arXiv:2302.09826v1 [cs.LG] 20 Feb 2023

On the Expressivity of Persistent Homology in Graph Learning

Bastian Rieck[1, 2]

[1]Helmholtz Munich
[2]Technical University of Munich

Abstract

Persistent homology, a technique from computational topology, has recently shown strong empirical performance in the context of graph classification. Being able to capture long range graph properties via higher-order topological features, such as cycles of arbitrary length, in combination with multi-scale topological descriptors, has improved predictive performance for data sets with prominent topological structures, such as molecules. At the same time, the *theoretical properties* of persistent homology have not been formally assessed in this context. This paper intends to bridge the gap between computational topology and graph machine learning by providing a brief introduction to persistent homology in the context of graphs, as well as a theoretical discussion and empirical analysis of its expressivity for graph learning tasks.

1 Introduction

Graph learning is a highly-active research domain in machine learning, fuelled in large parts by the *geometric deep learning* paradigm as well as the resurgence of new neural network architectures for handling graph data. Methods from computational topology, by contrast, have not yet been applied in this domain at large scales. Even though a large amount of prior work employs topological features to solve graph learning tasks [Car+20; CCG21; Hof+17; Hof+20; HKN19; Hor+22; RBB19; ZW19; Zha+20], a formal investigation relating expressivity in graph learning and topological machine learning is still lacking. We believe that this is largely driven by an issue of communication between the communities. This article serves to provide an introduction to topological methods for graph learning, while also proving new results about the *expressivity* of topological methods in this domain.

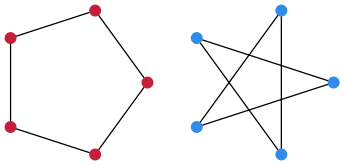


FIGURE 1: Two isomorphic graphs. The isomorphism can be understood as a relabelling, a shift of node identities, or, most intuitively, as being able to draw both graphs using a single line, thus highlighting the underlying *Hamiltonian cycle*.

Contributions. The main contribution of this paper is to provide a full characterisation of the expressivity of *persistent homology* in terms of the Weisfeiler–Leman hierarchy. We show that persistent homology is *at least as expressive* as a corresponding Weisfeiler–Leman test for graph isomorphism. Moreover, we show that there exist graphs that cannot be distinguished using k-FWL, the *folklore Weisfeiler–Leman algorithm* [Mor+21], for a specific number of iterations k but that can be distinguished by persistent homology (with or without access to k-cliques in the graph). Along the way, we also prove new properties of filtrations, hinting at their ability to capture information about graph substructures. Our theoretical expressivity discussions are complemented by an experimental suite that highlights the capabilities of different filtrations when it comes to distinguishing certain types of graphs.

Guide for readers. Section 2 briefly summarises the main concepts in graph learning. It should be accessible and informative to all readers. Section 3, by contrast, may well be skipped by readers that are already well versed in computational topology. New theoretical results are marked in the left margin. Section 4 outlines some advantageous properties of filtrations in the context of graph learning, while Section 5 and Section 6 discuss the expressivity of persistent homology with respect to the Weisfeiler–Leman hierarchy of graph isomorphism tests. The latter of these two sections contains novel proofs that extend our previous work [Hor+22].

2 Background & Notation

We will deal with undirected graphs in this paper. We consider an undirected graph G to be a pair $G = (V, E)$ of finite sets of n *vertices* and m *edges*, with $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$. We will often refer to an edge using tuple notation, with the understanding that (u, v) and (v, u) refer to the same edge. Moreover, we will say that a graph G is labelled if there is a function $\mathrm{l}\colon V \cup E \to \Sigma$ that assigns a vertex or an edge to

a label. Without loss of generality, we assume that $\Sigma \subset \mathbb{N}$.[1] We denote the space of all labelled graphs by $\mathcal{G}$. Two graphs $G = (V, E)$ and $G' = (V', E')$ are *isomorphic*, denoted by $G \simeq G'$, if there is a bijective function $\varphi \colon V \to V'$ that preserves adjacency, i.e. $(u, v) \in E$ if and only if $(\varphi(u), \varphi(v)) \in E'$. If G and G' are labelled, we also require φ to satisfy $\mathrm{l}(v) = \mathrm{l}(\varphi(v))$ for all $v \in V$ and, similarly, $\mathrm{l}(u, v) = \mathrm{l}(\varphi(u), \varphi(v))$ for all $(u, v) \in E$. The isomorphism φ is thus preserving edges and connectivity. Since φ is bijective, it has an inverse function, which we will denote by φ^{-1}. Fig. 1 depicts two isomorphic graphs. The problem of figuring out whether two graphs are isomorphic or not is referred to as the *graph isomorphism problem*. Presently, there is no known algorithm that solves this problem in polynomial time—efficient algorithms exist only for special families of graphs [Col81; Kel57]. Hence, all subsequently-discussed graph isomorphism tests are perforce limited with respect to their expressivity, i.e. there exist classes of non-isomorphic graphs that they cannot distinguish. In the context of graph isomorphism tests, we will often require the definition of a *multiset*, which is a set whose elements are included with multiplicities. We will denote such a multiset by $\{\!\{\}\!\}$.

Equivariance. Given two graphs G and G' with n nodes, let S_n refer to the permutation group on n letters. An element $\sigma \in S_n$ acts on a graph by permuting the order of vertices, and, by transitivity, the edges. If $G \simeq G'$, there is a permutation $\sigma \in S_n$ such that $\sigma(G) = G'$. Under the assumption that all graphs have the same number of vertices n, we call a function $f \colon V \to \mathbb{R}^n$ *permutation-equivariant* if $f(\sigma(G)) = \sigma(f(G))$ for a permutation σ, with the understanding that σ acts on $\mathbb{R}^n$ by permuting the order of coordinates of the vector.[2] The output of a permutation-equivariant function thus changes with the permutation in a predictable manner.

3 Topological Features of Graphs

Starting our investigation from first principles, we ask ourselves what types of topological features we can prescribe to graphs. The simplest answer—for now—will be to make use of *connected components* and *cycles*. If we assume that an input graph is planar, i.e. it affords an embedding in the plane such that there are no overlaps between edges, we have a formula that captures some of the structural properties of a graph via

$$V - E + F = 2, \tag{1}$$

[1] This is also for notational convenience. In practice, labels may be strings or more complex objects; as long as we can enumerate them, the natural numbers $\mathbb{N}$ are sufficient as the range of the label function. In practice, graphs can also have more complicated *attributes* assigned to their vertices or edges. Of particular importance in the application domains are continuous labels, such as measurements of a quantity. While we consider these cases to be of high importance, we want to focus on simple labelled graphs in this paper.

[2] We only consider functions operating on the vertices of the graph, but the concept can extended to edges as well.

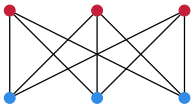


FIGURE 2: A depiction of the *utilities problem*. Three houses (top row) are supposed to be connected to three different utilities (bottom row; commonly referred to as water, gas, and electricity) without the connection lines crossing one another. The figure shows an incorrect solution in which crossings occur. Using Eq. (1), we can prove that no such solution can exist in the plane. To see this, we set $V = 3$ and $E = 9$, as required by the utilities problem. Eq. (1) now states that $F = 8$. However, we can see that F can be *at most* half the number of edges, i.e. $F \leq 4$. This is true because the edges of any face generated by a planar embedding have to alternate between the houses and utilities, respectively, meaning that every face has at least 4 edges, and each edge belongs to exactly 2 faces.

where F counts the number of faces (including the unbounded one) of the planar embedding. This remarkable formula, whose formulation and proof is originally due to Euler,[3] can be used to assess the feasibility of certain graph-theoretical problems—see Fig. 2 for an example—but its practical utility is limited. However, if we specifically focus on connected components and cycles, it turns out that more complicated assessments are possible. In fact, we only need to know the number of connected components to calculate the number of cycles in graph! Formally, referring to the number of connected components as β_0 and the number of cycles as β_1, we have

$$\beta_1 = m + \beta_0 - n, \tag{2}$$

where n and m denote the number of vertices and edges, respectively. The two quantities in this formula are also known as the first two *Betti numbers* of a graph, with β_1 also known as the *cyclomatic number* or the *circuit rank* [Ber01, pp. 27–30].

Example. The graphs in Fig. 1 have one connected component and $m = n = 5$. According to Eq. (2), we thus have $\beta_1 = 5 + 1 - 5 = 1$. While this is readily seen to be the case in the left graph, the formula helps us 'disentangle' the right graph rapidly—counting connected components in an embedding of a graph is typically easier than counting cycles. This example also hints at Betti numbers being an *isomorphism invariant*; we will discuss this later on in more detail.

Counting connected components. Since we deal with higher-order topological features later on, and such features afford a substantially less intuitive grasp, we want to briefly comment on how to obtain β_0, the number of connected components. As with many problems in computer science, this procedure turns out to be simple if we pick our data structures correctly. Here, we need a *union–find* data structure, also known as a disjoint set forest. This data structure is built on the vertices of a graph and affords

[3] See Wilson [Wil96, p. 66] for a simple proof by induction.

ALGORITHM 1 Using associative arrays to find connected components

```
function get_connected_components(V, E)
    UF ← {}
    for v ∈ V do
        UF[v] ← v
    end for
    for e = (v, w) ∈ E do
        merge(UF, v, w)
    end for
    return {v | UF[v] = v}
end function

function merge(UF, v, w)
    if UF[v] ≠ UF[w] then
        UF[v] ← w
    end if
end function
```

two operations, viz. `union` (or `merge`) and `find`. The `merge` operation assigns two vertices to the same connected component, while the `find` operation returns the current connected component of a vertex. Building such a data structure is reasonably easy in programming languages like `Python`, which offer *associative arrays*. Algorithm 1 shows one particular pseudo-code implementation of a simple union–find data structure. The pseudo-code assumes that all operations are changing objects 'in place.' Notice that the `find` operation is implemented implicitly via a lookup in the `merge` function. A proper object-oriented implementation of a union–find data structure should have these two operations in its public interface.

3.1 Simplicial Homology

The Betti numbers of a graph are actually a special instance of a more generic concept, known as *simplicial homology*. We will see that under this framework, the Betti numbers are the ranks of the zeroth and first homology group, respectively. Simplicial homology is not required in order to understand most of the results of this paper, but an appreciation for some of the concepts will be helpful in understanding connections to other concepts. We try to provide a self-contained introduction to the most relevant concepts and refer to the textbook by Munkres [Mun84] for a more in-depth exposition of these concepts. We start by introducing the central object of algebraic topology—the *simplicial complex*.[4]

[4]Technically, we will be working with *abstract simplicial complexes*. A definition of a simplicial complex in terms of convex subsets is also possible, but this necessitates understanding certain nuances that are irrelevant to this paper.

Definition 1 (Simplicial complex)**.** *A* simplicial complex K *is a system of sets that is closed under the subset operation. Thus, for any* $\sigma \in \mathrm{K}$ *and* $\tau \subseteq \sigma$*, we have* $\tau \in \mathrm{K}$*. An element* $\sigma \in \mathrm{K}$ *with* $|\sigma| = k+1$ *is also referred to as a* k*-simplex. We also express this by writing* $\dim \sigma = k$*. Moreover, if* k *is maximal among all the simplices of* K*, we say that the* K *is a* k*-dimensional simplicial complex.*

Note that there is an unfortunate shift in dimensions: a k-simplex has indeed $k+1$ elements. This convention makes sense when we relate it to the concept of *dimension*. A 0-simplex, i.e. a point or a vertex, should be assigned a dimension of 0. The reader should thus mentally equate the dimension of a simplex with its dimension. The text will aim to quell any confusion about such shifts. The quintessential example of a simplicial complex is a graph $G = (V, E)$. Setting $\mathrm{K} := V \cup E$, we obtain a 1-dimensional simplicial complex. We may calculate additional types of simplicial complexes from a graph, for instance by *expanding* each $(k+1)$-clique into a k-simplex [HMR09; Rie+18].

The simplicial complex on its own is only a set system; to perform calculations with this type of data structure, we need to imbue it with additional operations. One of the most common operations involves defining homomorphisms between the subsets of a simplicial complex K.

Definition 2 (Chain group of a simplicial complex)**.** *Given a simplicial complex* K*, the vector space generated over* $\mathbb{Z}_2$ *coefficients whose elements are the* k*-simplices of* K *is called the* kth chain group*, denoted by* $C_k(\mathrm{K})$*. The elements of a chain group are also referred to as* simplicial chains*.*

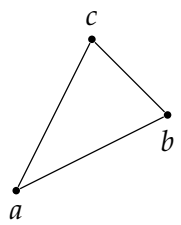
Elements of the chain group are thus sums of simplices of a compatible dimension. For instance, we may write the sum of all edges of a graph to obtain a valid simplicial chain. Operating over $\mathbb{Z}_2$ coefficients means that $\sigma + \sigma = 0$, the empty chain, for all $\sigma \in \mathrm{K}$.[5] Simplicial chains permit us to define homomorphisms between chain groups, which will ultimately permit us to treat topological questions with tools of linear algebra.

Definition 3 (Boundary homomorphism)**.** *Given* $\sigma = (v_0, \ldots, v_k) \in \mathrm{K}$*, we define the* kth boundary homomorphism $\partial_k \colon C_k(\mathrm{K}) \to C_{k-1}(\mathrm{K})$ *as*

$$\partial_k(\sigma) := \sum_{i=0}^{k} (v_0, \ldots, v_{i-1}, v_{i+1}, \ldots, v_k), \tag{3}$$

i.e. a sum of simplices with the i*th entry—vertex—of the simplex missing, respectively.*

[5] Readers familiar with algebraic topology will recognise $\mathbb{Z}_2$ as a deliberate choice of *coefficient field* for the subsequent calculations. Other choices are possible, but the computational topology community predominantly uses $\mathbb{Z}_2$ coefficients in practice, with very few exceptions [Gar+22]. However, all the proofs and concepts introduced in this paper apply, *mutatis mutandis*, for other coefficient sets as well.



The triangle is a simple simplicial complex, consisting of one 2-simplex, three 1-simplices and three 0-simplices, respectively. The boundary of the 2-simplex is non-zero: we have $\partial_2\{a,b,c\} = \{b,c\} + \{a,c\} + \{a,b\}$. The set of edges, on the other hand, does not have a boundary, i.e. $\partial_1(\{b,c\} + \{a,c\} + \{a,b\}) = \{c\} + \{b\} + \{c\} + \{a\} + \{b\} + \{a\} = 0$, because the simplices cancel each other out.

FIGURE 3: Calculating the boundaries of a 2-simplex and the boundary of a simplicial chain consisting of 1-simplices. Notice that the boundary of a boundary is always zero. This is a fundamental property of persistent homology. The figure is slightly adapted from Rieck [Rie17].

It is sufficient to define ∂_k on individual simplices; since it is a homomorphism, it extends to arbitrary simplicial chains. Fig. 3 shows an example of this calculation. The boundary operator already assigns some algebraic structure to K, but it turns out that we can use it to assign a set of groups to the simplicial complex.

Definition 4 (Homology group)**.** *We define the* kth homology group *of a simplicial complex* K *as*

$$\mathrm{H}_k(\mathrm{K}) := \ker \partial_k / \operatorname{im} \partial_{k+1}, \tag{4}$$

i.e. a quotient *group that we obtain from the two subgroups* $\ker \partial_k$ *and* $\operatorname{im} \partial_{k+1}$ *of* C_k*.*

The kth homology group of a simplicial complex contains its k-dimensional topological features in the form of *equivalence classes* of simplicial chains, also known as *homology classes*. This rather abstract definition is best understood by an additional simplification step that involves calculating the *rank* of a homology group.

Definition 5 (Betti number)**.** *The rank of the kth homology group* $\mathrm{H}_k(\mathrm{K})$ *is known as the* kth Betti number*, denoted by* β_k*.*

Despite the rank being a rather coarse summary, Betti numbers turn out to be of immense utility in comparing different simplicial complexes. We may even reproduce Eq. (2) by noting that the *Euler characteristic* $\chi(\mathrm{K}) := \sum_i (-1)^i \left|\{\sigma \mid \dim \sigma = i\}\right|$ can also be expressed as a sum of alternating Betti numbers, i.e. $\chi(\mathrm{K}) := \sum_i (-1)^i \beta_i$. For a proof of this surprising fact, see e.g. Munkres [Mun84, p. 124]. Using this equivalence, we see that we can calculate β_1 by reshuffling some of the terms, thus also explaining why Eq. (2) exhibits alternating signs.

At this point, we have introduced a large amount of algebraic machinery. Changing our focus back to graphs, we may reap some advantageous properties by noting that homology groups are somewhat preserved under graph isomorphism.[6]

☞ **Lemma 1.** *Let* G, G' *be two isomorphic graphs with* $\varphi\colon G \to G'$ *their corresponding isomorphism. Then the homology groups of* G *and* G' *are isomorphic, i.e.* $\mathrm{H}_p(G) \simeq \mathrm{H}_p(G')$ *for all* p*.*

[6]The reader well-versed in algebraic topology may be aware of this property directly, but we find it useful to mention this fact briefly.

Proof. This is a consequence of the *functoriality* of homology, which implies that any function $f\colon G \to G'$ induces a function $f_p\colon \mathrm{H}_p(G) \to \mathrm{H}_p(G')$ on the homology level. This function has the property that it composes in a natural manner with any other function $g\colon G \to G'$, namely $(f \circ g)_p = f_p \circ g_p$. In other words, functoriality implies that the evaluation order does not matter, or, intuitively, that the induced function commutes with function evaluation. In our case, setting $f = \varphi$ and $g = \varphi^{-1}$, this implies that $(\varphi \circ \varphi^{-1})_p = \varphi_p \circ \varphi_p^{-1}$, i.e. $\mathrm{id}_p = \varphi_p \circ \varphi_p^{-1}$, with id_p representing the identity function induced on the homology groups. The same calculations with f, g swapped show that the homology groups are isomorphic, with the isomorphism induced by φ, the isomorphism between G and G'. ■

As a direct corollary, the Betti numbers of G and G' do not change, and in fact, a similar property holds for isomorphic simplicial complexes. This may be seen as a hint about the popularity of simplicial homology in algebraic topology: the framework leads directly to characteristic descriptions that remain invariant under (graph) isomorphism.

3.2 Persistent Homology

Because of their conceptual simplicity—their calculation in low dimensions only involves knowledge about the connected components of a graph—Betti numbers are somewhat limited in their expressivity. Taking any graph $G = (V, E)$, even the addition of a single edge to G will change its Betti numbers, either by merging two connected components (thus decreasing β_0) or by creating an additional cycle (thus increasing β_1). This is a direct consequence of Eq. (2), which, effectively, states that the insertion of a new edge $e = (u, v)$ with $u, v \in V$ either causes β_1 to increase by 1 because m changes, or remain the same in case the number of connected components β_0 changes. However, a single edge may only merge two connected components into one, so β_0 may also at most decrease by 1. This indicates that Betti numbers are too coarse to be practically useful in large-scale graph analysis. It is possible to turn Betti numbers into a *multi-scale descriptor* of a graph. This requires certain modifications to the previously-introduced concepts. Similar to Section 3.1, we will formulate everything in terms of simplicial complexes, again pointing out that this results in a more general description.

Definition 6 (Filtration)**.** *Given a simplicial complex* K*, we call a sequence of simplicial complexes* filtration *if it affords a nesting property of the form*

$$\emptyset = \mathrm{K}_0 \subseteq \mathrm{K}_1 \subseteq \cdots \subseteq \mathrm{K}_{m-1} \subseteq \mathrm{K}_m = \mathrm{K}. \tag{5}$$

Since each element of this sequence is a valid simplicial complex, we can also think of this construction as 'growing' K *by adding simplices one after the other.*

Filtrations arise naturally when building simplicial complexes from point cloud data, but even in the context of graphs, we can imagine filtrations as *filtering* a graph based on some type of data, or function, assigned to its vertices. For instance, we may build a

filtration of a graph based on the degree of its vertices, defining K_i to be the subgraph consisting of all vertices satisfying the degree condition, plus all edges whose endpoints satisfy it, i.e.

$$\mathrm{K}_i := \{v \in V \mid \deg(v) \leq i\} \cup \{\{u, v\} \in E \mid \deg(u) \leq i \wedge \deg(v) \leq i\}. \tag{6}$$

Notice that we could also express the second condition more compactly by assigning to each 1-simplex (each edge) the *maximum* of the weight of its vertices. This construction is sometimes also referred to as a *lower-star filtration* since it extends a node-level function to higher-order simplices [DW22]. Not all filtrations have to be defined on the vertex level; as long as each edge in the filtration is preceded by its vertices, we can also build valid filtrations from functions that are primarily defined on edges.[7]

Setting aside further discussions about how to obtain filtrations for now, filtrations are compatible with the simplicial homology framework introduced above. The boundary operators $\partial(\cdot)$, together with the inclusion homomorphism between consecutive simplicial complexes, induce a homomorphism between corresponding homology groups of any filtration of m simplicial complexes. Given $i \leq j$, we write $\iota^{i,j} \colon \mathrm{H}_k(\mathrm{K}_i) \to \mathrm{H}_k(\mathrm{K}_j)$ to denote this homomorphism. This construction yields a sequence of homology groups

$$0 = \mathrm{H}_k(\mathrm{K}_0) \xrightarrow{\iota_k^{0,1}} \mathrm{H}_k(\mathrm{K}_1) \xrightarrow{\iota_k^{1,2}} \dots \xrightarrow{\iota_k^{m-2,m-1}} \mathrm{H}_k(\mathrm{K}_{m-1}) \xrightarrow{\iota_k^{m-1,m}} \mathrm{H}_k(\mathrm{K}_m) = \mathrm{H}_k(\mathrm{K}) \tag{7}$$

for every dimension k. We then define the *kth persistent homology group* as

$$\mathrm{H}_d^{i,j} := \ker \partial_k(\mathrm{K}_i) / \big(\operatorname{im} \partial_{k+1}(\mathrm{K}_j) \cap \ker \partial_k(\mathrm{K}_i)\big), \tag{8}$$

containing all topological features—homology classes—created in K_i that still exist in K_j. Following the definition of the ordinary Betti number, we then define the *kth persistent Betti number* to be the rank of this group, leading to $\beta_k^{i,j} := \operatorname{rank} \mathrm{H}_k^{i,j}$. It should be noted that this type of construction makes use of numerous deep mathematical concepts; for the sake of an expository article, we heavily summarise and compress everything to the most pertinent results.

The appeal of persistent homology can be seen when we start to make use of the features it captures. If we assume that our filtration is associated with a set of values $a_0 \leq a_1 \leq \dots \leq a_{m-1} \leq a_m$, such as the function values on the vertices, we can calculate *persistence diagrams*, i.e. simple topological feature descriptors.

Definition 7 (Persistence diagram)**.** *The* k-dimensional persistence diagram *of a filtration is the multiset of points in* $\mathbb{R}^2$ *that, for each pair* i, j *with* $i \leq j$*, stores the tuple* (a_i, a_j) *with multiplicity*

$$\mu_{i,j}^{(k)} := \left(\beta_k^{i,j-1} - \beta_k^{i,j}\right) - \left(\beta_k^{i-1,j-1} - \beta_k^{i-1,j}\right). \tag{9}$$

[7] In Section 7, we will make use of a filtration defined on edge-based curvature values.

(a) Example graph (b) $f^{-1}((-\infty, 0])$ (c) $f^{-1}((-\infty, 1])$ (d) $f^{-1}((-\infty, 2])$ (e) $f^{-1}((-\infty, 3])$

FIGURE 4: An example graph and three different steps of a degree-based filtration. The respective caption indicates the pre-image of the respective filtration function.

We will also assign a multiplicity to essential topological features *of the simplicial complex, setting*

$$\mu_{i,\infty}^{(k)} := \beta_k^{i,m} - \beta_k^{i-1,m}, \tag{10}$$

which denotes all features that are still present in the last simplicial complex of the filtration, i.e. in $\mathrm{K}_m = \mathrm{K}$. *The persistence diagram thus contain all the information carried by Betti numbers.*

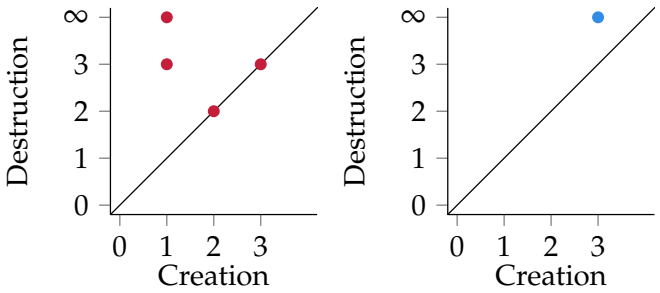
Persistence diagrams summarise the topological activity of a filtration. Given a persistence diagram $\mathcal{D}$, for any tuple (a_i, a_j), the quantity $\|a_j - a_i\|$ is called the *persistence* of the respective topological feature. Persistence indicates whether a feature, created in some simplicial complex during the filtration, is prominent or not. This notion was originally introduced by Edelsbrunner, Letscher, and Zomorodian [ELZ02] to analyse the relevance of topological features of a distance function; the terminology is supposed to indicate the prominence of a topological feature. Features with a high persistence are commonly taken to be relevant, whereas features with a low persistence used to be considered as noise; this assumption is changing as, depending on the filtration, low persistence may also just imply 'low reliability.' [Ben+16]

Persistence diagrams can be endowed with different metrics and kernels [Kwi+15; Rei+15], and it is known that the space of persistence diagrams is an Alexandrov space with curvature bounded from below [Tur+14]. The most common metric to compare two persistence diagrams is the *bottleneck distance*, defined as

$$\mathrm{d_B}\big(\mathcal{D}, \mathcal{D}'\big) := \inf_{\eta\colon \mathcal{D} \to \mathcal{D}'} \sup_{x \in \mathcal{D}D} \|x - \eta(x)\|_\infty, \tag{11}$$

where η ranges over all bijections between the two persistence diagrams. Eq. (11) is solved using optimal transport; different cardinalities are handled by permitting points in one diagram to be transported to their corresponding projection on the diagonal. Another metric is the *Wasserstein distance*, in which the sup calculation is replaced by a weighted sum over all distances between points in a diagram.

Example. Fig. 4 depicts an example filtration for a simple graph. Following Eq. (3), the degree of each vertex is used to filter the graph. The calculation of persistent homology along this filtration involves counting the connected components and cycles. We note that these features can only change whenever the filtration function changes. The



(a) $\mathcal{D}_0$ (b) $\mathcal{D}_1$

FIGURE 5: Persistence diagrams corresponding to the filtration depicted in Fig. 4. Notice that the *multiplicity* of points in the diagram is larger than one. For instance, the point $(2,2)$ has multiplicity 5, because the filtration step merges 7 new vertices into 2 connected components, respectively. The point $(1,3)$, by contrast, has multiplicity 2, corresponding to the vertices with degree 1, which only get merged into a larger connected component later on. We also show the *essential features* of the persistence diagram, denoting them using an ∞ symbol. Here, we have multiplicities of 1 and 2, respectively, denoting the single connected component of the graph and its two cycles. The persistence diagram, with the multiplicities of points, thus carries *at least* as much information as the Betti numbers of the graph on their own.

critical points of the degree filtration are thus the unique degrees of vertices occurring in the graph. Fig. 5 shows the persistence diagrams arising from the filtration. The more complex structure of persistence diagrams already hints at their capabilities in providing expressive graph descriptors.

Computations in practice. When dealing with more complex data, persistent homology is typically calculated using matrix reduction algorithms, once again outlining the strong relation between this concept and linear algebra. Efficient algorithms and their behaviour under various conditions are still an ongoing topic of research [Bau+22], but details are beyond the scope of this paper. We refer readers to Otter et al. [Ott+17] for a comprehensive introduction of computational strategies. In the general point cloud setting, a paper by Bauer [Bau21] provides a highly-efficient reference implementation while also stating details on combinatorial optimisation strategies.

4 Properties of Filtrations

Before we discuss how to create specific filtrations that are useful for graph learning tasks, we first discuss some of their general properties. Before discussing our new results, we briefly expand on the *stability properties* of filtrations. Given two filtrations f, g of the same graph, a seminal result by Cohen-Steiner, Edelsbrunner, and Harer [CEH07] proves the following bound.

Theorem 1 (Bottleneck stability)**.** *Let f, g refer to filtrations of a graph G, and let $\mathcal{D}_f$ and $\mathcal{D}_g$ denote their respective persistence diagrams. The* bottleneck distance *distance is upper-bounded by*

$$\mathrm{d_B}(\mathcal{D}_f, \mathcal{D}_g) \leq ||f - g||_\infty, \tag{12}$$

where $\|\cdot\|_\infty$ refers to the Hausdorff distance between the two functions.

An extension of this theorem, with $\mathrm{d_B}$ being replaced by the Wasserstein distance, shows that persistence diagrams are also stable in the Lipschitz sense [Coh+10]. These stability properties are remarkable because they link a topological quantity with a geometrical one, thus underscoring how persistent homology itself incorporates both geometrical and topological aspects of input data. Stability theorems are a crucial aspect of research in computational topology; readers are invited to read a recent work that provides a concise exposition and deposes of some erroneous assumptions in this context [ŠT20].

We continue our enumeration of filtration properties by proving that *any* filtration induced by an equivariant function is well-behaved under graph isomorphism.

☞ **Lemma 2.** *Let $G \simeq G'$ be two isomorphic graphs, and φ be the respective isomorphism. For any equivariant filtration f, the corresponding persistence diagrams are equal.*

Proof. Since $G \simeq G'$, there exists a permutation σ such that $\sigma(G) = G'$. Moreover, since f is equivariant, the image of G under f will just be permutation of vectorial coordinates. This, in turn, implies that all weights of vertices and edges of G are permuted. This does not change the calculation of persistent homology, though, because the *ordering* given by the filtration does not change. The persistence diagrams of G and G' thus coincide. ■

This lemma is again a specific consequence of a more general principle, viz. *functoriality*. It also shows that it is impossible to 'adversarially' pick an equivariant filtration function that leads to a non-zero topological dissimilarity between two non-isomorphic graphs. Furthermore, the lemma demonstrates that persistent homology is fundamentally compatible with equivariant function learning, pointing towards the utility of hybrid models that leverage different types of structural properties of graphs.

When working with filtrations in the subsequent proofs, it would be ideal to have filtrations that satisfy *injectivity* on the level of vertices, i.e. $f(v) \neq f(v')$ if $v \neq v'$. Such injective filtrations have the advantage of permitting gradient-based optimisation schemes [Hof+20]. The following lemma, first proved in Horn et al. [Hor+22], demonstrates that injectivity is not a strict requirement, though, as it is always possible to find an injective filtration function that is arbitrarily close (in the Hausdorff sense) to a non-injective filtration function.

Lemma 3. *For all $\epsilon > 0$ and a filtration function f defined on the vertices, i.e. $f\colon V \to \mathbb{R}^d$, there is an injective function $\tilde{f}\colon V \to \mathbb{R}^d$ such that $\|f - \tilde{f}\|_\infty < \epsilon$.*

Proof. Let $V = \{v_1, \ldots, v_n\}$ be the vertices of a graph and $\operatorname{im} f = \{u_1, \ldots, u_m\}$ be their images under f. Since f is not injective, we have $m < n$. We resolve non-injective vertex pairs iteratively. For $u \in \operatorname{im} f$, let $V' := \{v \in V \mid f(v) = u\}$. If V' only contains a single element, we do not have to do anything. Otherwise, for each $v' \in V'$, pick a new value from $\mathrm{B}_\epsilon(u) \setminus \operatorname{im} f$, where $\mathrm{B}_r(x) \subset \mathbb{R}^d$ refers to the open ball of radius r around a point x (for $d = 1$, this becomes an open interval in $\mathbb{R}$, but the same reasoning applies in higher dimensions). Since we only ever remove a finite number of points, such a new value always exists, and we can modify $\operatorname{im} f$ accordingly. The number of vertex pairs for which f is non-injective decreases by at least one in every iteration, hence after a finite number of iterations, we have modified f to obtain $\tilde{f}$, an *injective* approximation to f. By always picking new values from balls of radius ϵ, we ensure that $\|f - \tilde{f}\|_\infty < \epsilon$, as required. ■

☞ **Theorem 2.** *Given* any *vertex-based filtration f of a graph G with a single connected component, we can upper-bound* $\operatorname{diam}(G)$*, the diameter of G, based on $\mathcal{D}_0$, the resulting persistence diagram in dimension* 0.

Proof. We can provide a procedure to obtain an upper bound d of $\operatorname{diam}(G)$ alongside the calculation of $\mathcal{D}_0$. To this end, we set $d = 0$. While calculating $\mathcal{D}_0$ with Algorithm 1, we check for each edge whether it is a creator or destroyer. If the edge is a creator, we stop. Else, we increase d by one. This procedure works because $\operatorname{diam}(G)$ is the diameter of the minimum spanning tree of G. Our estimate d counts the number of edges in such a tree. We thus have $\operatorname{diam}(G) \leq d$. The bound is tight for some graphs, such as line graphs. ■

Theorem 2 indicates that persistent homology captures more than 'just' topological information about a graph. Similar to Vietoris–Rips filtrations, a topological construction for general metric spaces, persistence diagrams permit inferring additional properties about the input data [Bub+20; LMO20; TMO22]. This indicates that a topological perspective can complement and enrich graph learning tasks.

5 The One-Dimensional Weisfeiler–Leman Test

Having discussed the properties of specific filtrations, we now analyse the expressivity of persistent homology in the context of the Weisfeiler–Leman hierarchy of graph isomorphism tests. 1-WL, also known as *colour refinement*, constitutes a simple method for addressing the graph isomorphism problem. It is one of the backbones of modern graph learning research; readers are referred to Morris et al. [Mor+21] for a comprehensive survey of 1-WL, its higher-order variants, and their relation to machine learning research. We follow the terminology of this article and briefly introduce all relevant concepts.

Formally, 1-WL proceeds by iteratively calculating a node colouring function $C_i^{(1)} \colon V \to \mathbb{N}$. The output of this function depends on the neighbours of a given note. For a vertex v at iteration $i > 0$, we have

$$C_i^{(1)}(v) := \mathsf{RELABEL}\left(\left(C_{i-1}^{(1)}(v), \left\{\!\!\left\{ C_i^{(1)}(u) \mid u \in \mathcal{N}(v) \right\}\!\!\right\}\right)\right), \tag{13}$$

where RELABEL refers to an injective function that maps the tuple of colours to a unique colour, i.e. a unique number. The algorithm is initialised by either using existing labels or the degree of vertices.[8] After a finite number of steps, the colour assignments generated using Eq. (13) stabilise. If two graphs give rise to different colour sequences, the graphs are guaranteed to be non-isomorphic. The 1-WL test is computationally easy and, somewhat surprisingly, provides an upper-bound for the expressivity of many graph neural network (GNN) architectures [Mor+19; Xu+19]. In other words, if 1-WL cannot distinguish two non-isomorphic graphs, GNNs will also not be able to distinguish them.

We already showed that *any* 1-WL colouring can be reproduced by creating a 'bespoke' filtration [Hor+22]. We reiterate this argument briefly since it will provide a segue into our new result for k dimensions.

Theorem 3. *Given* 1*-WL colourings of two graphs G and G' that are different, there exists a filtration of G and G' such that their persistence diagrams in dimension* 0 *are also different.*

Proof. Since the colourings are different, there is an iteration h of 1-WL such that the label sequences of G and G' are different. We thus have at least one colour—equivalently, one label—whose count is different. Let $\mathcal{L}^{(h)} := \{l_1, l_2, \dots\}$ be an enumeration of the finitely many hashed labels at iteration h. We can build a filtration function f by assigning a vertex v with label l_i to its index, i.e. $f(v) := i$, and setting $f(v, w) := \max\{f(v), f(w)\}$ for an edge (v, w). The resulting 0-dimensional persistence diagrams for G and G', denoted by $\mathcal{D}_0$ and $\mathcal{D}_0'$, respectively, now contain tuples of the form (i, j). Moreover, each vertex is guaranteed to give rise to *exactly* one such pair since each vertex creates a connected component in 0-dimensional persistent homology. Letting $\mu^{(i,j)}(\mathcal{D}_0)$ refer to the multiplicity of a tuple in $\mathcal{D}_0$, we know that, since the label count is different, there is *at least* one tuple (k, l) with $\mu^{(k,l)}(\mathcal{D}_0) \neq \mu^{(k,l)}(\mathcal{D}_0')$. Hence, $\mathcal{D}_0 \neq \mathcal{D}_0'$. ■

The implication of Theorem 3 is that persistent homology is *at least as expressive* as 1-WL because there is a filtration that distinguishes all the graphs 1-WL can distinguish. Since 1-WL is oblivious to certain topological structures such as cycles [Arv+15], the existence of graphs with different Betti number counts proves that persistent homology is *strictly more expressive* than 1-WL. For example, consider a graph consisting of the union of two triangles, i.e. . This graph has $\beta_0 = \beta_1 = 2$ since it consists of two connected components and two cycles. If we change the connectivity slightly to obtain

[8] Note that Eq. (13) does not recognise an ordering of labels. Initialising 1-WL with a constant value thus leads to the *same* colouring—up to renaming—after the first iteration.

a hexagon, i.e. , we obtain a graph with $\beta_0 = \beta_1 = 1$. 1-WL is not able to distinguish between these graphs, but persistent homology can, since the Betti numbers of the graph are still encoded in the persistence diagram as essential features. Note that Theorem 3 does *not* apply to arbitrary filtrations since the theorem requires knowing the correct labels assigned by 1-WL. Finding filtration functions that are able to split graphs in a manner that is provably equivalent to 1-WL remains an open research question.

6 The k-Dimensional Weisfeiler–Leman Test

Since 1-WL is known to be unable to distinguish between graphs with different triangle counts or graphs with cycle information, it was generalised to include information about labelling *tuples* of k nodes (as opposed to only labelling a single node), leading to the k-FWL hierarchy. The variant we shall subsequently describe is also known as the *folklore Weisfeiler–Leman algorithm* [Mor+21]. It can be shown that there are non-isomorphic graphs that cannot be distinguished by k-FWL, but that can be distinguished by $(k+1)$-FWL.[9]

Our subsequent exposition follows Morris et al. [Mor+21]. k-FWL is based on the idea of assigning colours to *subgraphs* as opposed to assigning colours to *vertices*. To achieve this, k-FWL operates on k-tuples of vertices; for iteration $i = 0$, two tuples $\mathbf{v} = (v_1, \ldots, v_k)$ and $\mathbf{w} = (w_1, \ldots, w_k)$ are assigned the same colour if the map $v_j \mapsto w_j$ induces a homomorphism between the subgraphs induced by v and w, respectively. For subsequent iterations with $i > 0$, we relabel the tuples similar to 1-WL, i.e.

$$C_i^{(k)}(\mathbf{v}) := \mathsf{RELABEL}\Big(\Big(C_{i-1}^{(k)}(\mathbf{v}), \Big\{\!\!\Big\{ C_i^{(k)}(\phi_1(\mathbf{v}, u)), \ldots, C_i^{(k)}(\phi_k(\mathbf{v}, u)) \mid u \in \mathcal{N}(v) \Big\}\!\!\Big\}\Big)\Big), \tag{14}$$

where $\phi_j(\mathbf{v}, u) := (v_1, \ldots, v_{j-1}, u, v_{j+1}, \ldots, v_k)$ refers to the function that replaces the jth element of the k-tuple $\mathbf{v}$ with u. This induces a neighbourhood relation between tuples and just as in the case of 1-WL, we run the algorithm until the assigned colours of tuples stabilise for one graph. Similarly, if the colour sequences of two graphs differ, the graphs are non-isomorphic.

As a generalisation of Theorem 3, we can show that *any* k-FWL colouring can be reproduced with a specific filtration, thus proving that persistent homology is *at least* as expressive as k-FWL. The theorem will make use of the equivalence of k-tuples and $(k-1)$-simplices.

[9] There are also other variants, for instance the *oblivious Weisfeiler–Leman algorithm*. It slightly differs in the way tuples are being relabelled, but a paper by Grohe [Gro21] shows that the variant is essentially as powerful as k-FWL (with a minor shift in indices). The reader is referred to Morris et al. [Mor+21] and the references therein for an extended discussion of these aspects.

☞ **Theorem 4.** *Given k-FWL colourings of two graphs G and G' that are different, there exists a filtration of G and G' such that the corresponding persistence diagrams in dimension $k-1$ or dimension k are different.*

Proof. The main idea involves harnessing the colours of k-tuples. We first identify all colours $c_1, c_2, \ldots$ with natural numbers $1, 2, \ldots$ We then expand G and G' to a simplicial complex that contains all k-tuples as its faces.[10] Moreover, we assign *all* simplices in dimensions less than or equal to $k-2$ a weight of 0. Each $(k-1)$-simplex is assigned its colour according to the respective k-FWL colouring. As a consequence of the *pairing lemma* of persistent homology, every $(k-1)$-simplex is either a creator simplex or a destroyer simplex [EH08].

We handle the case of *creator simplices* first. Each creator simplex gives rise to an essential persistence pair in the $(k-1)$-dimensional persistence diagram. Each such pair is of the form (i, ∞), where c_i is a colour according to k-FWL.

Each *destroyer simplex*, by contrast, destroys a topological feature created by a $(k-2)$-simplex, i.e. a $(k-1)$-tuples, resulting in a pair of the form $(\cdot, j)$, with again c_j being the corresponding k-FWL colour. This pair is part of a $(k-2)$-dimensional persistence diagram.

By assumption, the k-FWL colours of G and G' are different, so there must exist a colour c whose count is *different* in G and G', respectively. Since the *sum* of all colours arising in the two types of persistence pairs above is the number of colours of k-tuples, there is either a difference in colour counts in the $(k-1)$-dimensional or the $(k-2)$-dimensional persistence diagrams of G and G', showing that they are not equal. ■

Notice that the argument made in the proof above does not depend on the actual pairing. In contrast to Theorem 3, we also see that Theorem 4 makes use of *two* types of persistence diagrams. The astute reader might wonder at this point to what extent the filtration described above is 'valid.' Indeed, it does not necessarily lead to differentiable persistence diagrams because its function values are not distinct. However, using Lemma 3, we can rectify this in practice. Both Theorem 3 and Theorem 4 may not be completely satisfactory because they only show the *existence* of such a filtration, but it makes no claims about the expressivity of existing filtrations. This is the nature of such existence proofs; we would hope our publication serves as an invitation to a more learning-theoretic approach of this issue.

On strictly higher expressivity. Ideally, we would want to extend Theorem 4 to state that persistent homology is *strictly* more expressive than k-FWL. This is not as straightforward as for $k = 1$, since we would have to construct families of non-isomorphic graphs that require $(k+1)$-FWL to be distinguished but that can be distinguished

[10] When dealing with tuples or simplices, there is always the risk of an off-by-one error. In this theorem, despite dealing with k-FWL, only the $(k-1)$-simplices, which have k vertices, will be relevant.

already with lower-dimensional persistent homology. Currently, we can provide one such counterexample, described in Table 2, consisting of the 4×4 rook's graph and the Shrikhande graph. With an appropriate filtration, persistent homology can distinguish these two graphs *without* requiring more than vertices and edges, whereas 2-FWL is unable to distinguish them. We leave a more general result for future work.

7 Experiments

The previous sections discussed the theoretical properties of filtrations. Here, we want to briefly comment on their *empirical* performance. We thus analyse to what extent two different filtrations are capable of distinguishing between different non-isomorphic graphs. Notice that we refrain from performing experiments that require training an additional classifier since such results would be inherently harder to interpret.

Experimental setup. We use different data sets of connected cubic graphs [CDG23] and strongly-regular graphs [McK]. The latter type of graph is known to be challenging for graph isomorphism tests; we know for instance that 2-FWL cannot distinguish between any such instances [Mor+21]. In the following, we will use three different filtrations for each graph:

1. A *degree* filtration, i.e. $v \mapsto \deg(v)$. The degree filtration is the most basic non-trivial filtration of a graph, showing nevertheless surprising empirical performance in graph classification tasks [Hof+20; ORB21; RSL20].

2. A filtration based on the eigenvalues of the *undirected graph Laplacian*, i.e. $v \mapsto \lambda_v$, where λ_v indicates the eigenvalue of the *undirected graph Laplacian* corresponding to vertex v. The graph Laplacian is known to capture characteristic properties of a graph; in the context of persistent homology it is often used in the form of a *heat kernel signature* [Car+20].

3. A filtration based on the *Ollivier–Ricci curvature* [Oll07] in the graph, setting $v \mapsto -1$ and $(u, v) \mapsto \kappa(u, v)$, with κ denoting the Ollivier–Ricci curvature

$$\kappa(u, v) := 1 - \mathrm{W}_1(\mu_u^\alpha, \mu_v^\alpha), \tag{15}$$

where W_1 denotes the first Wasserstein distance,[11] and $\mu_u^\alpha, \mu_v^\alpha$ denote *probability measures* based on lazy random walk in the graph:

$$\mu_u^\alpha(v) := \begin{cases} \alpha & u = v \\ (1-\alpha)\frac{1}{\deg(u)} & (u, v) \in E \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

[11] This metric is also known as the *Earth Mover's Distance* [LB01]. The Wasserstein distance is a fundamental concept in optimal transport; the monograph by Villani [Vil09] contains a comprehensive introduction to this topic.

Notice that Eq. (15) can be generalised to arbitrary pairs of vertices, which entails a slightly different weighting factor. Moreover, the probability measures μ_u^α may be adjusted; recent work investigates the utility of this perspective [CDR23; Sou+23]. We set $\alpha = 0$ for our subsequent experiments, thus obtaining a non-lazy random walk; the investigation of additional parameters impacts is left for future work.

After picking a filtration, we expand the graph by filling in all k-cliques and calculate persistent homology up to dimension k. Hence, for $k = 1$, we leave the graph 'as-is,' making use of connected components and cycles only. Our persistent homology calculations result in a set of persistence diagrams for each graph, which we compare in a pairwise manner using the bottleneck distance described by Eq. (11). We consider two graphs to be different whenever the distance between their persistence diagrams is $> 1 \times 10^{-8}$.

Connected cubic graphs. We start our investigation by distinguishing connected cubic graphs, i.e. 3-regular graphs. These graphs cannot be distinguished by 1-WL, but they can be distinguished by 2-FWL [Bol82; Mor+21]. As such, they provide a good example of how different filtrations harness different types of graph information. Table 1 shows the results. We first observe that the degree filtration is incapable of distinguishing graphs for $k = 1$. This is a direct consequence of the regularity—since the function is constant on the graph, persistent homology cannot capture any variability. This changes, however, when higher-order structures—triangles—are included for $k = 2$. We also observe that the Laplacian-based filtration exhibits strong empirical performance; in the absence of additional information, the spectral properties captured by the Laplacian help in distinguishing graphs. The Ollivier–Ricci curvature filtration is performing similarly well; in contrast to the Laplacian-based filtration, it does not require the calculation of eigenvalues, which may be prohibitive for larger graphs.[12]

Strongly-regular graphs. The second class of graphs we will be analysing consists of *strongly-regular graphs*. Such graphs are known to be extremely challenging to distinguish; 2-FWL is unable to distinguish *any* of these graphs, for instance. Our goal is again to highlight the utility of different filtration function choices. Table 2 summarises the performance.

We first observe that for $k = 1$, i.e. for the original graph without any cliques, few pairs of graphs can be distinguished. Notably, a curvature-based filtration is sufficient to distinguish the two graphs in the `sr16622` data set, colloquially also known as the 4×4 rook's graph and the Shrikhande graph. Distinguishing between these two graphs is usually said to require knowledge about cliques [Bod+21], but it turns out that a suitable filtration is sufficient. However, the empirical expressivity of the curvature-based filtration appears limited for $k = 1$, improving for higher-order clique complexes.

[12] Ollivier–Ricci curvature requires solving optimal transport problems, for which highly efficient approximative solvers are available [Cut13].

Table 1: Success rate (↑) of distinguishing pairs of connected cubic graphs when using three different filtrations (D: degree filtration, L: Laplacian filtration, C: Ollivier–Ricci curvature filtration) at varying expansion levels of the graph (denoted by k). Due to the regularity of each graph, $k = 3$ is omitted since the clique complex of the graph is exactly the same as for $k = 2$. These graphs can be distinguished by 2-FWL but not by 1-WL.

Data	$k = 1$			$k = 2$		
	Filtration					
	D	L	C	D	L	C
`cub06`	0.00	1.00	1.00	1.00	1.00	1.00
`cub08`	0.00	1.00	0.90	0.90	1.00	0.90
`cub10`	0.00	1.00	0.99	0.81	1.00	0.99
`cub12`	0.00	1.00	1.00	0.80	1.00	1.00
`cub14`	0.00	1.00	1.00	0.79	1.00	1.00

Table 2: Success rate (↑) for distinguishing pairs of *strongly-regular graphs* when using three different filtrations (D: degree filtration, L: Laplacian filtration, C: Ollivier–Ricci curvature filtration) at varying expansion levels of the graph (denoted by k). None of these pairs of graphs can be distinguished by 2-FWL [Bod+21].

Data	$k = 1$			$k = 2$			$k = 3$		
	Filtration								
	D	L	C	D	L	C	D	L	C
`sr16622`	0.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
`sr251256`	0.00	0.00	0.00	0.00	0.83	0.00	0.91	0.95	0.91
`sr261034`	0.00	0.73	0.00	0.20	0.84	0.20	0.93	0.98	0.93
`sr291467`	0.00	0.00	0.00	0.00	0.75	0.00	0.80	0.89	0.80
`sr281264`	0.00	0.67	1.00	0.00	0.83	1.00	1.00	1.00	1.00
`sr351899`	0.00	0.46	0.00	0.00	0.50	0.00	0.98	0.99	0.98
`sr361446`	0.00	0.49	0.00	0.02	0.96	0.02	0.92	0.99	0.92
`sr401224`	0.00	0.00	0.00	0.93	0.99	0.93	0.94	0.99	0.94

The Laplacian filtration, by contrast, exhibits strong empirical performance for $k = 2$, increasing to near-perfect performance for $k = 3$ in almost all data sets. It is clear that knowledge about higher-order cliques helps in driving performance here. Notice that in contrast to other algorithms [Bod+21], no additional embedding of the graphs is required; we are comparing 'raw' persistence diagrams directly.

8 Discussion

This paper provided an introduction to persistent homology in the context of graph learning. We discussed various aspects of the computation process and provided theoretical evidence of advantageous properties of persistent homology. Our primary insights are that persistent homology is *at least as expressive* as a corresponding 1-WL or k-FWL test, while in some cases even surpassing their discriminative power.[13] In the following, we give an overview of emerging research directions, ending with a list of open theoretical questions for the community and a brief conclusion.

8.1 Future Work: Beyond Expressivity of Filtrations

Expressivity in terms of distinguishing between special families of graphs is not the be-all and end-all of graph learning research. Considering recent work, an investigation of which other properties are being captured by persistent homology would be worthwhile. Theorem 2 provides a first glimpse here, and we hope that it will be accompanied by additional such theorems in the future. Any such research research can either focus on the overall properties of the persistent homology calculations, i.e. provide results that are *independent* of the respective filtration, or focus on the expressive power of special *classes of filtrations*.

Along these lines, Southern et al. [Sou+23] provide a first empirical analysis of the substructures that are being captured by curvature-based filtrations. Any further results are strongly contingent on the respective filtration, making a learning-theoretical investigation of which filtrations can be learnt in practice an interesting direction for future work.[14] Similarly, the success of the Laplacian filtration may hint at new filtrations based on spectral graph theory that provide a trade-off between utility and computational efficiency.

We find that this research direction is overlooked by the computational topology research community at the moment. The reasons for this are probably historical: persistent homology and related approaches are originally geared towards assessing high-dimensional point clouds. Most of the stability results are thus of a somewhat

[13] More specifically, our previous work [Hor+22] showed that persistent homology is *strictly more expressive* than 1-WL. In this article, when tackling $k \geq 2$, we only provide counterexamples that show that there are graphs that can be distinguished by *some* filtration but not by 2-FWL, for instance.

[14] We already showed how to learn graph filtrations in earlier work [Hof+20], but there are currently no results about the properties

geometric flavour [Cha+16; CSO14] and describe distance-based constructions such as the Vietoris–Rips complex filtration. This paper is but a first attempt at elucidating the theoretical utility of computational topology in a graph learning context; advancing the field will require many more insights.

8.2 Future Work: Extended Persistent Homology and Zigzag Persistence

When introducing the concept of a filtration in Definition 6, we only discuss a very limited part of computational topology research. There are two extensions that merit further analysis in the context of graph learning. The first one involves *extended persistent homology* [CEH09], a way to calculate persistent homology without essential features. The advantage of this formulation is that all tuples in the persistence diagrams are *finite*. Extended persistent homology thus permits a more nuanced view into a graph and does not require special handling for essential features. Its empirical performance in graph learning tasks has recently been investigated [Yan+22; YSX23].

The second extension is *zigzag persistent homology* [CS10; CSM09], which removes the rigid requirement of having a sequence of nested complexes. Instead, zigzag persistent homology permits filtrations that contain subset and superset relations in an alternating fashion. This is typically indicated by rewriting Eq. (5) as

$$\emptyset = \mathrm{K}_0 \leftrightarrow \mathrm{K}_1 \leftrightarrow \cdots \leftrightarrow \mathrm{K}_{m-1} \leftrightarrow \mathrm{K}_m = \mathrm{K}, \tag{17}$$

where $\leftrightarrow$ represents an inclusion relation, either to the left or to the right. Zigzag persistent homology can be calculated efficiently on graphs [DH21] and has shown promise in detecting critical points in dynamical systems; we are not aware of any uses in graph learning so far.

8.3 Future Work: Topology-Driven Baselines & Hybrid Models

While not explicitly assessed in the experimental setup at hand, we envision that persistent homology may well provide a strong baseline for graph learning applications. As previous work shows, even topology-inspired approaches, making use of concepts such as filtrations, can approximate the performance of highly-parametrised models at a fraction of the computational cost [ORB21]. All insights obtained using such topological methods hint at the overall utility of graph structural information for graph learning tasks. Current graph benchmark data sets tend to lack such an understanding, making it unclear to what extent empirical performance is driven by label information or by the actual graph structure. The strong empirical performance *without using node label information* that we observed in previous work [Hor+22] seems to suggest that some graph data sets are not 'graphical.' We thus hope that persistent homology and related techniques will also find more applications in *hybrid models*, which are able to incorporate geometrical–topological information about graphs. This is an emerging

research topic of crucial relevance since there are now numerous graph data sets that combine geometrical information (in the form of node coordinates, for instance) with topological information (in the form of a graph structure).

Researchers interested in this topic are invited to think about persistent homology as a different way of viewing an evolving function on their data, rather than 'just' another source of graph-based features. Despite us focusing only on static filtrations for the sake of exposition, the real power of a topological perspective for graph learning lies in *learning task-driven filtrations*. If the class of potential functions is sufficiently well-behaved from a mathematical point of view, such filtrations might even lead to better interpretable and more robust results.

8.4 Open Questions

This paper was motivated by our observation of a graph between theory and practice when it comes to the use of persistent homology in graph learning contexts. In light of our theorems, we would like to pose the following questions to the community:

1. Is there a way to simulate or describe k-FWL via persistent homology that does not involve the full k-simplex?

2. Can we show general expressivity results for *clique complexes*, i.e. for simplicial complexes that we obtain from graphs by expanding each k-clique into a $(k-1)$-simplex?

3. Is persistent homology based on k-dimensional simplicial complexes *strictly* more expressive than k-FWL?

4. Can we prove that there are classes of continuous filtration functions that are homotopy-equivalent to the Weisfeiler–Leman set of colours or an approximation thereof?

5. Which graph properties other than the diameter can be bounded or decided by persistent homology?

8.5 Conclusion

Our theoretical analysis of the properties of persistent homology for graph learning tasks show great potential for a topology-based perspective. We are confident that additional computational topology concepts will enrich and augment machine learning models, leading to new insights about their theoretical and empirical capabilities.

Acknowledgements

This paper was motivated in large parts by discussions with participants of the BIRS 2022 Workshop on 'Deep Exploration of Non-Euclidean Data with Geometric and Topological Representation Learning.' The author is also indebted to Leslie O'Bray for comments and discussions that helped substantially improve the main arguments and led to a simplified proof of Theorem 4.

References

[Arv+15] V. Arvind, J. Köbler, G. Rattan, and O. Verbitsky. 'On the Power of Color Refinement'. In: *Fundamentals of Computation Theory*. Ed. by A. Kosowski and I. Walukiewicz. Cham, Switzerland: Springer, 2015, pp. 339–350.

[Bau21] U. Bauer. 'Ripser: efficient computation of Vietoris–Rips persistence barcodes'. In: *Journal of Applied and Computational Topology* 5.3 (2021), pp. 391–423. DOI: 10.1007/s41468-021-00071-5.

[Bau+22] U. Bauer, T. B. Masood, B. Giunti, G. Houry, M. Kerber, and A. Rathod. 'Keeping it sparse: Computing Persistent Homology revisited'. 2022. arXiv: 2211.09075 [cs.CG].

[Ben+16] P. Bendich, J. S. Marron, E. Miller, A. Pieloch, and S. Skwerer. 'Persistent homology analysis of brain artery trees'. In: *The Annals of Applied Statistics* 10.1 (2016), pp. 198–218. DOI: 10.1214/15-AOAS886.

[Ber01] C. Berge. *The Theory of Graphs*. Dover Publications, 2001.

[Bod+21] C. Bodnar, F. Frasca, Y. Wang, N. Otter, G. F. Montufar, P. Lió, and M. Bronstein. 'Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks'. In: *International Conference on Machine Learning (ICML)*. Ed. by M. Meila and T. Zhang. Proceedings of Machine Learning Research 139. PMLR, 2021, pp. 1026–1037.

[Bol82] B. Bollobás. 'Distinguishing Vertices of Random Graphs'. In: *Graph Theory*. Ed. by B. Bollobás. North–Holland Mathematics Studies 62. Amsterdam, Netherlands: North–Holland, 1982, pp. 33–49. DOI: 10.1016/S0304-0208(08)73545-X.

[Bub+20] P. Bubenik, M. Hull, D. Patel, and B. Whittle. 'Persistent homology detects curvature'. In: *Inverse Problems* 36.2 (2020), p. 025008. DOI: 10.1088/1361-6420/ab4ac0.

[CS10] G. Carlsson and V. de Silva. 'Zigzag Persistence'. In: *Foundations of Computational Mathematics* 10.4 (2010), pp. 367–405. DOI: 10.1007/s10208-010-9066-0.

[CSM09] G. Carlsson, V. de Silva, and D. Morozov. 'Zigzag Persistent Homology and Real-Valued Functions'. In: *25th International Symposium on Computational Geometry (SoCG)*. 2009, pp. 247–256. DOI: 10.1145/1542362.1542408.

[Car+20] M. Carrière, F. Chazal, Y. Ike, T. Lacombe, M. Royer, and Y. Umeda. 'PersLay: A Neural Network Layer for Persistence Diagrams and New Graph Topological Signatures'. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by S. Chiappa and R. Calandra. Proceedings of Machine Learning Research 108. PMLR, 2020, pp. 2786–2796.

[Cha+16] F. Chazal, V. De Silva, M. Glisse, and S. Oudot. *The structure and stability of persistence modules*. Vol. 10. SpringerBriefs in Mathematics. Cham, Switzerland: Springer, 2016. DOI: 10.1007/978-3-319-42545-0.

[CSO14] F. Chazal, V. de Silva, and S. Oudot. 'Persistence stability for geometric complexes'. In: *Geometriae Dedicata* 173.1 (2014), pp. 193–214. DOI: 10.1007/s10711-013-9937-z.

[CCG21] Y. Chen, B. Coskunuzer, and Y. Gel. 'Topological Relational Learning on Graphs'. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan. Vol. 34. Curran Associates, Inc., 2021, pp. 27029–27042.

[CEH09] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. 'Extending persistence using Poincaré and Lefschetz duality'. In: *Foundations of Computational Mathematics* 9.1 (2009), pp. 79–103. DOI: 10.1007/s10208-008-9027-z.

[CEH07] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. 'Stability of persistence diagrams'. In: *Discrete & Computational Geometry* 37.1 (2007), pp. 103–120. DOI: 10.1007/s00454-006-1276-5.

[Coh+10] D. Cohen-Steiner, H. Edelsbrunner, J. Harer, and Y. Mileyko. 'Lipschitz functions have L_p-stable persistence'. In: *Foundations of Computational Mathematics* 10.2 (2010), pp. 127–139. DOI: 10.1007/s10208-010-9060-6.

[Col81] C. J. Colbourn. 'On testing isomorphism of permutation graphs'. In: *Networks* 11.1 (1981), pp. 13–21. DOI: 10.1002/net.3230110103.

[CDG23] K. Coolsaet, S. D'hondt, and J. Goedgebeur. 'House of Graphs 2.0: A database of interesting graphs and more'. In: *Discrete Applied Mathematics* 325 (2023), pp. 97–107. DOI: 10.1016/j.dam.2022.10.013. URL: https://houseofgraphs.org.

[CDR23] C. Coupette, S. Dalleiger, and B. Rieck. 'Ollivier–Ricci Curvature for Hypergraphs: A Unified Framework'. In: *International Conference on Learning Representations (ICLR)*. 2023. arXiv: 2210.12048 [cs.LG]. In press.

[Cut13] M. Cuturi. 'Sinkhorn Distances: Lightspeed Computation of Optimal Transport'. In: *Advances in Neural Information Processing Systems*. Ed. by C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger. Vol. 26. 2013.

[DH21] T. K. Dey and T. Hou. 'Computing Zigzag Persistence on Graphs in Near-Linear Time'. In: *37th International Symposium on Computational Geometry (SoCG)*. Ed. by K. Buchin and É. Colin de Verdière. Vol. 189. 2021, 30:1–30:15. DOI: 10.4230/LIPIcs.SoCG.2021.30.

[DW22] T. K. Dey and Y. Wang. *Computational Topology for Data Analysis*. Cambridge, United Kingdom: Cambridge University Press, 2022. DOI: 10.1017/9781009099950.

[EH08] H. Edelsbrunner and J. Harer. 'Persistent homology—a survey'. In: *Surveys on discrete and computational geometry: Twenty years later*. Ed. by J. E. Goodman, J. Pach, and R. Pollack. Contemporary Mathematics 453. Providence, RI, USA: American Mathematical Society, 2008, pp. 257–282.

[ELZ02] H. Edelsbrunner, D. Letscher, and A. J. Zomorodian. 'Topological persistence and simplification'. In: *Discrete & Computational Geometry* 28.4 (2002), pp. 511–533. DOI: 10.1007/s00454-002-2885-2.

[Gar+22] R. J. Gardner, E. Hermansen, M. Pachitariu, Y. Burak, N. A. Baas, B. A. Dunn, M.-B. Moser, and E. I. Moser. 'Toroidal topology of population activity in grid cells'. In: *Nature* 602.7895 (2022), pp. 123–128. DOI: 10.1038/s41586-021-04268-7.

[Gro21] M. Grohe. 'The Logic of Graph Neural Networks'. In: *Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 2021, pp. 1–17. DOI: 10.1109/LICS52264.2021.9470677.

[Hof+17] C. Hofer, R. Kwitt, M. Niethammer, and A. Uhl. 'Deep Learning with Topological Signatures'. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. 2017, pp. 1634–1644.

[Hof+20] C. D. Hofer, F. Graf, B. Rieck, M. Niethammer, and R. Kwitt. 'Graph Filtration Learning'. In: *International Conference on Machine Learning (ICML)*. Ed. by H. Daumé III and A. Singh. Proceedings of Machine Learning Research 119. PMLR, 2020, pp. 4314–4323. arXiv: 1905.10996 [cs.LG].

[HKN19] C. D. Hofer, R. Kwitt, and M. Niethammer. 'Learning Representations of Persistence Barcodes'. In: *Journal of Machine Learning Research* 20.126 (2019), pp. 1–45.

[HMR09] D. Horak, S. Maletić, and M. Rajković. 'Persistent homology of complex networks'. In: *Journal of Statistical Mechanics: Theory and Experiment* 2009.3 (2009), P03034. DOI: 10.1088/1742-5468/2009/03/P03034.

[Hor+22] M. Horn, E. De Brouwer, M. Moor, Y. Moreau, B. Rieck, and K. Borgwardt. 'Topological Graph Neural Networks'. In: *International Conference on Learning Representations (ICLR)*. 2022. URL: https://openreview.net/forum?id=oxxUMeFwEHd.

[Kel57] P. J. Kelly. 'A congruence theorem for trees'. In: *Pacific Journal of Mathematics* 7.1 (1957), pp. 961–968. DOI: pjm/1103043674.

[Kwi+15] R. Kwitt, S. Huber, M. Niethammer, W. Lin, and U. Bauer. 'Statistical Topological Data Analysis - A Kernel Perspective'. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. 2015.

[LB01] E. Levina and P. Bickel. 'The Earth Mover's distance is the Mallows distance: some insights from statistics'. In: *IEEE International Conference on Computer Vision (ICCV)*. Vol. 2. 2001, pp. 251–256. DOI: 10.1109/ICCV.2001.937632.

[LMO20] S. Lim, F. Memoli, and O. B. Okutan. *Vietoris–Rips Persistent Homology, Injective Metric Spaces, and The Filling Radius*. 2020. arXiv: 2001.07588 [math.AT].

[McK] B. McKay. *Collection of strongly-regular graphs*. URL: https://users.cecs.anu.edu.au/~bdm/data/graphs.html.

[Mor+21] C. Morris, Y. Lipman, H. Maron, B. Rieck, N. M. Kriege, M. Grohe, M. Fey, and K. Borgwardt. 'Weisfeiler and Leman go Machine Learning: The Story so far'. 2021. arXiv: 2112.09992 [cs.LG].

[Mor+19] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe. 'Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks'. In: *AAAI Conference on Artificial Intelligence*. 2019. DOI: 10.1609/aaai.v33i01.33014602.

[Mun84] J. R. Munkres. *Elements of Algebraic Topology*. Menlo Park, California, USA: Addison–Wesley Publishing Company, 1984.

[ORB21] L. O'Bray, B. Rieck, and K. Borgwardt. 'Filtration Curves for Graph Representation'. In: *International Conference on Knowledge Discovery & Data Mining (KDD)*. New York, NY, USA: Association for Computing Machinery, 2021, pp. 1267–1275. DOI: 10.1145/3447548.3467442.

[Oll07] Y. Ollivier. 'Ricci curvature of metric spaces'. In: *Comptes Rendus Mathematique* 345.11 (2007), pp. 643–646. DOI: 10.1016/j.crma.2007.10.041.

[Ott+17] N. Otter, M. A. Porter, U. Tillmann, P. Grindrod, and H. A. Harrington. 'A roadmap for the computation of persistent homology'. In: *EPJ Data Science* 6.1 (2017), 17:1–17:38. DOI: 10.1140/epjds/s13688-017-0109-5.

[Rei+15] J. Reininghaus, S. Huber, U. Bauer, and R. Kwitt. 'A stable multi-scale kernel for topological machine learning'. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 4741–4748. DOI: 10.1109/CVPR.2015.7299106.

[Rie17] B. Rieck. 'Persistent Homology in Multivariate Data Visualization'. PhD thesis. Heidelberg University, 2017. DOI: 10.11588/heidok.00022914.

[RBB19] B. Rieck, C. Bock, and K. Borgwardt. 'A Persistent Weisfeiler–Lehman Procedure for Graph Classification'. In: *International Conference on Machine Learning (ICML)*. Ed. by K. Chaudhuri and R. Salakhutdinov. Proceedings of Machine Learning Research 97. PMLR, 2019, pp. 5448–5458.

[Rie+18] B. Rieck, U. Fugacci, J. Lukasczyk, and H. Leitte. 'Clique Community Persistence: A Topological Visual Analysis Approach for Complex Networks'. In: *IEEE Transactions on Visualization and Computer Graphics* 24.1 (2018), pp. 822–831. DOI: 10.1109/TVCG.2017.2744321.

[RSL20] B. Rieck, F. Sadlo, and H. Leitte. 'Topological Machine Learning with Persistence Indicator Functions'. In: *Topological Methods in Data Analysis and Visualization V*. Ed. by H. Carr, I. Fujishiro, F. Sadlo, and S. Takahashi. Cham, Switzerland: Springer, 2020, pp. 87–101. DOI: 10.1007/978-3-030-43036-8_6. arXiv: 1907.13496 [math.AT].

[ŠT20] P. Škraba and K. Turner. 'Wasserstein Stability for Persistence Diagrams'. 2020. arXiv: 2006.16824 [math.AT].

[Sou+23] J. Southern, J. Wayland, M. Bronstein, and B. Rieck. 'Curvature Filtrations for Graph Generative Model Evaluation'. 2023. arXiv: 2301.12906 [cs.LG].

[TMO22] R. Turkeš, G. Montúfar, and N. Otter. 'On the Effectiveness of Persistent Homology'. In: *Advances in Neural Information Processing Systems*. Ed. by A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho. 2022. URL: https://openreview.net/forum?id=DRjUkfExCix.

[Tur+14] K. Turner, Y. Mileyko, S. Mukherjee, and J. Harer. 'Fréchet Means for Distributions of Persistence Diagrams'. In: *Discrete & Computational Geometry* 52.1 (2014), pp. 44–70. DOI: 10.1007/s00454-014-9604-7.

[Vil09] C. Villani. *Optimal Transport. Old and New*. Grundlehren der mathematischen Wissenschaften 338. Springer, 2009.

[Wil96] R. J. Wilson. *Introduction to Graph Theory*. 4th ed. Harlow, United Kingdom: Addison Wesley Longman Limited, 1996.

[Xu+19] K. Xu, W. Hu, J. Leskovec, and S. Jegelka. 'How Powerful are Graph Neural Networks?' In: *International Conference on Learning Representations (ICLR)*. 2019. URL: https://openreview.net/forum?id=ryGs6iA5Km.

[Yan+22] Z. Yan, T. Ma, L. Gao, Z. Tang, Y. Wang, and C. Chen. 'Neural Approximation of Graph Topological Features'. In: *Advances in Neural Information Processing Systems*. Ed. by A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho. 2022. URL: https://openreview.net/forum?id=qwjrO7Rewqy.

[YSX23] X. Ye, F. Sun, and S. Xiang. 'TREPH: A Plug-In Topological Layer for Graph Neural Networks'. In: *Entropy* 25.2 (2023). ISSN: 1099-4300. DOI: 10.3390/e25020331.

[ZW19] Q. Zhao and Y. Wang. 'Learning metrics for persistence-based summaries and applications for graph classification'. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. 2019, pp. 9855–9866.

[Zha+20] Q. Zhao, Z. Ye, C. Chen, and Y. Wang. 'Persistence Enhanced Graph Neural Network'. In: *International Conference on Artificial Intelligence and Statistics (AISTATS)*. Ed. by S. Chiappa and R. Calandra. Proceedings of Machine Learning Research 108. PMLR, 2020, pp. 2896–2906.