearly-linear time*, ACM Symposium on Theory of Computing, (2013), pp. 911–920.

[65] M. C. Kennedy and A. O'Hagan, *Bayesian calibration of computer models*, Journal of the Royal Statistical Society, Series B, 63 (2001), pp. 425–464.

[66] C. Klymko, D. Gleich, and T. G. Kolda, *Using triangles to improve community detection in directed networks*. Preprint, arXiv:1404.5874, Apr. 2014.

[67] G. Kusano, K. Fukumizu, and Y. Hiraoka, *Kernel method for persistence diagrams via kernel embedding and weight factor*, Journal of Machine Learning Research, 18 (2017), pp. 6947–6987.

[68] B. C. Kwon, B. Eysenbach, J. Verma, K. Ng, C. deFilippi, W. F. Stewart, and A. Perer, *Clustervision: Visual supervision of unsupervised clustering*, IEEE Transactions on Visualization and Computer Graphics, 24 (2018), pp. 142–151.

[69] R. Kyng, R. Peng, R. Schwieterman, and P. Zhang, *Incomplete nested dissection*, in Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, 2018, pp. 404–417.

[70] F. Lan, S. Palande, M. Young, and B. Wang, *Uncertainty quantification in graph reduction*. In preparation, 2020.

[71] T. Le and M. Yamada, *Persistence Fisher kernel: A Riemannian manifold kernel for persistence diagrams*, Advances in Neural Information Processing Systems, 31 (2018), pp. 10028–10039.

[72] H. Lee, M. K. Chung, H. Kang, B.-N. Kim, and D. S. Lee, *Discriminative persistent homology of brain networks*, in 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, 2011, pp. 841–844. doi:10.1109/ISBI.2011.5872535.

[73] H. Lee, H. Kang, M. K. Chung, B.-N. Kim, and D. S. Lee, *Persistent brain network homology from the perspective of dendrogram*, IEEE Transactions on Medical Imaging, 31 (2012), pp. 2267–2277.

[74] M. Lesnick and M. Wright, *Interactive visualization of 2-d persistence modules*. Preprint, arXiv:1512.00180 [math.AT], 2015.

[75] ——, *Computing minimal presentations and bigraded betti numbers of 2-parameter persistent homology*. Preprint, arXiv:1902.05708 [math.AT], 2019.

[76] M. Li, H. An, R. Angelovici, C. Bagaza, A. Batushansky, L. Clark, V. Coneva, M. J. Donoghue, E. Edwards, D. Fajardo, H. Fang, M. H. Frank, T. Gallaher, S. Gebken, T. Hill, S. Jansky, B. Kaur, P. C. Klahs, L. L. Klein, V. Kuraparthy, J. Londo, Z. Migicovsky, A. Miller, R. Mohn, S. Myles, W. C. Otoni, J. C. Pires, E. Rieffer, S. Schmerler, E. Spriggs, C. N. Topp, A. V. Deynze, K. Zhang, L. Zhu, B. M. Zink, and D. H. Chitwood, *Topological data analysis as a morphometric method: Using persistent homology to demarcate a leaf morphospace*, Frontiers in Plant Science, 9 (2018), p. 553. doi:10.3389/fpls.2018.00553.

[77] M. Li, M. H. Frank, V. Coneva, W. Mio, D. H. Chitwood, and C. N. Topp, *The persistent homology mathematical framework provides enhanced genotype-to-phenotype associations for plant morphology*, Plant Physiology, 177 (2018), pp. 1382–1395.

[78] J.-Y. Liu, S.-K. Jeng, and Y.-H. Yang, *Applying topological persistence in convolutional neural network for music audio signals*. Preprint, arXiv:1608.07373 [cs.NE], 2016.

[79] A. Lubotzky, *Ramanujan complexes and high dimensional expanders*, Japanese Journal of Mathematics, 9 (2014), pp. 137–169.

[80] P. Y. Lum, G. Singh, A. Lehman, T. Ishkanov, M. Vejdemo-Johansson, M. Alagappan, J. Carlsson, and G. Carlsson, *Extracting insights from the shape of complex data using topology*, Scientific Reports, 3 (2013), p. 1236. doi:10.1038/srep01236.

[81] C. J. Lynch, L. Q. Uddin, K. Supekar, A. Khouzam, J. Phillips, and V. Menon, *Default mode network in childhood autism: Posteromedial cortex heterogeneity and relationship with social deficits*, Biological Psychiatry, 74 (2013), pp. 212–219.

[82] M. W. Mahoney, L. Orecchia, and N. K. Vishnoi, *A local spectral method for graphs: With applications to improving graph partitions and exploring data graphs locally*, Journal of Machine Learning Research, 13 (2012), pp. 2339–2365.

[83] C. A. McAllister, M. R. McKain, M. Li, B. Bookout, and E. A. Kellogg, *Specimen-based analysis of morphology and the environment in ecologically dominant grasses: the power of the herbarium*, Philosophical Transactions of the Royal Society B: Biological Sciences, 374 (2018), p. 403. doi:10.1098/rstb.2017.0403.

[84] A. R. McIntosh, F. L. Bookstein, J. V. Haxby, and C. L. Grady, *Spatial pattern analysis of functional brain images using partial least squares*, NeuroImage, 3 (1996), pp. 143–157.

[85] M. Meilă, *Comparing clusterings – An information based distance*, Journal of Multivariate Analysis, 98 (2007), pp. 873–895.

[86] M. R. Meiss, F. Menczer, S. Fortunato, A. Flammini, and A. Vespignani, *Ranking web sites with real user traffic*, in Proceedings of the International Conference on Web Search and Web Data Mining - WSDM'08, ACM Press, 2008, p. 65–76. doi:10.1145/1341531.1341543.

[87] L. Minkova, S. B. Eickhoff, A. Abdulkadir, C. P. Kaller, J. Peter, E. Scheller, J. Lahr, R. A. Roos, A. Durr, B. R. Leavitt, S. J. Tabrizi, S. Klöppel, and T. R. A. C. K.-H. D. Investigators, *Large-scale brain network abnormalities in Huntington's disease revealed by structural covariance*, Human Brain Mapping, 37 (2016), pp. 67–80.

[88] M. Montembeault, I. Rouleau, J.-S. Provost, and S. M. Brambati, *Altered gray matter structural covariance networks in early stages of Alzheimer's disease*, Cerebral Cortex, 26 (2016), p. 2650. doi:10.1093/cercor/bhv105.

[89] S. Monti, P. Tamayo, J. Mesirov, and T. Golub, *Consensus clustering: A resampling-based methodfor class discovery and visualization of gene expression microarray data*, Machine Learning, 52 (2003), pp. 91–118.

[90] S. Mukherjee and J. Steenbergen, *Random walks on simplicial complexes and harmonics*, Random Structures & Algorithms, 49 (2016), pp. 379–405.

[91] J. R. Munkres, *Elements of Algebraic Topology*, Addison-Wesley, Redwood City, CA, USA, 1984.

[92] A. Y. Ng, M. I. Jordan, and Y. Weiss, *On spectral clustering: Analysis and an algorithm*, in Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic, NIPS'01, Vancouver, British Columbia, Canada, 2001, MIT Press, pp. 849–856.

[93] M. Nicolau, A. J. Levine, and G. Carlsson, *Topology based data analysis identifies a subgroup of breast cancers with a unique mutational profile and excellent survival*, Proceedings of the National Academy of Sciences, 108 (2011), pp. 7265–7270.

[94] J. A. Nielsen, B. A. Zielinski, P. T. Fletcher, A. L. Alexander, N. Lange, E. D. Bigler, J. E. Lainhart, and J. S. Anderson, *Multisite functional connectivity MRI classification of autism: ABIDE results*, Frontiers in Human Neuroscience, 7 (2013), p. 599. doi:10.3389/fnhum.2013.00599.

[95] M. Nolte, E. Gal, H. Markram, and M. W. Reimann, *Impact of higher order network structure on emergent cortical activity*, Network Neuroscience, 4 (2020), pp. 292–314.

[96] B. Osting, C. D. White, and E. Oudet, *Minimal Dirichlet energy partitions for graphs*, SIAM Journal on Scientific Computing, 36 (2014), pp. A1635–A1651.

[97] O. Parzanchevski and R. Rosenthal, *Simplicial complexes: Spectrum, homology and random walks*, Random Structures & Algorithms, 50 (2016), pp. 225–261.

[98] O. Parzanchevski, R. Rosenthal, and R. J. Tessler, *Isoperimetric inequalities in simplicial complexes*, Combinatorica, 36 (2016), pp. 195–227.

[99] G. Petri, P. Expert, F. Turkheimer, R. Carhart-Harris, D. Nutt, P. J. Hellyer, and F. Vaccarino, *Homological scaffolds of brain functional networks*, Journal of The Royal Society Interface, 11 (2014), p. 873. doi:10.1098/rsif.2014.0873.

[100] C. Poletto, M. Tizzoni, and V. Colizza, *Human mobility and time spent at destination: Impact on spatial epidemic spreading*, Journal of Theoretical Biology, 338 (2013), pp. 41–58.

[101] A. Poulenard, P. Skraba, and M. Ovsjanikov, *Topological function optimization for continuous shape matching*, Computer Graphics Forum, 37 (2018), pp. 13–25.

[102] J. D. Power, A. L. Cohen, S. M. Nelson, G. S. Wig, K. A. Barnes, J. A. Church, A. C. Vogel, T. O. Laumann, F. M. Miezin, B. L. Schlaggar, and S. E. Petersen, *Functional network organization of the human brain*, Neuron, 72 (2011), pp. 665–678.

[103] A. Rathore, N. Chalapathi, S. Palande, and B. Wang, *TopoAct: Exploring the shape of activations in deep learning.* In submission, arXiv:1912.06332, 2019.

[104] M. W. Reimann, M. Nolte, M. Scolamiero, K. Turner, R. Perin, G. Chindemi, P. Dłotko, R. Levi, K. Hess, and H. Markram, *Cliques of neurons bound into cavities provide a missing link between structure and function*, Frontiers in Computational Neuroscience, 11 (2017), p. 48.

[105] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt, *A stable multi-scale kernel for topological machine learning*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015), pp. 4741–4748.

[106] R. Rosipal and L. Trejo, *Kernel partial least squares regression in reproducing kernel Hilbert space*, JMLR, 2 (2002), pp. 97–123.

[107] M. Rosvall, A. V. Esquivel, A. Lancichinetti, J. D. West, and R. Lambiotte, *Memory in network flows and its effects on spreading dynamics and community detection*, Nature Communications, 5 (2014), p. 4630. doi:10.1038/ncomms5630.

[108] M. Rubinov and O. Sporns, *Complex network measures of brain connectivity: Uses and interpretations*, NeuroImage, 52 (2010), pp. 1059–1069.

[109] M. Rudelson and R. Vershynin, *Sampling from large matrices: An approach through geometric functional analysis*, Journal of the ACM, 54 (2007), p. 21. doi:10.1145/1255443.1255449.

[110] I. Safro, P. Sanders, and C. Schulz, *Advanced coarsening schemes for graph partitioning*, Journal of Experimental Algorithmics, 19 (2015), pp. 1–24.

[111] M. Saggar, O. Sporns, J. Gonzalez-Castillo, P. A. Bandettini, G. Carlsson, G. Glover, and A. L. Reiss, *Towards a new approach to reveal dynamical organization of the brain using topological data analysis*, Nature Communications, 9 (2018), p. 1399. doi:10.1038/s41467-018-03664-4.

[112] C. M. Schumann, C. S. Bloss, C. C. Barnes, G. M. Wideman, R. A. Carper, N. Akshoomoff, K. Pierce, D. Hagler, N. Schork, C. Lord, and E. Courchesne, *Longitudinal magnetic resonance imaging study of cortical development through early childhood in autism*, Journal of Neuroscience, 30 (2010), pp. 4419–4427.

[113] W. W. Seeley, R. K. Crawford, J. Zhou, B. L. Miller, and M. D. Greicius, *Neurodegenerative diseases target large-scale human brain networks*, Neuron, 62 (2009), pp. 42–52.

[114] W. W. Seeley, V. Menon, A. F. Schatzberg, J. Keller, G. H. Glover, H. Kenna, A. L. Reiss, and M. D. Greicius, *Dissociable intrinsic connectivity networks for salience processing and executive control*, Journal of Neuroscience, 27 (2007), pp. 2349–2356.

[115] G. Singh, F. Mémoli, and G. Carlsson, *Topological methods for the analysis of high dimensional data sets and 3d object recognition*, Eurographics Symposium on Point-Based Graphics, 22 (2007), pp. 91–100.

[116] A. E. Sizemore, C. Giusti, A. Kahn, J. M. Vettel, R. F. Betzel, and D. S. Bassett, *Cliques and cavities in the human connectome*, Journal of Computational Neuroscience, 44 (2017), pp. 115–145.

[117] D. A. Spielman and N. Srivastava, *Graph sparsification by effective resistances*, SIAM Journal on Computing, 40 (2011), pp. 1913–1926.

[118] D. A. Spielman and S.-H. Teng, *Spectral sparsification of graphs*, SIAM Journal on Computing, 40 (2011), pp. 981–1025.

[119] ——, *A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning*, SIAM Journal on Computing, 42 (2013), pp. 1–26.

[120] ——, *Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems*, SIAM Journal on Matrix Analysis and Applications, 35 (2014), pp. 835–885.

[121] J. Steenbergen, C. Klivans, and S. Mukherjee, *A Cheeger-type inequality on simplicial complexes*, Advances in Applied Mathematics, 56 (2014), pp. 56–77.

[122] K. A. Stigler, B. C. McDonald, A. Anand, A. J. Saykin, and C. J. McDougle, *Structural and functional magnetic resonance imaging of autism spectrum disorders*, Brain Research, 1380 (2011), pp. 146–161.

[123] M. Szummer and T. Jaakkola, *Partially labeled classification with markov random walks*, Advances in Neural Information Processing Systems, 14 (2002), pp. 945–952.

[124] A. Tahbaz-Salehi and A. Jadbabaie, *Distributed coverage verification in sensor networks without location information*, IEEE Transactions on Automatic Control, 55 (2010), pp. 1837–1849.

[125] L. Q. Uddin, K. Supekar, C. J. Lynch, A. Khouzam, J. Phillips, C. Feinstein, S. Ryali, and V. Menon, *Salience network-based classification and prediction of symptom severity in children with autism*, JAMA Psychiatry, 70 (2013), pp. 869–879.

[126] Y. van Gennip, N. Guillen, B. Osting, and A. L. Bertozzi, *Mean curvature, threshold dynamics, and phase field theory on finite graphs*, Milan Journal of Mathematics, 82 (2014), pp. 3–65.

[127] U. von Luxburg, *A tutorial on spectral clustering*, Statistics and Computing, 17 (2007), pp. 395–416.

[128] S. Wagner and D. Wagner, *Comparing clusterings - an overview*, Tech. Rep. 4, Universität Karlsruhe, Karlsruhe, 2007.

[129] M. Wilke, S. K. Holland, M. Altaye, and C. Gaser, *Template-O-Matic: A toolbox for creating customized pediatric templates*, NeuroImage, 41 (2008), pp. 903–913.

[130] D. P. Woodruff, *Sketching as a tool for numerical linear algebra*, Foundations and Trends in Theoretical Computer Science, 10 (2014), pp. 1–157.

[131] X. Zhu and Z. Ghahramani, *Learning from labeled and unlabeled data with label propagation*, Tech. Rep. CMU-CALD-02-107, Carnegie Mellon University, 2002.

[132] X. Zhu, Z. Ghahramani, and J. Lafferty, *Semi-supervised learning using gaussian fields and harmonic functions*, in Proceedings of the Twentieth International Conference on International Conference on Machine Learning - ICML'03, AAAI Press, 2003, pp. 912–919. doi:10.5555/3041838.3041953.

[133] B. A. Zielinski, J. S. Anderson, A. L. Froehlich, M. B. D. Prigge, J. A. Nielsen, J. R. Cooperrider, A. N. Cariello, P. T. Fletcher, A. L. Alexander, N. Lange, E. D. Bigler, and J. E. Lainhart, *scMRI reveals large-scale brain network abnormalities in autism*, PLOS ONE, 7 (2012), pp. 1–14.

[134] B. A. Zielinski, E. D. Gennatas, J. Zhou, and W. W. Seeley, *Network-level structural covariance in the developing brain*, Proceedings of the National Academy of Sciences, 107 (2010), pp. 18191–18196.

[135] B. A. Zielinski, M. B. D. Prigge, J. A. Nielsen, A. L. Froehlich, T. J. Abildskov, J. S. Anderson, P. T. Fletcher, K. M. Zygmunt, B. G. Travers, N. Lange, A. L. Alexander, E. D. Bigler, and J. E. Lainhart, *Longitudinal changes in cortical thickness in autism and typical development*, Brain, 137 (2014), pp. 1799–1812.