Finding the Cores of Higher Graphs Using Geometric and Topological Means: A Survey

Inés García-Redondo[0000−0001−8340−4235], Claudia Landi[0000−0001−8725−4844], Sarah Percival[0000−0003−1024−4618], Anda Skeja[0000−0002−5707−5420], Bei Wang[0000−0002−9240−0700], and Ling Zhou[0000−0001−6655−5162]

Abstract In this survey, we explore recent literature on finding the cores of higher graphs using geometric and topological means. We study graphs, hypergraphs, and simplicial complexes, all of which are models of higher graphs. We study the notion of a core, which is a minimalist representation of a higher graph that retains its geometric or topological information. We focus on geometric and topological methods based on discrete curvatures, effective resistance, and persistent homology. We aim to connect tools from graph theory, discrete geometry, and computational topology to inspire new research on the simplification of higher graphs.

Inés García-Redondo
London School of Geometry and Number Theory, Imperial College London, UK.
e-mail: i.garcia-redondo22@imperial.ac.uk

Claudia Landi
DISMI, University of Modena and Reggio Emilia, Italy.
e-mail: clandi@unimore.it

Sarah Percival
Department of Mathematics and Statistics, University of New Mexico, USA.
e-mail: spercival@unm.edu

Anda Skeja
Department of Mathematics, Uppsala University, Sweden.
e-mail: anda.skeja@math.uu.se

Bei Wang
School of Computing, Scientific Computing and Imaging Institute, University of Utah, USA.
e-mail: beiwang@sci.utah.edu

Ling Zhou
Department of Mathematics, Duke University, USA.
e-mail: ling.zhou@duke.edu

1 Introduction

Real-world networks are often modeled as graphs, where edges capture pairwise interactions between entities represented by vertices. However, in many social and biological scenarios, pairwise interactions are mere reductions from actual multiway interactions, which can be captured more adequately by hypergraphs and simplicial complexes. In studying pathogenic viral responses, multiple genes (vertices) coordinate with each other to form biological pathways (hyperedges) [50]. While investigating consensus formation, opinions of individuals (vertices) in a society are not formed in isolation but rather in large groups (hyperedges) [58]. We study graphs, hypergraphs, and simplicial complexes, all of which are considered as forms of higher graphs[1]. On the other hand, although a simplicial complex can be viewed as a special case of a hypergraph—specifically, as a downward-closed hypergraph—these two structures possess distinct algebraic properties (e.g., homology theories) and are therefore treated separately in this context.

In this paper, we survey the recent literature on finding the *cores* of higher graphs using geometric and topological means. Informally, the core of a higher graph is a minimalist representation that retains its geometric or topological information. Here, the concept of a core is more general than the *k-core* of a graph [124], which is a maximal connected subgraph in which each vertex has a degree of at least k. The concept of a core also goes beyond the (k, q)-*core* of a hypergraph [3], which is a maximal subgraph in which each vertex has a hypergraph degree of at least k and each hyperedge contains at least q vertices [86]. The notion of a core has been used in [95] to refer to a minimalistic simplicial filtration that retains the persistent homology information. Several notions from network science are closely related to this concept. *Coarse geometry* of networks preserves geometric or topological properties while ignoring small-scale features [145]. A *network backbone* retains short-range interactions while disregarding long-range ones [15]. The study of cores is relevant to graph reduction methods, including sparsification, coarsening, and condensation, some of which focus on preserving geometric or topological information within the reduced representations; see [63, 90] for surveys.

We focus on geometric and topological methods in extracting the cores of higher graphs, based on discrete curvatures, effective resistance, and persistent homology. We do not survey these three classes of concepts in general, but rather focus on using them for core extraction. We aim to connect tools from graph theory, discrete geometry, and computational topology to inspire new research on the simplification of higher graphs.

Paper classification. The annotation of each paper is guided primarily by three types of higher graphs, namely, **graphs**, **hypergraphs**, and **simplicial complexes**. It is guided secondarily by certain geometric and topological methods used to preserve the cores of higher graphs, which, admittedly, partially reflect our own

---

[1] Spivak [132] considered these objects to belong to the categories of higher graphs and studied them in a unifying framework. We simply borrow the notion of higher graphs in this paper to encompass the above objects of interest.

biases in constructing this survey: **discrete curvatures**, which include Forman-Ricci, Ollivier-Ricci, and resistance curvatures; **effective resistance**, which arises from various notions of Laplacians; and **homology**, which includes simplicial homology and persistent homology. For hypergraphs and simplicial complexes, we also survey geometric methods from **percolation** and **spectral clustering**.

Connection to existing surveys. There is a vast literature on graph reduction, also known as graph summarization or graph simplification; see [29, 31, 63, 66, 90, 144] for surveys. *Graph sparsification* approximates a given graph by a sparse graph with a subset of vertices and/or edges [31]. *Graph coarsening* groups nodes into super-nodes and aggregates the intergroup edges into super-edges [29]. *Graph condensation* condenses a graph by synthesizing a smaller graph with comparable performance for graph neural networks (GNNs) [71]; see [56] for a survey. Different from existing surveys, we focus on higher graph sparsification and coarsening techniques based on tools in combinatorics (e.g., Laplacians, discrete curvatures) and computational topology (e.g., persistent homology).

Both graph coarsening and graph clustering involve grouping nodes, but with different goals: graph coarsening seeks to reduce the size of the graph while maintaining its essential properties, whereas graph clustering aims to identify meaningful communities or patterns within the data. The literature on spectral clustering of graphs is extensive (e.g., [126, 101, 94]), focusing on detecting communities among nodes by clustering the eigenvectors of specific Laplacians; for a comprehensive survey, see [144]. For the purpose of this survey, we exclude spectral graph clustering due to its long history and popularity; instead, we include recent advancements on the spectral clusterings of hypergraphs and simplicial complexes as they create new research opportunities.

In network science, percolation theory has received considerable attention, describing the behaviors of a network (e.g, giant clusters, cluster distribution) when nodes or edges are randomly designated either occupied or unoccupied [87]. We exclude network percolation (including an in-depth discussion on k-cores [80]) from this survey; see [117, 84, 87] for recent reviews. Instead, we pay special attention to the recent theory of percolation on hypergraphs (e.g. [17]), whereas the theory of percolation on simplicial complexes remains scarce and does not yet lead to the computation of well-defined cores (e.g. [154, 153]).

Finally, there are a number of surveys on the combinatorics and topological tools we utilize: see [96] for a classic review of graph Laplacians and [146] for a recent survey on persistence Laplacians derived from computational topology; [44, 45] for introductory texts on persistent homology.

2 Technical Background on Discrete Curvatures

To study the geometry of graphs, discrete curvatures are natural measures that quantify the local geometry around nodes and edges of a graph. In this survey, we are

interested in finding the cores of higher graphs using discrete curvatures. Therefore, in this section we review various notions of discrete curvatures for higher graphs, followed by discussions of their connections to core-findings in graphs (**Section** 3), hypergraphs (**Section** 4), and simplicial complexes (**Section** 5), respectively.

The concept of Ricci curvature, originating from differential geometry, has seen extensive research dedicated to its discretization, for application in graphs and similar structures [135, 33, 97, 91, 62, 21, 69, 5]. We begin with the discretization of Ricci curvature, namely, the Forman-Ricci curvature (**Section** 2.1), the Ollivier-Ricci curvature (**Section** 2.2), and their variants. We also discuss the more recent notion of resistance curvature (**Section** 2.3), which is related to the effective resistance and the sparsification of graphs and simplicial complexes.

2.1 Forman-Ricci Curvature

History. One of the simplest discrete curvatures to compute in graphs is the Forman-Ricci curvature, originally formulated for general CW-complexes by Forman [52] employing a reinterpretation of the Bochner-Weitzenböck formula. For an oriented, compact Riemannian manifold, the Hodge Laplacian can be expressed as a sum of the Bochner Laplacian, i.e. $\nabla^* \nabla$, where ∇ is the Levi-Civita connection of the manifold and ∇^* its adjoint, and an endomorphism of the bundle of differential forms only involving the Ricci curvature of the manifold [109, Theorem 9.4.1].

Forman adapted this approach to a combinatorial setting and demonstrated that, for a weighted CW complex, a similar decomposition could be achieved, and thus defined a Ricci curvature from the corresponding term within it [52]. Sreejith et al. [134] introduced the Forman-Ricci curvature to network analysis, particularized for undirected, possibly weighted graphs (seen as 1-dimensional simplicial complexes). Forman-Ricci curvature was later extended to possibly directed hypergraphs by Leal et al. [83]. All these notions are reviewed and unified by Eidi et al. [46].

Definitions for graphs and simplicial complexes. We begin by introducing the notion of curvature introduced by Forman [52] applied to the case of *simplicial complexes* (instead of CW complexes), as they constitute one of our primary objects of study.

Definition 1 ([52]) For a weighted simplicial complex S and any dimension p, the *Forman-Ricci curvature* is a function

$$\mathcal{F}_p : S_p \to \mathbb{R}$$

defined for each p-simplex $\alpha \in S_p$ as

$$\mathcal{F}_p(\alpha) = w_\alpha \left(\left[\sum_{\beta > \alpha} \frac{w_\alpha}{w_\beta} + \sum_{\gamma < \alpha} \frac{w_\gamma}{w_\alpha} \right] - \sum_{\tilde{\alpha} \neq \alpha} \left| \sum_{\beta > \alpha, \, \tilde{\alpha}} \frac{\sqrt{w_\alpha w_{\tilde{\alpha}}}}{w_\beta} - \sum_{\gamma < \alpha, \, \tilde{\alpha}} \frac{w_\gamma}{\sqrt{w_\alpha w_{\tilde{\alpha}}}} \right| \right),$$

where $\alpha < \beta$ denotes that α is contained in the boundary of the $(p+1)$-simplex β. In other words, α is a face of β or β is a coface of α of relative dimension 1.

This definition can be particularized for unweighted simplicial complexes as

$$\mathcal{F}_p(\alpha) = \#\{(p+1)\text{-cells } \beta > \alpha\} + \#\{(p-1)\text{-cells } \gamma < \alpha\} \\ - \#\{\text{parallel neighbours of } \alpha\}, \tag{1}$$

where a parallel neighbour of a p-simplex α is defined as another p-simplex $\tilde{\alpha}$ sharing only one face or one coface of relative dimension 1 with α.

Assuming an undirected, unweighted graph $G = (V, E)$, upon substituting $p = 1$ into **Definition** 1, one derives the following curvature expression [134, 46] for an edge $e = (u, v)$,

$$\mathcal{F}(e) = 2 - \deg(e). \tag{2}$$

Here, $\deg(e)$ denotes the number of parallel neighbours of an edge e, that is, the number of edges sharing a vertex with e. Note that one can also write $\deg(e) = \deg_u(e) + \deg_v(e) = \deg(u) + \deg(v) - 2$ where $\deg(v)$ is the degree of a vertex v and $\deg_v(e)$ is the number of edges sharing the vertex v with $e = (u, v)$. This relation on degrees gives the following equivalent expression for the Forman-Ricci curvature in terms of the degrees of the vertices: $\mathcal{F}(e) = 4 - \deg(u) - \deg(v)$. From this notion, Sreejith et al. [134] also defined a *node Forman curvature* as

$$\mathcal{F}(v) = \frac{1}{\deg(v)} \sum_{e_v} \mathcal{F}(e_v), \tag{3}$$

where e_v denotes an edge incident on the node v.

In [133, 123], the Forman-Ricci curvature for nodes and edges was extended to directed graphs. Borrowing notations from [46], let $G = (V, E)$ be an unweighted, directed graph, where we denote the directed edge going from u (the tail) to v (the head) as $e = [u, v]$. For an edge $e = [u, v]$, we call the *input* of e to the set of all edges having u as their head, and analogously, the output of e is the set of all edges having v as its tail. The degree of the input $\deg_{in}(e)$ is the number of elements in the input, and similarly we can define the degree of the output $\deg_{out}(e)$. Using these notions, we can define the Forman-Ricci curvature of the edge as $\mathcal{F}(e) = 2 - \deg_{in}(e) - \deg_{out}(e)$.

Definitions for hypergraphs. From the formulation above, the Forman-Ricci curvature can be very naturally extended to (directed) hypergraphs $H = (V, E)$ [83] where V is the set of vertices and E a set of ordered pairs of subsets of V, called hyperedges. For $e = (e_1, e_2) \in E$, let e_1 be the tail of the hyperedge, its nodes being input nodes, and e_2 the head of the hyperedge, composed by output nodes. We now define

$$\deg_{in}(e) = \#\{\text{hyperedges with an input node of } e \text{ as their head}\}$$

and

$$\deg_{out}(e) = \#\{\text{hyperedges with an output node of } e \text{ as their tail}\}.$$

Notice that another hyperedge $\tilde{e}$ from H might have several input nodes of e as their head, so that the formula above is actually counting the hyperedges with multiplicity given by this number of "shared" nodes; and similarly for the output nodes of e and the hyperedges having them as their tail.

Definition 2 ([83]) Given a directed hypergraph $H = (V, E)$ the Forman-Ricci curvature of an hyperedge $e = (e_1, e_2) \in E$ is

$$\mathcal{F}(e) = |e_1| + |e_2| - \deg_{in}(e) - \deg_{out}(e).$$

Note that [83] also generalized the definition of Forman–Ricci curvature to the setting of weighted undirected hypergraphs. Their formulation is motivated by the following expression for the curvature of an edge $e = (e_1, e_2)$ in a weighted graph:

$$\mathcal{F}(e) = w_e \left(\left(\frac{w_{e_1}}{w_e} - \sum_{e' \ni e_1, e' \neq e} \frac{w_{e_1}}{\sqrt{w_e w_{e'}}} \right) + \left(\frac{w_{e_2}}{w_e} - \sum_{e' \ni e_2, e' \neq e} \frac{w_{e_2}}{\sqrt{w_e w_{e'}}} \right) \right).$$

Definition 3 ([83]) Given a weighted undirected hypergraph $H = (V, E)$, the Forman–Ricci curvature of a hyperedge $e \in E$ is defined as

$$\mathcal{F}(e) = w_e \sum_{v \in e} \left(\frac{w_v}{w_e} - \sum_{e' \ni v, e' \neq e} \frac{w_v}{\sqrt{w_e w_{e'}}} \right).$$

Properties. From its definition, one of the key advantages of the Forman-Ricci curvature becomes apparent: it has a direct, inexpensive computation for networks, in contrast to other notions of curvature that will be introduced subsequently. It is also clear from **Equation** (2) that the Forman-Ricci curvature of an edge should be very negative (have high absolute value) when the vertices defining the edge are well-connected within the network. To investigate in more detail which features of a graph this function captures, Sreejith et al. [134] examined the Forman-Ricci curvature for nodes and edges in both theoretical and real-world networks, including transportation and protein interaction networks. They studied their distributions and investigated correlations between Forman-Ricci curvature and centrality/connectivity measures, such as degree, clustering coefficient, and betweenness centrality. As expected, strong positive correlations with *degree* are found in random and small-world networks, whereas weaker negative correlations exist in scale-free networks. Their study also suggested an observable correlation between Forman-Ricci curvature and the degree of nodes, as a function of degree assortativity. Negative correlations were discovered between Forman-Ricci curvature and various centrality measures in different network types, such as *betweenness centrality*, which measures the number of shortest paths passing through an edge and is expensive to compute. Interestingly, the *clustering coefficient*—a quantity typically used as a notion of curvature in network—showed no correlation with Forman-Ricci curvature. Finally, the paper examined the impact of removing nodes based on their Forman-Ricci curvature on network connectivity, finding that such removal accelerates network disintegration compared to random or

clustering coefficient-based node removal. In a follow-up paper, Sreejith et al. [133] studied similar notions for directed graphs, arriving at the same conclusions.

2.2 Ollivier-Ricci Curvature

History. Recall that in Riemannian geometry, the Ricci curvature is positive when "small spheres are closer (in transportation distance) than their centers are" [111]. Inspired by this, Ollivier used a probability measure m_x that depends on x serving as an analog for the sphere S_x centered at x. Rather than corresponding points between two close spheres S_x and S_y, the transportation distance between measures is utilized. Ollivier introduced a notion of *coarse Ricci curvature* [103, 104], now commonly referred to as the *Ollivier-Ricci curvature*, of Markov chains valid on arbitrary metric spaces, such as graphs.

With a modification of Ollivier's definition, Lin et al. [89] introduced another notion of discrete curvature on graphs. By incorporating gradients of the graph Laplacian, Münch and Wojciechowski [98, Theorems 2.1 and 2.6] obtained a limit-free formula of the definition of Lin et al. In these definitions, the Ollivier-Ricci curvature of an edge is a local quantity that depends on the degrees of its endpoints, which may result in a lack of robustness of the Ollivier-Ricci curvature for sparse networks. For this reason, instead of structural neighbourhoods, Gosztolai and Arnaudon [60] considered distributions generated by diffusion processes across scales.

In a further extension, Eidi and Jost [47] generalized the Ollivier-Ricci curvature to (possibly directed and/or weighted) hypergraphs. More recently, Coupette et al. [34] provided a unified framework for several generalizations of Ollivier-Ricci curvatures [6, 47, 82, 8] on hypergraphs and their computations. Recently, Yamada [149] extended the Ollivier-Ricci curvature to simplicial complexes.

Definitions for graphs. Let (X, d) be a metric space and ν_1, ν_2 probability measures on X. An L_1 transportation distance between ν_1 and ν_2 is

$$W_1(\nu_1, \nu_2) := \inf_{\xi \in \Pi(\nu_1, \nu_2)} \int_{(x,y) \in X \times X} d(x,y) \, \mathrm{d}\xi(x,y),$$

where $\Pi(\nu_1, \nu_2)$ is the set of measures on $X \times X$ projecting onto ν_1 and ν_2, respectively.

A *random* walk m on X is defined as a family of probability measures $m_x(\cdot)$ on X for each $x \in X$, satisfying the following conditions:

1. The measure m_x depends measurably on the point $x \in X$;
2. Each measure m_x has a finite first moment, i.e., for some (hence any) $o \in X$ and for any $x \in X$, $\int d(o, y) dm_x(y)$ is finite.

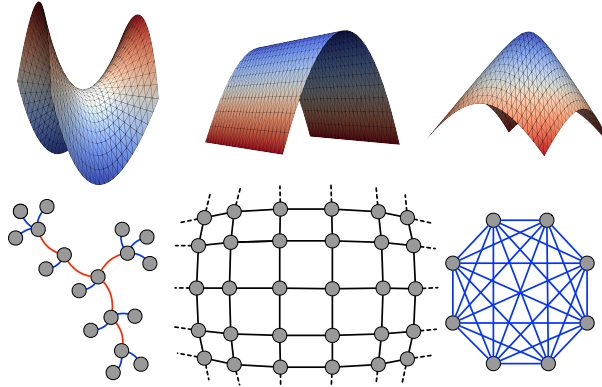The data $(m_x)_{x \in X}$ allow us to define a notion of curvature as follows:

Fig. 1: An illustration of Ricci curvatures for surfaces (top) and Ollivier-Ricci curvatures (**Equation** (7)) for graphs (bottom). Top: from left to right, surfaces of negative, zero, and positive Ricci curvature, respectively. Bottom: red for negative Ollivier-Ricci curvature, black for zero Ollivier-Ricci curvature, and blue for positive Ollivier-Ricci curvature. Bottom left: a tree graph with negative Ollivier-Ricci curvature everywhere except the edges connecting the leaves. Bottom middle: an infinitely sized grid graph with all edges of zero Ollivier-Ricci curvature. Bottom right: a complete graph with all edges of positive Ollivier-Ricci curvature. Notice the difference with respect to the Forman-Ricci curvature (**Equation** (2)) which would be negative in all the edges of the graphs on the bottom. Image reproduced from [102, Figure 4] with modifications, licensed under CC BY 4.0. https://creativecommons.org/licenses/by/4.0/.

Definition 4 ([103, Definition 3]) Let (X, d_X) be a metric space with a random walk m. Let $x, y \in X$ be two distinct points. The *Ollivier-Ricci curvature* (briefly, *OR curvature*, originally called the coarse Ricci curvature) of (X, d_X, m) at (x, y) is

$$\kappa(x, y) := 1 - \frac{W_1(m_x, m_y)}{d_X(x, y)}. \tag{4}$$

Similar to the classical Ricci case, this curvature will be positive or negative depending on whether the measures m_x and m_y are closer or further apart than the points x and y themselves. See **Figure** 1 for a discussion on spaces with positive, zero or negative Ricci curvature; and their analogue graphs where curvature is measured using Ollivier-Ricci curvature.

Consider a metric measure space (X, d_X, μ) where balls in X have finite measure, and μ is fully supported on X. Choose some $\varepsilon > 0$ and let $B(x, \varepsilon)$ denote the open ε-ball of $x \in X$. The *ε-step random walk* on X, starting from a point x, consists in randomly jumping in the ball of radius ε around x, with probability proportional to μ, i.e., it is defined to be $m_x = \frac{\mu|_{B(x,\varepsilon)}}{\mu(B(x,\varepsilon))}$. In the case of unweighted graphs, $\varepsilon = 1$ is a natural choice.

Explicitly, for an unweighted graph G, the neighbourhoods of vertices x, y adjacent to an edge $e = (x, y)$ are endowed with a uniform probability measure, i.e., for z adjacent to x (denoted $z \sim x$),

$$\mu_x(z) := \frac{1}{\deg_G(x)}.$$

In [89], still focusing on the case of graphs, the authors proposed not to use a uniform measure but rather the following one: for any $t \in [0, 1]$ and $x \in V$:

$$m_x^t(z) := \begin{cases} t, & \text{if } z = x; \\ \frac{1-t}{\deg_G(z)}, & \text{if } z \sim x; \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Here, the parameter t controls whether the random walk is likely to revisit a node. The original definition by Ollivier can be retrieved with $t = 0$.

The parametric definition for the measure can be further generalized as follows for a weighted graph (cf. [7]): for any $t \in [0, 1]$ and $x \in V$,

$$m_x^t(z) := \begin{cases} t, & \text{if } z = x; \\ \frac{(1-t)w_{xz}}{\sum_{z' \sim x} w_{xz'}}, & \text{if } z \sim x; \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Here, w_{xz} denotes the weight of the edge between vertices x and z.

Another way of defining the measures for weighted graphs is by Ni et al. [102]: for any $t \in [0, 1]$, $p \geq 0$ and $x \in V$, define the probability measure,

$$m_x^{t,p}(z) := \begin{cases} t, & \text{if } z = x; \\ \frac{1-t}{C_x} e^{-d_G(x,z)^p}, & \text{if } z \sim x; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Here, $C_x = \sum_{z \sim x} e^{-d_G(x,z)^p}$ and d_G is the graph distance. For any $x \sim z$, $d_G(x, z)$ is the weight between x and z, with a default value of 1 when the graph is unweighted. Note that when $t = 0$ and $p = 0$, or if the graph is unweighted, this measure reduces to the uniform measure. In this context, t determines the probability of staying at x, and p controls the extent to which the neighbour z of x is discounted based on the weight $d_G(x, z)$. The idea of the Ollivier-Ricci curvature definition compared to the Forman-Ricci curvature is illustrated in [141, Figure 5]. In the particular case when we use the probability measure following **Equation** (5), which depends on a parameter $t \in [0, 1]$, we can consider the asymptotic curvature for t tending to 1.

Definition 5 ([89, Section 2]) The t-Ollivier-Ricci curvature is

$$\kappa_t(x, y) := 1 - \frac{W_1(m_x^t, m_y^t)}{d_G(x, y)}. \tag{8}$$

The *LinLuYau-Ricci curvature*[2] at (x, y) is

$$\kappa^{LLY}(x, y) := \lim_{t \to 1} \frac{\kappa_t(x, y)}{1 - t}. \tag{9}$$

Following **Equation** (9), we may consider a filtration of a graph using t as a filtration parameter, giving rise to multiscale cores of a graph. This consideration is left for future discussions.

Remark 1 We review examples of LinLuYau-Ricci curvatures and Forman-Ricci curvatures for various simple graphs. For the computation of LinLuYau-Ricci curvatures, see [89, page 610] for details. For Forman-Ricci curvatures, the computation can be done using the formula $F(e) = 4 - \deg(u) - \deg(v)$, where e is an edge connecting vertices u and v.

- The complete graph K_n has a constant LinLuYau-Ricci curvature on each edge equal to $n/(n-1)$. This is the only graph with a constant Ricci curvature greater than 1. In K_n, the Forman-Ricci curvature is also a constant on each edge with a value of $6 - 2n$.
- The cycle C_n for $n \geq 6$ has a constant LinLuYau-Ricci curvature on each edge equal to 0. For small cycles C_3, C_4, and C_5, we have constant LinLuYau-Ricci curvature on each edge equal to $3/2$, 1, and $1/2$, respectively. In C_n for any $n \geq 3$, the Forman-Ricci curvature is a constant on each edge with a value of 0.
- The hypercube Q^n has a constant LinLuYau-Ricci curvature $2/n$ and a constant Forman-Ricci curvature $4 - 2n$.

To deal with the lack of a resolution parameter of Ollivier-Ricci curvature and to reveal multiscale structures in real-world networks, Gosztolai and Arnaudon [60] further modified the definition of the probability measure. They introduced the idea of initiating a diffusion process at each node i to generate a set of measures $p_i(\tau)$, where τ represents the time parameter of the diffusion process. The *dynamic Ollivier-Ricci curvature* of an edge is then defined as the distance between the pair of measures started at its endpoints, normalized by the weight of the edge:

$$\kappa_\tau(u, v) := 1 - \frac{W_1(p_u(\tau), p_v(\tau))}{w_{uv}}, \tag{10}$$

whenever $e = (u, v)$ is an edge, and 0 otherwise.

Following [141], for a graph $G = (V, E)$, we can further define the Ollivier-Ricci curvature for vertices (rather than edges) with respect to the curvature of its adjacent edges. Formally, if for a vertex $x \in V$ the set $E_x := \{e \in E : x \in e\}$ contains the edges adjacent to x, then the *Ollivier-Ricci curvature at a vertex x* is given by $\kappa(x) = \sum_{e \in E_x} \kappa(e)$.

Definitions for hypergraphs. To generalize the Ollivier-Ricci curvature to hypergraphs, Eidi et al. [47] observed that **Definition** 4 can also be expressed by the following formula for an edge $e = (v, w)$ of a graph:

---

[2] We use this notion to differentiate it from the classical Ricci curvature.

$$\kappa(e) = \mu_0 - \mu_2 - 2\mu_3, \tag{11}$$

where μ_0 represents the amount of mass that remains unmoved in an optimal transport plan, corresponding to the stable mass in directed 3-cycles $(u \to v \to w \to u)$ for any vertex u adjacent to v and w. The terms μ_2 and μ_3 are the amount of mass that should be moved with distance 2 and 3, respectively.

The above formula **Equation** (11) is obtained as follows: consider the edge $e = (v, w)$ and let $e_v = (v, v_1)$ and $e_w = (w, w_1)$ be edges emanating from v and w, respectively. Then define their distance w.r.t. e as $d_e(e_v, e_w) := d(v_1, w_1)$ where $d(v_1, w_1)$ denotes the distance between v_1 and w_1 in the graph, specifically the minimum number of edges that must be traversed to travel from v_1 to w_1 (by construction, not greater than 3: distance 0, when v_1, w_1 participates in a triangle, distance 1 for a square, and so on up to 3). Let E_v be the set of edges that have v as a vertex, and let $|E_v|$ be its cardinality. Define a probability measure m_v on the set of all edges E of the graph by assigning each edge $e_v \in E_v$ a weight of $1/|E_v|$, and assigning a weight of 0 to all edges not in E_v. For an edge $e = (v, w)$, from $d(v, w) = 1$ it follows that the Ollivier–Ricci curvature of **Definition** 4 is equal to

$$\kappa(e) = 1 - W_1(m_v, m_w), \tag{12}$$

where W_1 is the 1-Wasserstein distance between m_v and m_w:

$$W_1(m_v, m_w) = \inf_{p \in \Pi(m_v, m_w)} \sum_{(e_1, e_2) \in E \times E} d_e(e_1, e_2) p(e_1, e_2), \tag{13}$$

where $\Pi(m_v, m_w)$ is the set of measures on $E \times E$ that project to m_v and m_w, respectively. The key idea is to optimally arrange the two collections E_v, E_w of edges sharing one of their endpoints with e, so that the average distances between the paired edges are minimized. Here, the sets E_v and E_w both include the edge $e = (v, w)$ that we are evaluating. Let μ_i $(0 \le i \le 3)$ be the fraction of edges in E_v that are moved at distance i in some optimal transport plan. According to **Equation** (13), $W_1(m_u, m_v) = \mu_1 + 2\mu_2 + 3\mu_3$. Observing that $\sum_{i=0}^{3} \mu_i = 1$, **Equation** (12) then yields **Equation** (11).

The above formula **Equation** (11) is taken as the definition for the curvature of a hyperedge in a directed hypergraph in [47]. Precisely, if we denote $u \to e_i$ when there exists a hyperedge $e = (e_k, e_i)$ such that $u \in e_k$, and similarly, $e_j \to v$ when there is a hyperedge $e = (e_j, e_k)$ such that $v \in e_k$, then for a given hyperedge $e = (e_i, e_j)$, we can define two sets: $\mathcal{M} = \{u \colon u \to e_i\}$ referred to as the *masses* and $\mathcal{H} = \{v \colon e_j \to v\}$, referred to as the *holes*. A probability measure is assigned to each set, denoted by $\mu_{\mathcal{M}}$ and $\mu_{\mathcal{H}}$, respectively. For $u \in \mathcal{M}$ and $v \in \mathcal{H}$, the distance $d(u, v)$ between each mass $\mu_{\mathcal{M}}(u)$ and each hole $\mu_{\mathcal{H}}(v)$ of a hyperedge e is defined as the minimum number of directed hyperedges connecting them. This distance is at most 3: it is exactly 3 if the shortest path from u to v requires passing through e; it is 0 when $u = v$, meaning that u is both a mass and a hole of e.

A *transport plan* is a matrix $\mathcal{E}$ whose entries represent the amount of mass, out of $\mu_{\mathcal{M}}(u)$, to be moved from vertex u to vertex v, denoted by $\mathcal{E}(u, v)$. Assume

we are given an optimal transport plan $\mathcal{E}$, i.e., one that minimizes the quantity $\sum_{u \to e_i} \sum_{e_j \to v} d(u, v) \mathcal{E}(u, v)$. Let μ_i denote the amount of mass that is moved at distance i, with $i \in \{0, 1, 2, 3\}$. The Ollivier-Ricci curvature κ of e for a hypergraph is then defined as $\kappa(e) = \mu_0 - \mu_2 - 2\mu_3$. It is bounded above by $\kappa = 1$ (achieved when $\mu_0 = 1$, meaning each mass coincides with a hole of its same size) and below by $\kappa = -2$ (achieved when $\mu_3 = 1$, meaning each mass must be moved a distance of 3).

Coupette et al. [34] presented a flexible and unifying framework, ORCHID, which generalizes the Ollivier–Ricci curvature to hypergraphs by extending both the measure μ and the distance metric in several ways. The distance metric is replaced by a function aggregating measures (AGG):

$$\kappa^{AGG}(u, v) := 1 - \frac{\mathsf{AGG}(\mu_u, \mu_v)}{d(u, v)}. \tag{14}$$

and they identified three ways to generalize the probability measure induced by the lazy random walk on graphs: the *equal-nodes random walk* on the unweighted clique expansion of H that results from picking a neighbour v of u uniformly at random; the *equal-edges random walk* that follows a two-step random walk on the unweighted star expansion of H; and the *weighted-edges random walk* that can be thought as a random walk on a weighted clique expansion of H.

Treating a hyperedge as a set of nodes, the AGG function can be taken to be:

$$\mathsf{AGG}_{\mathrm{A}}(e) := \frac{2}{|e|(|e| - 1)} \sum_{\{u,v\} \subseteq e} W_1(\mu_u, \mu_v),$$

where $W_1(\cdot, \cdot)$ corresponds to the Wasserstein distance; $\mathsf{AGG}_{\mathrm{A}}$ measures the average amount of effort to transport the probability mass from one node to another in the edge e. In general, the AGG function should reduce to the Ollivier–Ricci curvature on graphs, while remaining computationally tractable and invariant under permutations of node indices.

The curvature at node u, independent of the choice for AGG, can be obtained as the mean of all curvatures of edges containing u:

$$\kappa^E(u) := \frac{1}{\deg(u)} \sum_{e \ni u} \kappa^{AGG}(e).$$

Since the hypergraph H is connected, Coupette et al. [34] defined the curvature of an arbitrary subset of nodes $s \subseteq V$ to be:

$$\kappa^{AGG}(s) := 1 - \frac{\mathsf{AGG}(s)}{d(s)},$$

where $d(s) := \max\{d(u, v) | \{u, v\} \subseteq s\}$. Coupette et al. [34] observed that the above definition aligns with their notions of hypergraph curvature for $s \in E, d(s) = 1$.

Definitions for simplicial complexes. In [149], Yamada generalized the definition of Ollivier-Ricci curvature to weighted simplicial complexes. Let S be a weighted simplicial complex with weight function w. Define the degree of a p-simplex α to be

$$\deg(\alpha) := \sum_{\alpha < \gamma \in S_{p+1}} w_\gamma.$$

Two distinct p-simplices α, β in S are *connected* if they share a $(p + 1)$-coface γ, denoted as $\alpha \overset{\gamma}{\sim} \beta$. If γ exists, it is unique. A path from a p-simplex α to another p-simplex β is a finite sequence of connected p-simplices $\alpha = \alpha_0, \ldots, \alpha_m = \beta$. We call m the length of the path. We define the distance between two p-simplices α and β to be the length of the shortest path from α to β, and denote it by $d_S(\alpha, \beta)$.

Definition 6 ([149, Definitions 2.8 and 2.9]) Let S be a weighted simplicial complex with weight function w. For a p-simplex α in S, define the probability measure

$$m_\alpha(\alpha') := \begin{cases} \frac{w_\gamma}{(p+1)\deg(\alpha)}, & \text{if } \alpha \overset{\gamma}{\sim} \alpha' \text{ for some } \gamma \in S_{p+1}; \\ 0, & \text{otherwise.} \end{cases}$$

For any two distinct p-simplex α and β in S, the Yamada-Ricci curvature at (α, β) is defined as

$$\kappa(\alpha, \beta) := 1 - \frac{W_1(m_\alpha, m_\beta)}{d_S(\alpha, \beta)}.$$

Properties. When X is a Riemannian manifold and ε is small, the Ollivier-Ricci curvature derived from the ε-step random walks captures the Ricci curvature in the following manner. Consider X a smooth, complete, N-dimensional Riemannian manifold. Let $x \in X$ and v be a unit tangent vector at x. Let y be a point on the geodesic that starts from v. As stated in [104, Example 7],

$$\kappa(x, y) = \frac{\varepsilon^2 \operatorname{Ric}(v, v)}{2(N + 2)} + O(\varepsilon^3 + \varepsilon^2 d(x, y)),$$

where Ric denotes the Ricci curvature and $O(\cdot)$ represents the big O notation.

The Ollivier-Ricci curvature bounds eigenvalues of the Laplacian of the graph (cf. [104], see also [12]), in the sense that if k is a lower bound for the Ollivier-Ricci curvature of edges (i.e., $\kappa(e) \geq k$, for all edges e), then the eigenvalues λ_i satisfy

$$k \leq \lambda_1 \leq \cdots \leq \lambda_{N-1} \leq 2 - k, \tag{15}$$

where λ_1 is the first non-zero eigenvalue. It follows that the Ollivier-Ricci curvature provides formal bounds on the local clustering coefficient, which measures the connectivity of the network around a specific node. Thus, in the case of graphs, the Ollivier-Ricci curvature controls the amount of overlap between neighbourhoods of adjacent vertices.

In other words, the Ollivier-Ricci curvature gives us information about how close we are to increasing the number of connected components in a graph: the smaller

the eigenvalue gap, the "thinner" the bottleneck among two large components. This observation motivates the application of the Ollivier-Ricci curvature to community detection, finding bottlenecks, and clustering (cf. **Section** 3.2). In [48], Eidi and Mukherjee focused on this bridge between topology and geometry via random walks and the Laplacian. In particular, they considered random walks on p-simplices where $p \geq 1$ with the goal of connecting the spectral gap of the Laplacian in degree p to the p-dimensional homology. The authors then speculated about a connection with the Ollivier-Ricci curvature of simplicial complexes, which could potentially pave the way for using this curvature to detect how close a simplicial complex is to exhibiting higher-dimensional holes.

According to Gosztolai and Arnaudon [60], Ollivier-Ricci curvature measures how much a graph differs from being locally grid-like, analogous to the concept of being *flat* in continuous spaces. The flatness of a network can be understood in terms of its local connectivity: the distance of a pair of nodes is the same as the average distance of their neighbourhoods. Thus, an edge with positive (or negative) Ollivier-Ricci curvature suggests that it is located in a part of the graph that is more (or less) connected than a grid.

Regarding the dynamic Ollivier-Ricci curvature in **Equation** (10), initially, as τ approaches zero—when all nodes support disjoint point masses and the diffusion has not yet mixed—the dynamic curvature tends toward 0. At the other extreme, as τ approaches infinity, i.e., as the diffusion reaches a stationary state, the dynamic curvature tends to 1. At intermediate scales, the curvature can vary between 1 and some finite negative number, depending on the structure of the graph. According to [60], as the curvature of an edge evolves, the scale at which it approaches unity reflects how easily information can be propagated between clusters.

Interestingly, for applications of the Ollivier-Ricci curvature of hypergraphs to sparsification, removing vertices from a hyperedge has the following effect on the curvature:

Proposition 1 ([47, Proposition 2.4]) *Given a hyperedge $e\colon A = \{x_1, \ldots, x_n\} \to B = \{y_1, \ldots, y_m\}$, by removing $l \leq n$ vertices from the set A and $l' \leq m$ vertices from B, the following relation holds between the curvature of the resulting hyperedge e' and e:*

$$|\kappa(e) - \kappa(e')| \leq \min\{3, 3\,(l/n, l'/m)\}.$$

However, in the graph setting, there are lower and upper bounds for the Ollivier-Ricci curvature of an edge that enjoy a combinatorial formula [139]: a lower bound for the Ollivier-Ricci curvature $\kappa(e)$ of the edge e with vertices u and v is given by

$$\kappa^{low}(e) := -2 \left(1 - \frac{1}{\deg_G(u)} - \frac{1}{\deg_G(v)} \right),$$

whereas an upper bound is given by

$$\kappa^{up}(e) := \frac{\Delta(e)}{\max\{\deg_G(u), \deg_G(v)\}}.$$

where $\Delta(e)$ denotes the number of triangles in G that include the edge e.

Given these bounds, the arithmetic mean of the lower and upper bounds yields an approximation of the Ollivier-Ricci curvature computable with a combinatorial formula:

$$\hat{\kappa}(e) = \frac{1}{2}\left(\kappa^{low}(e) + \kappa^{up}(e)\right). \tag{16}$$

2.3 Resistance Curvature

We now discuss a definition of curvature that is derived from the physical property of electric circuits called *effective resistance*. The effective resistance between two vertices in a weighted graph $G = (V, E, w)$ can be computed via the graph Laplacian [77], which we review here. Let $n = |V|$ and $m = |E|$. Using the notation of Spielman and Srivastava [130], we first define W to be an $m \times m$ diagonal matrix containing the weights w of G. We then define the *signed edge-vertex incidence matrix* $B \in \mathbb{R}^{m \times n}$ as

$$B(v, e) = \begin{cases} 0 & \text{if vertex } v \text{ is not on the boundary of edge } e, \\ 1 & \text{if vertex } v \text{ is edge } e\text{'s head,} \\ -1 & \text{if vertex } v \text{ is edge } e\text{'s tail.} \end{cases} \tag{17}$$

From here, we can define the *graph Laplacian* $L \in \mathbb{R}^{n \times n}$ as

$$L = B^\top W B, \tag{18}$$

noting that L is symmetric.

Letting L^+ denote the Moore-Penrose pseudoinverse of L, i.e., the unique matrix L^+ such that $LL^+ = L^+L = \text{proj}(\ker(L)^\perp)$, Spielman and Srivastava [130] defined the *effective resistance* of an edge e to be the entry on the diagonal corresponding to e of the matrix $R \in \mathbb{R}^{m \times m}$ defined by

$$R := BL^+B^\top = B(B^\top W B)^+ B^\top. \tag{19}$$

That is, the *effective resistance* of and edge e is given by:

$$r_e := R(e, e).$$

The pseudoinverse L^+ can also be used to define a notion of effective resistance between pairs of vertices [77]:

$$r_{v_i v_j} = (\mathbf{e}_i - \mathbf{e}_j)^\top L^+ (\mathbf{e}_i - \mathbf{e}_j)$$

where $\mathbf{e}_i \in \mathbb{R}^n$ is a column vector of length $n = |V|$ with a one in the ith spot and zeros everywhere else. Alternatively, if we let $\Gamma = L + \frac{J}{n}$, where J is the $n \times n$ matrix of all ones, we have that the effective resistance between two vertices v_i and

v_j is given by

$$r_{v_i v_j} = (\mathbf{e}_i - \mathbf{e}_j)^\top \Gamma^{-1} (\mathbf{e}_i - \mathbf{e}_j),$$

since Γ in this case is invertible. We denote by Ω the matrix whose (i, j)-th entry is $r_{v_i v_j}$; this matrix is known as the *effective resistance matrix*.

Devriendt and Lambiotte [37] introduced discrete curvatures on both vertices and edges of weighted graphs based on effective resistance. They called them *nodal* and *link resistance curvature*, respectively. Given a weighted graph $G = (V, E, w)$, we denote these curvatures as $k^{up} : V \to \mathbb{R}$ and $k^{down} : E \to \mathbb{R}$, respectively.

Definition 7 (Nodal Resistance Curvature [37]) The *nodal resistance curvature* of a vertex $v_i \in V$ in a graph G is defined as

$$k^{up}(v_i) := 1 - \frac{1}{2} \sum_{j \sim i} w_{v_i v_j} r_{v_i v_j}. \tag{20}$$

Here, $r_{v_i v_j}$ denotes the effective resistance between vertices v_i and v_j, and $w_{v_i v_j}$ the weight of the edge (v_i, v_j), where $j \sim i$ is the set such that there is an edge $e \in E$ connecting v_i and v_j.

Devriendt and Lambiotte [37] called the quantity $w_{v_i v_j} r_{v_i v_j}$ the *relative resistance* of the edge $e = (v_i, v_j)$ as it quantifies how important the edge e is for the connectivity of G.

We note that Devriendt et al. [38] introduce a slightly different definition of resistance curvature, namely

$$k^{up} = \Omega^{-1} \mathbf{1}. \tag{21}$$

Observe that the expression in **Equation** (20) can also be written as

$$k^{up} = \frac{\Omega^{-1} \mathbf{1}}{\langle \mathbf{1}, \Omega^{-1} \mathbf{1} \rangle}. \tag{22}$$

It is evident the two definitions are equal up to a factor of $\langle \mathbf{1}, \Omega^{-1} \mathbf{1} \rangle$.

As with the curvatures mentioned in previous sections, there is also a definition of resistance curvature on edges:

Definition 8 (Link Resistance Curvature [37]) The link resistance curvature of an edge $e = (v_1, v_2) \in E$ in a graph G is defined as

$$k^{down}(e) = \frac{2(k^{up}(v_1) + k^{up}(v_2))}{r_{v_1 v_2}}. \tag{23}$$

Intuitively speaking, the resistance curvature of a vertex is derived from the effective resistance of its adjacent edges, whereas the resistance curvature of an edge comes from the resistance curvature of its vertices. Let κ denote the Ollivier-Ricci curvature and $\mathcal{F}$ denote the Forman-Ricci curvature, it was shown in [37] that for an edge $e = (v_i, v_j)$,

Table 1: A summary of recent works in finding cores of graphs, as surveyed in **Section** 3. Ref.: references. V: vertices (nodes). E: edges. FRC: Forman–Ricci curvature. A-FRC: augmented FRC. ORC: Ollivier–Ricci curvature. ER: effective resistance. NMI: normalized mutual information. ARI: adjusted Rand index. MAE: mean absolute error. "Threshold": a subsampling method that removes vertices or edges above/below a chosen value. "Probability": a subsampling method that removes vertices or edges based on a probability distribution. "Spectral": properties related to the Laplacian matrix of a graph.

Ref.	Graph	Method	Criteria	Property Preserved
[145]	weighted	threshold, E	FRC	backbone
[10]	weighted	threshold, V or E	FRC	backbone
[121]	weighted	threshold, E	FRC	backbone
[152]	unweighted	threshold, E	Balanced FRC	clusters
[76]	unweighted	threshold, E	FRC	clusters (modularity, NMI)
[51]	unweighted	threshold, E	A-FRC	clusters (accuracy)
[141]	weighted	threshold, E	FRC, ORC, A-FRC	single- & mixed-membership clusters (NMI)
[107]	unweighted	threshold, E	lower FRC	clusters
[127]	weighted	threshold, E	ORC	clusters (accuracy)
[102]	weighted	threshold, E	ORC	clusters (modularity, ARI)
[147]	weighted	threshold, V and E	ORC	clusters (MAE)
[60]	weighted	N/A	dynamical ORC	multiscale clusters (geometric modularity)
[140]	weighted	threshold, E	ORC	clusters (NMI)
[130]	weighted	probability, E	ER	spectral

$$\kappa(e) \geq k^{down}(e) \geq \frac{\mathcal{F}(e)}{\omega_e}.$$

We believe it is easy to generalize resistance curvature to hypergraphs and simplicial complexes, which is left for future work.

3 Finding Cores of Graphs

In this section, we survey recent approaches in finding cores of graphs using geometric or topological means. We begin with the applications of discrete curvatures, in particular, Forman-Ricci Curvature (**Section** 3.1) and Ollivier-Ricci Curvature (**Section** 3.2), followed by a discussion of graph sparsification based on effective resistance (**Section** 3.3). References surveyed in this paper are classified in **Table** 1.

3.1 Forman-Ricci Curvature

A number of Forman-Ricci curvature-based graph sparsification methods have been proposed to preserve the *backbone* (e.g., [145, 10, 121, 152]). Weber et al. [145] defined the backbone of a network (graph) as a subnetwork (subgraph) that captures important nodes (called *hubs*) and edges (called *bridges*). Hubs are nodes with high degree and high betweenness centrality, whereas bridges are edges governing the mesoscale structure of the graph (such as those forming long-range connections) [10]. According to Barkanass et al. [10], a backbone is considered to be *structure-preserving* if it preserves the structural features such as the node degree distribution or community structure of the original graph. The notions of a *network backbone* and the *core of a network* are sometimes used interchangeably [100]. In this survey, however, we define the *core of a higher graph* more generally as a minimalist representation that retains its geometric or topological information.

On the other hand, graph clustering or community detection can also be regarded as core finding by grouping (thus preserving) the highly interconnected regions of a graph, while discarding its small-scale information. To that end, the Forman-Ricci curvature has also been used as a criterion to guide graph clustering (e.g., [51, 141, 107]).

3.1.1 Curvature-Based Sampling for Preserving Backbones

Using the Forman-Ricci curvature, Weber et al. [145] proposed a sampling method to reduce the size of a given graph while preserving its backbone. The key idea is that "high absolute curvature is strongly related to the structural importance of an edge." [145]. Their sampling procedure applies thresholding with respect to the Forman-Ricci curvature, where edges with high absolute curvature are selected and edges with low absolute curvature are eliminated. They experimentally showcased that some desirable structure of the graph (such as clusters or communities) are preserved after the sampling. As an example, they showed that in the weighted graph of co-occurrences in Les Misérables, the backbone graph maintains relationships between Valjean and other characters central to the storyline, and preserves a cluster of revolutionaries around Gavroche.

Barkanass et al. [10] further extended the above sampling procedure by considering Forman-Ricci curvatures for graphs and 2-dimensional simplicial complexes, as well as the Haantjes-Ricci curvature, i.e., a form of curvature in general metric spaces. They showed that sampling by discrete versions of Ricci curvature is mathematically justified by the relation between the Ricci curvature of a manifold and the Forman curvatures of its discretization. They further studied coarse embeddings of a network into a metric space using kernels based on Forman-Ricci curvature.

For real-world applications, Salamatian et al. [121] discussed challenges in accurately assessing network connectivity within large cloud provider infrastructures, due to potential manipulation of probe packets. They demonstrated the success of the above sampling method based on Forman-Ricci curvature, using latency

measurements from RIPE Atlas anchors and virtual machines in data centers of three major cloud providers.

3.1.2 Graph Sparsification for Graph Learning

By employing Ricci curvature-based sampling, Zhang et al. [152] proposed a graph sparsification method based on the *balanced Forman-Ricci curvature* [142]— a modification of the Forman-Ricci curvature—to study over-squashing in deep networks. The balanced Forman-Ricci curvature takes into account the presence of triangles and cycles of length four. To perform graph representation learning, Zhang et al. stored the topological information in the form of a sparsified computation subgraph around each node, where such a sparsified subgraph is obtained by iteratively sampling the edges with the highest balanced Forman-Ricci curvature in neighbourhoods (of increasing radius) around the node.

For theoretical foundations of graph sparsification, Spielman and Teng [131] introduced nearly-linear time algorithms for spectral sparsification, leveraging reduced edge sets to preserve graph Laplacians. Benczúr and Karger [14] introduced randomized edge sampling to approximate minimum s-t cuts, establishing a practical sparsification technique. Later, Batson et al. [11] proposed deterministic spectral sparsifiers and proved that every graph has a spectral sparsifier with a number of edges linear in its number of vertices.

Graph sparsification also appears frequently in graph neural networks (GNNs), where a number of approaches have been developed to leverage or learn sparse graph structures. Peng et al. [108] discussed two primary approaches to reducing the training and inference complexity of GNNs through sparsification: sparsifying either the input graph or the model itself. Specifically, they formulated model sparsification for GNNs using both the train-and-prune paradigm and sparse training. Chen et al. introduced FastGCN [28] that addresses scalability by introducing an importance sampling technique that sparsifies the neighbourhood aggregation process, leading to efficient graph convolutional network (GCN) training. Jin et al. [70] proposed a method to jointly learn sparse graph structures and node features for robust GNN performance under noise. Franceschi et al. [53] treated the graph structure as a discrete variable in an optimization setting, enabling the discovery of sparse dependency structures during training. See [157] for a survey of graph structure learning.

3.1.3 Forman-Ricci Curvature in Community Detection

A number of applications within the field of community detection and graph clustering [76, 51, 141, 107] employ the Forman-Ricci curvature as a pruning or clustering criterion.

Kim et al. [76] proposed *link pruning* as a preprocessing step for community detection. Their pruning procedure eliminates edges below a certain threshold according to edge attributes such as the Forman-Ricci curvature. The threshold is

chosen to preserve a proportion of the edges in the graph, in a similar manner to the sampling algorithms described in **Section** 3.1.1. After preprocessing, Kim et al. performed community detection using standard methods such as the Louvain method [19]. They used modularity (for real-world social networks) and normalized mutual information (NMI) for synthetic networks with ground-truth communities to evaluate the performance of these community detection methods. Roughly speaking, modularity measures the density of connections within communities, and NMI describes the similarity between the predicted communities with the ground-truth. The authors demonstrated empirically that community detection with link pruning in most cases achieved better performance in terms of modularity and NMI, in comparison with traditional graph sparsification.

As we will see in **Section** 3.2, community detection using the Ollivier-Ricci curvature is well-founded and has been proven to perform well in synthetic and real-world data. These results are based on the fact that positively curved edges, with respect to the Ollivier-Ricci curvature, tend to lie within communities, whereas negatively curved edges tend to serve as bridges between communities. The Forman-Ricci curvature behaves similarly. Although the Forman-Ricci curvature is advantageous for computational purposes compared to the Ollivier-Ricci curvature, it may be too simple in some cases to perform well in community detection.

Consequently, Fesser et al. [51] proposed to use the *augmented Forman-Ricci curvature* [67] for community detection and explored the connection between different augmentations and the expressive power and computational complexity for community detection. The augmented Forman-Ricci curvature is obtained from augmentations of the graphs. That is, by adding 2-dimensional cells as cofaces of all cocycles to obtain an unweighted cellular complex, and applying **Definition** 1 to the edges, we arrive at

$$\mathcal{AF}(e) = 2 + \Gamma_{ee} - \sum_{e' \sim e} |\Gamma_{ee'} + \Gamma_{e'e} - 1| - \sum_{e \not\sim e'} |\Gamma_{ee'} - \Gamma_{e'e}|, \qquad (24)$$

where $e \sim e'$ denotes two edges sharing a vertex and $e \not\sim e'$ two edges not sharing a vertex, and $\Gamma_{ee'}$ is the number of the cycles containing both e and e'. See **Figure** 2 for an example of the computation of this curvature. An alternative notion of augmented Forman-Ricci curvature can be obtained by restricting the length of the cycles considered in the definition of the augmentations of the graph:

$$\mathcal{AF}_n(e) = \mathcal{AF}(e) \text{ taking into account all cycles of at most length } n.$$

This definition is supported by the idea that curvature should be a local construction. Fesser et al. [51] mostly worked with $\mathcal{AF}_n$ with $n \in \{3, 4\}$ for community detection.

Several authors have studied the similarities between Ollivier-Ricci curvature and Forman-Ricci curvature [138, 72, 122], establishing empirical and theoretical equivalences and connections between these two notions. Fesser et al. [51] performed an extensive correlation analysis of the augmented Forman-Ricci curvature against the Ollivier-Ricci and the Forman-Ricci curvatures. They applied augmented Forman-Ricci curvature to community detection and demonstrated performance improvement against the Ollivier-Ricci curvature-based method. In general, community
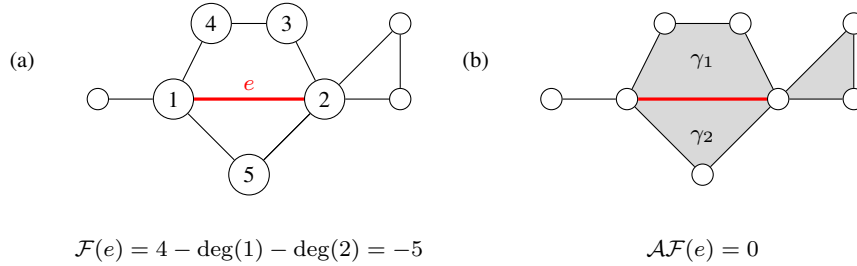
$$\mathcal{F}(e) = 4 - \deg(1) - \deg(2) = -5 \qquad\qquad \mathcal{AF}(e) = 0$$

Fig. 2: Forman-Ricci curvature vs. augmented Forman-Ricci curvature. Left: Forman-Ricci curvature of the edge e. Right: augmented Forman-Ricci curvature of the same edge. We see that the edge is contained in the cycles $\gamma_1 = 1234$ and $\gamma_2 = 125$. Looking at **Equation** (24), we then see that $\mathcal{AF}(e) = 2 + 2 - 3 - 1 = 0$. This image is a reproduction of Figure 1 from [51], licensed under CC BY (Creative Commons Attribution) and generated with the assistance of ChatGPT.

detection methods using the augmented Forman-Ricci curvature are faster and less computationally expensive than those based on the Ollivier-Ricci curvature.

Tian et al. [141] recently brought forward a unified framework for Ollivier-Ricci curvature-based community detection, including Ollivier and Forman-Ricci curvatures, in order to study their respective weaknesses and strengths. As part of this framework, they implemented a community detection algorithm based on the augmented Forman-Ricci curvature, with a weighting scheme for the edges (cf., [51]). In practice, they implemented $\mathcal{AF}_3$, that is, they considered only triangles and worked with simplicial complexes of dimension 2. They empirically showed that this reduction achieves a similar performance to the one with the full augmented Forman-Ricci curvature, while providing a significant speed-up in computations. In addition, they addressed a gap in the literature by incorporating mixed membership community detection to their analysis. In mixed-membership communities, one node might belong to more than one community, so that there might be overlaps between communities. Mixed membership is inspired by real-life situations. For instance, in a social network, a person might belong to more than one group of friends, such as school friends and university friends; or a protein might have several functional roles in a biochemical network (see [141, Figure 1] for an illustrative example). In order to incorporate this framework with curvature-based community detection methods, Tian et al. proposed to turn to the dual or line graph.

Park and Li [107] recently introduced the *lower Ricci curvature* for community detection. This new notion of curvature is inspired by the balanced Forman-Ricci curvature (already mentioned in **Section** 3.1.2), a curvature built upon the Forman-Ricci curvature to address its unboundedness and scale-dependent nature, in addition to its skewness toward negative values. As an extension of the Forman-Ricci curvature, the lower Ricci curvature has linear computational complexity. Park and Li also draw connections between the lower Ricci curvature and the Cheeger constant, which motivates the use of this curvature in community detection. In more detail, they

proved that if there exists a positive constant $\alpha > 0$, that is, a lower bound of the lower Ricci curvature for all the edges in the graph, then the diameter of the graph is bounded from above by $\alpha/2$ and the Cheeger inequality of the graph is bounded from below by the same quantity. This observation can be interpreted as follows: for larger values of α, the graph tends to be fully connected, hence having a smaller diameter and a larger Cheeger constant, implying more interconnectivity and less well-separated communities within the graph. Given these theoretical guarantees, Park and Li proposed a preprocessing algorithm that removes edges with lower Ricci curvature below a certain threshold, computed as the local minimum of a Gaussian mixture model fitted to the distribution of the lower Ricci curvatures. Using this preprocessing algorithm, they demonstrated improvement in efficiency and performance of a number of community detection methods across simulated and real-world data.

3.2 Ollivier-Ricci Curvature

The most popular application of the Ollivier-Ricci curvature for core finding is in clustering and community detection. Sia et al. [127] explored community detection using the Ollivier-Ricci curvature (**Equation** (4)) on weighted graphs. They considered probability measures $m_x(v) = \frac{w_{xv}}{\sum_{v' \sim x} w_{xv'}}$ if v is adjacent to x and $m_x(v) = 0$ otherwise. Here, w_{xv} denotes the weight of the edge between vertices x and v. In their study, negatively curved edges act as *bottlenecks* or *bridges* linking different communities, whereas positively curved edges represent within-community connections. Their algorithm removes edges with the most negative Ollivier-Ricci curvature, and then re-calculates the curvature for the affected edges. The process is repeated until all edges have non-negative curvature. Compared with other community detection methods based on modularity or edge betweenness, this curvature-based method showed comparable or better accuracy.

Ni et al. [102] introduced a family of probability distributions $m_x^{t,p}$ with $t \in [0, 1]$ and $p \geq 0$; see **Equation** (7). In most computations, they set $t = 0.5$ and $p = 2$, and utilized the resulting random walk $\{m_x^{0.5,2}\}_x$ to define the Ollivier-Ricci curvature; see **Figure** 1 for an illustration of this curvature in the case of tree graphs, grid graphs and complete graphs. They introduced a discrete Ricci flow for a weighted graph iteratively defined as $w_{xy}^{(i+1)} := \left(1 - \kappa^{(i)}(x, y)\right) d^{(i)}(x, y)$, with the superscript representing the iteration step, $d^{(0)}(x, y) = d(x, y)$, and $w_{xy}^{(0)}$ being the initial weight between x and y. The algorithm runs by first applying the graph with 20 to 50 iterations of discrete Ricci flow processes, and then removing edges from highest curvature to lowest. The hierarchical community structure of the graph is revealed through careful adjustment of the cut-off threshold. Experiments were conducted on both synthetic networks and real-world datasets with ground-truth communities. Compared to several other community detection methods based on modularity or edge betweenness, their algorithm demonstrates competitive or better performance,

measured using modularity and adjusted Rand index (ARI), which measures the accuracy of clustering results against the ground-truth.

Wu et al. [147] utilized the t-Ollivier-Ricci curvature of graphs (defined in **Equation** (8)) with $t = 0.5$ to develop a graph sampling algorithm that reduces the number of nodes and edges while preserving the number of communities. Their algorithm starts by randomly selecting an edge and calculating its Ollivier-Ricci curvature, then then expands the subgraph by choosing the next edge with the largest difference in Ollivier-Ricci curvature from the previously selected edge. Their algorithm captures community structure by differentiating between within-community edges and between-community edges, which is supported by their finding that edges within a community have larger Ollivier-Ricci curvature than edges between that community and some other communities. Compared with degree-based graph sampling methods via the mean absolute error (MAE), this curvature-based method is more effective at preserving minor communities.

Different from [127, 102] where clusters are identified based on finding negatively curved edges between clusters, Gosztolai and Arnaudon [60] showed that along its evolution, the distribution of the dynamical Ollivier-Ricci curvature of edges exhibits gaps, i.e., differences in the relative magnitude of curvatures, which at characteristic timescales indicate bottleneck-edges that limit information spreading. These curvature gaps are claimed to be robust against large fluctuations in node degrees, preserving community structures up to the phase transition of detectability. As the diffusion process advances, these gaps capture progressively coarser community features; and they are used to uncover multiscale communities by analyzing deviations from constant curvature.

Tian et al. [140] interpreted the graph clustering algorithms of Ni et al. [102] and Sia et al. [127] as specific instances of a broader algorithm. These algorithms are based on geometric flows associated with discrete curvature [104], under which the edge weights of the graph evolve in a way that reveals its community structure. Starting with a possibly unweighted graph, let the edge weights evolve under the curvature flow, up to a re-normalization at each iteration: calling $G^T = (V, E^T, w^T)$ the weighted graph at the iteration T, having fixed cut-off thresholds h_T, the new edge set is

$$E^{T+1} := \{e \in E^T : w^T(e) > h_T\},$$

and the updated weights for any edge $e \in E^{T+1}$ with $e = \{u, v\}$,

$$w^{T+1}(e) := (1 - \kappa(e))d_G(u, v),$$

where κ is the discrete curvature of G^T and similarly d_G is the graph distance of G^T.

The intuition is that, over time, the negative curvature of edges connecting different communities intensifies: edges with lower curvature experience a reduction in weight under the flow. On the other hand, edges with higher curvature get larger weights. Thus, interpreting weights as lengths of edges, the internal edges will contract faster over time, and the bridges will contract slower. The above algorithm is independent of the particular discrete curvature chosen. In any case, the success of the method depends crucially on identifying a good weight threshold for cutting edges at each

step. To speed up the algorithm in the case of the Ollivier-Ricci curvature, Tian et al. used the combinatorial approximation of the Ollivier-Ricci curvature given in **Equation** (16).

3.3 Graph Sparsification with Effective Resistance

Another avenue to find the core of a graph that preserves its spectral properties utilizes the notion of effective resistance (see **Equation** (19)), which is directly linked to the Laplacian of a graph.

Spielman and Srivastava [130] introduced a nearly-linear time algorithm that, given a weighted graph $G = (V, E, w)$, produces a sparsified graph $H = (V, F, u)$ that approximately preserves the graph Laplacian of G. H and G share the same set of vertices, but H contains fewer edges. Let $R(e, e)$ be the effective resistance of an edge e. The algorithm samples a predetermined number q of edges e from G with probability $p(e)$ proportional to $w(e)R(e, e)$ (the relative effective resistance, as named in [37]) and adds them to H with weight $w(e, e)/(qp(e))$. The sampling is done independently with replacement, summing the weights if an edge is sampled more than once. Under such a construction, for $0 < \varepsilon \leq 1$, the algorithm produces a sparsified graph H after q sampling steps, such that the Laplacian of the sparsified graph L_H with (re-weighted) edges satisfies the following relation with respect to the Laplacian of the original graph L_G:

$$(1 - \varepsilon)\mathbf{x}^\top L_G \mathbf{x} \leq \mathbf{x}^\top L_H \mathbf{x} \leq (1 + \varepsilon)\mathbf{x}^\top L_G \mathbf{x}, \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

The Courant-Fischer theorem, provides a variational characterization of the eigenvalues λ_i, $1 \leq i \leq n$ of the Laplacian L in terms of the following min-max expression:

$$\lambda_i = \max_{\substack{V \subseteq \mathbb{R}^n \\ \dim(V) = i}} \min_{\substack{\mathbf{x} \in V \\ \mathbf{x} \neq \mathbf{0}}} \frac{\mathbf{x}^\top L \mathbf{x}}{\mathbf{x}^\top \mathbf{x}}.$$

The above expression allows us to conclude that

$$(1 - \varepsilon)\,\lambda_i(G) \leq \lambda_i(H) \leq (1 + \varepsilon)\,\lambda_i(G), \quad \forall 1 \leq i \leq n.$$

The strategy followed in this work was later extended to simplicial complexes by Osting et al. [105] (reviewed in **Section** 5.5).

Our survey of methods for finding graph cores shows that curvature-based approaches predominantly employ thresholding techniques to remove vertices or edges. This preference likely stems from theoretical insights into discrete curvatures, which suggest that edges with relatively high or low absolute curvatures convey certain structural information, thereby justifying their removal based on threshold criteria. On the other hand, the spectral method that leverages effective resistance employs a probability-based sampling strategy, which removes vertices or edges according

Table 2: A summary of recent works in finding cores of hypergraphs, as surveyed in **Section** 4. V: vertices (nodes). E: hyperedges. ER: effective resistance. "robustness": hypergraph robustness.

Ref.	Hypergraph	Method	Criteria	Property Preserved
[136]	undirected	percolation, V and E	none	robustness
[86]	undirected	percolation, V and E	degree	robustness, subhypergraph
[17]	undirected	percolation	none	robustness, subhypergraph
[129]	(un)directed	sparsification	Laplacian	spectral
[73]	(un)directed	sparsification	Laplacian	spectral
[85]	undirected	sparsification	Laplacian	spectral
[68]	undirected	sparsification	Laplacian	spectral
[27]	undirected	cut sparsification	max cut	global maximum cut
[30]	undirected	cut sparsification	auxiliary graph edge strength	spectral
[2]	weighted	threshold, V and E	ER	clusters
[156]	undirected	normalized cut	Laplacian	clusters
[150]	weighted	normalized cut	intra-weights, E	clusters
[34]	undirected	threshold, E	ORC, edge and node features	clusters

to a predefined probability distribution. For future work, it would be interesting to extract graph cores by sampling with resistance curvature, investigate the properties preserved, and study the connection between effective resistance based sparsification and resistance curvature based sampling.

A number of research works utilize probability-based sampling methods (often referred to as network percolation) to extract cores from graphs or to study the effect of random removal of nodes and edges. Since network percolation is a well-established field, we do not review relevant works in this paper, instead, we refer readers interested in this topic to a recent survey [118]. On the other hand, we believe that hypergraph percolation for core finding is a fairly new topic and deserves some attention, which is surveyed in **Section** 4.

4 Finding Cores of Hypergraphs

Numerous studies have focused on identifying the cores of hypergraphs. Techniques include finding the (k, q)-core of the hypergraph [86], preserving the *quadratic form of the Laplacian* of the resulting hypergraph [129], and approximately preserving the cuts [27]; see **Table** 2 for a summary. There is a plethora of clustering and community detection studies for hypergraphs (see, for example, [32, 116, 155], among others). However, few of them present a topological or geometric perspective,

favoring probabilistic or statistical approaches. We close the section reviewing the only two—to the best of our knowledge—geometric-based clustering method for hypergraphs: an extension of the Normalized Cut algorithm [126, 94, 101] for spectral clustering in graphs [156, 150]; and an application of spectral clustering based on the Ollivier-Ricci curvature [34].

4.1 Preliminaries on Hypergraphs

Given a hypergraph $H = (V, E)$, we let $n = |V|$, $m = |E|$, and $p = \sum_{e \in E} |e|$, where $|e|$ denotes the hyperedge size, that is, the number of vertices (or nodes) it contains. We let the rank of the hypergraph r to be the maximum hyperedge size. A hypergraph is *capacitated* if it is equipped with a capacity function $c : E \to \mathbb{R}^+$, where $c(e)$ assigns a positive real number to each hyperedge $e \in E$, representing the capacity of that hyperedge. If all capacities are equal to one, the hypergraph is called *uncapacitated*. The (k, q)-*core* of a hypergraph, defined by Ahmed et al. in [3], is a maximal subgraph in which each node has at least k hypergraph degrees and each hyperedge contains at least q nodes [86].

Several notions of hypergraph Laplacians exist in the literature, each motivated by different applications or structural properties [1, 64, 88, 120, 119]. Here, we present one such formulation, based on the submodular Laplacian framework [151], which is commonly used in recent works on spectral sparsification. The Laplacian $L_H : \mathbb{R}^V \to \mathbb{R}^V$ of an undirected weighted hypergraph H with weight function $w \in \mathbb{R}_+^E$ is defined [92, 151] such that its quadratic form $\mathbf{x}^\top L_H \mathbf{x}$ satisfies

$$\mathbf{x}^\top L_H \mathbf{x} = \sum_{e \in E} w(e) \max_{u,v \in e} (\mathbf{x}(u) - \mathbf{x}(v))^2,$$

for every $\mathbf{x} \in \mathbb{R}^V$.

We say that a node set $S \subseteq V$ *cuts* a hyperedge e if $e \cap S \neq \emptyset$ and $e \cap (V \setminus S) \neq \emptyset$. Specifically, for a set $S \subseteq V$, the quadratic form $\mathbf{1}_S^\top L_H \mathbf{1}_S$ coincides with the cut size of S, where $\mathbf{1}_S \in \mathbb{R}^V$ is the characteristic vector of S.

A directed hypergraph $H = (V, E, w)$ as defined by Gallo et al. in [54] consists of a node set V, a set of hyperedges E, and a weight function $w \in \mathbb{R}_+^E$, where each hyperedge e is a pair (T_e, H_e) of (not necessarily disjoint) sets of nodes, where $T_e, H_e \subseteq V$ are called the tail and head of e, respectively. The Laplacian $L_H : \mathbb{R}^V \to \mathbb{R}^V$ for a directed hypergraph H can be derived using the submodular Laplacian framework. Using the submodular form of the Laplacian [151], Soma and Yoshida [129] derived the quadratic form $\mathbf{x}^\top L_H \mathbf{x}$ for a directed hypergraph to be as follows:

$$\mathbf{x}^\top L_H \mathbf{x} = \sum_{e=(T_e, H_e) \in E} w(e) \max_{u \in T_e} \max_{v \in H_e} ([\mathbf{x}(u) - \mathbf{x}(v)]^+)^2,$$

where $[x]^+ = \max(x, 0)$.

A weighted subgraph H' of a hypergraph H on a node set V is said to be an ε-spectral sparsifier of H if the following holds for every $x \in \mathbb{R}^V$ [129][3],

$$(1 - \varepsilon)\mathbf{x}^\top L_{H'}\mathbf{x} \leq \mathbf{x}^\top L_H \mathbf{x} \leq (1 + \varepsilon)\mathbf{x}^\top L_{H'}\mathbf{x}.$$

4.2 Percolation Methods for Finding Hypergraph Cores

Percolation processes describe the size of the largest connected component when nodes or hyperedges are randomly removed [17]. In the context of studying the dependence of hyperedges on their nodes, also known as hypergraph percolation, as noted by Bianconi and Dor in [17], two types of hyperedges capture different classes of higher-order interactions. The first type, found in networks such as social interactions, remains intact unless all but one of its nodes are removed, or fail, meaning that the hyperedge collapses only when it loses all but one of its constituent nodes. This failure indicates that the functionality or connection of the hyperedge is lost when it is no longer supported by a sufficient number of nodes. Thus, these hyperedges can withstand the failure of one or more nodes. The second type consists of hyperedges that fail once one of their nodes is damaged, such as in supply chains or protein-interaction networks. In the first category, Bianconi and Dorogovtsev [17] noted that current theories and models that treat hypergraphs as factor graphs—which represent hypergraphs as bipartite networks of nodes and hyperedges—are completely effective.

We start by outlining contributions to the theory of percolation for hypergraphs with hyperedges of varying cardinalities. Sun and Bianconi [136] studied *hypergraph pruning* (cf., link pruning from **Section** 3.1.3) at the node and hyperedge levels separately, proposing percolation methods for finding the $(k, 1)$- and $(1, k)$-cores of random hypergraphs, respectively; this percolation procedure extends percolation on factor graphs. Higher-order percolation on hypergraphs is related to percolation on multiplex networks, that is, networks with multiple layers sharing the same set of nodes. Random hypergraphs can exhibit complex multiplex topologies, which are characterized by layers representing hyperedges of specific cardinalities, leading to the concept of random multiplex hypergraphs [136]. This approach avoids the strict assumption of fixed cardinality hyperedges. The authors argued that multiplex hypergraphs are excellent tools in statistical mechanics for examining a wide range of higher-order percolation processes, applicable across various fields. They defined ensembles of random multiplex hypergraphs, where each node has a generalized degree vector representing the number of hyperedges of different cardinalities incident to it. These hypergraphs show significant interlayer hyperdegree correlations and differ from multiplex bipartite networks, although mapping between the two is possible under certain conditions. The study demonstrates that the complex topology and interlayer correlations of multiplex hypergraphs influence percolation processes, allowing adjustment of the percolation threshold. The research reveals that the

---

[3] Notice that there is a difference between the formulation from [129] vs. the formulation from [130].

multiplex nature of hypergraph ensembles can be used to explore higher-order percolation problems, uncovering complex behaviors such as discontinuous hybrid transitions and multiple percolation transitions.

Lee et al. [86] extended this concept by defining the (k, q)-core of a hypergraph, which is obtained by iteratively removing nodes with degrees less than k and hyperedges smaller than q, leading to a repeated pruning process. As the process continues, any nodes and hyperedges that subsequently fall below these thresholds are also removed. This process results in a core subhypergraph where every node has at least k connections, and every hyperedge connects to at least q nodes. To describe this pruning process for uncorrelated hypergraphs, that is, hypergraphs where there is no correlation between node degrees and hyperedge sizes, the authors derived evolution equations for the degree and size distributions. These equations, framed within a bipartite network representation of hypergraphs, track the fractions of nodes and hyperedges pruned at each step. The self-consistency equations for the core structure were obtained using generating functions, which facilitated the analysis of (k, q)-core percolation transitions.

The analysis of Lee et al. revealed that there is a hybrid phase transition of (k, q)-core percolation when $k \geq 3$ or $q \geq 3$ and the second moments of the degree and size distributions are finite. This hybrid phase transition is characterized by a discontinuous jump in the order parameter, accompanied by critical behaviour at the transition point. Additionally, the relaxation dynamics of the fractions of nodes with degree $z = k - 1$ and hyperedges of size $n = q - 1$ at the transition point exhibit a universal behaviour proportional to t^{-2} for $k \geq 3$ or $q \geq 3$, where t denotes the time step. When $k = q = 2$, Lee et al. analytically derived and numerically confirmed a novel degree-dependent critical relaxation dynamics, showing that $P^{(u)}(z, t) \sim t^{-z}$ and $P^{(e)}(n, t) \sim t^{-n}$ for z and $n \geq 2$, with both decaying as t^{-3} for $z = n = 1$. Here, $P^{(u)}(z, t)$ represents the probability distribution of nodes with degree z at time t, and $P^{(e)}(n, t)$ represents the probability distribution of hyperedges with size n at time t.

Lee et al. further demonstrated that (k, q)-core decomposition is more effective in eliminating redundant modular structures compared to traditional k-core decomposition. This obervation was highlighted through a case study on a coauthorship hypergraph, where (k, q)-core decomposition successfully identified high-impact teams by eliminating modular structures with large hyperedges and many low-degree nodes. Lee et al. also noted that (k, q)-core decomposition has advantages in classifying groups (hyperedges) based on their properties more effectively than the previous k-core decomposition via this case study. Whereas the theoretical analysis was confined to hypergraphs with homogeneous distributions, the authors remarked that their theory can be generalized to heterogeneous cases due to the convergence of the infinite series, irrespective of the moments' divergence in the degree distribution. Furthermore, the method allows for extensions to cases with degree-size correlation. A key shortcoming of the existing theory, as pointed out in [86], is its reliance on a tree-like assumption. This relaxation and the development of broader theoretical models represent promising directions for future research.

Percolation theory for hypergraphs with hyperedges belonging to the second type, i.e., hyperedges that fail once one of their nodes is damaged, was recently developed by Bianconi and Dorogovtsev [17]. Whereas hypergraphs can be efficiently represented by factor graphs, k-core percolation can significantly differ between the two. To address these disparities, Bianconi and Dorogovtsev introduced a set of pruning processes targeting either nodes or hyperedges based on connectivity in the second neighbourhood. They constructed a message-passing theory for hypergraph percolation, utilizing the generating function formalism and supporting their analysis with Monte Carlo simulations on both random and real-world hypergraphs. They further derived message-passing equations for percolation on both factor graphs and hypergraphs. Next, they applied message-passing theory to random hypergraphs described by a degree distribution for nodes and a cardinality distribution for hyperedges, and a joint degree distribution where each degree is a vector listing the numbers of hyperedges of each cardinality adjacent to a node.

Bianconi and Dorogovtsev further investigated the critical behavior of hypergraph percolation, revealing that the node percolation threshold for hypergraphs is higher than that for factor graphs. Additionally, unlike ordinary graphs where node and edge percolation thresholds coincide, in hypergraphs the situation is different, with the threshold for node percolation being higher than that for hyperedge percolation. The authors determined the presence of a percolation cluster (i.e., a giant connected component) and its relative size through self-consistent equations, which are derived using generating functions. The study shows that in hypergraphs, distributions of hyperedge sizes with heavy tails do not result in hyperresilience, which is a contrast to factor graphs. In factor graphs, if the size distribution has a diverging second moment, it leads to a percolation threshold of zero. The distinction between node and hyperedge percolation in hypergraphs is significant, especially when hyperedges have large cardinalities. Notably, when pruning focuses solely on hyperedges, the phase diagram converges to that of factor graph k-cores.

Importantly, the message-passing algorithm developed by Bianconi and Dorogovtsev does not assume the absence of correlations in hypergraphs, which allows the problem to be treated numerically. The final formulas are obtained for hypergraphs without degree-degree correlations between different nodes. The authors suggested that the case of correlated hypergraphs can also be treated analytically within their framework.

Recently, Bianconi and Dorogovtsev [16] studied k-core percolation processes on hypergraphs and contrasted these with the analogous processes on factor graphs. The critical distinction arises in the integrity of hyperedges: in hypergraphs, every node within a hyperedge must be intact for the hyperedge to be considered intact, reflecting the all-or-none characteristic observed in systems such as supply chains, protein-interaction networks, and chemical reactions. The authors formulated a message-passing theory tailored to the k-core percolation on hypergraphs, leveraging the theory of critical phenomena on networks to underline the main differences from factor graph-based percolation where hyperedges can still function despite partial node failures. They explored phase transitions and phase diagrams in these systems, noting more complex behaviors in hypergraphs compared to factor graphs, especially

in scenarios where node or hyperedge pruning is based on broader neighbourhood connectivity. To resolve discrepancies between k-core percolation on hypergraphs and factor graphs, they introduced a series of pruning processes, designed to either exclusively target nodes or hyperedges, with a dependence on their connectivity to the second neighbourhood.

4.3 Spectral Sparsification Methods

Soma and Yoshida [129] extended spectral sparsification of graphs to hypergraphs by providing an ε-spectral sparsifier for a hypergraph H. They presented a polynomial-time algorithm that constructs an ε-spectral sparsifier of an undirected or directed hypergraph $H = (V, E)$ with $O(n^3 \log n/\varepsilon^2)$ hyperedges, where $n = |V|$. Their main contributions include the development of a randomized algorithm for spectral sparsification and its application to various computational problems involving hypergraph Laplacians. For an undirected hypergraph, their algorithm outputs an ε-spectral sparsifier H with $O(n^3 \log n/\varepsilon^2)$ hyperedges with high probability in $O(pn + m \log(1/\varepsilon^2) + n^3 \log n/\varepsilon^2)$ time, where $n = |V|$, $m = |E|$, $p = \sum_{e \in E} |e|$.

Soma and Yoshida further showed that spectral sparsification of directed hypergraphs can also be achieved with $O(n^3 \log n/\varepsilon^2)$ hyperedges. This result is significant because even directed graphs with $O(n^2)$ edges do not admit nontrivial cut/spectral sparsification of size $o(n^2)$ ($o(\cdot)$ representing little-O notation), whereas directed hypergraphs, which could have $O(4^n)$ hyperedges, do admit such sparsification. Their study also demonstrated how spectral sparsification can be applied to enhance the time and space efficiency of algorithms that work with the quadratic forms of hypergraph Laplacians. This study includes tasks such as calculating eigenvalues, solving systems based on Laplacians, and conducting semi-supervised learning.

Kapralov et al. [73] presented a polynomial-time algorithm that constructs an ε-spectral sparsifier for a hypergraph with only $O^*(nr)$ hyperedges, where r is the maximum hyperedge size, and the notation $O^*(\cdot)$ suppresses $(\varepsilon^{-1} \log n)^{O(1)}$ factors [73]. This result improves upon the previous bounds, i.e., $O^*(n^3)$ set by Soma and Yoshida [129] and $O^*(nr^3)$ by Bansal et al. [9]. The algorithm achieves this sparsification with high probability and a running time of $O(mr^2) + n^{O(1)}$. Additionally, the authors established lower bounds on the bit complexity for any compression scheme that approximates all cuts in a hypergraph within a factor of $1 \pm \varepsilon$. We refer the reader to **Section** 4.1 for the definition of a hypergraph cut. They demonstrated that $\Omega(nr)$ bits are required to represent all cut values of an r-uniform hypergraph where r equals $n^{O(1/\log \log n)}$. For directed hypergraphs, Kapralov et al. further presented an algorithm that computes an ε-spectral sparsifier with $O^*(n^2 r^3)$ hyperedges, which represents a significant improvement over the previously known $O^*(n^3)$ bound by Soma and Yoshida. This result is particularly impactful when the rank r is small. They also introduced a new approach for proving concentration of the nonlinear version of the quadratic form associated with the Laplacians in hypergraph expanders.

Independently, Lee [85] and Jambulapati et al. [68] further improved the bound in the undirected hypergraph with a near-linear time algorithm. They introduced an algorithm that constructs an ε-spectral sparsifier with $O(n\varepsilon^{-2} \log n \log r)$ hyperedges for an n-node, m-edge hypergraph of rank r. This algorithm operates in nearly linear $O^*(mr)$ time, significantly enhancing both the size and efficiency over previous works (e.g., [9, 73]). The algorithm by Jambulapati et al. [68] utilizes a novel approach involving *group leverage score* overestimates and generic chaining, enhancing the calculation of sampling weights in nearly linear time to achieve optimal sparsification bounds. In the context of hypergraphs, group leverage scores assess the importance or influence of hyperedges, typically derived from spectral properties. By overestimating these scores, the algorithm simplifies the calculations, using approximations to determine the importance of each hyperedge efficiently. This approximation results in adjusted sampling probabilities for retaining significant hyperedges in the sparsifier, which maintains the structural integrity of the original hypergraph.

Furthermore, the technique of generic chaining, developed by Talagrand [137], was adapted to manage the accumulation of error in sampling. This method constructs multiple layers of increasingly finer nets over the probability space, chaining these layers to control the maximum deviation of the stochastic processes involved. This rigorous control allows for a more refined process in determining which hyperedges to retain, tightening the bound on the number of necessary hyperedges. These innovations offer improvements over previous methods, demonstrating that every hypergraph admits a sparsifier with nearly linear time complexity and fewer hyperedges. Specifically, this method effectively reduces the bound on the number of hyperedges to $O(n\varepsilon^{-2} \log n \log r)$, improving over the previous bounds and approaching the theoretical limits of sparsification.

Given a hypergraph $H = (V, E)$ with $n = |V|$, $m = |E|$, and $p = \sum_{e \in E} |e|$, Chekuri and Xu [27] extended sparsification techniques from graphs to uncapacitated hypergraphs, focusing on constructing a k-trimmed certificate. A *k-trimmed certificate* is a (sparse) subhypergraph H' obtained via hyperedge deletion and trimming that preserves all cuts of value up to k [27]. The authors improved on previously bounds in [112, 93, 78] to find the min-cut (or minimum cut) of an arbitrary symmetric submodular function. For a hypergraph, their method adapts the *maximum adjacency ordering* to arrange nodes in a sequence where each node has high connectivity to previously ordered nodes. Edges are then evaluated, and for each node v, only the first k significant backward edges (i.e., those connecting v to previously ordered nodes) are retained. This process trims the hypergraph by removing less critical connections while maintaining essential connectivity properties. The construction of the k-trimmed certificate involves creating a data structure in $O(p)$ time. The practical implications include significantly faster algorithms for computing the global min-cut in hypergraphs. For example, the global min-cut can be computed in $O(p + \lambda n^2)$ time, where λ is the min-cut value of the hypergraph. Chekuri and Xu proposed a split oracle that, given a hypergraph H and its min-cut value λ, determines if a split exists between any two nodes $\{s, t\}$. The oracle operates in near-linear time by converting H into an equivalent directed graph H_D and computing the maximum s-t flow in H_D. If this flow exceeds λ, no s-t split exists; otherwise, a nontrivial

min-s-t cut implies a split. Canonical and prime decompositions are introduced to further break down a hypergraph. A hypergraph is *prime* if it contains no splits, with all min-cuts being trivial. A *canonical* decomposition captures all min-cut information and is constructed by refining the hypergraph iteratively using splits. This decomposition helps in identifying cores, as they correspond to components maintaining the hypergraph connectivity. To compute the canonical decomposition, an algorithm starts with a prime decomposition and iteratively glues components. This algorithm operates in $O(np + n^2 \log n)$ time for capacitated hypergraphs and $O(np)$ time for uncapacitated hypergraphs.

Chekuri and Xu [27] discussed a compact representation of a hypergraph called the *hypercactus representation*, which captures all min-cuts and is derived from the canonical decomposition. For a hypergraph with $\lambda = 1$, the hypercactus is constructed by identifying marker nodes and combining canonical decomposition components. The hypercactus representation can be computed in $O(n(p + n \log n))$ time and $O(p)$ space for capacitated hypergraphs, and in $O(p + \lambda n^2)$ time for uncapacitated hypergraphs. The authors then extended Matula's algorithm to hypergraphs, obtaining a $(2 + \varepsilon)$-approximation for the global min-cut of a capacitated hypergraph in $O(\frac{1}{\varepsilon}(p \log n + n \log^2 n))$ time and for uncapacitated hypergraphs in $O(p/\varepsilon)$ time. This generalized version of Matula's algorithm incorporates the use of node orderings, inspired by Nagamochi and Ibaraki's [99] maximum adjacency ordering method for graphs. Unlike graphs, hypergraphs have several possible orderings, and these yield different insights into their structure. The authors present an algorithm that computes approximate strengths for all edges in a hypergraph, running in $O(p \log^2 n \log p)$ time. The *strength* $\gamma_H(e)$ of an edge e is defined as the maximum min-cut value over all node-induced subhypergraphs containing e. The algorithm ensures that the estimated strength $\gamma'(e)$ is less than or equal to the actual strength $\gamma_H(e)$, and that the sum of the inverses of the approximate strengths satisfies certain bounds. This edge strength estimation facilitates the construction of a $(1 + \varepsilon)$-cut sparsifier, a reduced hypergraph that preserves cut size within a factor of $1 \pm \varepsilon$. This cut sparsifier can be found in near-linear time, significantly reducing the number of edges while maintaining essential connectivity properties. This approach extends the work of Kogan and Krauthgamer [79] and leads to faster algorithms for solving various cut and flow problems in hypergraphs of small rank. By using the approximate strengths to sample edges and create sparsifiers, the authors achieved efficient cut approximations.

Chen et al. [30] improved on the work of Chekuri and Xu, providing an algorithm that runs in $\tilde{O}(mn + n^{10}/\varepsilon^7)$ time that constructs an approximate sparsifier of size $O\left(\varepsilon^{-2} n \log n\right)$. The authors claimed that this size bound is the best possible bound within a logarithmic factor by providing an example of a hypergraph such that any sparsifier must include every hyperedge. Their algorithm works by constructing a sparsifier via an auxiliary graph G as follows. First, for each hyperedge e in H, add a clique F_e to G whose node set is the same as the nodes of e, then assign weights to the edges in G according to a balanced weight scheme, where the weights of the edges in F_e range from 0 to the weight of the hyperedge e. With this setup, the probability of sampling a hyperedge e in H is determined by the balanced weight scheme on the clique $F_e \in G$.

By generalizing the notion of effective resistance to hypergraphs, Aghdaei and Feng [2] introduced a scalable algorithmic framework, called HyperEF, for spectral coarsening of large-scale hypergraphs. This generalization relies on the nonlinear quadratic form introduced by Chan et al. in [26]: for a weighted undirected hypergraph $H = (V, E, w)$ with non-negative weights, and a vector $\mathbf{x} \in \mathbb{R}^{|V|}$,

$$Q(\mathbf{x}) := w_e \sum_{e \in E} \max_{v_k, v_l \in e} (\mathbf{x}(k) - \mathbf{x}(l))^2,$$

where $\mathbf{x}(k)$ is the k-th element of the vector $\mathbf{x}$. In the graph setting, the effective resistance between two nodes v_i and v_j satisfies the following property, as shown in Theorem 1 in [2]:

$$r_{v_i v_j} = \max_{\mathbf{x} \in \mathbb{R}^{|V|}} \frac{(\mathbf{x}^\top (\mathbf{e}_i - \mathbf{e}_j))^2}{\mathbf{x}^\top L \mathbf{x}}.$$

By extending the above formula to the hypergraph setting while replacing $\mathbf{x}^\top L \mathbf{x}$ with the nonlinear quadratic form $Q(\mathbf{x})$, Aghdaei and Feng [2] defined the effective resistance of a hyperedge e as

$$R_e := \max_{\mathbf{x} \in \mathbb{R}^{|V|}} \frac{\left(\mathbf{x}^\top (\mathbf{e}_i - \mathbf{e}_j)\right)^2}{Q(\mathbf{x})}, \tag{25}$$

assuming v_i and v_j are some nodes within e. To enhance computational efficiency, R_e is computed approximately as follows. For a vector $\mathbf{x} \in \mathbb{R}^{|V|}$, we call $\tilde{R}_e(\mathbf{x}) := \frac{(\mathbf{x}^\top (\mathbf{e}_i - \mathbf{e}_j))^2}{Q(\mathbf{x})}$ the resistance ratio associated with $\mathbf{x}$. Then, R_e is approximated as the sum of the resistance ratios associated with a few Laplacian eigenvectors $\{\xi_i\}$ of a certain bipartite graph G_b converted from the original hypergraph. This approximation is motivated by the following property of graph effective resistance: assuming $\{\xi_i\}_{i=1}^{|V|}$ is an orthonormal eigenbasis of the graph Laplacian associated with eigenvalues $\{\lambda_i\}_{i=1}^{|V|}$, then

$$r_{v_i v_j} = \sum_{\lambda_i \neq 0} \frac{(\mathbf{x}^\top (\mathbf{e}_i - \mathbf{e}_j))^2}{\xi_i^\top L \xi_i}.$$

Here, the vectors $\{\xi_i\}$ are computed approximately as the orthonormal basis of the Krylov subspace $K := \mathrm{span}\{x, Ax, \ldots, Ax^\rho\}$, for a non-negative integer ρ, a random vector x and the adjacency matrix A of the bipartite graph G_b. Nodes v_i and v_j in **Equation** (25) are selected as the maximally separated nodes within this ρ-dimensional embedding space K.

The HyperEF algorithm iteratively contracts the hyperedges and nodes via the following steps: (1) compute the approximate effective resistance as described above; (2) compute the node weights via a specified formula; (3) compute the hyperedge weights using both the approximate effective resistance and the node weights; and (4) contract the hyperedges and cluster nodes when their weights are below a threshold. The performance of the algorithm is tested by its preservation of the so-called average conductance of clusters. Experiment results on real-world VLSI designs

(i.e., very-large-scale integration, the process of creating an integrated circuit by combining thousands to millions of transistors on a single chip) showed that HyperEF is faster than HMetis [74], a well-known hypergraph partitioner, while maintaining comparable or better average conductance values in most cases.

4.4 Spectral Clustering for Hypergraphs

A successful clustering technique in the graph domain is *spectral clustering*, based on the eigenvalues and eigenvectors of some matrix containing geometric information about the graph. A well-established method for spectral clustering is the normalized cut algorithm [126, 94], which intuitively minimizes the weights of the edges connecting different clusters without favoring clusters of isolated nodes in the graph.

This normalized cut algorithm was extended to hypergraphs by Zhou et al. [156]. The authors showed that, much like in the graph case, a relaxation in the hypergraph cut criterion allows to express the optimal cut from a decomposition of a positive semidefinite matrix, which can be regarded as a hypergraph Laplacian. Nonetheless, their approach had two limitations. On the one hand, it assumed that nodes within the same hyperedge have equal relevance, which can be restrictive. On the other hand, the need to solve an eigenproblem made the approach time and storage consuming.

GraphLSHC [150] was proposed by Yang et al. as an extension of the previous method to overcome such limitations. For the importance of edges problem, the authors proposed to use weighted hypergraphs, and extended the normalized cut algorithm to such a framework. Here, the weights refer to "intra-weights" within hyperedges. To address the computational issues, they proposed several optimizations, including an "eigen-trick" that is particularly designed to accelerate the eigenproblem solver.

To the best of our knowledge, the only existing curvature-based spectral clustering algorithm for hypergraphs was proposed by Coupette et al. [34], who proposed an flexible framework to extend the Ollivier-Ricci curvature to hypergraphs (see **Section** 2.2). As part of their experimental exploration on *hypergraph learning*, they performed spectral clustering using curvatures and other local edge features (e.g., the number of neighbours of a given edge) and node features (e.g., the average curvature of the incident edges in a node, or the size of its neighbourhood). They evaluated the quality of the clustering through the *Wasserstein clustering coefficient*, which measures average intra-cluster Wasserstein distances to inter-cluster Wasserstein distances; and observed that curvature-based clustering consistently obtains better results.

Table 3: A table summarizing recent works in finding cores of simplicial complexes, as surveyed in **Section** 5. Ref.: references. SC: simplicial complex. WSSD: well-separated simplicial decomposition. PD: persistence diagram. VR: Vietoris–Rips complex. Čech: Čech complex. ER: effective resistance.

Ref.	SC	Key component(s)	Criteria	Property Preserved
[125]	VR	point collapse/removal	metric ball cover	PD
[39]	VR	batch collapse with pairwise distance	metric ball cover	PD
[40]	VR	batch collapse with set distance	metric ball cover	PD
[75]	Čech	WSSD	metric ball cover	PD
[95]	VR	vertex quasi-distance	codensity	PD
[13]	SC	simplicial collapse	none	PD
[24]	VR	subsample smaller point clouds	none	PD
[59]	VR	subsample smaller point clouds	none	PD
[128]	VR	subsample smaller point clouds	none	PD
[105]	SC	Sampling with replacement	ER	spectral
[43]	SC	harmonic embedding, subspace clustering	none	homology generators (experimentally)

5 Finding Cores of Simplicial Complexes

Finally, we review recent works in finding cores of simplicial complexes. Majority of these works arise from the fields of computational topology; thus, they have a strong focus on preserving homological information of the complexes. After reviewing various notions of simplicial complexes in **Section** 5.1, we discuss sparsification methods that preserve (persistent) homology (**Sections** 5.3 and 5.4) and spectral properties (**Section** 5.5) of the complexes. We finish this section with a discussion on simplicial spectral clustering (**Section** 5.6). Whereas there have been recent works on percolation theory for simplicial complexes, e.g., *homological percolation* by Bobrowski and Skraba [20], different from hypergraphs, these works have yet to produce well-defined cores. Therefore, in this paper, we choose not to survey simplicial percolation.

5.1 Preliminaries on Simplicial Complexes

We first review the definitions of various simplicial complexes popular in computational topological, including Čech, Vietoris–Rips, Delaunay, Alpha, Delaunay-Čech, and Wrap complexes.

Given a finite set $X \subseteq \mathbb{R}^n$ and a radius $r \geq 0$, the *Čech complex* of X is the nerve of the r-balls centered at points of X, that is,

$$\check{C}ech_r(X) = \{Q \subseteq X \mid \bigcap_{x \in Q} B_x(r) \neq \emptyset\}.$$

The Vietoris–Rips complex for a scale r is

$$VR_r(X) = \{Q \subseteq X \mid \mathrm{diam}(Q) \leq r\}.$$

The *Voronoi cell* of a point $x \in X$ (w.r.t. X) contains all points closer to x than to any other point,

$$\mathrm{Vor}(x) = \{y \in \mathbb{R}^n \mid d(y, x) \leq d(y, p), \ \forall\, p \in X\}.$$

The *Voronoi ball* of x (w.r.t. X) for a radius r is

$$\mathrm{Vor}_r(x) = B_r(x) \cap \mathrm{Vor}(x).$$

The *Delaunay complex* of X for a radius r is

$$\mathrm{Del}_r(X) = \{Q \subseteq X \mid \bigcap_{x \in Q} \mathrm{Vor}_r(x) \neq \emptyset\}.$$

$\mathrm{Del}_r(X)$ is also called the *alpha complex*. The *Delaunay triangulation* of X is $Del(X) := Del_\infty(X)$.

The *Delaunay-Čech* complex for a radius r restricts the Čech complex to the Delaunay triangulation [13]:

$$\mathrm{Del}\check{C}ech_r(X) = \{Q \in \mathrm{Del}(X) \mid \bigcap_{x \in Q} B_r(x) \neq \emptyset\}.$$

The *Wrap complex* is defined using the gradient of the Delaunay radius function $s_X : \mathrm{Del}(X) \to \mathbb{R}$, see [13] for its technical definition.

5.2 Persistent Homology

In this section, we introduce the basics pertaining to the theory of persistence that are necessary in **Section** 5. The fundamental idea is to build a geometric representation of the input data in the form of a *filtration*, from which one computes invariants capturing the topological features at multiple scales. For the purpose of this survey, given a simplicial complex K, a filtration is an increasing sequence of simplicial complexes connected by inclusions,

$$\emptyset = K_0 \hookrightarrow K_1 \hookrightarrow \cdots \hookrightarrow K_N = K.$$

To extract the topological information from a filtration, persistent homology studies the p-dimensional homology groups H_p of the simplicial complexes in the filtration. For

our purposes, these are vector spaces that contain information about the topological features in a simplicial complex: connected components (for $p = 0$), tunnels (for $p = 1$), voids (for $p = 2$), and so on. By computing p-homology groups for all simplicial complexes in a filtration, one obtains a family of vector spaces connected by linear maps that satisfy a commutativity relation,

$$\emptyset = \mathsf{H}_p(K_0) \to \mathsf{H}_p(K_1) \to \cdots \to \mathsf{H}_p(K_N) = \mathsf{H}_p(K).$$

This structure is an example of a *persistence module*. Algebraically, a persistence module, denoted as (M, ϕ), is a functor $M : \mathbf{P} \to \mathbf{Vec}$, where $\mathbf{P}$ is a poset category (e.g., $\mathbf{P} = \mathbb{R}^n$), and $\mathbf{Vec}$ is the category of finite-dimensional vector space over a field [18]. The *transition map* between a pair of such vector spaces is a linear transformation $\phi_s^t : M_s \to M_t$ for $s \leq t \in \mathbf{P}$. When $\mathbf{P} = \mathbb{R}$, by the Structure Theorem [158, 35, 22], a persistence module is fully explained by a family of intervals called the *persistence barcode* of the module. An interval or *bar* in a barcode, denoted as (x, y), represents the birth time x and the death time y of a homological feature in the filtration. An equivalent representation of the barcode is given by the *persistence diagram*, a multiset of points in the extended real plane $(\mathbb{R} \cup \pm\infty)^2$. A point (x, y) in the persistence diagram corresponds to a bar in the barcode. The persistence diagram has been extensively investigated and successfully applied to study sensor networks [36], protein interactions [55, 81, 148], DNA structures [49], robot trajectories [110], and bipedal walks [143], to name a few.

The interleaving distance describes the proximity between two persistence modules. Two persistence modules, (M, ϕ) and (N, ψ), are ε-*interleaved* if there are families of functions $f_t : M_t \to N_{t+\varepsilon}$ and $g_t : N_t \to M_{t+\varepsilon}$ that commute with the internal transition maps, i.e., the following diagrams commute for any $s, t \in \mathbb{R}$ where $s \leq t$:

$$
\begin{array}{ccc}
M_s & \xrightarrow{\phi_{s,t}} & M_t \\
\downarrow{f_s} & & \downarrow{f_t} \\
N_{s+\varepsilon} & \xrightarrow{\psi_{s+\varepsilon,t+\varepsilon}} & N_{t+\varepsilon},
\end{array}
\qquad
\begin{array}{ccc}
N_s & \xrightarrow{\psi_{s,t}} & N_t \\
\downarrow{g_s} & & \downarrow{g_t} \\
M_{s+\varepsilon} & \xrightarrow{\phi_{s+\varepsilon,t+\varepsilon}} & M_{t+\varepsilon},
\end{array}
$$

$$
\begin{array}{ccc}
M_t & \xrightarrow{\phi_{t,t+2\varepsilon}} & M_{t+2\varepsilon}, \\
{}_{f_t}\searrow & \nearrow_{g_{t+\varepsilon}} & \\
& N_{t+\varepsilon} &
\end{array}
\qquad
\begin{array}{ccc}
N_t & \xrightarrow{\psi_{t,t+2\varepsilon}} & N_{t+2\varepsilon}. \\
{}_{g_t}\searrow & \nearrow_{f_{t+\varepsilon}} & \\
& M_{t+\varepsilon} &
\end{array}
$$

The interleaving distance between two persistence modules is

$$\inf\{\varepsilon \mid (M, \phi) \text{ and } (N, \psi) \text{ are } \varepsilon\text{-}interleaved\}.$$

Given two filtrations $\mathcal{K} := (K_t)$, $\mathcal{L} := (L_t)$, the persistence diagram of $\mathcal{K}$ is a c-approximation [125] to that of $\mathcal{L}$ if there is a bijection π between the diagrams such that the birth (resp. death) time of a point x in the persistence diagram of $\mathcal{K}$ and $\pi(x)$ in the persistence diagram of $\mathcal{L}$ differ by at most a factor of c.

5.3 Sparse Filtrations Approximating Persistent Homology

A significant issue in computational topology is that the number of simplices in a filtration may explode as the number of data points increases. For example, for large enough radii, both the Čech and the Vietoris–Rips complexes have k-skeletons (i.e., simplices up to dimension k) of size $O(n^{k+1})$ [125], and both complexes contain up to $2^n - 1$ (i.e., $O(2^n)$) simplices for n data points (see Table 1 in [106]). Therefore, many works focus on sparsifying the simplicial filtrations (i.e., removing points or simplices) while approximately preserving their homological information. We review representative works based on obtaining sparse filtrations [125, 25] and/or applying batch collapse [39].

Given n number of points, Sheehy [125] constructed a linear-size (i.e., $O(n)$) filtered simplicial complex whose persistence diagram approximates that of a Vietoris–Rips filtration. The key idea is the introduction of a *relaxed Vietoris–Rips filtration* based on the notion of a *relaxed distance* that is provably close to the input metric. This new relaxed distance adds a weight to each point, which in turn shrinks the metric ball centered at the point. If a ball is covered by nearby balls, its center point can be deleted without changing the topology of the input point set, resulting in a sparsified complex. Formally, given a finite subset P of a metric space (X, d) and a user-defined parameter $\varepsilon \leq \frac{1}{3}$, the algorithm assigns a deletion time t_x (based on a hierarchical net-tree construction, see [125] for details) to each point $x \in P$ and defines its weight w_x based on a scale parameter r:

$$w_x(r) := \begin{cases} 0 & \text{if } r \leq (1 - 2\varepsilon)t_x; \\ \frac{1}{2}(r - (1 - 2\varepsilon)t_x) & \text{if } (1 - 2\varepsilon)t_x < r < t_x; \\ \varepsilon r & \text{if } t_x \leq r. \end{cases}$$

Using these weights, the *relaxed distance* at scale r between two points $x, y \in P$ is

$$\hat{d}_r(x, y) := d(x, y) + w_x(r) + w_y(r).$$

The *relaxed Vietoris–Rips complex* is defined to be $\hat{\mathcal{R}}_r := \mathrm{VR}_r(P, \hat{d}_r)$, which gives rise to a filtration with varying r. Let $\mathcal{R}_r := \mathrm{VR}_r(P, d)$. The multiplicative interleaving is

$$\mathcal{R}_{\frac{r}{c}} \subseteq \hat{\mathcal{R}}_r \subseteq \mathcal{R}_r,$$

where $c = \frac{1}{1 - 2\varepsilon}$ (see Lemma 4.2. in [125]).

The algorithm then constructs a *sparse zigzag Vietoris–Rips filtration* with $O(n)$ total number of simplices, whose persistence diagram is identical to that of the relaxed Vietoris–Rips filtration. To obtain such a sparsified filtration, the algorithm deletes vertices and all the corresponding incident simplices from the relaxed Vietoris–Rips filtration, when they become unimportant at the scale being considered. The algorithm makes use of an *open net* $\mathcal{N}_r$ and a *closed net* $\overline{\mathcal{N}}_r$ at scale r, defined to be the points in P with deletion time above r,

$$\mathcal{N}_r := \{x \in P : t_x > r\}; \ \ \overline{\mathcal{N}}_r := \{x \in P : t_x \geq r\}.$$

The *sparse zigzag Vietoris–Rips complex* at scale r is the subcomplex of $\hat{\mathcal{R}}_r$ induced by the vertices of $\mathcal{N}_r$,

$$\mathcal{Q}_r := \{\sigma \in \hat{\mathcal{R}}_r : \sigma \subseteq \mathcal{N}_r\} = \mathrm{VR}_r(\mathcal{N}_r, \hat{d}_r).$$

Its closed version is defined similarly, $\overline{\mathcal{Q}}_r := \mathrm{VR}_r(\overline{\mathcal{N}}_r, \hat{d}_r)$.

The *sparse zigzag Vietoris–Rips filtration* is

$$\cdots \hookrightarrow \overline{\mathcal{Q}}_{r'} \hookleftarrow \mathcal{Q}_{r'} \hookrightarrow \overline{\mathcal{Q}}_r \hookleftarrow \cdots$$

This leads to a filtration without the zigzag, defined by the union $\mathcal{S}_r = \bigcup_{r' \leq r} \overline{\mathcal{Q}}_{r'}$, which gives rise to the so-called *sparse Vietoris–Rips filtration*, having the same persistence diagram as the zigzag one.

Sheehy provided theoretical guarantees that (i) the persistence diagram of a sparse Vietoris–Rips filtration is a multiplicative c-approximation (see **Section** 5.2 for the definition) of the standard one and (ii) the filtration has linear size [125].

Dey et al. [39] provided an efficient algorithm to compute persistence diagrams of filtrations where the simplicial complexes are connected by simplicial maps instead of inclusions (cf., **Section** 5.2). They applied their algorithm to approximate the persistence diagram of a Vietoris–Rips filtration via a sparsified filtration. The key idea is that a simplicial map can be decomposed into elementary inclusions and vertex collapses, and these atomic operations can be used to collapse input points in batches with an increasing radius, thereby controlling the size of the filtration during persistence computation.

Using a mapping cylinder construction, a simplicial map $f : K_s \to K_t$ can be simulated with a zigzag connected by inclusions $K_s \hookrightarrow \hat{K} \hookleftarrow K_t$, via a third, significantly larger simplicial complex $\hat{K}$. Dey et al. provided an improved construction that converts a zigzag module connected by simplicial maps into a zigzag module connected only by inclusions, where the intermediate modules are not as big as the naive construction of $\hat{K}$. They also showed that for a monotone sequence of simplicial maps, one can further improve the construction by using the notion of annotation [23], which are binary vectors assigned to simplices. This improved construction can be used to compute the persistence of the sparse zigzag Vietoris–Rips filtration of Sheehy [125], without straightening out the zigzag. Whereas the approach by Sheehy [125] allows points to be deleted with a weighting scheme, Dey et al. provided an alternative method of sparsifying a Vietoris–Rips filtration via a more aggressive subsampling by collapsing input points in batches, thus avoiding the weighting scheme completely. Their construction produces a sparsified filtration that is a $(3 \log(1 + \varepsilon)/2)$-approximation of the original filtration (see **Section** 5.2) and can be computed efficiently, where $0 < \varepsilon \leq 1$ is a user-defined parameter.

As a follow-up, Dey at al. [40] further provided a thorough computational comparison of the methods in [125] and [39]. They observed that, despite the linear size guarantee, the approach by Sheehy still ends up producing very large complexes due to the union step, and becomes unfeasible for high-dimensional data. On the

other hand, the batch collapse from Dey et al. [39] is more space efficient, but still becomes prohibitively large for high-dimensional data. Inspired by their experimental findings, Dey at al. [40] proposed a new algorithm called SimBa, which obtains sparsified Vietoris–Rips filtrations through *simplicial batch collapse*. Comparing with the approach in [39], the key difference is that, instead of collapsing vertices based on their pairwise distance, SimBa builds another filtration by collapsing w.r.t. a *set distance*. For two sets of points $A, B \subset P$, this set distance is defined as $d(A, B) = \min_{a \in B, \, b \in B} d(a, b)$. Although this may not seem a major modification, in practice, it ends up reporting significant improvements in memory and runtime.

We have discussed finding cores only of Vietoris–Rips filtrations as sparsified filtrations. Kerber and Sharathkumar [75] extended the above ideas to sparsify Čech filtrations in two constructions. The first construction yields, for a fixed homological dimension, an approximate Čech filtration of linear size—just like in the Vietoris–Rips case—which is $(1 + \varepsilon)$-interleaved with the original filtration. The key technical ingredient for this construction is a generalization to higher dimension of the *well-separated pair decomposition* (WSPD), resulting in a *well-separated simplicial decomposition* (WSSD). A WSSD decomposes a given point set X into $O(n/\varepsilon^d)$ tuples; each p-tuple can be viewed as p clusters of points satisfying that, if a ball contains at least one point of each cluster, a small expansion of the ball will contain all points [75]. In addition, for any p-simplex that can be formed with the points in S, there is a decomposition in $p + 1$ clusters of S where each cluster contains one vertex of the p-simplex. A drawback of this construction is that the constant in the size of the filtration depends exponentially on the dimension of the ambient space of the point cloud, whereas for the Vietoris–Rips construction, it depends on its doubling constant. Intuitively, the *doubling constant* of a metric space is the minimum number of metric balls of some radius needed to cover any ball of double that radius.

For the second construction, Kerber and Sharathkumar [75] generalized the Vietoris–Rips Lemma [45], stating that the Vietoris–Rips complex at scale r is contained in the Čech complex at scale $\sqrt{2}r$. This allows for the definition of a family of *completion complexes*, verifying that the Čech complex at scale r is contained in a completion complex at scale $(1 + \varepsilon)r$. These complexes are parametrized by an integer p and they are completely determined by their p-skeleton—of size at most $O(n^p)$—meaning that any higher-dimensional simplex is obtained combinatorially from the p-simplices. For a specific choice of p, one obtains a $(1 + \varepsilon)$-interleaving with the original Čech filtration; see [75, Theorem 4] for a precise statement. In this case, the construction does not depend on the dimension of the ambient space. For the proof of the approximation to the Čech complex, the key point is using coresets of the minimum enclosing ball of the point cloud. Here, the *coreset* of a point cloud is a subset from which one can obtain an approximation of the minimum enclosing ball.

Mémoli and Okutan [95] continued the above line of work and studied the simplification of *filtered simplicial complexes*. A filtered simplicial complex indexed over $I \subset \mathbb{R}$ is a family of simplicial complexes $(K_t)_{t \in I}$ such that K_s is contained in K_t ($K_s \hookrightarrow K_t$) for all $s \leq t$. It arises from the family of simplicial complexes that forms a filtration in persistent homology computation (see **Section** 5.2), such as the Vietoris–Rips or the Čech complexes. The authors aimed to simplify a filtered

simplicial complex (i.e., reducing the number of simplices) while retaining its persistent homology. A key idea is the introduction of *codensity*, defined on the vertices of a filtered simplicial complex, which quantifies the contribution of each vertex to the persistent homology (see [95] for its technical definition based on vertex quasi-distance). The authors showed that removing a vertex with zero codensity does not affect the homology. If one considers the Vietoris–Rips complex of a finite metric space, then the codensity of a point equals its distance to the nearest neighbour [95, Remark 4.1]. Similar to previous approaches [125, 25], the algorithm produces a simplified filtered complex by removing a vertex and all simplices containing it in an iterative way (i.e., based on codensity). The authors further introduced the notion of a *simple filtered simplicial complex*, which is the one whose vertices all have positive codensity. They showed the existence of a unique (up to isomorphism) simple filtered simplicial complex (referred to as the *core*) for any filtered simplicial complex such that their interleaving type distance is zero.

Another way to obtain sparsified filtrations is via sequences of simplicial collapses. Bauer and Edelsbrunner [13] proved that, for a fixed parameter r, there is a collapsing sequence between the Čech, Čech-Delaunay, Delaunay (alpha), and the Wrap complexes. Since these complexes are homotopy equivalent, we may consider a collapsed complex as the core of the original complex by preserving both homological and homotopic properties. In particular, they proved the following theorem.

Theorem 1 (Čech-Delaunay Collapsing Theorem, [13]) *Let X be a finite set of possibly weighted points in general position in $\mathbb{R}^n$. Then, for every $r \in \mathbb{R}$,*

$$\check{C}ech_r(X) \searrow \mathrm{Del}\check{C}ech_r(X) \searrow \mathrm{Del}_r(X) \searrow \mathrm{Wrap}_r(X).$$

This theorem implies that filtrations built from these four complexes have isomorphic persistent homology [13, Corollary 6.1].

5.4 Subsampling Methods Approximating Persistent Homology

We now discuss sparsification of simplicial complexes based on subsampling or bootstrapping. Cao and Monod [24] introduced a bootstrapping technique to estimate the persistence diagram of a large input point cloud, whose direct persistence computation would be otherwise intractable. The authors proposed the following algorithm: given a finite subset P of a metric space (X, d) with a large number of points $|P| = N$, the algorithm takes a number of subsamples of X, each with $n \ll N$ points, and computes the persistence diagram of the Vietoris-Rips filtration of each subsample. The authors showed that the mean of the persistence diagrams of these bootstrap subsamples is a good approximation of the persistence diagram of the large data. Here, the *mean* of persistence diagrams refers to the *mean persistence measure*, obtained by first turning the persistence diagram into a discrete measure, where each point is substituted by a Dirac measure weighted by the multiplicity of the point in the diagram [41, 42], and then taking the mean of the measures. Building on Divol

and Lacombe's work on the convergence of the sample mean to the population mean of persistence diagrams [42], Cao and Monod derived explicit expressions for the approximation error of the convergence of the mean persistence measure through a bias-variance decomposition, justifying their bootstrap approach.

In a similar vein, Gómez and Mémoli [59] studied Vietoris–Rips filtrations of subsamples. Given a compact metric space (X, d), for a fixed dimension p and an integer $n \geq 1$, they considered the persistence diagrams of p-dimensional Vietoris–Rips filtrations of all subsets of (X, d) with cardinality at most $n \in \mathbb{N}$. They worked with the notion of a *curvature set* of a compact metric space (X, d), first introduced by Gromov [61]. For a fixed $n \in \mathbb{N}$, the n-th curvature set of X is a collection of *all $n \times n$* submatrices of the distance matrix of points in X (with possible repetitions) [61]. Starting from (X, d), the algorithm takes subsamples of the distance matrix of X, and applies (Vietoris–Rips) persistent homology to each subsample, and aggregates the persistence diagrams of the subsamples by overlaying them into a single set of axes. The authors showed that the aggregated diagram is easy to compute, stable, and enjoys good discriminating power in classification tasks. Solomon et al. [128] also explored the idea of *distributed persistence*, and demonstrated that the collection of persistence diagrams of *many* small subsets of (X, d) serves as a better invariant than a single persistence diagram of the entire space (X, d).

Although these approaches do not give rise to a single sparsified filtration, we may argue that the collection of subsamples and their corresponding filtrations serve as a replacement of the original data filtration, and approximately preserve its persistence diagram. It remains open to develop concrete, geometric representations of such a collection.

5.5 Sparsification Methods Approximating Spectral Properties

As an extension of spectral graph sparsification by Spielman and Srivastava [130], Osting et al. [105] introduced a subsampling algorithm for the p-skeleton of a simplicial complex that approximately preserves its p-dimensional up Laplacian. It relies on the effective resistance of simplices, generalizing **Equation** (19) using incidence matrix between simplices of adjacent dimensions. Given a simplicial complex K, and a fixed dimension $p \leq \dim(K)$, the algorithm runs as follows. It fixes all the simplices in K up to dimension $p - 1$ and then samples a number q of p-simplices independently with replacement, according to a probability distribution defined by their effective resistance. That way, the algorithm produces a sparsified simplicial complex J. Fixing a parameter $1/\sqrt{n_{p-1}} < \varepsilon \leq 1$, where n_{p-1} is the number of $p - 1$ simplices in K, for a sufficiently large value of n_{p-1} and a precise value of q (see [105, Theorem 3.1] for the full statement), Osting et al. proved that with probability at least $1/2$, they have,

$$(1 - \varepsilon)\mathbf{x}^T L_K \mathbf{x} \leq \mathbf{x}^T L_J \mathbf{x} \leq (1 + \varepsilon)\mathbf{x}^T L_K \mathbf{x},$$

for all $\mathbf{x} \in \mathbb{R}^{n_{p-1}}$, where L_K and L_J are the up Laplacian matrices for the original and sparsified simplicial complexes.

5.6 Spectral Clustering of Simplicial Complexes Preserving Homology

Similar to the case of graphs (**Section** 3), certain clustering methods of simplicial complexes may be considered as core finding as they preserve geometric or topological information of the complexes. Inspired by spectral clustering of graphs, Ebli and Spreemann [43] introduced a spectral clustering of simplicial complexes. Given a p-dimensional simplicial complex K of *low homological complexity* (that is, $\beta_p(K) := \mathrm{rank}(\mathsf{H}_p(K)) \leq 10$), the algorithm produces a clustering of the simplices in K of a fixed degree. The key idea is defining a *harmonic embedding* h of K_p, the p-skeleton of K, that is,

$$\varphi : K_p \to \mathbb{R}^{\beta_p}, \varphi = \xi \circ j \circ i,$$

where $i : K_p \hookrightarrow \mathsf{C}_p(K)$ is the inclusion of p-skeleton K_p to the p-chain group $\mathsf{C}_p(K)$, $j : \mathsf{C}_p(K) \to \mathcal{H}_p(K)$ is the orthogonal projection onto the harmonic group $\mathcal{H}_p(K)$ (i.e., the kernel of the Hodge Laplacian isomorphic to the homology group of K, $\mathsf{H}_p(K)$), and $\xi : \mathcal{H}_p(K) \to \mathbb{R}^{\beta_p(K)}$ is a choice of basis. In practice, Ebli and Spreemann simply chose an orthonormal basis $h_1, \ldots, h_{\beta_p(K)}$ for $\mathcal{H}_p(K)$ and defined

$$\varphi(\sigma) = \left(\langle \sigma, h_1 \rangle_p, \ldots, \langle \sigma, h_{\beta_p(K)} \rangle_p \right).$$

In a lower-dimensional subspace of $\mathbb{R}^{\beta_p(K)}$, the algorithm performs a subspace clustering method (e.g., independent component analysis [65]) of $\mathrm{im}(\varphi)$ and treats the output as the clustering of the original simplices. However, there are no theoretical guarantees for the above algorithm. The authors experimentally demonstrated that the simplices assigned to each cluster tend to reflect the presence of homology generators in the chosen dimension. For instance, when clustering the edges, they observed that edge clusters tend to respect the independent loops of the underlying space. The authors considered their algorithm to be complementary to the one presented by Osting et al. [105].

6 Future Research Opportunities

6.1 Cores of Graphs

An aspect common to many methods using discrete curvatures to find graph cores through sampling is the lack of theoretical guarantees. Numerous experimental studies qualitatively show that these sampling techniques produce graphs similar to

the originals and retain key structures. However, rigorous quantitative descriptions are scarce.

In practice, the primary sampling technique for sparsifying graphs using Forman-Ricci curvature involves simple thresholding to retain a certain percentage of edges. Several methods have been implemented to determine such a threshold, but it remains unclear which approach is optimal. A first direction for future research could be comparing all the methods proposed and evaluating in which scenarios they are more useful. In addition, future research could explore whether different sampling techniques, perhaps using the empirical distributions of edge curvatures, result in graph cores with distinct properties or impact the preservation of original graph structures.

As already mentioned, the Ollivier-Ricci curvature is mainly used for clustering. One reason it works for a graph with positive curvature can be explained by the inequalities in **Equation** (15), which ensure that it bounds the spectral gap of the Laplacian. However, as reviewed in **Section** 3.2, the edges representing the bottlenecks between clusters are those with the lowest negative curvature. A question one may ask is if and how this negative curvature also relates to the spectral gap.

Samal et al. [122] performed a comparative analysis of two discrete Ricci curvatures, the Forman-Ricci curvature and the Ollivier-Ricci curvature. They empirically showed a correlation between these two measures, which increases using the augmented Forman-Ricci curvature. However, the theoretical reasons for such a correlation require further study, which may shed some light on extracting graph cores with discrete curvatures.

Finally, resistance curvature [37, 38], as a newer notion of discrete curvature for graphs, may be employed for finding graph cores, similar to the ways where Forman-Ricci the Ollivier-Ricci curvatures are employed.

6.2 Cores of Hypergraphs

We have primarily reviewed methods for finding cores of hypergraphs based on sparsification and percolation techniques. However, given that hypergraphs inherently possess a complex underlying geometry, an outstanding gap in this area is the utilization of geometric methods, such as curvature-based approaches (e.g., [83]), for identifying the cores of hypergraphs.

The notion of effective resistance was originally defined for graphs by Spielman and Srivastava [130], and then generalized to simplicial complexes by Osting et al. [105], and more recently to hypergraphs by Aghdaei and Feng [2]. There are two possible future venues. First, whereas effective resistance has been used as a sampling criterion for sparsifying graphs [130] and simplicial complexes [105], its generalization to the sparsification of hypergraphs remains underexplored. Second, it would be interesting to define resistance curvature for hypergraphs and obtain curvature-based hypergraph cores. To define such a notion, we would need to define effective resistance of hypergraphs. Despite recent progress [2], the question is: does

there exist a canonical notion of effective resistance for hypergraphs, defined using an effective resistance matrix? In other words, is there an effective resistance matrix R for hypergraph, in the form of $R := BL^+B^\top$, similar to **Equation** (19), where B is a boundary matrix for hypergraphs and L is a hypergraph Laplacian? Whereas multiple generalizations from graph Laplacian to hypergraph Laplacian have appeared in the literature [1, 64, 88, 120, 119], to the best of our knowledge, there does not appear to be a consensus as to the canonical notion of a hypergraph Laplacian. For instance, Rodríguez [113, 114] introduced a version of the Laplacian matrix of a hypergraph based on the *Laplacian degree* of a vertex and used it to obtain spectral-like results on partition problems in hypergraphs [115]. Agarwal et al. [1] defined higher order Laplacians as operators that measure variations on functions defined on p-chains formed by the vertex set. Hein et al. [64] made a connection between p-Laplacians of homogeneous hypergraphs with the total variation. Aktas and Akbas [4] introduced hypergraph Laplacians inspired by the simplicial Laplacian. It would be interesting to explore how various notions of resistance curvature (based on different Laplacians) affect the properties of hypergraph cores.

Another potential direction for future research is the sparsification of hypergraphs while preserving their homological properties. Given the existence of various homology theories for hypergraphs (see [57] for a survey), we anticipate the development of distinct frameworks tailored to preserving different notions of homology.

6.3 Cores of Simplicial Complexes

Within the area of finding the cores of simplicial complexes, there is still room for improvement in sparsification methods for filtrations approximately preserving the persistence diagram. Some ideas could be trying out different schemes for the batch collapse proposed by Dey et al. [39] or trying to introduce some probabilistic techniques, useful in the sparsification techniques preserving spectral properties [105]. In addition, there are other typical filtration constructions, known to have fewer simplices, that still pose computational constraints due to their size, such as filtrations based on the alpha complex. It could be interesting to see how the techniques described in **Section** 5.3 extend to this setting, which is already favorable for the purpose of reducing the size of the final complex.

On the other hand, concerning the method for sparsifying simplicial complexes preserving spectral properties described by Osting et al. [105], there are a few directions open for future research. First, from the condition obtained relating the Laplacians of the original and the sparsified complexes, it is not straightforward to see how the eigenvalues of the sparsified complex actually change. Finding a more direct relation in this context would be useful and illuminating. Additionally, the method allows sparsification only for simplices of dimension p, keeping the entire skeleton of the complex fixed up to dimension $p - 1$. A potential direction of future research, already mentioned in [105], would be developing a methodology to sparsify across

dimensions, without the need to fix the simplices in any dimension, and obtaining theoretical guarantees for it.

Finally, as already mentioned in **Section** 5.6, a clear open question regarding the harmonic clustering algorithm described in [43] involves the derivation of theoretical guarantees for this method.

7 Conclusion

In this paper, we survey geometric and topological methods to extract the cores of graphs, hypergraphs, and simplicial complexes. After studying the collection of recent works, we found that there are noticeable imbalances among core-finding methods for higher graphs, revealing research gaps and hence opportunities. While many core-finding methods for graphs focus on sampling/thresholding with discrete curvatures (**Table** 1), similar methods are underdeveloped for hypergraphs and simplicial complexes with theoretical guarantees. Whereas there are a few recent developments in percolation theory for hypergraphs (**Table** 2), there is plenty of room to grow. The percolation theory for simplicial complexes does not yet lead to the computation of well-defined cores. Sparse filtrations of simplicial complexes have been developed for preserving persistent homology (**Table** 3), so the question is, can one obtain sparsifications that preserve homological and spectral properties simultaneously? Spectral sparsification methods have been extended from graphs to hypergraphs and to simplicial complexes; however, the extracted cores may vary depending on different notions of Laplacians for higher graphs. We aim to use this survey (cf. **Tables** 1 to 3) to inspire new research on the simplification of higher graphs.

References

1. Sameer Agarwal, Kristin Branson, and Serge Belongie. Higher order learning with graphs. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 17–24. ACM Press, 2006.
2. Ali Aghdaei and Zhuo Feng. HyperEF: Spectral hypergraph coarsening by effective-resistance clustering. In *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design*, pages 1–9, San Diego California, October 2022. ACM.
3. Adel Ahmed, Vladimir Batagelj, Xiaoyan Fu, Seok-Hee Hong, Damian Merrick, and Andrej Mrvar. Visualisation and analysis of the Internet Movie Database. In *2007 6th International Asia-Pacific Symposium on Visualization*, pages 17–24. IEEE, 2007.
4. Mehmet Emin Aktas and Esra Akbas. Hypergraph Laplacians in diffusion framework. In Rosa Maria Benito, Chantal Cherifi, Hocine Cherifi, Esteban Moro, Luis M. Rocha, and Marta Sales-Pardo, editors, *Complex Networks & Their Applications X*, pages 277–288, Cham, 2022. Springer International Publishing.
5. Paul M. Alsing, Jonathan R. McDonald, and Warner A. Miller. The simplicial Ricci tensor. *Classical and Quantum Gravity*, 28(15):155007, 2011.
6. Shahab Asoodeh, Tingran Gao, and James Evans. Curvature of hypergraphs via multi-marginal optimal transport. In *2018 IEEE Conference on Decision and Control*, pages 1180–1185. IEEE, 2018.
7. Shuliang Bai, An Huang, Linyuan Lu, and Shing-Tung Yau. On the sum of Ricci-curvatures for weighted graphs. *Pure and Applied Mathematics Quarterly*, 2020.
8. Anirban Banerjee. On the spectrum of hypergraphs. *Linear Algebra and its Applications*, 614:82–110, April 2021.
9. Nikhil Bansal, Ola Svensson, and Luca Trevisan. New notions and constructions of sparsification for graphs and hypergraphs. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science*, pages 910–928. IEEE, 2019.
10. Vladislav Barkanass, Jürgen Jost, and Emil Saucan. Geometric sampling of networks. *Journal of Complex Networks*, 10(4):cnac014, August 2022.
11. Joshua Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-Ramanujan sparsifiers. *SIAM Journal on Computing*, 41(6):1704–1721, 2012.
12. Frank Bauer, Jürgen Jost, and Shiping Liu. Ollivier–Ricci curvature and the spectrum of the normalized graph Laplace operator. *Mathematical Research Letters*, 19(6):1185–1205, 2012.
13. Ulrich Bauer and Herbert Edelsbrunner. The Morse theory of Čech and Delaunay complexes. *Transactions of the American Mathematical Society*, 369(5):3741–3762, 2017.
14. András A. Benczúr and David R. Karger. Approximating $s - t$ minimum cuts in $\tilde{O}(n^2)$ time. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, pages 47–55, 1996.
15. Austin R. Benson, David F. Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.

16. Ginestra Bianconi and Sergey N Dorogovtsev. Nature of hypergraph k-core percolation problems. *Physical Review E*, 109(1):014307, 2024.
17. Ginestra Bianconi and Sergey N Dorogovtsev. Theory of percolation on hypergraphs. *Physical Review E*, 109(1):014306, 2024.
18. Håvard Bakke Bjerkevik. On the stability of interval decomposable persistence modules. *Discrete & Computational Geometry*, 66(1):92–121, 2021.
19. Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.
20. Omer Bobrowski and Primoz Skraba. Homological percolation and the Euler characteristic. *Physical Review E*, 101(3), 2020.
21. Anca-Iuliana Bonciocat and Karl-Theodor Sturm. Mass transportation and rough curvature bounds for discrete spaces. *Journal of Functional Analysis*, 256(9):2944–2966, 2009.
22. Magnus Botnan and William Crawley-Boevey. Decomposition of persistence modules. *Proceedings of the American Mathematical Society*, 148(11):4581–4596, August 2020.
23. Oleksiy Busaryev, Sergio Cabello, Chao Chen, Tamal K. Dey, and Yusu Wang. Annotating simplices with a homology basis and its applications. In Fedor V. Fomin and Petteri Kaski, editors, *Algorithm Theory – SWAT 2012*, pages 189–200, Berlin, Heidelberg, 2012. Springer.
24. Yueqi Cao and Anthea Monod. Approximating persistent homology for large datasets. arXiv preprint arXiv:2204.09155, May 2022.
25. Nicholas J. Cavanna, Mahmoodreza Jahanseir, and Donald R. Sheehy. A geometric perspective on sparse filtrations. *Proceedings of the Canadian Conference on Computational Geometry*, 2015.
26. T-H Hubert Chan, Anand Louis, Zhihao Gavin Tang, and Chenzi Zhang. Spectral properties of hypergraph Laplacian and approximation algorithms. *Journal of the ACM*, 65(3):1–48, 2018.
27. Chandra Chekuri and Chao Xu. Minimum cuts and sparsification in hypergraphs. *SIAM Journal on Computing*, 47(6):2118–2156, 2018.
28. Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *International Conference on Learning Representations*, 2018.
29. Jie Chen, Yousef Saad, and Zechen Zhang. Graph coarsening: from scientific computing to machine learning. *SeMA Journal. Boletin de la Sociedad Española de Matemática Aplicada*, 79(1):187–223, 2022.
30. Yu Chen, Sanjeev Khanna, and Ansh Nagda. Near-linear size hypergraph cut sparsifiers. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science*, pages 61–72, 2020.
31. Yuhan Chen, Haojie Ye, Sanketh Vedula, Alex Bronstein, Ronald Dreslinski, Trevor Mudge, and Nishil Talati. Demystifying graph sparsification algorithms in graph properties preservation. *Proceedings of the VLDB Endowment*, 17(3):427–440, 2023.
32. I (Eli) Chien, Chung-Yi Lin, and I-Hsiang Wang. Community detection in hypergraphs: Optimal statistical limit and efficient algorithms. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 871–879. PMLR, 2018.
33. Bennett Chow and Feng Luo. Combinatorial Ricci flows on surfaces. *Journal of Differential Geometry*, 63(1):97–129, 2003.
34. Corinna Coupette, Sebastian Dalleiger, and Bastian Rieck. Ollivier-Ricci curvature for hypergraphs: A unified framework. In *The 11th International Conference on Learning Representations*, 2023.
35. William Crawley-Boevey. Decomposition of pointwise finite-dimensional persistence modules. *Journal of Algebra and Its Applications*, 14(05):1550066, June 2015. Publisher: World Scientific Publishing Co.
36. Vin de Silva and Robert Ghrist. Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology*, 7(1):339–358, April 2007.
37. Karel Devriendt and Renaud Lambiotte. Discrete curvature on graphs from the effective resistance. *Journal of Physics: Complexity*, 3(2), 2022.

38. Karel Devriendt, Andrea Ottolini, and Stefan Steinerberger. Graph curvature via resistance distance. *Discrete Applied Mathematics*, 348:68–78, 2024.

39. Tamal K. Dey, Fengtao Fan, and Yusu Wang. Computing topological persistence for simplicial maps. In *Proceedings of the 30th Annual Symposium on Computational Geometry*, pages 345–354. ACM, 2014.

40. Tamal K. Dey, Dayu Shi, and Yusu Wang. SimBa: An efficient tool for approximating Rips-filtration persistence via Simplicial Batch Collapse. *ACM Journal of Experimental Algorithmics*, 24:1–16, 2019.

41. Vincent Divol and Frédéric Chazal. The density of expected persistence diagrams and its kernel based estimation. *Journal of Computational Geometry*, 10(2):127–153, 2019.

42. Vincent Divol and Théo Lacombe. Estimation and quantization of expected persistence diagrams. In *International Conference on Machine Learning*, pages 2760–2770. PMLR, 2021.

43. Stefania Ebli and Gard Spreemann. A notion of harmonic clustering in simplicial complexes. In *2019 18th IEEE International Conference On Machine Learning And Applications*, pages 1083–1090, December 2019.

44. Herbert Edelsbrunner and John Harer. Persistent homology-a survey. *Contemporary Mathematics*, 453:257–282, 2008.

45. Herbert Edelsbrunner and John L Harer. *Computational topology: an introduction*. American Mathematical Society, 2022.

46. Marzieh Eidi, Amirhossein Farzam, Wilmer Leal, Areejit Samal, and Jürgen Jost. Edge-based analysis of networks: curvatures of graphs and hypergraphs. *Theory in Biosciences*, 139(4):337–348, December 2020.

47. Marzieh Eidi and Jürgen Jost. Ollivier Ricci curvature of directed hypergraphs. *Scientific Reports*, 10(1):12466, 2020.

48. Marzieh Eidi and Sayan Mukherjee. Irreducibility of Markov chains on simplicial complexes, the spectrum of the discrete Hodge Laplacian and homology. arXiv preprint arXiv:2310.07912, 2023.

49. Kevin Emmett, Benjamin Schweinhart, and Raul Rabadan. Multiscale topology of chromatin folding. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS)*, BICT'15, pages 177–180, Brussels, BEL, May 2016. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

50. Song Feng, Emily Heath, Brett Jefferson, Cliff Joslyn, Henry Kvinge, Hugh D. Mitchell, Brenda Praggastis, Amie J. Eisfeld, Amy C. Sims, Larissa B. Thackray, Shufang Fan, Kevin B. Walters, Peter J. Halfmann, Danielle Westhoff-Smith, Qing Tan, Vineet D. Menachery, Timothy P. Sheahan, Adam S. Cockrell, Jacob F. Kocher, Kelly G. Stratton, Natalie C. Heller, Lisa M. Bramer, Michael S. Diamond, Ralph S. Baric, Katrina M. Waters, Yoshihiro Kawaoka, Jason E. McDermott, and Emilie Purvine. Hypergraph models of biological networks to identify genes critical to pathogenic viral response. *BMC Bioinformatics*, 22(1), May 2021.

51. Lukas Fesser, Sergio Serrano de Haro Iváñez, Karel Devriendt, Melanie Weber, and Renaud Lambiotte. Augmentations of Forman's Ricci curvature and their applications in community detection. *Journal of Physics: Complexity*, 5(3):035010, 2024.

52. Robin Forman. Bochner's method for cell complexes and combinatorial Ricci curvature. *Discrete & Computational Geometry*, 29:323–374, 2003.

53. Luca Franceschi, Matthias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, pages 1972–1982, 2019.

54. Giorgio Gallo, Giustino Longo, Stefano Pallottino, and Sang Nguyen. Directed hypergraphs and applications. *Discrete Applied Mathematics*, 42(2-3):177–201, 1993.

55. Marcio Gameiro, Yasuaki Hiraoka, Shunsuke Izumi, Miroslav Kramár, Konstantin Mischaikow, and Vidit Nanda. A topological measurement of protein compressibility. *Japan Journal of Industrial and Applied Mathematics*, 32:1–17, March 2014.

56. Xinyi Gao, Junliang Yu, Tong Chen, Guanhua Ye, Wentao Zhang, and Hongzhi Yin. Graph condensation: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 37(4):1819–1837, 2025.

57. Ellen Gasparovic, Emilie Purvine, Radmila Sazdanovic, Bei Wang, Yusu Wang, and Lori Ziegelmeier. A survey of simplicial, relative, and chain complex homology theories for hypergraphs. arXiv preprint arXiv:2409.18310, 2024.

58. Anastasia Golovin, Jan Mölter, and Christian Kuehn. Polyadic opinion formation: The adaptive voter model on a hypergraph. *Annalen der Physik*, 2024.

59. Mario Gómez and Facundo Mémoli. Curvature sets over persistence diagrams. *Discrete & Computational Geometry*, 72:91–180, 2024.

60. Adam Gosztolai and Alexis Arnaudon. Unfolding the multiscale structure of networks with dynamical Ollivier-Ricci curvature. *Nature Communications*, 12(1):4561, 2021.

61. Misha Gromov. *Metric structures for Riemannian and non–Riemannian spaces*. Modern Birkhäuser Classics. Birkhäuser Boston, Inc., Boston, MA, english edition, 2007. Based on the 1981 French original, With appendices by M. Katz, P. Pansu and S. Semmes, Translated from the French by Sean Michael Bates.

62. David Xianfeng Gu and Emil Saucan. Metric Ricci curvature for manifolds. *Geometry*, 2013, 2013.

63. Mohammad Hashemi, Shengbo Gong, Juntong Ni, Wenqi Fan, B. Aditya Prakash, and Wei Jin. A comprehensive survey on graph reduction: sparsification, coarsening, and condensation. In *Proceedings of the 33rd International Joint Conference on Artificial Intelligence*, 2024.

64. Matthias Hein, Simon Setzer, Leonardo Jost, and Syama Sundar Rangapuram. The total variation on hypergraphs - learning on hypergraphs revisited. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, pages 2427–2435, Red Hook, NY, USA, 2013. Curran Associates Inc.

65. Aapo Hyvärinen and Erkki Oja. Independent component analysis: Algorithms and applications. *Neural Networks*, 13(4):411–430, June 2000.

66. Roberto Interdonato, Matteo Magnani, Diego Perna, Andrea Tagarelli, and Davide Vega. Multilayer network simplification: Approaches, models and methods. *Computer Science Review*, 36:100246, 2020.

67. Sergio Serrano de Haro Ivánez. Comparative analysis of Forman-Ricci curvature versions applied to the persistent homology of networks. *arXiv preprint arXiv:2212.01357*, 2022.

68. Arun Jambulapati, Yang P. Liu, and Aaron Sidford. Chaining, group leverage score overestimates, and fast spectral hypergraph sparsification. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 196–206, New York, NY, USA, 2023. Association for Computing Machinery.

69. Miao Jin, Junho Kim, and Xianfeng David Gu. Discrete surface Ricci flow: Theory and applications. In *Mathematics of Surfaces XII: 12th IMA International Conference, Sheffield, UK, September 4-6, 2007. Proceedings 12*, pages 209–232. Springer, 2007.

70. Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. Graph structure learning for robust graph neural networks. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 66–74, 2020.

71. Wei Jin, Lingxiao Zhao, Shichang Zhang, Yozen Liu, Jiliang Tang, and Neil Shah. Graph condensation for graph neural networks. In *International Conference on Learning Representations*, 2022.

72. Jürgen Jost and Florentin Münch. Characterizations of Forman curvature. *arXiv preprint arXiv:2110.04554*, 2021.

73. Michael Kapralov, Robert Krauthgamer, Jakab Tardos, and Yuichi Yoshida. Towards tight bounds for spectral sparsification of hypergraphs. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 598–611, 2021.

74. George Karypis and Vipin Kumar. A hypergraph partitioning package. *Army HPC Research Center, Department of Computer Science & Engineering, University of Minnesota*, 2:1–20, 1998.

75. Michael Kerber and R. Sharathkumar. Approximate Čech complex in low and high dimensions. In Leizhen Cai, Siu-Wing Cheng, and Tak-Wah Lam, editors, *Algorithms and Computation*, volume 8283, pages 666–676. Springer Berlin Heidelberg, 2013.

76. Jeongseon Kim, Soohwan Jeong, and Sungsu Lim. Link pruning for community detection in social networks. *Applied Sciences*, 12(13):6811, 2022.

77. Douglas J. Klein and Milan Randic. Resistance distance. *Journal of Mathematical Chemistry*, 12:81–95, 1993.

78. Regina Klimmek and Frank Wagner. A simple hypergraph min cut algorithm. *Freie Universität Berlin, Fachbereich Mathematik und Informatik; Serie B, Informatik*, 1996.

79. Dmitry Kogan and Robert Krauthgamer. Sketching cuts in graphs and hypergraphs. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 367–376, 2015.

80. Yi-Xiu Kong, Gui-Yuan Shi, Rui-Jie Wu, and Yi-Cheng Zhang. k-core: Theories and applications. *Physics Reports*, 832:1–32, 2019. k-core: Theories and Applications.

81. Violeta Kovacev-Nikolic, Peter Bubenik, Dragan Nikolić, and Giseon Heo. Using persistent homology and dynamical distances to analyze protein binding. *Statistical Applications in Genetics and Molecular Biology*, 15(1), January 2016.

82. Wilmer Leal, Marzieh Eidi, and Jürgen Jost. Ricci curvature of random and empirical directed hypernetworks. *Applied network science*, 5:1–14, 2020.

83. Wilmer Leal, Guillermo Restrepo, Peter F. Stadler, and Jürgen Jost. Forman–Ricci curvature for hypergraphs. *Advances in Complex Systems*, 24(01):2150003, February 2021. Publisher: World Scientific Publishing Co.

84. Deokjae Lee, B. Kahng, Y. S. Cho, K.-I. Goh, and D.-S. Lee. Recent advances of percolation theory in complex networks. *Journal of the Korean Physical Society*, 73:152–164, 2018.

85. James R. Lee. Spectral hypergraph sparsification via chaining. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 207–218, New York, NY, USA, 2023. Association for Computing Machinery.

86. Jongshin Lee, Kwang-Il Goh, Deok-Sun Lee, and B. Kahng. (k, q)-core decomposition of hypergraphs. *Chaos, Solitons & Fractals*, 173:113645, 2023.

87. Ming Li, Run-Ran Liu, Linyuan Lü, Mao-Bin Hu, Shuqi Xu, and Yi-Cheng Zhang. Percolation on complex networks: Theory and application. *Physics Reports*, 907:1–68, 2021.

88. Pan Li and Olgica Milenkovic. Submodular hypergraphs: p-Laplacians, Cheeger inequalities and spectral clustering. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 3014–3023, 2018.

89. Yong Lin, Linyuan Lu, and Shing-Tung Yau. Ricci curvature of graphs. *Tohoku Mathematical Journal*, 63(4):605 – 627, 2011.

90. Yike Liu, Tara Safavi, Abhilash Dighe, and Danai Koutra. Graph summarization methods and applications: A survey. *ACM Computing Surveys*, 51(3):1–34, 2018.

91. John Lott and Cédric Villani. Ricci curvature for metric-measure spaces via optimal transport. *Annals of Mathematics*, pages 903–991, 2009.

92. Anand Louis. Hypergraph Markov operators, eigenvalues and approximation algorithms. In *Proceedings of the forty-seventh annual ACM symposium on theory of computing*, pages 713–722, 2015.

93. Wai-Kei Mak and DF Wong. A fast hypergraph min-cut algorithm for circuit partitioning. *Integration*, 30(1):1–11, 2000.

94. Marina Meilă and Jianbo Shi. A random walks view of spectral segmentation. In *International Workshop on Artificial Intelligence and Statistics*, pages 203–208. PMLR, January 2001.

95. Facundo Mémoli and Osman Berat Okutan. Quantitative simplification of filtered simplicial complexes. *Discrete & Computational Geometry*, 65(2):554–583, 2021.

96. Russell Merris. Laplacian matrices of graphs: A survey. *Linear Algebra and its Applications*, 197–198:143–176, 1994.

97. Frank Morgan. Manifolds with density. *Notices of the AMS*, 52(8):853–858, 2005.

98. Florentin Münch and Radoslaw K. Wojciechowski. Ollivier Ricci curvature for general graph Laplacians: Heat equation, Laplacian comparison, non-explosion and diameter bounds. *Advances in Mathematics*, 356:106759, November 2019.

99. Hiroshi Nagamochi and Toshihide Ibaraki. A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. *Algorithmica*, 7(1):583–596, 1992.

100. Mark Newman. *Networks*. Oxford University Press, 2018.

101. Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in Neural Information Processing Systems*, 14, 2001.

102. Chien-Chun Ni, Yu-Yao Lin, Feng Luo, and Jie Gao. Community detection on networks with Ricci flow. *Scientific Reports*, 9(1):9984, July 2019.
103. Yann Ollivier. Ricci curvature of metric spaces. *Comptes Rendus Mathématique. Académie des Sciences. Paris*, 345(11):643–646, 2007.
104. Yann Ollivier. Ricci curvature of Markov chains on metric spaces. *Journal of Functional Analysis*, 256(3):810–864, 2009.
105. Braxton Osting, Sourabh Palande, and Bei Wang. Spectral sparsification of simplicial complexes for clustering and label propagation. *Journal of Computational Geometry*, 11(1):176–211, 2020.
106. Nina Otter, Mason A Porter, Ulrike Tillmann, Peter Grindrod, and Heather A Harrington. A roadmap for the computation of persistent homology. *EPJ Data Science*, 6:1–38, 2017.
107. Yun Jin Park and Didong Li. Lower Ricci curvature for efficient community detection. arXiv preprint arXiv:2401.10124, 2024.
108. Hongwu Peng, Deniz Gurevin, Shaoyi Huang, Tong Geng, Weiwen Jiang, Orner Khan, and Caiwen Ding. Towards sparsification of graph neural networks. In *IEEE 40th International Conference on Computer Design (ICCD)*, pages 272–279, 2022.
109. Peter Petersen. *Riemannian Geometry*, volume 171 of *Graduate Texts in Mathematics*. Springer International Publishing, Cham, 2016.
110. Florian T. Pokorny, Majd Hawasly, and Subramanian Ramamoorthy. Topological trajectory classification with filtrations of simplicial complexes and persistent homology. *The International Journal of Robotics Research*, 35(1-3):204–223, January 2016.
111. Liudmila Prokhorenkova, Egor Samosvat, and Pim van der Hoorn. Global graph curvature. In Bogumił Kamiński, Pawel Pralat, and Przemyslaw Szufel, editors, *Algorithms and Models for the Web Graph*, pages 16–35, Cham, 2020. Springer International Publishing.
112. Maurice Queyranne. Minimizing symmetric submodular functions. *Mathematical programming*, 82:3–12, 1998.
113. J.A. Rodríguez. On the Laplacian eigenvalues and metric parameters of hypergraphs. *Linear and Multilinear Algebra*, 50(1):1–14, 2002.
114. J.A. Rodríguez. On the Laplacian spectrum and walk-regular hypergraphs. *Linear and Multilinear Algebra*, 51(3):285–297, 2003.
115. J.A. Rodríguez. Laplacian eigenvalues and partition problems in hypergraphs. *Applied Mathematics Letters*, 22(6):916–921, 2009.
116. Nicolò Ruggeri, Martina Contisciani, Federico Battiston, and Caterina De Bacco. Community detection in large hypergraphs. *Science Advances*, 9(28):eadg9159, 2023.
117. Abbas Ali Saberi. Recent advances in percolation theory and its applications. *Physics Reports*, 578:1–32, 2015. Recent advances in percolation theory and its applications.
118. Muhammad Sahimi. *Applications of Percolation Theory*. Springer Cham, 2023.
119. Shota Saito and Mark Herbster. Generalizing p-Laplacian: Spectral hypergraph theory and a partitioning algorithm. *Machine Learning*, 112(1):241–280, 2023.
120. Shota Saito, Danilo Mandic, and Hideyuki Suzuki. Hypergraph p-Laplacian: A differential geometry view. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018.
121. Loqman Salamatian, Scott Anderson, Joshua Matthews, Paul Barford, Walter Willinger, and Mark Crovella. Curvature-based analysis of network connectivity in private backbone infrastructures. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 6(1):5:1–5:32, February 2022.
122. Areejit Samal, RP Sreejith, Jiao Gu, Shiping Liu, Emil Saucan, and Jürgen Jost. Comparative analysis of two discretizations of Ricci curvature for complex networks. *Scientific Reports*, 8(1):8650, 2018.
123. Emil Saucan, R. P. Sreejith, R. P. Vivek-Ananth, Jürgen Jost, and Areejit Samal. Discrete Ricci curvatures for directed networks. *Chaos, Solitons & Fractals*, 118:347–360, January 2019.
124. Stephen B. Seidman. Network structure and minimum degree. *Social Networks*, 5(3):269–287, 1983.
125. Donald R. Sheehy. Linear-size approximations to the Vietoris–Rips filtration. In *Proceedings of the 28th Annual Symposium on Computational Geometry*, pages 239–248. ACM, 2012.

126. Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 2000.
127. Jayson Sia, Edmond Jonckheere, and Paul Bogdan. Ollivier-Ricci curvature-based method to community detection in complex networks. *Scientific Reports*, 9(1):9800, July 2019.
128. Elchanan Solomon, Alexander Wagner, and Paul Bendich. From geometry to topology: Inverse theorems for distributed persistence. In Xavier Goaac and Michael Kerber, editors, *38th International Symposium on Computational Geometry*, volume 224 of *Leibniz International Proceedings in Informatics*, pages 61:1–61:16, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.
129. Tasuku Soma and Yuichi Yoshida. Spectral sparsification of hypergraphs. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2570–2581, 2019.
130. Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926, 2011.
131. Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 81–90, 2004.
132. David I. Spivak. Higher-dimensional models of networks. arXiv preprint arXiv:0909.4314, 2009.
133. R. P. Sreejith, Jürgen Jost, Emil Saucan, and Areejit Samal. Forman curvature for directed networks. arXiv preprint arXiv:1605.04662, February 2017.
134. RP Sreejith, Karthikeyan Mohanraj, Jürgen Jost, Emil Saucan, and Areejit Samal. Forman curvature for complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(6):063206, 2016.
135. David A Stone. A combinatorial analogue of a theorem of Myers. *Illinois Journal of Mathematics*, 20(1):12–21, 1976.
136. Hanlin Sun and Ginestra Bianconi. Higher-order percolation processes on multiplex hypergraphs. *Physical Review E*, 104(3):034306, 2021.
137. Michel Talagrand. *Upper and lower bounds for stochastic processes*, volume 60. Springer Berlin, Heidelberg, 2014.
138. Philip Tee and Carlo A Trugenberger. Enhanced Forman curvature and its relation to Ollivier curvature. *Europhysics Letters*, 133(6):60006, 2021.
139. Yu Tian, Zachary Lubberts, and Melanie Weber. Mixed-membership community detection via line graph curvature. In *Proceedings of the 1st NeurIPS Workshop on Symmetry and Geometry in Neural Representations*, pages 219–233. PMLR, February 2023.
140. Yu Tian, Zachary Lubberts, and Melanie Weber. Mixed-membership community detection via line graph curvature. In *NeurIPS Workshop on Symmetry and Geometry in Neural Representations*, pages 219–233. PMLR, 2023.
141. Yu Tian, Zachary Lubberts, and Melanie Weber. Curvature-based clustering on graphs. *Journal of Machine Learning Research*, 26(52):1–67, 2025.
142. Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.
143. Ramanarayan Vasudevan, Aaron Ames, and Ruzena Bajcsy. Persistent homology for automatic determination of human-data based cost of bipedal walking. *Nonlinear Analysis: Hybrid Systems*, 7(1):101–115, February 2013.
144. Ulrike Von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, December 2007.
145. Melanie Weber, Jurgen Jost, and Emil Saucan. Detecting the coarse geometry of networks. In *32nd Conference on Neural Information Processing Systems Workshop on Relational Representation Learning*, 2018.
146. Xiaoqi Wei and Guo-Wei Wei. Persistent topological laplacians—a survey. *Mathematics*, 13(2), 2025.
147. Shushan Wu, Huimin Cheng, Jiazhang Cai, Ping Ma, and Wenxuan Zhong. Subsampling in large graphs using Ricci curvature. In *The 11th International Conference on Learning Representations*, 2023.

148. Kelin Xia, Zhiming Li, and Lin Mu. Multiscale persistent functions for biomolecular structure characterization. *Bulletin of Mathematical Biology*, 80(1):1–31, January 2018.

149. Taiki Yamada. The Ricci curvature on simplicial complexes. *Theory and Applications of Graphs*, 10(2):1–18, 2023.

150. Yiyang Yang, Sucheng Deng, Juan Lu, Yuhong Li, Zhiguo Gong, Leong Hou U, and Zhifeng Hao. GraphLSHC: Towards large scale spectral hypergraph clustering. *Information Sciences*, 544:117–134, January 2021.

151. Yuichi Yoshida. Cheeger inequalities for submodular transformations. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2582–2601. SIAM, 2019.

152. Xikun Zhang, Dongjin Song, and Dacheng Tao. Ricci curvature-based graph sparsification for continual graph representation learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2024.

153. Dandan Zhao, Runchao Li, Hao Peng, Ming Zhong, and Wei Wang. Higher-order percolation in simplicial complexes. *Chaos, Solitons & Fractals*, 155:111701, 2022.

154. Dandan Zhao, Runchao Li, Hao Peng, Ming Zhong, and Wei Wang. Percolation on simplicial complexes. *Applied Mathematics and Computation*, 431:127330, 2022.

155. Yaoming Zhen and Junhui Wang. Community detection in general hypergraph via graph embedding. *Journal of the American Statistical Association*, 118(543):1620–1629, 2023.

156. Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

157. Yanqiao Zhu, Weizhi Xu, Jinghao Zhang, Yuanqi Du, Jieyu Zhang, Qiang Liu, Carl Yang, and Shu Wu. A survey on graph structure learning: Progress and opportunities. arXiv preprint arXiv:2103.03036, 2022.

158. Afra Zomorodian and Gunnar Carlsson. Computing persistent homology. *Discrete & Computational Geometry*, 33(2):249–274, 2005.