Dist2Cycle: A Simplicial Neural Network for Homology Localization

Alexandros D. Keros[1], **Vidit Nanda**[2], **Kartic Subr**[1]

[1]The University of Edinburgh
[2]University of Oxford
a.d.keros@sms.ed.ac.uk, nanda@maths.ox.ac.uk, ksubr@ed.ac.uk

Abstract

Simplicial complexes can be viewed as high dimensional generalizations of graphs that explicitly encode multi-way ordered relations between vertices at different resolutions, all at once. This concept is central towards detection of higher dimensional topological features of data, features to which graphs, encoding only pairwise relationships, remain oblivious. While attempts have been made to extend Graph Neural Networks (GNNs) to a simplicial complex setting, the methods do not inherently exploit, or reason about, the underlying topological structure of the network. We propose a graph convolutional model for learning functions parametrized by the k-homological features of simplicial complexes. By spectrally manipulating their combinatorial k-dimensional Hodge Laplacians, the proposed model enables learning topological features of the underlying simplicial complexes, specifically, the distance of each k-simplex from the nearest "optimal" k-th homology generator, effectively providing an alternative to homology localization.

Introduction

Tremendous advancements in sensor technology and data-driven machine learning have enabled exciting applications such as automatic health monitoring and autonomous cars. In many cases, the lack of data in certain regions of the domain reveals important structure. For instance, the sensors on a car driving through a parking lot might have dense observation points in 3D except inside pillars. Such *voids* in the data are ubiquitous across applications whether it is a subspace of unattainable configurations for a robot (Farber 2018), regions without network coverage (Ghrist and Muhammad 2005) or missing measurements in an experimentally-determined chemical structure (Townsend et al. 2020), to name a few (Aktas, Akbas, and El Fatmaoui 2019).

A standard approach to extract structural information from data proceeds by first encoding pairwise relationships in a problem via a graph and then analysing its properties. Recent advances in Graph Neural Networks have enabled practical learning in the domain of graphs and have provided approximate solutions to difficult graph problems (Hamilton, Ying, and Leskovec 2017). Despite the wealth of techniques underpinned by solid graph theory, this approach is fundamentally misaligned with problems where the relationships involve multiple points, and topological & geometric structure must be encoded beyond pairwise interactions.

Fortunately, higher dimensional combinatorial structures come to the rescue in the form of simplicial complexes, the powerhorse of topological data analysis (Chazal and Michel 2017). Interfacing between combinatorics and geometry, simplicial complexes capture multi-scale relationships and facilitate the passage from local structure to global invariant features. These features occur in the form of homology groups, intuitively perceived as *holes*, or *voids*, in any desired dimension. Alas, this expressive power comes with a burden, that of high computational complexity, and difficulty in localization of said voids (Chen and Freedman 2011).

For every hard computational problem there seems to exist a neural network approximation (Xu et al. 2018). Nevertheless, homology and simplicial complexes have only recently started to follow suit (Bodnar et al. 2021b; Ebli, Defferrard, and Spreemann 2020; Bunch et al. 2020), with inference *of* homological information still lacking.

The key insight in this paper is to guide learning on simplicial complexes by flipping the conventional view of approximation. We propose a GNN model for localizing homological information in the form of a distance function of each point of a complex to its the nearest homology generating features, a bird's-eye view of which is illustrated in Figure 1. Instead of using the most-significant eigenvectors of the relevant Laplacian operator we focus on the subspace spanned by the eigenvectors corresponding to the lowest eigenvalues. The justification is that homology-related information is contained in its nullspace. We implement this idea by calculating the most-significant subspace of an inverted version of the operator (see Sec.). Figure 2 shows the result of twelve diffusion iterations performed using the conventional view and compares it with our inverted operator. Although diffusion is insufficient to localise homology, it highlights the tendency of the inverted operator to localize cycles.

The main contributions in the paper are:

1. A novel way to represent simplicial complexes as computational graphs suitable for inference via GNNs, the *Hodge Laplacian graphs*, focusing on the dimension of interest (see Section), and

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

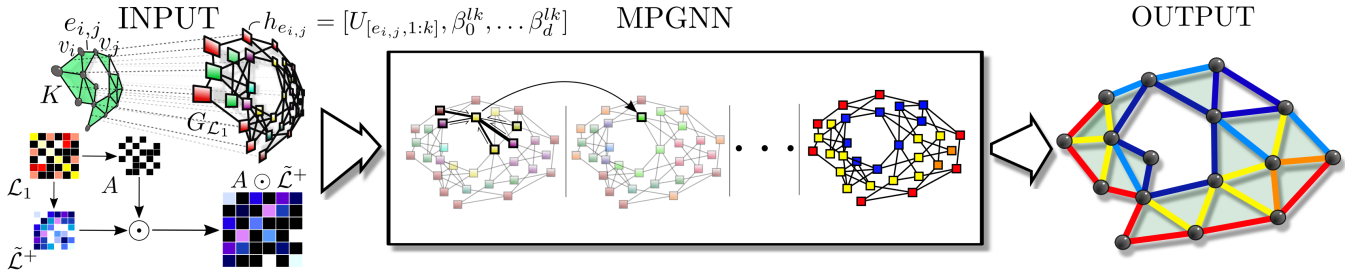


Figure 1: Overview of Dist2Cycle model. An arbitrary input complex K is transformed into a Hodge Laplacian graph $G_{\mathcal{L}_1}$, via its Hodge Laplacian appropriately shift-inverted ($\tilde{\mathcal{L}}^+$) (Section), and zero-masked ($A \odot \tilde{\mathcal{L}}^+$) (Section), suitable for downstream processing as a GNN by the proposed Dist2Cycle model (Section). The model outputs the distance of each simplex to its nearest optimal homology generator (cooler colors indicate closeness).

2. A new homology-aware graph convolution framework operating on the proposed Hodge Laplacian graphs, taking advantage of the spectral properties of a shifted-inverted version of the Hodge Laplacian (see Section , Eq. (8)).

The rest of the paper is structured as follows: first all the necessary theoretical background is presented in the "Preliminaries" section, followed by a literature review of relevant work. We then describe our proposed model in detail in the "Dist2Cycle" section. We end the paper with a thorough evaluation and discussion of our model.

Preliminaries

The Simplicial Laplacian Operators

An *abstract simplicial complex* is a collection K of subsets of a finite set S satisfying two axioms: first, for each v in S the singleton set $\{v\}$ lies in K, and second, whenever some $\sigma \subset S$ lies in K, every subset of σ must also lie in K. The constituent subsets $\sigma \subset S$ which lie in K are called *simplices*, and the dimension of each such σ is one less than its cardinality, i.e., $\dim \sigma = |\sigma| - 1$. By far the most familiar examples of simplicial complexes are (undirected, simple) *graphs*; each graph $G = (V, E)$ forms a simplicial complex whose 0-dimensional simplices are given by the vertex set V and 1-dimensional simplices constitute the edge set E. The passage from graphs to simplicial complexes is motivated by the compelling desire to model phenomena beyond pairwise interactions using higher-dimensional simplices.

Homology Groups To each directed graph $G = (V, E)$ one can associate an *incidence matrix*, which is best viewed as a linear map $A : \mathbb{R}[E] \to \mathbb{R}[V]$ from a real vector space spanned by edges to the vector space spanned by the vertices. The entry of A in the column corresponding to a directed edge $e : v \to v'$ and the row corresponding to a vertex u is prescribed by

$$A_{u,e} = \begin{cases} -1 & \text{if } u = v, \\ 1 & \text{if } u = v', \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

Writing r for the rank of A, the number of connected components and loops in G equals $|V| - r$ and $|E| - r$, respectively. Thus, one can learn the geometry of G from the linear algebraic data given by its adjacency matrix.

This linear algebraic success story admits a remarkable simplicial sequel. Fix a simplicial complex K and write K_d to indicate the set of all d-simplices in K. We seek linear maps $\partial_d : \mathbb{R}[K_d] \to \mathbb{R}[K_{d-1}]$ to play the role of the d-dimensional incidence matrices. To build these *boundary operators*, one first orders the vertices in K_0 so that each d-simplex $\sigma \in K$ can be uniquely expressed as a list $\sigma = [v_0, \dots, v_d]$ of vertices in increasing order. The desired matrix ∂_d is completely prescribed by the following action on each such σ:

$$\partial_d(\sigma) = \sum_{i=0}^{d} (-1)^i \cdot \sigma_{-i} \tag{1}$$

where $\sigma_{-i} := [v_0, \dots, \hat{v}_i, \dots, v_d]$ is the $(d-1)$-simplex obtained by removing the i-th vertex v_i from σ.

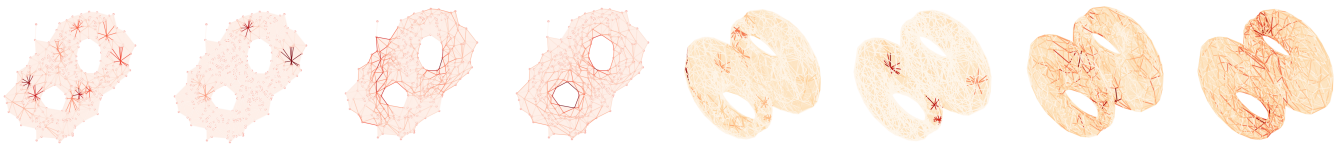
These higher incidence operators assemble into a sequence of vector spaces and linear maps:

$$\dots \xrightarrow{\partial_{d+1}} \mathbb{R}[K_d] \xrightarrow{\partial_d} \mathbb{R}[K_{d-1}] \xrightarrow{\partial_{d-1}} \cdots . \tag{2}$$

It follows from (1) that for each $d > 0$ the composite $\partial_d \circ \partial_{d+1}$ is the zero map, so the kernel of ∂_d contains the image of ∂_{d+1} as a subspace, $\operatorname{im} \partial_{d+1} \subseteq \ker \partial_d$.

For each $d \geq 0$, the d-th **homology group** of K is the quotient vector space $\mathcal{H}_d(K) := \ker \partial_d / \operatorname{im} \partial_{d+1}$ of k-cycles $\mathcal{Z}_k = \ker \partial_d$ by $(k+1)$-boundaries $\mathcal{B}_k = \operatorname{im} \partial_{d+1}$. The basis of $\mathcal{H}_d(K)$ contains equivalence classes of d-dimensional *voids* or *loops* $[g_i]$, i.e. $\mathcal{H}_d(K) = \operatorname{span}\{[g_1], \dots, [g_k]\}$, each $[g_i]$ describing a family of loops that cannot be contracted to a point, and cannot be continuously deformed into another family $[g_j]$, $i \neq j$. Consequently, the dimension of $\mathcal{H}_d(K)$ provides us with a topological invariant, namely, the k-th **betti number** $\beta_d = \operatorname{rank}(\mathcal{H}_d(K))$, which counts the number of d-dimensional voids in K.

Each d-cycle $g \in \mathcal{Z}_d$ is a formal sum of d-simplices satisfying $\partial_d(g) = 0$. By assigning weights $w : K_d \to \mathbb{R}_+$ to these simplices, one can thus define the length of g by adding together weights of its constituent simplices, i.e., $\operatorname{len}(g) = \sum_{\sigma \in g} w(\sigma)$. An *optimal* homology basis is the one whose generators have minimum length among all possible



(a) $\mathcal{L}_1$ (b) $\mathcal{L}_1$ (rank-3) (c) $\tilde{\mathcal{L}}_1^+$ (d) $\tilde{\mathcal{L}}_1^+$(rank-3) (e) $\mathcal{L}_1$ (f) $\mathcal{L}_1$ (rank-5) (g) $\tilde{\mathcal{L}}_1^+$ (h) $\tilde{\mathcal{L}}_1^+$(rank-5)

Figure 2: Twelve diffusion iterations based on the 1-dimensional Hodge Laplacian $\mathcal{L}_1$, and its shifted pseudoinverse $\tilde{\mathcal{L}}_1^+$ of a double annulus in 2D (four leftmost) and a double torus in 3D (four rightmost). (b), (d), (f), and (h) are low-rank approximations of the respective Laplacians based on the top 3 and 5 eigenpairs corresponding to the largest magnitude eigenvalues.

bases. Assuming unit, or Euclidean, edge weights, Figure 3 depicts the optimal $\mathcal{H}_1$ basis of a torus, in blue, along with a cycle homologous to the generator inscribing the central "hole", in red, and a trivial, contractible 1-cycle belonging to $\mathcal{B}_1$, in green.

Replacing each matrix ∂_d in (2) by its transpose ∂_d^T, one similarly obtains the d-th **cohomology group** of K, denoted $\mathcal{H}^d(K;\mathbb{R})$. It is a straightforward consequence of the rank-nullity theorem that there are isomorphisms $\mathcal{H}_d(K;\mathbb{R}) \cong \mathcal{H}^d(K;\mathbb{R})$ between homology and cohomology groups.

Hodge Laplacians *Hodge Laplacians* (Horak and Jost 2013) are to graph Laplacians what simplicial boundary operators are to adjacency matrices. Given a simplicial complex K and the corresponding sequence (2), both composites $\mathcal{L}_d^{\text{up}} := \partial_{d+1}\partial_{d+1}^T$ and $\mathcal{L}_d^{\text{down}} := \partial_d^T\partial_d$ furnish linear maps $\mathbb{R}[K_d] \to \mathbb{R}[K_d]$. The d-th Hodge Laplacian is their sum:

$$\mathcal{L}_d := \mathcal{L}_d^{\text{up}} + \mathcal{L}_d^{\text{down}}. \tag{3}$$

An immediate consequence of this definition is that the standard graph Laplacian agrees with the 0-th Hodge Laplacian. The nullity of the graph Laplacian equals the number of connected components of the underlying graph. Similarly, the kernel of the d-th Hodge Laplacian of a simplicial complex K is isomorphic the corresponding d-th homology group (Eckmann 1944):

$$\ker \mathcal{L}_d(K) \cong \mathcal{H}_d(K;\mathbb{R}). \tag{4}$$

The aforementioned isomorphism still holds for the case of weighted simplicial complexes, where simplices are endowed with non-trivial weights, provided appropriately *weighted* Hodge Laplacians (Horak and Jost 2013) are employed.

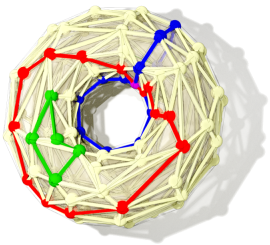


Figure 3: Optimal $\mathcal{H}_1$ homology basis (blue) against an arbitrary cycle homologous to the central loop (red), and a trivial, contractible, boundary cycle (green).

Graph Neural Networks

Graph Neural Networks (GNNs) provide a general framework for *Geometric Deep Learning* (Bronstein et al. 2021), where the input domain, represented by a *graph* $G = (V, E)$, is allowed to vary together with the signals that are defined on it. More concretely, the *Message Passing Graph Neural Network* (MPGNN) framework generalizes the convolution operation on the edges of a graph G by employing a simple message passing scheme between features of nodes $h_u, u \in V$, and their neighbors $v \in \mathcal{N}_u$.

The output of each layer ℓ for each node u can be broadly formulated as:

$$h_u^{\ell+1} = \phi\left(h_u^\ell, \bigoplus_{v\in\mathcal{N}_u} w_{u,v} \cdot \psi\left(h_u^\ell, h_v^\ell\right)\right), \tag{5}$$

with $\bigoplus$ being a *permutation invariant* aggregation, ϕ and ψ learnable functions, and $w_{u,v}$ the weight of edge $(u, v) \in E$. Under this formulation, learnable parameters of ϕ and ψ are shared across all nodes in the graph network.

Each message passing layer with summation aggregation can be described more compactly using matrix notation:

$$H^{\ell+1} = \phi\left(\tilde{L}\psi(H^\ell)\right), \tag{6}$$

where $\tilde{L} = AWA^T$ is the weighted graph Laplacian matrix, and H^0 the $|V| \times F$ matrix of initial node features. This formulation highlights the similarities of GNNs with Laplacian diffusion operations, a fact that we will largely exploit.

The output of a number of message passing iterations results to latent *node embeddings*, largely based on the local graph topology at each node. Such embeddings can be subsequently used for node regression, node classification, or, via feature aggregation of all nodes, for graph classification and aggregation tasks.

Related Work

Homology Localization The *minimum basis problem* in computational topology involves extracting optimal homology generators, with optimality usually expressed in terms of norm or length minimization of cycles. In dimensions exceeding one, this is an NP-hard problem (Chambers, Erickson, and Nayyeri 2009; Chen and Freedman 2011), whereas the 1-dimensional case succumbs to a polynomial time algorithm (Dey, Sun, and Wang 2010; Dey, Li, and Wang 2018).

This latter fact spawned a significant body of work examining special cases and computational improvements (Borradaile et al. 2017; Chen and Freedman 2010; Dey, Hirani, and Krishnamoorthy 2011; Erickson and Whittlesey 2005; Busaryev et al. 2011; Chen and Meilă 2021).

While the aforementioned methods generally output sets of simplices that form optimal homology generators in their respective class, the rest of the simplices in the complex remain largely oblivious to the location of such optimal cycles in relation to themselves. In our work we attempt to characterize each simplex in the complex with respect to its nearest homology generator, while gaining in efficiency (once the model is sufficiently trained). More similar to our line of work, (Ebli and Spreemann 2019) implements a homology-aware clustering method for point data.

Topological Methods in ML With the marriage of homology and ML (Hensel, Moor, and Rieck 2021; Love et al. 2021; Montúfar, Otter, and Wang 2020; Hofer, Kwitt, and Niethammer 2019), it did not take long for GNNs to meet their higher dimensional counterparts in the form of simplicial (Bodnar et al. 2021b; Ebli, Defferrard, and Spreemann 2020; Bunch et al. 2020), cell (Hajij, Istvan, and Zamzmi 2021; Bodnar et al. 2021a), hypergraph (Feng et al. 2019), and sheaf (Hansen and Gebhart 2020) neural networks. Most higher dimensional extensions of GNNs aim to operate on the full complex, and redefine the convolution operation in terms of the corresponding Laplacian operator. Contrary to such generalizations, we still operate on a graph. The key difference is that our graph is derived from adjacency and Hodge Laplacian information of the complex at the dimension of interest.

Pseudoinverse & Hodge Laplacians in GNNs The pseudoinverse and shifted versions of the Laplacian operator are not new in the context of GNNs (Klicpera, Weißenberger, and Günnemann 2019; Wu et al. 2019; Alfke and Stoll 2021). Nevertheless, they only consider spectral manipulations of the "classic" graph Laplacian, whose kernel is usually of no practical interest, as long as the graph is connected.

More closely to our work, (Roddenberry and Segarra 2019; Schaub and Segarra 2018) consider edge-flows for signal denoising, interpolation, and source localization based on the *linegraph Laplacian*, and the 1-dimensional down Hodge Laplacian $\mathcal{L}_1^{\text{down}}$, based on a proxy graph resulting from interchanging edges and nodes. Nevertheless, their analysis remains restricted on graph structures, disregarding any homological features.

The baseline works considered in the present paper are counterposed in Table 1. The two methods we experimentally compare against, shortloop (Dey, Sun, and Wang 2010) and hom_emb (Chen and Meilă 2021), expect complexes embedded in a metric space (first column). Ours can operate purely on the combinatorial structure, and additional structure, such as simplex weights, can be encoded via a weighted Hodge Laplacian, if desired. Although our method depends on training (second column), it is virtually independent of the number of simplices N, as long as the complex, or local neighborhoods of simplices, can fit in GPU memory. The

	embed	training	time	cycles	dist.
shortloop	yes	no	$O(N^4)$	yes	no
distr_cover_loc	no	no	N/A	yes	no
hom_emb	yes	no	$O(n_1^\omega)$	yes	no
ours	no	yes	$O(1)$	no	yes

Table 1: Qualitative comparison of our proposed method against three baselines, shortloop (Dey, Sun, and Wang 2010),distr_cover_loc (Tahbaz-Salehi and Jadbabaie 2010), and hom_emb (Chen and Meilă 2021). N and n_1 are the total number of simplices, and edges, respectively, ω is the matrix multiplication time exponent.

reference baseline, shortloop, requires $O(N^4)$ time, whereas hom_emb requires $O(n_1^{2.37\cdots})$ (third column). The alternative baseline considered, distr_cover_loc (Tahbaz-Salehi and Jadbabaie 2010), does not provide runtime complexity or computation times, but their method relies on L_1-relaxation minimization. Finally, post-processing is required to calculate distances to optimal homology generators using the baselines (columns 4-5). In our case, post-processing will be required to identify the generators in terms of the simplices comprising them.

Dist2Cycle

Here we present a model for learning homology-aware distance functions on simplicial complexes.

Shifted Inverted Hodge Laplacians

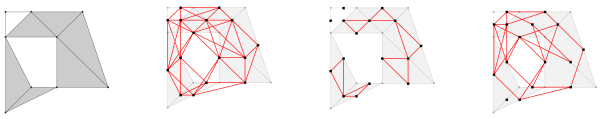
The spectral properties of the Hodge Laplacian matrices provide salient information regarding the geometry and topology of a simplicial complex, as hinted in Section . Furthermore, Laplacian flow dynamical systems on simplicial complexes tend to stabilize towards specific spectral regions of the Laplacian (Muhammad and Egerstedt 2006). Nevertheless, the choice of the Laplacian operator with which diffusion is performed impacts greatly the energy distribution on the simplices of interest.

In Figures 2(a),(b),(e),(f) we perform 12 diffusion steps according to the Hodge Laplacian $\mathcal{L}_1$ (and its low-rank approximation using the top 3 and 5 eigenpairs, respectively), namely,

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \mathcal{L}_1 \boldsymbol{x}_i, \tag{7}$$

with $\boldsymbol{x}_0 = [1\ 1\ \dots\ 1]^T$ as the initial signal on the 1-simplices of the complex. We show the absolute value of the resulting flow vector at each simplex, on two basic examples. Energy tends to concentrate at well connected simplices, while ignoring the homological features that we are interested in.

The Laplacian diffusion of (7) can be seen as a simplified version of a graph convolution that takes place in GNNs (5), with all nonlinearities and learnable parameters pruned, and trivial initial features. Thus, in order to focus our attention on optimal homology generators, we must invert the spectrum of the Hodge Laplacian $\mathcal{L}_1$, while making sure that its kernel will replace the part of the spectrum corresponding to its top



(a) K. (b) $G_{\mathcal{L}_1^{\text{down}}}$. (c) $G_{\mathcal{L}_1^{\text{up}}}$. (d) $G_{\mathcal{L}_1}$.

Figure 4: Laplacian graph constructions on example complex K for 1-simplices (square nodes with red edges overlaid on top of the original complex).

eigenvalues. For this purpose we employ a *shifted inverted* version of the Hodge Laplacian

$$\tilde{\mathcal{L}}_d^+ = (\mathbb{1} + \mathcal{L}_d)^+,$$

which makes the $\ker \mathcal{L}_d$ the prominent part of the spectrum of $\tilde{\mathcal{L}}_d^+$ with eigenvalue 1, onto which the diffusion asymptotically converges. The effect this modified Laplacian matrix has on diffusion is shown in 2(c),(d),(g),(h).

From Simplicial Complexes to Graph Neural Networks

In order to employ the GNN framework for inference on the complex, we need to express the space of k-simplices accordingly. Furthermore, we desire the resulting graph structure to retain the spectral properties of the Laplacian operator of interest.

We interpret the $|K_d| \times |K_d|$ Hodge Laplacian operator $\mathcal{L}_d$ (or $\tilde{\mathcal{L}}_d^+$) as the weighted graph Laplacian $\tilde{L}_{\text{GNN}} = AWA^T$ of a computational graph $G_{\text{GNN}} = (V_{\text{GNN}}, E_{\text{GNN}})$. Under this lens, each d-simplex σ of the original complex K becomes a node of G_{GNN}, i.e. $\sigma \in V_{\text{GNN}}$, and weighted edges are drawn according to the adjacency information encoded in $\mathcal{L}_d$ ($\tilde{\mathcal{L}}_d^+$). Namely, a weighted edge $(\sigma, \tau) \in E_{\text{GNN}}$ is drawn between nodes corresponding to k-simplices σ and τ, whenever the (potentially normalized) Laplacian operator $\mathcal{L}_d$ ($\tilde{\mathcal{L}}_d^+$) contains a nonzero entry in the respective position. This entry is also used to weight the corresponding edge $w_{\sigma,\tau} = \mathcal{L}_{\sigma,\tau}$, allowing self-loops. Figure 4 provides an example of the resulting graph, what we call the *Hodge Laplacian graph*, when using $\mathcal{L}_1^{\text{down}}$, $\mathcal{L}_1^{\text{up}}$, and $\mathcal{L}_1$ for extracting adjacency relations on 1-simplices (sans self-loops, for easier vizualization). A somewhat similar approach is followed in (Roddenberry and Segarra 2019), with their mapping akin to the graph in Figure 4(b) minus the 2-simplices, as they are only dealing with graphs.

As mentioned in Section , we are interested in capturing the spectrum, and thus the connectivity information of $\tilde{\mathcal{L}}^+$, which is in general a dense matrix and hence computationally prohibitive to work with directly. To overcome this issue, we impose the sparsity structure of $\mathcal{L}$ to $\tilde{\mathcal{L}}^+$, masking all entries of $\tilde{\mathcal{L}}^+$ that are zero in the original, sparse, Hodge Laplacian $\mathcal{L}$. If we denote by A the adjacency matrix encoding the connectivity of $\mathcal{L}$, with

$$A_{u,v} = \begin{cases} 1 & \text{if } \mathcal{L}_{u,v} \neq 0, \text{ and} \\ 0 & \text{otherwise} \end{cases},$$

this can be achieved with the Hadamard product $A \odot \tilde{\mathcal{L}}^+$. The resulting graph is called the **Hodge Laplacian graph** throughout this paper.

While more sophisticated methods for spectral sparsification exist (Spielman and Srivastava 2011), imposing the connectivity dictated by $\mathcal{L}$ or $\mathcal{L}^{\text{down}}$ seems to preserve all important adjacency information required for the task at hand, while not annihilating important spectral information. Furthermore, in the context of learning, this approach is reminiscent to inference with missing values, which GNNs are known to handle well (You et al. 2020).

Shifted Inverted Laplacian GNNs for Homology Localization

We are now ready to propose a *Simplicial Neural Network* model for homology localization. By following the construction described in Section we obtain a weighted computational graph $G_{\text{GNN}} = (V_{\text{GNN}}, E_{\text{GNN}})$, with weights according to $\tilde{\mathcal{L}}^+$ and adjacency dictated by $\mathcal{L}$ (or $\mathcal{L}^{\text{down}}$). Thus, graph convolution (message passing) on the G_{GNN} can be summarized as:

$$H^{\ell+1} = \phi\left(A \odot \tilde{\mathcal{L}}_d^+ H^\ell W^\ell\right), \tag{8}$$

where A is the adjacency matrix describing the selected sparsification regime according to $\mathcal{L}$ (or $\mathcal{L}^{\text{down}}$), $\tilde{\mathcal{L}}_d^+$ the *shifted inverted Hodge Laplacian* in dimension d (Section), and $\odot$ denoting the Hadamard product. The learnable weights of the model at layer ℓ are denoted as W^ℓ, and ϕ can be any activation function, such as ReLU, Sigmoid, etc. Finally, H^ℓ is the $|V_{\text{GNN}}| \times F$ feature matrix having the F-dimensional features of each node (i.e. d-simplex) as rows.

To aid the task of homology localization, we encode both local and global information at each node. Locality is incorporated by computing betti numbers $[\beta_0, \ldots, \beta_{d+1}]$ of the *link* at each d-simplex σ — this is the subcomplex consisting of all simplices τ for which $\sigma \cap \tau$ is empty and $\sigma \cup \tau$ is a simplex in K. Global features manifest in the form of spectral embeddings of the d-simplices in the space spanned by singular vectors corresponding to the largest k singular values of $\tilde{\mathcal{L}}_d^+$. Denoting the appropriately permuted singular value decomposition (SVD) of $\tilde{\mathcal{L}}_d^+$ as $\tilde{\mathcal{L}}_d^+ = U\Sigma V^T$ with Σ containing in its diagonal the singular values of $\tilde{\mathcal{L}}^+$ in descending order, the rows of the matrix $U_{1:k}$ formed by the first k singular vectors constitute coordinates of the simplices in the eigenspace of $\tilde{\mathcal{L}}_d^+$. Due to the shift-invert operation of $\tilde{\mathcal{L}}^+$, this scheme effectively embeds the d-simplices in the spectral subspace corresponding to the kernel of $\mathcal{L}_d$, i.e. the space encoding homological information.

Evaluation

In this section we describe the function learned by our model, the dataset we developed to train and evaluate our model (Section), the experiments conducted (Section) and an analysis of the results (Section)[1].

[1]Code & models: https://github.com/alexdkeros/Dist2Cycle

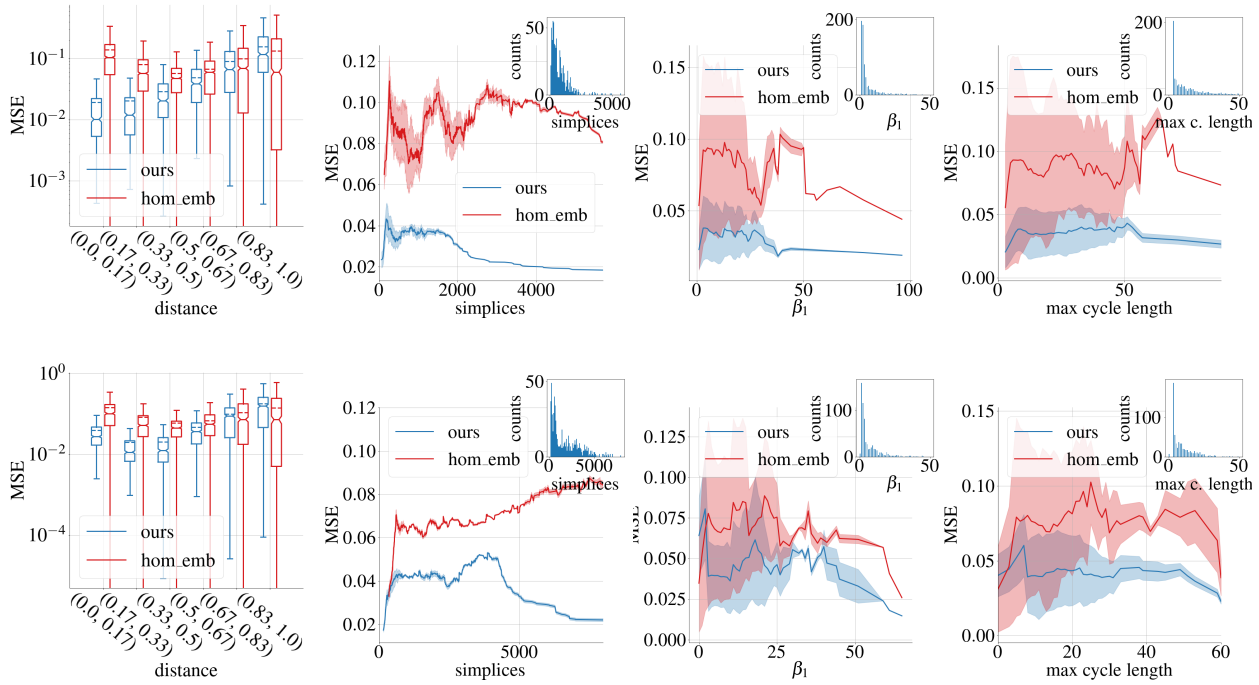


Figure 5: MSE error plots comparing our method against hom_emb for the TORI dataset in 2D (top) and 3D (bottom). Due to hardness of computing a "ground truth", the best known baseline appoximation, shortloop, acts as reference. Column 1 shows MSE against stratified distance values (x-axis). Columns 2, 3 and 4 plot MSE against the number of simplices, the homology rank β_1, and the maximum cycle length, respectively. Dashed line: mean, solid horizontal line: median, box limits: quartiles, whiskers: error range, shaded regions: standard deviation, insets: histograms for the respective parameters in the test set.

Nearest Optimal Homology Generator

Our model approximates a function that depends on the optimal homology generators. To validate our model we only consider 1-simplices, since efficient algorithms acting as proxies to ground truth, and combinatorial, baseline, methods only exist for $d = 1$. We denote by $\mathcal{Q}_1$ the set of optimal generators of $\mathcal{H}_1$. For each simplex $\sigma \in K_1$ we seek to learn its distance from the nearest optimal $g \in \mathcal{Q}_1$,

$$f(\sigma) = \min_{g \in \mathcal{Q}_1} \hat{d}(\sigma, g). \tag{9}$$

As distance $d(\sigma, g)$ between a k-simplex σ and a k-dimensional homology generator we consider the minimum number of k-simplices required to reach any k-simplex $\rho \in g$ participating in a k-cycle g. To keep the function complex-independent, we then normalize the distance in the range $[0, 1]$, obtaining $\hat{d}(\cdot)$, with simplices near an optimal homology generator attaining values close to zero.

TORI Dataset

Our TORI datasets consist of Alpha complexes (Edelsbrunner 2010) that originate from considering "snapshots" of filtrations (Edelsbrunner and Harer 2010) on points sampled from tori manifolds of diverse topological characteristics, in 2 and 3 dimensions. We seek to capture richness of homological information, controlability in terms of scalability in the number of simplices and homology cycles, as well as ease of visualization.

We first sampled 400 point clouds from randomly generated configurations of tori and pinched tori, with number of "holes" ranging from 1 to 5, to which Gaussian noise is added. We then constructed Alpha filtrations on the collection of point clouds, i.e. sequences of simplicial complexes dictated by a monotonically increasing distance parameter α. Tracking homological changes in the sequence of complexes results in a *barcode* representation, with one bar per homology feature, that spans a range of α values. The longer the bar, the more persistent, and possibly "significant", a homological feature is. From these barcodes we considered the 5 most persistent features, expressed by the longest bars. The birth and death values of these features were deemed as appropriate points to capture "snapshots" of the complexes, guaranteed to contain interesting large and small scale homological information. This pipeline yields a collection of 2000 complexes for each dataset. The generality of the datasets stems from the spurious homological features occuring while considering filtrations of noisy point clouds, as confirmed by the histogram insets describing the test sets in Figure 5. The number of simplices ranges from tens to thousands, and betti numbers, i.e. number of homology cycles, from 0 up to 66. Furthermore, homology generators present in the dataset can contain from 3 up to 60

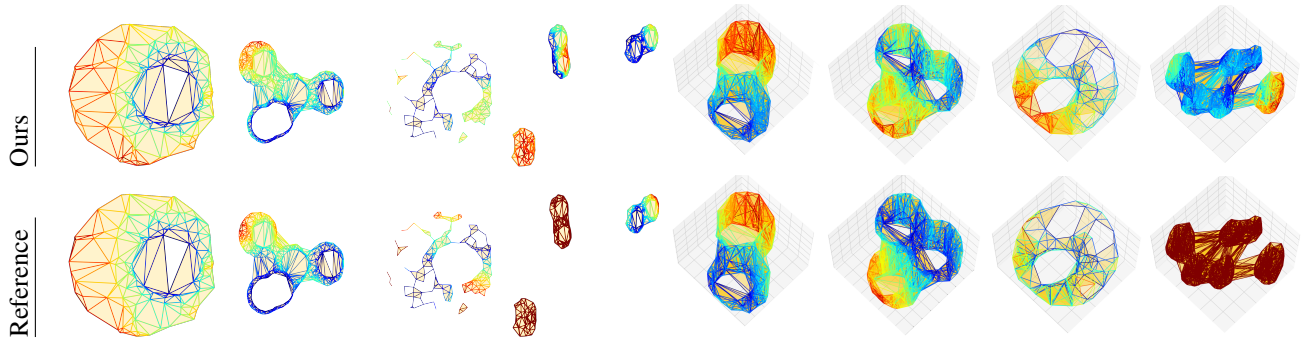


Figure 6: Qualitative comparisons of selected complexes from the 2D TORI (four left) and 3D (four right) test set. The 1-simplices of the complexes are color-coded according to their distance from the nearest homology cycle, with blue indicating close proximity to an optimal homology cycle, and red indicating large distance from a homology cycle.

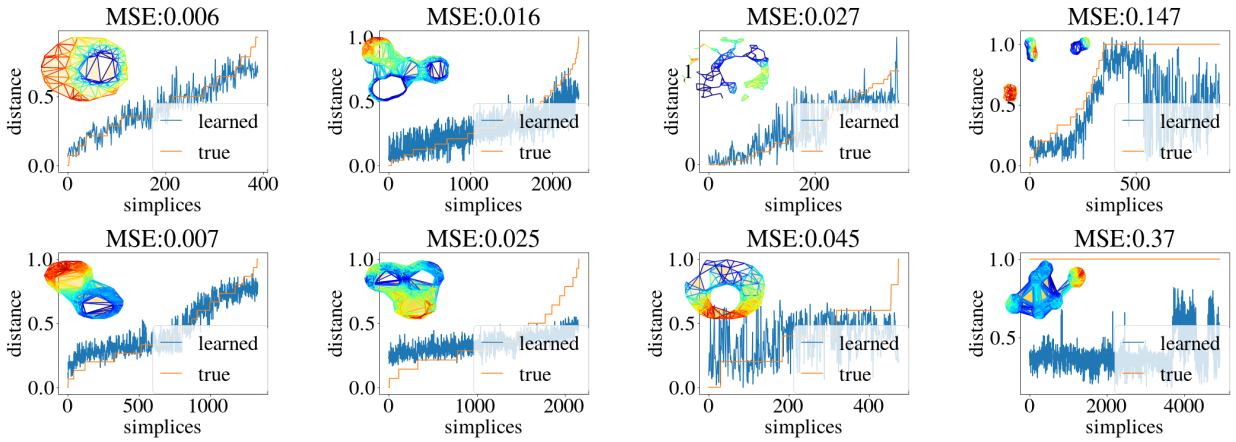


Figure 7: Plots of predicted distances (blue) of the simplices sorted in increasing ground truth distance value (orange) for the Tori dataset in 2D (top) and 3D (bottom). The plots show four cases (columns) ranging from the best (left) to the worst (right).

1-simplices.

Unfortunately, constructing a general enough dataset that contains rich homological information, whilst explicitly knowing ground-truth optimal homology generators, is not feasible. Thus, we calculate the reference function using optimal homology generators of $\mathcal{H}_1$ discovered via *shortloop* (Dey, Sun, and Wang 2010), by calculating the normalized "hop" distance of each simplex to its nearest optimal generator (see Eq.(9)). Thus, shortloop algorithm acts both as a baseline, and a *ground truth proxy*.

Experimental Settings

We used a GNN with 12 graph convolutional layers (for 2D as well as 3D), as described by Eq. (8), and 128 hidden units. We chose LeakyReLU activations (ϕ in Eq. (5)) with negative slope $r = 0.02$ for the layers, and a hyperbolic tangent Tanh for the output. Neighbor activations are aggregated via a summation ($\bigoplus$ in Eq. (5)). Learnable weights undergo Kaiming uniform initialization (He et al. 2015). Finally, node features are the result of concatenating the betti numbers describing the homology of the link at each simplex, with its 5-dimensional spectral embedding.

The dataset is split into training (80%) and testing (20%) sets and the models were trained for 1000 epochs, with a mini-batch size of 5 complexes using an Intel Xeon E5-2630 v.4 processor, a TITAN-X 64GB GPU and 64GB of RAM, using CUDA 10.1. The GNN model was implemented using the dgl library (Wang et al. 2019) with the Torch backend (Paszke et al. 2017). All simplicial and homology computations were handled by the Gudhi library (The GUDHI Project 2021).

We apply a Laplacian smoothing post-processing step. Let $\boldsymbol{x}$ the output of the model, i.e. the inferred distances for each 1-simplex, and $\hat{L} = D^{-1/2}(D-A)D^{-1/2}$ the normalized graph Laplacian of the 1-skeleton of the complex K, i.e. the underlying graph spanned by the 0 and 1-simplices of K. The signal at the simplices are smoothed using $\boldsymbol{x}' = \boldsymbol{x} - \hat{L}\boldsymbol{x}$.

Results

The main quantitative results can be found in Figure 7 and Figure 5 with qualitative examples in Figure 6. We report mean squared error (MSE) between the predicted and reference relative distances, with distances based on shortloop (Dey, Sun, and Wang 2010) acting as ground truth. We

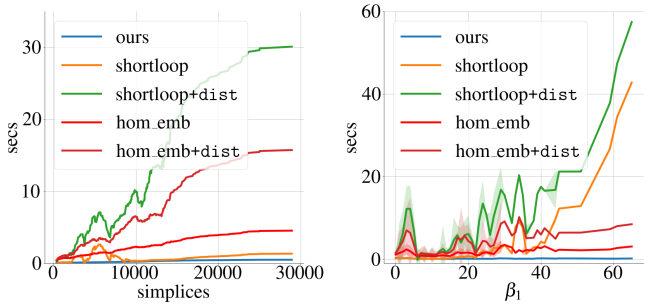


Figure 8: Computation time as a function of the number of simplices (left), and the number of generators (right) for the 3D TORI dataset. We compare our trained model against the reference baseline, shortloop, and hom_emb. +dist denotes additional post-processing required for obtaining distances.

experimentally compare our method against a combinatorial baseline, hom_emb (Chen and Meilă 2021). Since the range of $\hat{d}(\cdot)$ is within $[0, 1]$, MSE can never exceed 1, i.e. 100% error. While additional baselines were considered, such as distr_cover_loc (Tahbaz-Salehi and Jadbabaie 2010), unfortunately they do not provide code or empirical analysis that is easy to compare with.

In 2D as well as 3D our model learns the homology-parametrized distance function by achieving an MSE of 3.81% (0.0381) (2D), and 4.47% (0.0447) (3D), respectively, compared to 7.86% (0.0786) (2D), and 8.89% (0.0889) (3D), obtained by hom_emb. Figure 5 (first column) provides insights into the distribution of error at different distances (x-axis) from optimal generators. In 2D the error is monotonically increasing with the distance from the homology generators, whereas in 3D the model performs slightly better at relative distances of about a third from the optimal cycles. Our method consistently outperforms hom_emb at small and moderate distances (< 0.67), whereas for the farthest distance range hom_emb performs slightly better. In both settings, areas far away from the homology cycles attain the maximum mean MSE but never in excess of 15%.

Figure 5 also investigates the scalability of the model as the number of simplices, homology features (β_1) and maximum cycle lengths, increase (columns 2-4). The insets provide histograms of the respective parameter counts in the test sets to shed light into the standard deviation (shaded). The parameter values are non-uniformly represented in the test sets, partly explaining the larger variance towards the lower ends of the value ranges. Our model scales well in all three parameters and we observe that the error decreases for larger numbers of simplices. The baseline, hom_emb, overall scales similarly to our method (with the exception of scalability in terms of number of simplices), albeit exhibiting consistently higher MSE across all parameter scales. The maximum MSE of our method across all parameters never exceeded 12%.

Qualitative assessment of the model's performance is provided in Figure 6 for examples from the test sets. The comparisons are arranged from lowest error (left) to maximum error (right) within our dataset. The top row of figures visualize the predicted distances projected onto the complex while the bottom shows the ground truth. Cool areas indicate close proximity to an optimal homology generator.

Figure 7 plots reference distance values (orange) and our model's output (blue) for each simplex against the rank of the simplex's distance from an optimal generator (X axis). Ideally, the blue curve should be monotonically increasing and should closely match the orange curve.

Both in 2D and 3D the model can handle multiple homology cycles of various lengths, even greater than the number of convolutional layers that usually dictate the receptive field of each simplex. Problematic appear to be the cases where components of the complexes have trivial homology, evident from the rightmost column in Figure 7. In such cases the model attempts to detect homological structure, where none exists.

Timing results are provided in Figure 8, where computation time is presented as a function of the number of simplices, as well as the number of generators. Our trained model (blue) is more efficient than the two iterative baseline methods, shortloop (orange), and hom_emb (red). We also exhibit time taken for other methods to post-process the results in order to obtain relative distances to the homology generators of interest (+dist). The cost of post-processing is independent of generators, but scales poorly with the number of simplices.

Limitations and Conclusion

Our model entifies simplices that are distant to homology cycles, aided by the shifted-inverted Hodge Laplacian based graph convolution and the simplified simplex adjacency construction. We experimented with using a Hasse graph analogue, as well as the original, non-shifted, non-inverted, Hodge Laplacians of the complex, but these models failed to learn.

Another advantage of our model is that both large and small scale homology cycles are consistently localized. Although we restricted our analysis to $\mathcal{H}_1$ for ease of evaluation, the generality of our model allows direct extension to higher dimensional simplices.

One drawback of our model is that it performs poorly when components of complexes contain no higher dimensional homological information whatsoever. In such cases, it hallucinates homology cycles while they do not exist.

Another limitation is the variance of the inferred function on the complex, as shown in Figure 7. This variance stems from two sources: First, the target function is piecewise constant; and second, the adjacency structure that we use to sparsify an otherwise complete graph (for creating the GNN computational graph) alters the spectrum of the kernel.

The choice of a spectral sparsification method with theoretical guarantees is an interesting avenue of future work. Needless to say, our results are promising even with a basic GNN architecture.

Acknowledgements

VN is supported by EPSRC grant EP/R018472/1. Kartic Subr was supported by a Royal Society University Research Fellowship.

References

Aktas, M. E.; Akbas, E.; and El Fatmaoui, A. 2019. Persistence homology of networks: methods and applications. *Applied Network Science*, 4(1): 1–28.

Alfke, D.; and Stoll, M. 2021. Pseudoinverse graph convolutional networks. *Data Mining and Knowledge Discovery*, 1–24.

Bodnar, C.; Frasca, F.; Otter, N.; Wang, Y. G.; Liò, P.; Montúfar, G.; and Bronstein, M. 2021a. Weisfeiler and Lehman Go Cellular: CW Networks. *arXiv preprint arXiv:2106.12575*.

Bodnar, C.; Frasca, F.; Wang, Y. G.; Otter, N.; Montúfar, G.; Lio, P.; and Bronstein, M. 2021b. Weisfeiler and lehman go topological: Message passing simplicial networks. *arXiv preprint arXiv:2103.03212*.

Borradaile, G.; Chambers, E. W.; Fox, K.; and Nayyeri, A. 2017. Minimum cycle and homology bases of surface embedded graphs. *Journal of Computational Geometry*, (8(2)).

Bronstein, M. M.; Bruna, J.; Cohen, T.; and Veličković, P. 2021. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*.

Bunch, E.; You, Q.; Fung, G.; and Singh, V. 2020. Simplicial 2-complex convolutional neural nets. *arXiv preprint arXiv:2012.06010*.

Busaryev, O.; Cabello, S.; Chen, C.; Dey, T. K.; and Wang, Y. 2011. Annotating Simplices with a Homology Basis and Its Applications. 1–14.

Chambers, E. W.; Erickson, J.; and Nayyeri, A. 2009. Minimum cuts and shortest homologous cycles. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, 377–385.

Chazal, F.; and Michel, B. 2017. An introduction to topological data analysis: fundamental and practical aspects for data scientists. *arXiv preprint arXiv:1710.04019*.

Chen, C.; and Freedman, D. 2010. Measuring and computing natural generators for homology groups. *Computational Geometry*, 43(2): 169–181.

Chen, C.; and Freedman, D. 2011. Hardness results for homology localization. *Discrete & Computational Geometry*, 45(3): 425–448.

Chen, Y.-C.; and Meilă, M. 2021. The decomposition of the higher-order homology embedding constructed from the k-Laplacian. *arXiv preprint arXiv:2107.10970*.

Dey, T. K.; Hirani, A. N.; and Krishnamoorthy, B. 2011. Optimal homologous cycles, total unimodularity, and linear programming. *SIAM Journal on Computing*, 40(4): 1026–1044.

Dey, T. K.; Li, T.; and Wang, Y. 2018. Efficient algorithms for computing a minimal homology basis. In *Latin American Symposium on Theoretical Informatics*, 376–398. Springer.

Dey, T. K.; Sun, J.; and Wang, Y. 2010. Approximating loops in a shortest homology basis from point data. In *Proceedings of the twenty-sixth annual symposium on Computational geometry*, 166–175.

Ebli, S.; Defferrard, M.; and Spreemann, G. 2020. Simplicial neural networks. *arXiv preprint arXiv:2010.03633*.

Ebli, S.; and Spreemann, G. 2019. A notion of harmonic clustering in simplicial complexes. *Proceedings - 18th IEEE International Conference on Machine Learning and Applications, ICMLA 2019*, 1083–1090.

Eckmann, B. 1944. Harmonische funktionen und randwertaufgaben in einem komplex. *Commentarii Mathematici Helvetici*, 17(1): 240–255.

Edelsbrunner, H. 2010. Alpha shapes—a survey. *Tessellations in the Sciences*, 27: 1–25.

Edelsbrunner, H.; and Harer, J. 2010. *Computational topology: an introduction*. American Mathematical Soc.

Erickson, J.; and Whittlesey, K. 2005. Greedy optimal homotopy and homology generators. In *SODA*, volume 5, 1038–1046.

Farber, M. 2018. Configuration spaces and robot motion planning algorithms. In *Combinatorial And Toric Homotopy: Introductory Lectures*, 263–303. World Scientific.

Feng, Y.; You, H.; Zhang, Z.; Ji, R.; and Gao, Y. 2019. Hypergraph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3558–3565.

Ghrist, R.; and Muhammad, A. 2005. Coverage and hole-detection in sensor networks via homology. *2005 4th International Symposium on Information Processing in Sensor Networks, IPSN 2005*, 2005(1): 254–260.

Hajij, M.; Istvan, K.; and Zamzmi, G. 2021. Cell Complex Neural Networks. arXiv:2010.00743.

Hamilton, W. L.; Ying, R.; and Leskovec, J. 2017. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*.

Hansen, J.; and Gebhart, T. 2020. Sheaf Neural Networks. *arXiv preprint arXiv:2012.06333*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

Hensel, F.; Moor, M.; and Rieck, B. 2021. A Survey of Topological Machine Learning Methods. *Frontiers in Artificial Intelligence*, 4: 52.

Hofer, C. D.; Kwitt, R.; and Niethammer, M. 2019. Learning Representations of Persistence Barcodes. *Journal of Machine Learning Research*, 20(126): 1–45.

Horak, D.; and Jost, J. 2013. Spectra of combinatorial Laplace operators on simplicial complexes. *Advances in Mathematics*, 244: 303–336.

Klicpera, J.; Weißenberger, S.; and Günnemann, S. 2019. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 32: 13354–13366.

Love, E. R.; Filippenko, B.; Maroulas, V.; and Carlsson, G. 2021. Topological Deep Learning. *arXiv preprint arXiv:2101.05778*.

Montúfar, G.; Otter, N.; and Wang, Y. 2020. Can neural networks learn persistent homology features? *arXiv preprint arXiv:2011.14688*.

Muhammad, A.; and Egerstedt, M. 2006. Control using higher order Laplacians in network topologies. In *Proc. of 17th International Symposium on Mathematical Theory of Networks and Systems*, 1024–1038. Citeseer.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in PyTorch.

Roddenberry, T. M.; and Segarra, S. 2019. HodgeNet: Graph neural networks for edge data. In *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, 220–224. IEEE.

Schaub, M. T.; and Segarra, S. 2018. Flow smoothing and denoising: Graph signal processing in the edge-space. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 735–739. IEEE.

Spielman, D. A.; and Srivastava, N. 2011. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6): 1913–1926.

Tahbaz-Salehi, A.; and Jadbabaie, A. 2010. Distributed Coverage Verification in Sensor Networks Without Location Information. *IEEE Transactions on Automatic Control*, 55(8): 1837–1849.

The GUDHI Project. 2021. *GUDHI User and Reference Manual*. GUDHI Editorial Board, 3.4.1 edition.

Townsend, J.; Micucci, C. P.; Hymel, J. H.; Maroulas, V.; and Vogiatzis, K. D. 2020. Representation of molecular structures with persistent homology for machine learning applications in chemistry. *Nature communications*, 11(1): 1–9.

Wang, M.; Zheng, D.; Ye, Z.; Gan, Q.; Li, M.; Song, X.; Zhou, J.; Ma, C.; Yu, L.; Gai, Y.; Xiao, T.; He, T.; Karypis, G.; Li, J.; and Zhang, Z. 2019. Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks. *arXiv preprint arXiv:1909.01315*.

Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; and Weinberger, K. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*, 6861–6871. PMLR.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

You, J.; Ma, X.; Ding, Y.; Kochenderfer, M. J.; and Leskovec, J. 2020. Handling Missing Data with Graph Representation Learning. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 19075–19087. Curran Associates, Inc.