Weisfeiler–Leman Is Incomplete on Simple Spectrum Graphs, so Canonicalize Them

Snir Hordan
Faculty of Mathematics
Technion - Israel Institute of Technology

Nadav Dym
Faculty of Mathematics
Technion - Israel Institute of Technology

Tim Seppelt
IT University of Copenhagen

Abstract

Graphs with a simple spectrum admit cubic-time isomorphism testing, yet we prove that for every natural number k, the k-Weisfeiler–Leman (k-WL) test cannot distinguish all non-isomorphic multi-graphs with a simple spectrum. As the WL hierarchy upper-bounds the distinguishing power of widely-used Graph Neural Networks (GNNs), this incompleteness applies to all such GNNs, ruling out completeness for every k-WL-aligned GNN family. To close this gap, we introduce PRiSM (Partition, Refine, Solve, Match), the first provably complete canonicalization of simple-spectrum eigendecompositions. PRiSM obtains the completeness guarantee that prior canonicalizations provably lack, and resolves the open problem of achieving complete expressivity on simple-spectrum graphs. When composed with DeepSets or a Transformer, PRiSM achieves universal approximation on simple-spectrum graphs, justifying the use of canonicalized Laplacian positional encodings. Empirically, PRiSM performs comparably to or outperforms existing spectral canonicalizations on graph regression, classification, and expressivity benchmarks.

1 Introduction

Graph Neural Networks (GNNs) are a leading machine learning paradigm for learning on graph-structured data such as chemical and molecular graphs [1–3]. These networks typically maintain node representations and refine them using their local graph neighborhoods [4–6]. To understand GNNs' expressive power, or their ability to distinguish among non-isomorphic graphs, it is common practice to compare the distinguishing power of GNNs with that of the Weisfeiler–Leman (WL) graph isomorphism test hierarchy [7–9]. The WL hierarchy [10] consists of combinatorial graph isomorphism tests that, analogously to GNNs, operate by a color refinement procedure, a procedure naturally aligned with the inner workings of many widely-used GNNs [7, 8, 11–13]. Consequently, any incompleteness result on the WL hierarchy directly translates into expressivity limitations of the GNN architectures aligned with it.

We focus on understanding the expressivity of WL tests on graphs with a simple spectrum. Graphs with a simple spectrum are graphs whose adjacency matrix has distinct eigenvalues. These graphs are known to be distinguishable in cubic time [14, 15], yet it has been unclear whether k-WL tests can distinguish among such graphs [16]. Our first main result in the manuscript provides a definitive answer: we prove that no test in the WL hierarchy, despite the hierarchy's ever-increasing expressive power and computational complexity, can distinguish among all graphs with a simple spectrum.

Our proof is constructive. For each k, we start from a 3-regular base graph of sufficiently high treewidth and apply the Cai–Fürer–Immerman (CFI) [17] construction (Fig. 1) to obtain a pair of

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

non-isomorphic graphs that k-WL cannot distinguish. Unfortunately, these graphs are not simple spectrum. To circumvent this, we introduce a novel mechanism of encoding these CFI adjacency patterns to obtain a pair of simple-spectrum multigraphs, on which k-WL still fails. This reveals a fundamental gap as each k-WL test runs in $O(k^2 n^{k+1} \log(n))$ time [18], yet these tests are not able to solve a combinatorial problem that can be solved in cubic time.

Our incompleteness result justifies the common practice of incorporating Laplacian positional encodings to GNNs and Graph Transformers [19, 13], as these encodings add spectral information that WL tests and corresponding GNNs natively lack. However, naively using positional encoding introduces a sign ambiguity in addition to the inherent permutation ambiguity of the model. To address this ambiguity, sign-invariant models such as SignNet [20] and canonicalizations such as OAP and MAP [21, 22] are used. Unfortunately, the sign invariance of these models comes with restricted expressivity, and such models provably cannot distinguish all simple-spectrum graphs [16]. Accordingly, Hordan et al. [16] conclude that achieving a complete invariant model for simple-spectrum graphs is an open question.

Our second main result in the manuscript resolves this open question, by defining a novel complete canonicalization for eigendecompositions of graphs with a simple spectrum, which we name PRiSM. Composing our canonicalization with a universal set network, such as DeepSets [23] or a Transformer [24, 25], yields a model that universally approximates any invariant function on simple-spectrum graphs.

To allow PRiSM to process arbitrary graphs, we define a heuristic extension of our canonicalization to graphs with repeated eigenvalues. Empirically, we find that PRiSM outperforms competing canonicalizations and sign and basis invariant models.

In summary, our main contributions are as follows:

1. **Entire WL hierarchy is incomplete on graphs with simple spectrum** (Theorem 1). We prove that for every natural number k, there exist non-isomorphic simple-spectrum multigraphs on $\Theta(k)$ nodes that k-WL cannot distinguish. In a single step, this proves the incompleteness of every WL-aligned GNN family (PPGN, k-GNNs, k-WL-aligned Graph Transformers, N2GNNs, d-DRFWL(2)) on the class of simple-spectrum graphs (see Corollary 1).
2. **First complete canonicalization** (Theorem 2, Theorem 3). We introduce PRiSM, the first provably complete sign canonicalization of simple-spectrum eigendecompositions. Composed with DeepSets or a Transformer, PRiSM achieves universal approximation on simple-spectrum graphs (Theorem 3), supplying the completeness guarantee absent from Ma et al. [21, 22], Lim et al. [20] and resolving the open problem of universality on simple-spectrum graphs raised by Hordan et al. [16].
3. **Empirical results** (Section 6). PRiSM improves ability to distinguish non-isomorphic graph pairs on the BREC [26] expressivity benchmark, and performs on par or better compared with competing canonicalizations on widely-used molecular datasets (ogbg [27], ZINC [28], Alchemy [29]).

The remainder of this paper is organized as follows. Section 2 surveys related work and previous canonicalizations for graph positional encodings. Section 3 introduces the necessary background on graphs with bounded eigenvalue multiplicity, the Weisfeiler–Leman hierarchy, and the CFI construction. Section 4 presents our incompleteness result, showing that no level of the WL hierarchy suffices for simple-spectrum graphs. Section 5 develops the canonicalization procedure and its completeness and universality guarantees. Section 6 evaluates our method empirically on expressivity and molecular benchmarks.

2 Related Work

Weisfeiler–Leman expressivity. The WL hierarchy is the primary framework for GNN expressivity: message-passing GNNs are equivalent to 1-WL [7, 9], higher-order architectures simulate higher levels (PPGN [8], k-GNNs [9], N2GNNs [30], d-DRFWL(2) [31]), and Subgraph GNNs are upper-bounded by 2-WL in the numbering of Definition 4 (3-WL in the convention of [32]). The Cai–Fürer–Immerman construction proves the k-WL incompleteness on general graphs for every k [17, 33]. We

show that this incompleteness exists even for graphs with a simple spectrum, a class for which graph isomorphism is polynomial-time solvable [14, 34].

Spectral positional encodings ambiguity groups and Spectral Graph Neural Networks. Laplacian eigenvector positional encodings underlie modern graph transformers [35, 36, 19], but are defined only up to sign (simple eigenvalues) or orthogonal basis (repeated eigenvalues) ambiguity. To address these ambiguities, methods such as SignNet, BasisNet, MAP and OAP were introduced [20–22], but they are incomplete on simple-spectrum graphs. Gai et al. [37], Zhang et al. [38] characterize spectral invariant GNNs via homomorphism counts and prove that spectral invariant GNNs lie strictly between 1- and 3-WL, offering an expressivity hierarchy between spectral and combinatorial GNNs, whereas Vasileiou et al. [39] aim to unify message-passing and Spectral GNNs. Hordan et al. [16] show Spectral GNNs are incomplete on simple spectrum multi-graphs and this paper extends that result to high-order combinatorial GNNs.

3 Preliminaries

3.1 Graphs with Bounded Eigenvalue Multiplicity

We use the standard notation $G = (V, E)$ for combinatorial graphs. All graphs in this manuscript are undirected. A weighted graph is a triplet $g = (V, E, A)$ where A is a symmetric $|V| \times |V|$ matrix with real values, satisfying that $A_{uv} = 0$ if $(u, v) \notin E$. A *multigraph* is a weighted graph with edge weights $A_{u,v} \in \mathbb{N}$ for all $(u, v) \in E$. We note that our definitions allow a weighted graph matrix A to have negative entries, and non-zero diagonal entries. Thus the diagonal elements of A can encode node features, and A can represent the graph's Laplacian, or normalized Laplacian, as well as its adjacency matrix. We denote the set of weighted graphs with n vertices by $\mathcal{G}_n$.

An *eigendecomposition* of a weighted graph $g = (V, E, A)$ in $\mathcal{G}_n$ is an eigendecomposition of the matrix A. Namely, it is a triplet $(U, \vec{\lambda}, \vec{m})$, where $U \in \mathbb{R}^{n\times n}$ is an orthogonal matrix of eigenvectors (rows indexed by nodes, columns by eigenvectors), $\vec{\lambda} = (\lambda_1, \ldots, \lambda_r) \in \mathbb{R}^r$ is the vector of $r \leq n$ distinct eigenvalues with $\lambda_1 < \cdots < \lambda_r$, and $\vec{m} = (m_1, \ldots, m_r) \in [n]^r$ is the *multiplicity vector* with $\sum_{i=1}^{r} m_i = n$, such that

$$A = U \operatorname{diag}(\underbrace{\lambda_1, \ldots, \lambda_1}_{m_1}, \ldots, \underbrace{\lambda_r, \ldots, \lambda_r}_{m_r}) U^\top.$$

For $k = \sum_{i=1}^{r_k} m_i$ with $r_k \in \{1, \ldots, r\}$, the *k-eigendecomposition* $(U_k, \vec{\lambda}_k, \vec{m}_k)$ consists of the first k columns $U_k \in \mathbb{R}^{n\times k}$, the first r_k distinct eigenvalues, and their full multiplicities. Thus, truncation retains whole eigenspaces.

An eigendecomposition is not unique: for each eigenvalue λ_i with multiplicity m_i, any orthogonal transformation $Q_i \in O(m_i)$ applied to the corresponding eigenvectors yields an equally valid eigenbasis. The ambiguity group is therefore $\mathcal{A}(\vec{m}_k) = \prod_{i=1}^{r_k} O(m_i)$.

When all eigenvalues are distinct ($m_i = 1$ for all i), this reduces to $\mathcal{A}(\vec{m}_k) = \{-1, 1\}^k \cong \mathbb{Z}_2^k$, which can be viewed as a sign flip $u_i \mapsto -u_i$ ambiguity on each eigenvector, often called *sign ambiguity*. We say such a graph has a *simple spectrum* and write $g \in \mathcal{G}_n^{\text{simple}}$. Graph isomorphism is solvable in cubic time on $\mathcal{G}_n^{\text{simple}}$ [14, 15], and Erdős–Rényi random graphs almost surely lie in $\mathcal{G}_n^{\text{simple}}$ as $n \to \infty$ [40].

3.2 The Weisfeiler–Leman Test Hierarchy

The subject of our incompleteness results is the Weisfeiler–Leman test hierarchy [10]. It is a classic combinatorial heuristic for graph isomorphism testing, central to analysing the expressivity of graph neural networks [7]. We give the explicit 1-dimensional case, also known as *color refinement*, and defer the general k-WL formulation to Section A.1.

Definition 1 (1-WL / Color Refinement)**.** Let $g = (V, E, A) \in \mathcal{G}_n$ be a weighted graph. The 1-WL algorithm maintains a coloring $C^{(t)} : [n] \to \mathcal{C}$ over a discrete color set $\mathcal{C}$. It is initialized by $C^{(0)}(v) = \text{HASH}\,(A(v, v))$, where HASH is an injective encoding of the node features $A(v, v)$.

This coloring is then refined via

$$C^{(t+1)}(v) = \mathrm{HASH}\Big(C^{(t)}(v),\ \big\{\!\!\big\{\,(C^{(t)}(w),\ A_{vw}) \;\Big|\; w \in [n]\,\big\}\!\!\big\}\Big)\,,$$

where HASH is a fixed injective map, the same for all graphs. The algorithm terminates at the smallest T for which $C^{(T+1)}$ induces the same partition of $[n]$ as $C^{(T)}$, and outputs the graph-level label

$$C_{\mathrm{global}}(g) = \mathrm{HASH}\Big(\big\{\!\!\big\{\,C^{(T)}(v) \;\Big|\; v \in [n]\,\big\}\!\!\big\}\Big)\,.$$

Two graphs $g, g' \in \mathcal{G}_n$ are 1-*WL indistinguishable*, written $g \equiv_1 g'$, if $C_{\mathrm{global}}(g) = C_{\mathrm{global}}(g')$.

For each $k \geq 1$ the k-dimensional generalisation k-WL colors k-tuples of vertices and yields a strictly finer indistinguishability relation $\equiv_k$, where $g \equiv_{k+1} g'$ implies $g \equiv_k g'$, with the converse failing in general [17].

3.3 CFI graph construction

We review the Cai–Fürer–Immerman [17] (CFI) graph construction, a standard method for producing pairs of graphs indistinguishable by k-WL, and the entrance point for our proof of Theorem 1.

Definition 2 (CFI construction [41])**.** Let G be a combinatorial graph and $U \subseteq V(G)$. Write $E(v) \coloneqq \{e \in E(G) : v \in e\}$ for edges incident to v. We define the *CFI graph* G_U to be the graph with vertex set $\{(v, S) \mid v \in V(G),\ S \subseteq E(v),\ |S| \equiv |\{v\} \cap U| \pmod 2\}$. We connect two vertices (v, S) and (u, T) in G_U by an edge if and only if $(u, v) \in E(G)$ and $(u, v) \notin S \triangle T$. We call G the *base graph*, and its vertices the *base vertices*, while vertices (v, S) in G_U we call *fiber vertices*.

The subset U controls a parity twist: fibers at $v \notin U$ use even-cardinality subsets and at $v \in U$ odd-cardinality subsets. By [41, Corollary 3.7], for any connected G, there are exactly two non-isomorphic CFI graphs: G_0 ($|U|$ even) and G_1 ($|U|$ odd). When G has treewidth at least $k+1$, G_0 and G_1 are k-WL indistinguishable [33, Lemma 12, Theorems 10 and 14].

Example 1 (CFI construction on C_3). Consider $G = C_3$ on $\{0, 1, 2\}$. Each vertex has degree 2, yielding two fiber vertices per base vertex and six CFI vertices in total. In the trivial lift ($U = \varnothing$), all fibers use even-cardinality subsets. The vertices split into two groups, the $\varnothing$-type and the full-set type, with each forming an independent triangle, so $G_0 \cong 2\,C_3$. In the non-trivial lift ($U = \{0\}$), the parity constraint at vertex 0 shifts to odd-cardinality subsets, rewiring edges incident to vertex 0 and bridging the two triangles into a single 6-cycle: $G_{\{0\}} \cong C_6$. See Fig. 1 for an illustration. We note that G_0 and G_1 are the classic examples of graphs that are 1-WL indistinguishable. Full details are given in Section A.3.

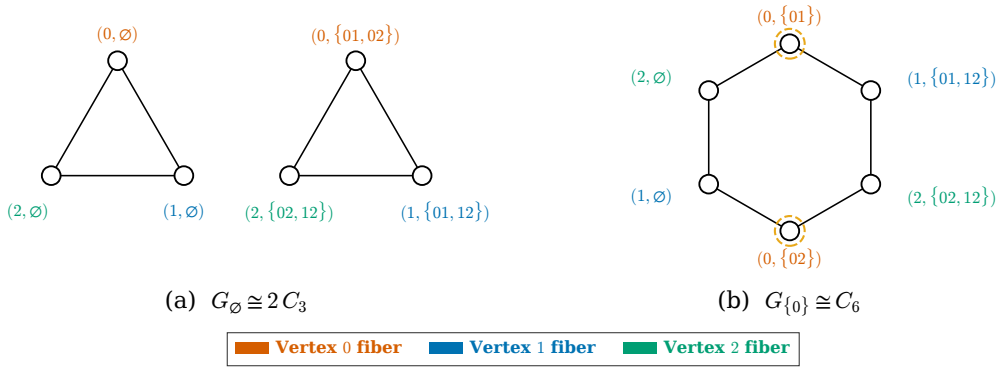

Figure 1: *The CFI construction applied to the base graph $G = C_3$.* ***(a)*** *Trivial lift $G_\varnothing \cong 2\,C_3$: the $(i, \varnothing)$ vertices and $(i, E(i))$ vertices each form an independent triangle.* ***(b)*** *Non-trivial lift $G_{\{0\}} \cong C_6$: the parity twist at vertex 0 bridges the two triangles into a single 6-cycle (dashed circles mark the twisted fiber). Colors indicate base-graph vertex correspondence and are for visualization purposes only.*

4 Incompleteness of the WL hierarchy on graphs with a simple spectrum

We introduce our main result, which proves that for any k-WL test, there exists a simple-spectrum pair of multigraphs that it cannot distinguish.

Theorem 1. *For every $k \in \mathbb{N}$, there exist non-isomorphic simple-spectrum multigraphs with $\Theta(k)$ nodes that are not distinguished by the k-dimensional Weisfeiler–Leman algorithm.*

The full proof is rather involved, and is provided in Section A. Our main goal in this section is to describe the main ideas of the proof.

Proof outline. For every $k \in \mathbb{N}$, we will construct a pair of simple-spectrum graphs by encoding the adjacency patterns of the CFI graphs, which are constructed with a carefully chosen base graph, into a valid simple-spectrum eigendecomposition via an integral encoding scheme. The novelty over the classical CFI construction lies in Step 2: an integral encoding $\tilde{X} = [X \mid X' \mid I]$ that turns CFI's k-WL-indistinguishable adjacency pattern into an orthogonal eigenvector matrix paired with pairwise distinct integer eigenvalues, simultaneously a valid eigendecomposition and one that inherits the corresponding CFI graphs' k-WL indistinguishability. Steps 1 and 3 are otherwise standard CFI machinery.

Step 1: CFI Graph Construction. Our proof begins with a CFI base graph G that is 3-regular, has $2n$ vertices, and has treewidth of at least $\Omega(n)$. Such graphs exist by [42, 43]. By Neuen [33], the $\Omega(n)$ treewidth of the base graph G implies that the CFI pair (G_0, G_1) is $\Omega(n)$-WL-indistinguishable. By [41], the two CFI graphs are non-isomorphic. The reason for the assumption that G is 3-regular will be apparent later on.

We note that if the CFI graphs G_0, G_1 were already simple spectra, the proof would be immediate. However, we find that the adjacency matrix in these constructions is typically far from having a simple spectrum (see Section A.5 for a formal proof). Therefore, we encode CFI graphs as orthogonal matrices, and use these matrices to define a new multigraph, for which these orthogonal matrices will serve as an eigendecomposition.

Step 2: Orthogonal Encoding of the CFI Graphs In this step we construct an *orthogonal integral encoding* of a CFI graph G_U as a matrix

$$\tilde{X}_{G,U} = [\, X \mid X' \mid I \,] \in \mathbb{Z}^{8n \times 8n}.$$

The blocks X and X' are indexed by a fiber vertex (w, S) and a base edge e, and are defined via

$$X((w,S),e) = \begin{cases} +1 & e \in E(w) \cap S, \\ -1 & e \in E(w) \setminus S, \\ 0 & e \notin E(w), \end{cases} \qquad X'((w,S),e) = \begin{cases} +X((w,S),e) & w = u, \\ -X((w,S),e) & w = v, \\ 0 & \text{otherwise}, \end{cases} \qquad \text{where } e = (u,v)$$

The matrix $I_{G,U}$ is indexed by a fiber vertex (w, S) and a base vertex v, and is of the general form $I((w,S),v) = W(w,v)$, where the matrix W is defined in Eq. (4) in the appendix. An important property of this matrix is that its first column is the all-ones vector.

An example of the matrix $\tilde{X}_{G,U} = [X \mid X' \mid I]$ for the graph $G = C_3$ is provided in the appendix in Section A.4 and Fig. 2.

The columns of the matrix $\tilde{X}_{G,U}$ are orthogonal. This is shown in Lemma 5 in the appendix, and this result relies on the fact that the base graph G is 3-regular. To get some intuition for the importance of the regularity assumption, we explain here why the 3-regularity assumption leads to the matrix $\tilde{X}_{G,U}$ being a square $8n \times 8n$ dimensional matrix.

In general, the matrix $\tilde{X}_{G,U}$ has $|V(G_U)|$ rows and $2|E|+|V|$ columns. Since the base graph G is a 3-regular graph with $|V| = 2n$ vertices, it has $|E| = 3n$ edges, and so the number of columns of $\tilde{X}_{G,U}$ will be $2|E| + |V| = 8n$. The matrix will also have $|V(G_U)| = 8n$ rows. This is because for each of the $2n$ vertices v in the base graph, there are in total $2^3 = 8$ sets S in $E(v)$, as $|E(v)| = 3$. Half of these sets S can be paired with v to form a fiber vertex (v, S). Therefore $|V(G_U)| = 2n \cdot 4 = 8n$.

Step 3: Multigraph Construction. We now use the orthogonal matrix $\tilde{X}_{G,U}$ to form a new *weighted graph* encoded by normalizing the columns of $\tilde{X}_{G,U}$, and then defining an adjacency matrix $A_{G,U} =$

$\tilde{X}_{G,U}\,\mathrm{diag}(\vec{\lambda})(\tilde{X}_{G,U})^\top$, where we choose pairwise distinct integer eigenvalues $\lambda_1 < \cdots < \lambda_{8n}$. This yields a symmetric matrix $A_{G,U}$ with integer values and a simple spectrum (see the proof of Theorem 1 in Section A.2). By assigning a sufficiently large eigenvalue to the constant eigenvector, the entries of $A_{G,U}$ become non-negative integers, producing a proper multigraph.

In conclusion, we obtain a pair of symmetric matrices $A_{G,0}$ and $A_{G,1}$ which encode simple-spectrum multigraphs. These graphs were obtained from non-isomorphic graphs G_0, G_1 which are k-WL-indistinguishable, and in the appendix we show how this implies that the multigraphs represented by $A_{G,0}$ and $A_{G,1}$ are also k-WL-indistinguishable, which then concludes the proof of the theorem. □

As many GNNs are upper-bounded in their expressivity by a k-WL graph isomorphism test, we obtain an immediate corollary.

Corollary 1 (Incompleteness of GNNs on Graphs with a Simple Spectrum)**.** *The widely-used Graph Neural Networks PPGN [8], k-GNNs [9], k-WL-aligned Graph Transformers [13], N2GNNs [30], and d-DRFWL(2) GNNs [31] cannot distinguish all simple-spectrum graphs.*

5 Canonicalizing eigendecompositions of graphs with a simple spectrum

We saw in Corollary 1 that even powerful high order GNNs cannot distinguish between all simple-spectrum graphs. This can be seen as theoretical justification for the necessity to integrate the spectral eigendecomposition directly into GNNs to obtain improved expressivity. This is exemplified by the prevalent practice of using spectral positional encodings (e.g. Laplacian Positional Encodings [19]) to enhance the expressivity of GNNs and graph transformers [19, 13], as this spectral encoding adds information which k-WL methods cannot capture. However, the spectral encoding contains sign ambiguities (in the simple-spectrum case), and previous sign-invariant methods for handling these ambiguities cannot distinguish between all simple-spectrum graphs [16].

In this section we provide a solution: a complete canonicalization for the eigendecomposition of simple-spectrum graphs. We begin by defining such a canonicalization.

The input to a canonicalization of a simple-spectrum eigendecomposition will be a column-wise orthonormal $n \times k$ matrix U, where $n \geq k$. The columns of the matrix represent the top k eigenvectors of a symmetric matrix representing the graph, such as the graph adjacency or Laplacian matrices. Under the simple-spectrum assumption, the matrix U is uniquely defined up to a sign ambiguity in choosing each of the eigenvectors, and a permutation ambiguity coming from the ambiguity in ordering the nodes of the graph. Thus, the ambiguity group of U is $S_n \times \{-1, 1\}^k$, where $\{-1, 1\}^k$ corresponds to the ambiguity group $\mathcal{A}(\vec{m}_k)$ when all eigenmultiplicities are 1 (see Section 3).

Definition 3. Let $k \leq n$ be natural numbers. A canonicalization of a simple-spectrum eigendecomposition $c : \mathbb{R}^{n\times k} \to \mathbb{R}^{n\times k}$ is a mapping which is $S_n \times \{-1, 1\}^k$ invariant, and satisfies that $c(U)$ is in the orbit of U, for all column-wise orthonormal matrices $U \in \mathbb{R}^{n\times k}$.

A canonicalization c can be composed with any 'backbone' model ψ to obtain a model $\psi \circ c$ which is invariant. Such models will be able to distinguish between simple-spectrum graphs, which also leads to universality guarantees, as we will discuss in Theorem 3 below. Motivated by these observations, we now discuss how to construct a canonicalization.

$$U = \begin{pmatrix} +1 & -1 & +1 & 0 \\ 0 & -1 & +1 & -1 \\ -1 & +1 & -1 & 0 \\ 0 & +1 & -1 & +1 \end{pmatrix} \xrightarrow{\text{Partition}} \left(\begin{array}{cccc} 0 & -1 & +1 & -1 \\ 0 & +1 & -1 & +1 \\ \hline +1 & -1 & +1 & 0 \\ -1 & +1 & -1 & 0 \end{array}\right) \xrightarrow{\text{Refine}} \left(\begin{array}{cccc} 0 & -1 & +1 & -1 \\ 0 & +1 & -1 & +1 \\ \hline +1 & -1 & +1 & 0 \\ -1 & +1 & -1 & 0 \end{array}\right)$$
$$\xrightarrow{\text{Solve}} \left(\begin{array}{cccc} 0 & +1 & +1 & +1 \\ 0 & -1 & -1 & -1 \\ \hline +1 & +1 & +1 & 0 \\ -1 & -1 & -1 & 0 \end{array}\right) \xrightarrow{\text{Match}} \begin{pmatrix} -1 & -1 & -1 & 0 \\ 0 & -1 & -1 & -1 \\ 0 & +1 & +1 & +1 \\ +1 & +1 & +1 & 0 \end{pmatrix} = \mathrm{PRiSM}(U) \tag{1}$$

Stage 1: PARTITION (Algorithm 3 in the appendix).

In the first stage, we define a (partial) order on the vertices of the graph. Each vertex v is assigned the sign-invariant signature $\sigma(v) = (|U_{v,1}|, \ldots, |U_{v,k}|)$, and then the vertices are sorted lexicographically

according to this signature. When each vertex is assigned a unique signature, this defines a unique ordering on the vertices. In the general case, this only groups vertices into classes $\{C_\ell\}$ on which $|U_i| = |U_j|$ for all $i, j \in C_\ell$. Thus, there is a canonical ordering between the different C_ℓ, but not within each class.

In our example, after applying the partial ordering according to σ we remain with a matrix whose first two rows share absolute-value signature $(0, 1, 1, 1)$ while the remaining rows share the absolute value signature $(1, 1, 1, 0)$, thus yielding a partition of the nodes into two classes $C_1 = \{1, 2\}, C_2 = \{3, 4\}$.

Stage 2: REFINE (Algorithm 4 in the appendix). In the second stage, we refine the partition from Stage 1 by running a spectral variant of the Weisfeiler–Leman (WL) color refinement procedure [38]: each node v is initialized with the absolute-value signature $h_v^{(0)} = U_v \odot U_v$ (where $\odot$ represents elementwise multiplication, which is invariant to sign ambiguities), and colors are then iteratively refined using the multiset of edge features $\{(h_u^{(t)}, U_v \odot U_u) : u \in [n]\}$ until the partition stabilizes (Algorithm 4). At the fixed point, every two nodes i, j in the same class share the same multiset of pairwise products $\{U_i \odot U_k : k \in C_\ell\} = \{U_j \odot U_k : k \in C_\ell\}$, which is the structural property we will use in the canonicalization proof. In our running example, this stage does not further refine the initial partition from Stage 1.

Stage 3: SOLVE (Algorithm 5 in the appendix). The Solve step determines the signs of the eigenvectors by requiring that each node partition C contains a node whose corresponding row has only non-negative entries. In our example, this requirement on C_1 and C_2 simultaneously is satisfied if $s \in \{-1, 1\}^4$ is one of the two vectors satisfying $s_1 = s_3 = -s_2 = -s_4$. We note that while this solution is not determined uniquely, the 4×4 output is determined uniquely, up to permutation!

In general, we will show in Section B that the constraints that each C_k contains a row with non-negative entries can be written as linear equations in the field $\mathbb{Z}_2 = \{0, 1\}$, and that (after eliminating linearly dependent equations) these equations uniquely determine the sign of all eigenvectors, up to graph automorphisms. As a result, we obtain a valid canonicalization.

Stage 4: MATCH Once the sign of the eigenvectors is determined, we can apply standard lexicographical sorting to the rows of the matrix to fix the vertex permutation. This concludes the canonicalization process.

This procedure induces a general canonicalization of eigendecompositions of any simple-spectrum graph. We call the resulting canonicalization PRiSM, short for **P**artition **R**efine **S**olve **M**atch, naming the four pipeline stages above (the final MATCH stage uses lex-sort to match each row to its canonical position). In the appendix (Section B) we give the full formal construction (assembling per-class parity-check matrices, embedding them in a global linear system over $\mathbb{Z}_2$, and lex-minimizing over its solution coset) and prove invariance to vertex permutations and sign flips, encapsulated by the theorem below.

Theorem 2 (Simple Spectrum Canonicalization)**.** *For every natural $k \leq n$, the base procedure* PRiSM *of Algorithm 1, applied to a k-eigenvector matrix $U \in \mathbb{R}^{n \times k}$ of a simple-spectrum graph, is a valid canonicalization in the sense of Definition 3.*

Universality. The theoretical significance of canonicalization is that it implies universal approximation, as shown in previous papers on canonicalizations and frames [44, 45]. In our context, we can prove universality on simple-spectrum graphs, when composing the canonicalization with any function which is universal on sets of vector features, such as DeepSets [23, 46, 47] and Transformers [25].

Theorem 3 (Universal Approximation over Graphs with a Simple Spectrum)**.** *Fix $M > 0$ and $\delta > 0$. Let $\mathcal{K}(N, M, \delta)$ denote the set of simple-spectrum weighted graphs with at most N vertices, edge weights bounded by M, and minimum eigenvalue gap at least δ (i.e. $|\lambda_i - \lambda_j| \geq \delta$ for $i \neq j$). Let $f : \mathcal{K}(N, M, \delta) \to \mathbb{R}$ be a continuous, isomorphism-invariant function and let $\epsilon > 0$. Denote the eigendecomposition of a graph g by $(U_g, \vec{\lambda}_g)$. Then for any universal set model there exists a model instantiation ψ such that*

$$\forall g \in \mathcal{K}(N, M, \delta), \quad |f(g) - \psi \circ \mathrm{PRiSM}(U_g, \vec{\lambda}_g)| < \epsilon.$$

The eigengap assumption $\delta > 0$ is necessary, not restrictive, since simple-spectrum graphs form an open, non-compact subset of $\mathcal{G}_n$ while uniform universal approximation requires compactness. In

Section D we formalize the notion of universal set models and prove Theorem 3, settling an open problem of Hordan et al. [16].

Relation to previously proposed canonicalizations. Other canonicalizations for eigendecompositions of simple-spectrum graphs were proposed in the past, such as MAP [21] and OAP [22]. However, these canonicalizations operate on each eigenvector independently, and as a result they are not able to choose a canonical sign for *uncanonicalizable eigenvectors*, which are eigenvectors v which can be reordered to obtain $-v$. For example, the last three eigenvectors (columns) of the matrix U in (1) are uncanonizable, so that MAP and OAP will not be able to choose a canonical sign to either of them. In contrast, our canonicalization, which is valid for all simple-spectrum graphs, does define a global canonical sign for all eigenvalues as shown in (1).

Practical aspects. The canonicalization PRiSM can be computed using $O(nk \log n + k^3 + nk^2)$ basic operations (see Section C.1). While this complexity is cubic in k, we note that k is typically not very large in our experiments, and, for fixed k, the complexity is $O(n \log n)$ in n. Moreover, the canonicalization is only carried out as a preprocessing step and not during training, and its complexity is comparable with the $O(km + k^2 n)$ complexity of computing the eigendecomposition of a graph with m edges using the Lanczos algorithm, which is necessary for all canonicalizations.

Moreover, we note that the $O(nk \log n + k^3 + nk^2)$ complexity is a worst-case bound which is typically not attained. When the absolute-value signature is injective, which holds on 3.3–45.3% of validation graphs across ZINC and the eleven OGB-MOL datasets (see Section C.1 and Table 4 in the appendix), canonicalization reduces to an algorithm we call FASTSIGN (see Algorithm 2 in the appendix for a full description) that has only $O(nk \log n)$ run-time. On the remaining graphs, the $\mathbb{Z}_2$ system solved by SOLVE has median row-count at most 3 across every dataset, so the cubic k^3 factor reduces to a small constant in practice, see Section C.1 for details.

Finally, we extend PRiSM heuristically to graphs which are not simple spectrum: we first apply PRiSM on all $m = 1$ eigenspaces, producing a canonical sign assignment for these eigenspaces, together with a node ordering. This node ordering is further refined by generalizations of Stages 1 and 2 to higher-order eigenspaces. The rotation degree of freedom in high-order eigenspaces ($m > 1$) is then canonicalized according to the first m nodes obtained from the ordering, using a QR decomposition. On a simple-spectrum graph this extension coincides with the base PRiSM of Algorithm 1 and inherits its completeness, whereas on graphs with eigenvalue multiplicities greater than 1 it is heuristic. Full pseudocode for the extension is in Algorithm 6 in the Appendix.

6 Experiments

We evaluate PRiSM on graph learning benchmarks, and compare against vanilla Laplacian Positional Encodings [19] (which ignore sign ambiguity), two canonicalizations: MAP [21], OAP [22], and the sign-invariant SignNet [20]. **Across all three benchmarks** PRiSM **performs comparably to or beats every baseline**. We hypothesize that PRiSM outperforms the incomplete canonicalizations because they either lose information [20] or are non-deterministic [21, 22] on graphs with simple spectrum. We first benchmark BREC graph distinguishability (Table 1), then molecular property prediction with GNN backbones (Table 2) and Transformer backbones (Table 3). Full experiment details are in Section E. Code is available at `https://github.com/snirhordan/prism-canonicalization`.

6.1 BREC expressivity benchmark

We evaluate the impact of the completeness or lack thereof of canonicalizations on the downstream base model's ability to distinguish among highly regular and 1-WL indistinguishable graphs in the BREC dataset [26]. For that end, using BREC's Basic, Regular, Extension, and CFI [17] graph categories, we measure graph-pair distinguishability of each canonicalization using a base DeepSets model with matched parameter budgets, see Table 1. PRiSM scores $212/360$ vs. $199/360$ for MAP and OAP, with gains on Regular ($+8$) and Extension ($+5$) categories. We note that distinguishing CFI graphs requires high-order k-WL (and equivalently expressive GNNs), and that CFI graphs have eigenvalue multiplicities up to $\Theta(n)$ (Theorem 4), and therefore we do not expect PRiSM, MAP, or OAP to distinguish them well.

Table 1: *BREC expressivity benchmark (360 pairs). All methods use matched parameter budget (∼12K) with a spectral positional encoding processed by a DeepSets embedding.*

Method	Basic /60	Regular /100	Ext. /100	CFI /100	Total /360
PRiSM **(ours)**	**60**	**50**	**99**	3	**212**
MAP	**60**	42	94	3	199
OAP	**60**	42	94	3	199

6.2 Molecular Property Prediction

Following the setup by Ma et al. [22], we compare molecular property prediction accuracy of PRiSM against MAP, OAP and SignNet on ZINC 12K [28, 19] and OGB MOLTOX21, MOLPCBA [27] across GatedGCN [48] and PNA [49] backbones at matched parameter budgets (Table 2). We report mean ± std over four seeds. The per-cell hyperparameter configuration is in Tables 5 and 6. We find that PRiSM is best or tied-best on five of six categories (Table 2), supporting the hypothesis that a complete, deterministic canonicalization translates to downstream gains.

Table 2: *Molecular property prediction of competing canonicalizations with GatedGCN/PNA backbones.*

	ZINC MAE ↓		MOLTOX21 AUC ↑		MOLPCBA AP ↑	
Method	**GatedGCN**	**PNA**	**GatedGCN**	**PNA**	**GatedGCN**	**PNA**
PRiSM **(ours)**	$\mathbf{0.110 \pm 0.002}$	$\mathbf{0.103 \pm 0.003}$	$\mathbf{0.783 \pm 0.006}$	0.762 ± 0.004	$\mathbf{0.272 \pm 0.003}$	$\mathbf{0.278 \pm 0.001}$
SignNet	0.121 ± 0.002	0.105 ± 0.007	0.769 ± 0.007	0.744 ± 0.002	0.260 ± 0.002	OOM
MAP	0.123 ± 0.003	0.104 ± 0.003	0.777 ± 0.004	$\mathbf{0.763 \pm 0.004}$	0.268 ± 0.002	0.274 ± 0.003
OAP	0.126 ± 0.004	0.106 ± 0.002	$\mathbf{0.783 \pm 0.006}$	0.761 ± 0.008	0.270 ± 0.002	0.272 ± 0.002

6.3 Transformer Backbone on OGB and Alchemy

We next evaluate the efficacy of our complete PRiSM-canonicalized Laplacian Positional Encoding (LapPE) on tasks with a Transformer backbone. Moreover, we leverage an artifact from our canonicalization, a node ordering that is used as positional encoding supplementary to the canonicalized LapPE. Concretely, we pair PRiSM with the graph Transformer of Kim et al. [25] on OGB [27] (MOLHIV, MOLCLINTOX, MOLBBBP) and ALCHEMY [29], a 12-task quantum-chemistry regression benchmark. Per-dataset hyperparameter configuration is in Section E.3. As shown in Table 3, PRiSM is best on all four datasets.

Table 3: *Transformer backbone results on OGB [27] and* ALCHEMY *[29] molecular datasets.*

	ROC-AUC ↑			MAE ↓
Method	**MOLHIV**	**MOLCLINTOX**	**MOLBBBP**	**ALCHEMY**
PRiSM **(ours)**	$\mathbf{0.7430 \pm 0.0233}$	$\mathbf{0.8576 \pm 0.0038}$	$\mathbf{0.7088 \pm 0.0293}$	$\mathbf{0.11956 \pm 0.00130}$
LPE	0.6891 ± 0.0068	0.7905 ± 0.0444	0.6798 ± 0.0179	0.12655 ± 0.00056
MAP	0.7221 ± 0.0190	0.7900 ± 0.0440	0.7029 ± 0.0019	0.13058 ± 0.00038
OAP	0.6878 ± 0.0197	0.7820 ± 0.0530	0.6825 ± 0.0323	0.13216 ± 0.00488

7 Conclusion, Limitations and Future Work

In this paper, we uncovered that k-WL tests are incomplete on graphs with a simple spectrum and presented a novel canonicalization to achieve completeness. Empirically, we show that our complete canonicalization outperforms competing canonicalizations. One limitation is that we do not address completeness for higher-order eigenmultiplicities. An additional limitation is that our incompleteness result is for the class of multi-graphs. Attaining similar results for the smaller class of combinatorial graphs is an interesting avenue for future work.

Acknowledgements

N.D. and S.H. are supported by the Israeli Science Foundation grant no. 272/23. S.H. is supported by the Gloria and Ken Levy Foundation Fellowship.

T.S. is supported by the European Union (CountHom, 101077083). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

References

[1] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28, 2015.

[2] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. Pmlr, 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

[3] Gabriele Corso, Hannes Stark, Stefanie Jegelka, Tommi Jaakkola, and Regina Barzilay. Graph neural networks. *Nature Reviews Methods Primers*, 4(1):17, 2024. doi:10.1038/s43586-024-00294-7.

[4] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008. doi:10.1109/TNN.2008.2005605.

[5] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

[6] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[7] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*, 2018. URL https://arxiv.org/abs/1810.00826.

[8] Haggai Maron, Heli Ben-Hamu, Hadar Serviansky, and Yaron Lipman. Provably powerful graph networks. *Advances in neural information processing systems*, 32, 2019.

[9] Christopher Morris, Martin Ritzert, Matthias Fey, William L. Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. Weisfeiler and leman go neural: Higher-order graph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4602–4609, Jul. 2019. doi:10.1609/aaai.v33i01.33014602. URL https://ojs.aaai.org/index.php/AAAI/article/view/4384.

[10] Boris Weisfeiler and A. A. Lehman. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsia*, 9, 1968. URL https://www.iti.zcu.cz/wl2018/pdf/wl_paper_translation.pdf.

[11] Christopher Morris, Gaurav Rattan, and Petra Mutzel. Weisfeiler and leman go sparse: Towards scalable higher-order graph embeddings. *Advances in Neural Information Processing Systems*, 33:21824–21840, 2020.

[12] Christopher Morris, Yaron Lipman, Haggai Maron, Bastian Rieck, Nils M Kriege, Martin Grohe, Matthias Fey, and Karsten Borgwardt. Weisfeiler and leman go machine learning: The story so far. *Journal of Machine Learning Research*, 24(333):1–59, 2023. URL https://jmlr.org/papers/v24/22-0240.html.

[13] Luis Müller and Christopher Morris. Aligning transformers with weisfeiler-leman. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 36654–36704. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/muller24c.html.

[14] F. Thomson Leighton and Gary l. Miller. Certificates for graphs with distinct eigen values. Original Manuscript, 1979.

[15] Daniel Neuen. Parameterized complexity of graph isomorphism testing. *Computer Science Review*, 60:100918, May 2026. ISSN 15740137. doi:10.1016/j.cosrev.2026.100918. URL https://linkinghub.elsevier.com/retrieve/pii/S1574013726000274.

[16] Snir Hordan, Maya Bechler-Speicher, Gur Lifshitz, and Nadav Dym. Spectral graph neural networks are incomplete on graphs with a simple spectrum. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[17] Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992. ISSN 1439-6912. doi:10.1007/BF01305232. URL https://doi.org/10.1007/BF01305232.

[18] Neil Immerman and Eric Lander. Describing Graphs: A First-Order Approach to Graph Canonization. In Alan L. Selman, editor, *Complexity Theory Retrospective*, pages 59–81. Springer New York, New York, NY, 1990. ISBN 978-1-4612-4478-3. doi:10.1007/978-1-4612-4478-3_5. URL https://doi.org/10.1007/978-1-4612-4478-3_5.

[19] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023. URL https://jmlr.org/papers/v24/22-0567.html.

[20] Derek Lim, Joshua David Robinson, Lingxiao Zhao, Tess Smidt, Suvrit Sra, Haggai Maron, and Stefanie Jegelka. Sign and basis invariant networks for spectral graph representation learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Q-UHqMorzil.

[21] George Ma, Yifei Wang, and Yisen Wang. Laplacian canonization: A minimalist approach to sign and basis invariant spectral embedding. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=1mAYtdoYw6.

[22] George Ma, Yifei Wang, Derek Lim, Stefanie Jegelka, and Yisen Wang. A canonicalization perspective on invariant and equivariant learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=jjcY92FX4R.

[23] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/f22e4747da1aa27e363d86d40ff442fe-Paper.pdf.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[25] Jinwoo Kim, Saeyoon Oh, and Seunghoon Hong. Transformers generalize deepsets and can be extended to graphs & hypergraphs. *Advances in Neural Information Processing Systems*, 34: 28016–28028, 2021.

[26] Yanbo Wang and Muhan Zhang. An empirical study of realized GNN expressiveness. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 52134–52155. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/wang24cl.html.

[27] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *Advances in neural information processing systems*, 33:22118–22133, 2020.

[28] John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52 (7):1757–1768, 2012. doi:10.1021/ci3001277.

[29] Guangyong Chen, Pengfei Chen, Chang-Yu Hsieh, Chee-Kong Lee, Benben Liao, Renjie Liao, Weiwen Liu, Jiezhong Qiu, Qiming Sun, Jie Tang, et al. Alchemy: A quantum chemistry dataset for benchmarking ai models. *arXiv preprint arXiv:1906.09427*, 2019. URL `https://arxiv.org/abs/1906.09427`.

[30] Jiarui Feng, Lecheng Kong, Hao Liu, Dacheng Tao, Fuhai Li, Muhan Zhang, and Yixin Chen. Extending the design space of graph neural networks by rethinking folklore weisfeiler-lehman. *Advances in Neural Information Processing Systems*, 36:9029–9064, 2023.

[31] Junru Zhou, Jiarui Feng, Xiyuan Wang, and Muhan Zhang. Distance-restricted folklore weisfeiler-leman gnns with provable cycle counting power. *Advances in Neural Information Processing Systems*, 36:14293–14337, 2023.

[32] Fabrizio Frasca, Beatrice Bevilacqua, Michael Bronstein, and Haggai Maron. Understanding and extending subgraph gnns by rethinking their symmetries. *Advances in Neural Information Processing Systems*, 35:31376–31390, 2022.

[33] Daniel Neuen. Homomorphism-Distinguishing Closedness for Graphs of Bounded Tree-Width. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science (STACS 2024)*, volume 289 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 53:1–53:12, Dagstuhl, Germany, 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. ISBN 978-3-95977-311-9. doi:10.4230/LIPIcs.STACS.2024.53. URL `https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.STACS.2024.53`. ISSN: 1868-8969.

[34] László Babai, D. Yu. Grigoryev, and David M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, page 310–324, New York, NY, USA, 1982. Association for Computing Machinery. ISBN 0897910702. doi:10.1145/800070.802206. URL `https://doi.org/10.1145/800070.802206`.

[35] Devin Kreuzer, Dominique Beaini, Will Hamilton, Vincent Létourneau, and Prudencio Tossou. Rethinking graph transformers with spectral attention. *Advances in neural information processing systems*, 34:21618–21629, 2021.

[36] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021. URL `https://arxiv.org/abs/2012.09699`.

[37] Jingchu Gai, Yiheng Du, Bohang Zhang, Haggai Maron, and Liwei Wang. Homomorphism expressivity of spectral invariant graph neural networks. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=rdv6yeMFpn`.

[38] Bohang Zhang, Lingxiao Zhao, and Haggai Maron. On the expressive power of spectral invariant graph neural networks. In *Forty-first International Conference on Machine Learning*. URL `https://proceedings.mlr.press/v235/zhang24ck.html`.

[39] Antonis Vasileiou, Juan Cervino, Pascal Frossard, Charilaos I. Kanatsoulis, Christopher Morris, Michael T. Schaub, Pierre Vandergheynst, Zhiyang Wang, Guy Wolf, and Ron Levie. Position: Message-passing and spectral gnns are two sides of the same coin, 2026. URL `https://arxiv.org/abs/2602.10031`.

[40] Terence Tao and Van Vu. Random matrices have simple spectrum. *Combinatorica*, 37(3): 539–553, 2017. doi:10.1007/s00493-016-3363-4.

[41] David E. Roberson. Oddomorphisms and homomorphism indistinguishability over graphs of bounded degree. *Journal of Combinatorial Theory, Series B*, 181:1–61, November 2026. ISSN 00958956. doi:10.1016/j.jctb.2026.07.003. URL https://linkinghub.elsevier.com/retrieve/pii/S0095895626000432.

[42] Zdeněk Dvořák and Sergey Norin. Strongly Sublinear Separators and Polynomial Expansion. *SIAM Journal on Discrete Mathematics*, 30(2):1095–1101, January 2016. ISSN 0895-4801, 1095-7146. doi:10.1137/15M1017569.

[43] Brett Kolesnik and Nick Wormald. Lower Bounds for the Isoperimetric Numbers of Random Regular Graphs. *SIAM Journal on Discrete Mathematics*, 28(1):553–575, January 2014. ISSN 0895-4801, 1095-7146. doi:10.1137/120891265.

[44] Omri Puny, Matan Atzmon, Edward J. Smith, Ishan Misra, Aditya Grover, Heli Ben-Hamu, and Yaron Lipman. Frame averaging for invariant and equivariant network design. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zIUyj55nXR.

[45] Sékou-Oumar Kaba, Arnab Kumar Mondal, Yan Zhang, Yoshua Bengio, and Siamak Ravanbakhsh. Equivariance with learned canonicalization functions. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 15546–15566. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/kaba23a.html.

[46] Tal Amir, Steven Gortler, Ilai Avni, Ravina Ravina, and Nadav Dym. Neural injective functions for multisets, measures and graphs via a finite witness theorem. *Advances in Neural Information Processing Systems*, 36:42516–42551, 2023.

[47] Edward Wagstaff, Fabian B Fuchs, Martin Engelcke, Michael A Osborne, and Ingmar Posner. Universal approximation of functions on sets. *Journal of Machine Learning Research*, 23(151):1–56, 2022. URL https://jmlr.org/papers/v23/21-0730.html.

[48] Xavier Bresson and Thomas Laurent. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553*, 2017. URL https://arxiv.org/abs/1711.07553.

[49] Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *Advances in neural information processing systems*, 33:13260–13271, 2020.

[50] Martin Grohe. The Logic of Graph Neural Networks. In *36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS*, pages 1–17, Rome, Italy, 2021. IEEE. doi:10.1109/LICS52264.2021.9470677. URL https://doi.org/10.1109/LICS52264.2021.9470677.

[51] Tim Seppelt. *Homomorphism Indistinguishability*. Dissertation, RWTH Aachen University, Aachen, 2024. URL https://publications.rwth-aachen.de/record/998785.

[52] Daniel A. Spielman. Spectral and algebraic graph theory. 2025. URL http://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf.

[53] Hermann Von Weyl. Das asymptotische verteilungsgesetz der eigenwerte linearer partieller differentialgleichungen (mit einer anwendung auf die theorie der hohlraumstrahlung). *Mathematische Annalen*, 71:441–479, 1912. URL https://api.semanticscholar.org/CorpusID:120278241.

[54] Ladislav Rampášek, Michael Galkin, Vijay Prakash Dwivedi, Anh Tuan Luu, Guy Wolf, and Dominique Beaini. Recipe for a general, powerful, scalable graph transformer. *Advances in Neural Information Processing Systems*, 35:14501–14515, 2022.

[55] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. doi:10.1016/j.neucom.2023.127063.

A Incompleteness of Weisfeiler–Leman on Simple Spectrum Graphs

A.1 Weisfeiler–Leman Definitions

For completeness we record here the general k-dimensional Weisfeiler–Leman test and its associated indistinguishability relation, both used throughout this appendix and referenced from the main text.

Definition 4 (k-Weisfeiler–Leman Algorithm, see [50])**.** Let $k \geq 1$ and let $g \in \mathcal{G}_n$ with matrix $A \in \mathbb{R}^{n\times n}$, whose diagonal entries A_{vv} are the node features. For $\ell \geq 1$ and $\bar{v} = (v_1, \ldots, v_\ell) \in [n]^\ell$, the *atomic type* of $\bar{v}$ is

$$\mathrm{atp}(\bar{v}) = \Big(\big(A_{v_i v_j}\big)_{i,j\in[\ell]},\ \big([v_i = v_j]\big)_{i,j\in[\ell]} \Big),$$

where $[\cdot]$ denotes the Iverson bracket. For $\bar{v} \in [n]^k$, $w \in [n]$, and $j \in [k]$, write $\bar{v}w = (v_1, \ldots, v_k, w) \in [n]^{k+1}$ and $\bar{v}[w/j] = (v_1, \ldots, v_{j-1}, w, v_{j+1}, \ldots, v_k)$. The *$k$-WL algorithm* computes colorings $C^{(t)} \colon [n]^k \to \mathcal{C}$ via

$$C^{(0)}(\bar{v}) = \mathrm{HASH}\big(\mathrm{atp}(\bar{v})\big),$$
$$C^{(t+1)}(\bar{v}) = \mathrm{HASH}\Big(C^{(t)}(\bar{v}),\ \big\{\!\!\big\{ \big(\mathrm{atp}(\bar{v}w), C^{(t)}(\bar{v}[w/1]), \ldots, C^{(t)}(\bar{v}[w/k])\big) \bigm| w \in [n] \big\}\!\!\big\}\Big),$$

where HASH is a fixed injective map, the same for all graphs.

Since $C^{(t+1)}(\bar{v})$ determines $C^{(t)}(\bar{v})$, every $C^{(t)}(\bar{v})$ determines $\mathrm{atp}(\bar{v})$. For $k = 1$, the colorings of Definition 4 and Definition 1 determine each other via a bijection of colors that does not depend on the graph. For $k \geq 2$, every entry of $\mathrm{atp}(\bar{v}w)$ already occurs in $\mathrm{atp}(\bar{v})$ or in some $\mathrm{atp}(\bar{v}[w/j])$, so Definition 4 coincides with the classical formulation that omits $\mathrm{atp}(\bar{v}w)$ from the multiset [17, 33].

Definition 5 (k-WL Indistinguishability)**.** Two graphs $g, g' \in \mathcal{G}_n$ are *k-WL indistinguishable*, written $g \equiv_k g'$, if $\{\!\{C^{(t)}(g, \bar{v}) \mid \bar{v} \in [n]^k\}\!\} = \{\!\{C^{(t)}(g', \bar{v}) \mid \bar{v} \in [n]^k\}\!\}$ for all $t \geq 0$. The relation $\equiv_k$ is an equivalence relation on $\mathcal{G}_n$, and satisfies the strict refinement hierarchy: for all $k \geq 1$,

$$g \equiv_{k+1} g' \implies g \equiv_k g',$$

but the converse fails in general [17], meaning that $(k+1)$-WL distinguishes strictly more pairs of graphs than k-WL.

For $k = 1$, $\equiv_1$ coincides with the relation of Definition 1, since both colorings determine each other via a graph-independent bijection and two graphs with equal global labels stabilize at the same round.

A.2 Proof of the Main Incompleteness Theorem

In this subsection, we give a proof of Theorem 1. The bulk of the argument is to translate a pair of graphs obtained via the CFI construction considered in [41] to a pair of orthonormal matrices. The CFI construction builds, given a highly connected graph G, two graphs G_0 and G_1 that cannot be distinguished by highly dimensional Weisfeiler–Leman. More precisely, as shown in [33], if G has *treewidth* at least $k+1$, then G_0 and G_1 are not distinguished by the k-dimensional Weisfeiler–Leman algorithm. We will transfer this result to our matrix encoding of CFI graphs. Finally, we obtain the desired matrices by instantiating the CFI construction for 3-regular graphs G of high treewidth, so-called expanders, as constructed and analyzed in [42, 43]. Throughout, one should think of G as being such a graph, although we will be explicit about the particular properties of G needed for each lemma. More precisely, the proof of Theorem 1 proceeds as follows.

1. For some suitable base graph G and $U \in \{0, 1\}$, we define a matrix $\tilde{X}_{G,U}$ based on the CFI graph G_U. This matrix will serve as the matrix of eigenvectors for the multigraphs, which we construct subsequently.
2. In order for $\tilde{X}_{G,U}$ to encode an eigenbasis of a simple-spectrum graph, we show in Lemma 5 that the columns of $\tilde{X}_{G,U}$ are orthogonal.
3. Now we choose a diagonal matrix with distinct diagonal entries D and consider the matrices $A_{G,U} \coloneqq \tilde{X}_{G,U} D (\tilde{X}_{G,U})^\top$. Our construction will be such that $A_{G,U}$ can be ensured to have only non-negative integral entries. Hence, $A_{G,U}$ is a multigraph with simple spectrum.

4. To argue that $A_{G,0}$ and $A_{G,1}$ are non-isomorphic, we show that $A_{G,0} \cong A_{G,1}$ would imply that the CFI graphs G_0 and G_1 are isomorphic, which is known not to hold [41].
5. Lastly, we show that the Weisfeiler–Leman algorithm fails to distinguish $A_{G,0}$ and $A_{G,1}$. To that end, we show that running Weisfeiler–Leman on $A_{G,0}$ and $A_{G,1}$ can be simulated by running Weisfeiler–Leman on G_0 and G_1. By the choice of G as a high-treewidth graph, it follows that G_0 and G_1 are not distinguished by Weisfeiler–Leman [33] and hence the same holds for $A_{G,0}$ and $A_{G,1}$.

A.2.1 Definition of matrices from CFI graphs

Let G be a connected graph. Let G_U denote the CFI graph of G for $U \subseteq V(G)$ as defined in Definition 2. By [41, Corollary 3.7], there are two CFI graphs for G up to isomorphism. That is, it holds that $G_U \cong G_{U'}$ if, and only if, $|U| \equiv |U'| \pmod 2$. The two graphs are typically denoted by G_0, i.e., $|U| = 0$, and G_1, i.e., $|U| = 1$. Subsequently, we will nevertheless work with G_U in order to be able to define matrices based on CFI graphs.

Fix an ordering on $V(G)$. We think of the edges of G as being oriented according to this ordering, i.e. every edge points from the vertex with lower number to the vertex of higher number. Fix $U \subseteq V(G)$. We define the following four matrices whose rows are indexed by $V(G_U)$. See Fig. 2 in Section A.4 for an example.

1. The columns of $X_{G,U}$ are indexed by $E(G)$. The entries are given by

$$X_{G,U}((v,S),e) \coloneqq \begin{cases} 1, & \text{if } e \in E(v) \text{ and } e \in S, \\ -1, & \text{if } e \in E(v) \text{ and } e \notin S, \\ 0, & \text{if } e \notin E(v). \end{cases} \tag{2}$$

2. The columns of $X'_{G,U}$ are indexed by $E(G)$. The entries are given by

$$X'_{G,U}((v,S),e) \coloneqq \begin{cases} X_{G,U}((v,S),e), & \text{if } v \text{ is the first vertex in } e, \\ -X_{G,U}((v,S),e), & \text{if } v \text{ is the second vertex in } e, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

3. The columns of $I_{G,U}$ are indexed by $V(G)$. Let W be the following $V(G) \times V(G)$ matrix; its first column is the all-ones vector and its remaining columns are pairwise orthogonal and orthogonal to the first column, hence form a basis for the orthogonal complement of the all-ones direction in $\mathbb{R}^{|V(G)|}$.

$$W \coloneqq \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 & 1 \\ 1 & -1 & 1 & 1 & \cdots & 1 & 1 \\ 1 & 0 & -2 & 1 & \cdots & 1 & 1 \\ 1 & 0 & 0 & -3 & \cdots & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & -(|V(G)|-2) & 1 \\ 1 & 0 & 0 & 0 & \cdots & 0 & -(|V(G)|-1) \end{pmatrix}. \tag{4}$$

   The entries of $I_{G,U}$ are given by

$$I_{G,U}((v,S),u) \coloneqq W(v,u). \tag{5}$$

4. Define $\tilde{X}_{G,U}$ as the concatenation of $X_{G,U}$, $X'_{G,U}$, and $I_{G,U}$. The rows of $\tilde{X}_{G,U}$ are indexed by $V(G_U)$. The columns of $\tilde{X}_{G,U}$ are indexed by $E(G) \uplus E(G) \uplus V(G)$.

We observe that $\tilde{X}_{G,U}$ is a square matrix when G is a 3-regular graph.

Lemma 1. *If G is a 3-regular $2n$-vertex graph, then $\tilde{X}_{G,U}$ is an $8n \times 8n$ matrix.*

Proof. The number of edges in a 3-regular $2n$-vertex graph is $3n$. For every $u \in V(G)$, the graph G_U contains $2^{3-1} = 4$ vertices (u,S). Hence, there are $8n$ vertices in G_U. It follows that $\tilde{X}_{G,U}$ has $8n$ rows and $3n + 3n + 2n = 8n$ columns. □

A.2.2 Orthogonality

We show that the columns of $\tilde{X}_{G,U}$ are orthogonal. Hence, they can serve as the eigenbasis of a simple-spectrum multigraph.

Lemma 2. *If every vertex in G has degree at least 3, then the columns of $X_{G,U}$ are orthogonal.*

Proof. Let $e_1, e_2 \in E(G)$ be distinct edges. If e_1 and e_2 do not share a vertex, then there is no row index (v, S) such that the columns of e_1 and e_2 are both non-zero. Hence, in this case, the columns of e_1 and e_2 are clearly orthogonal.

So suppose that $e_1 = uv$ and $e_2 = vw$ for $u \neq w$. The column of vw is zero on the block for u and the column of uv is zero on the block for w. Hence, it remains to consider the block for v. Thus, the inner product of the columns of e_1 and e_2 is

$$\sum_{\substack{S \subseteq E(v) \\ |S| \equiv |\{v\} \cap U| \pmod 2}} (-1)^{vu \in S} (-1)^{vw \in S} = 2^{d-3} - 2 \cdot 2^{d-3} + 2^{d-3} = 0 \tag{6}$$

for $d := |E(v)|$. Here, the sum is over all row indices (v, S) in the block for v. The condition $|S| \equiv |\{v\} \cap U| \pmod 2$ comes from the definition of the vertex set of G_U. By Eq. (2), we need to distinguish cases whether $vu \in S$ and $vw \in S$. If both edges or neither of them is contained in S, then the product of the two entries is $+1$, otherwise it is -1.

Finally, we count how often each of these four scenarios arises. Here, we use that the sets S are solutions to a system of linear equations over $\mathbb{Z}_2$. The variables x_e are indexed by edges $e \in E(v)$ incident to v. The constraints are

$$\sum_{e \in E(v)} x_e = |\{v\} \cap U|$$

and $x_{vu} = 1$ or $x_{vu} = 0$, and $x_{vw} = 0$ or $x_{vw} = 1$, depending on the given scenario. In other words, the sets S are in bijection with solutions to a d-variable system of 3 linearly independent equations over $\mathbb{Z}_2$. Hence, the number of solutions is 2^{d-3}. □

Lemma 3. *If every vertex in G has degree at least 3, then the columns of $X'_{G,U}$ are orthogonal.*

Proof. The argument is similar to Lemma 2. As above, we need to consider only the case of two columns indexed by $e_1 = uv$ and $e_2 = vw$ for $u \neq w$ (with v the shared vertex). The definition of $X'_{G,U}$ negates each entry depending on whether v is the first or second endpoint of the edge in the fixed total ordering of $V(G)$. Concretely, if $p_1 \in \{+1, -1\}$ is the sign contributed by edge e_1 (i.e., $p_1 = +1$ if v is the first vertex of e_1, and $p_1 = -1$ otherwise), and p_2 is defined analogously for e_2, then the inner product over the block for v is

$$\begin{aligned} &\sum_{\substack{S \subseteq E(v) \\ |S| \equiv |\{v\} \cap U| \pmod 2}} \big(p_1 \cdot (-1)^{vu \in S}\big)\big(p_2 \cdot (-1)^{vw \in S}\big) \\ &\quad = p_1 p_2 \sum_{\substack{S \subseteq E(v) \\ |S| \equiv |\{v\} \cap U| \pmod 2}} (-1)^{vu \in S} (-1)^{vw \in S}. \end{aligned}$$

The sum on the right equals $2^{d-3} - 2 \cdot 2^{d-3} + 2^{d-3} = 0$ by the same four-case counting as in Eq. (6), regardless of the values of $p_1, p_2 \in \{+1, -1\}$ (i.e., regardless of the ordering of u, v, w). This parity-symmetry argument covers all six orderings of $\{u, v, w\}$ simultaneously: the global sign factor $p_1 p_2$ is irrelevant to the cancellation, which depends only on the balanced count of subsets S with $vu \in S$, $vw \in S$, both, or neither. □

Lemma 4. *The columns of W are pairwise orthogonal. If, in addition, G is d-regular, the columns of $I_{G,U}$ are pairwise orthogonal as well.*

Proof. The columns of W are orthogonal by direct computation of the inner products in Eq. (4). When G is d-regular, each base vertex v contributes the same number 2^{d-1} of fiber rows replicating $W(v, \cdot)$, so $I_{G,U}^\top I_{G,U} = 2^{d-1} W^\top W$ and the orthogonality transfers to $I_{G,U}$. □

Lemma 5. *If G is a d-regular graph for $d \geq 3$, then the columns of $\tilde{X}_{G,U}$ are orthogonal.*

Proof. By Lemmas 2 to 4, the columns of $X_{G,U}$, $X'_{G,U}$, and $I_{G,U}$ are respectively orthogonal. Thus, it remains to consider inner products of pairs of columns stemming from different matrices $X_{G,U}$, $X'_{G,U}$, and $I_{G,U}$. Distinguish cases:

1. One column $e \in E(G)$ from $X_{G,U}$ or $X'_{G,U}$ and one column $u \in V(G)$ from $I_{G,U}$.

   The crucial observation is that the columns of $I_{G,U}$ are constant on sets of vertices (v, S) belonging to the same vertex $v \in V(G)$. We argue that the columns of $X_{G,U}$ and $X'_{G,U}$ are orthogonal to the subspace of vectors in $\mathbb{R}^{V(G_U)}$ that are constant on these fibers. Let $u \in V(G)$ and consider the inner product of a column indexed by $e \in E(G)$ of $X_{G,U}$ or $X'_{G,U}$ with the vector in $\mathbb{R}^{V(G_U)}$ that is 1 on vertices belonging to u and zero otherwise.

   If e is not incident to u, then the inner product is zero. Otherwise, up to sign, the inner product is equal to

   $$\sum_{\substack{S \subseteq E(u) \\ |S| \equiv |U \cap \{u\}| \pmod 2}} (-1)^{e \in S} = 2^{d-2} - 2^{d-2} = 0.$$

   Here, as in Lemma 2, we observe that $S \subseteq E(u)$ such that $|S| \equiv |U \cap \{u\}| \pmod 2$ and $e \in S$ correspond to solutions to a system of linear equations over $\mathbb{Z}_2$ with d variables and 2 linearly independent constraints.

2. One column $e \in E(G)$ from $X_{G,U}$ and one column $e' \in E(G)$ from $X'_{G,U}$.

   The case when e and e' intersect only in one vertex is similar to what has been argued in Lemma 3. If $e = e' = uv$ where wlog u precedes v in the ordering on G, then the inner product is

   $$\begin{aligned}
   &\sum_{\substack{S \subseteq E(u) \\ |S| \equiv |U \cap \{u\}| \pmod 2}} (-1)^{e \in S}(-1)^{e' \in S} - \sum_{\substack{T \subseteq E(v) \\ |T| \equiv |U \cap \{v\}| \pmod 2}} (-1)^{e \in T}(-1)^{e' \in T} \\
   &= \sum_{\substack{S \subseteq E(u) \\ |S| \equiv |U \cap \{u\}| \pmod 2}} 1 - \sum_{\substack{T \subseteq E(v) \\ |T| \equiv |U \cap \{v\}| \pmod 2}} 1 \\
   &= 2^{d-1} - 2^{d-1} = 0.
   \end{aligned}$$

   Here, the assumption that G is regular is crucial. □

A.2.3 Definition of the multigraphs

By Lemma 5, the matrix $\tilde{X}_{G,U}$ has orthogonal columns. Furthermore, by Eq. (4), the all-ones vector is one of these columns. All entries of $\tilde{X}_{G,U}$ are integers.

We define a matrix

$$A_{G,U} \coloneqq \tilde{X}_{G,U} D (\tilde{X}_{G,U})^\top \tag{7}$$

by fixing some diagonal matrix D with diagonal entries

$$D_{ii} = \lambda_i \prod_{j \neq i} N_j,$$

where N_i denotes the squared norm of the i-th column of $\tilde{X}_{G,U}$, and the λ_i are pairwise distinct natural numbers. The eigenvalues of this matrix are then $\lambda_1 \cdot \prod_{i=1}^n N_i, \ldots, \lambda_n \cdot \prod_{i=1}^n N_i$. By choosing a sufficiently large positive entry for the index corresponding to the all-ones vector in $\tilde{X}_{G,U}$, we can ensure that the matrix $A_{G,U}$ has only positive integer entries.

Define $A_{G,0} \coloneqq A_{G,\emptyset}$ and $A_{G,1} \coloneqq A_{G,\{v\}}$ for an arbitrary vertex $v \in V(G)$. Here, we use the same matrix D for both $A_{G,0}$ and $A_{G,1}$.

Thus, we have defined a pair of symmetric simple-spectrum matrices with positive integer entries. These will be the adjacency matrices of the multigraphs stipulated by Theorem 1.

A.2.4 No isomorphisms

Lemma 6. *The multigraphs $A_{G,0}$ and $A_{G,1}$ are non-isomorphic.*

Proof. Towards a contradiction, let $\pi\colon V(G_1) \to V(G_0)$ be the isomorphism witnessing $A_{G,0} \cong A_{G,1}$. We argue that this map gives an isomorphism $G_1 \to G_0$. No such map exists by [41, Corollary 3.7].

Write $P \in \{0,1\}^{V(G_0)\times V(G_1)}$ for the permutation matrix encoding π. It holds that

$$A_{G,0} = PA_{G,1}P^\top.$$

It follows from this equation that P maps eigenvectors of $A_{G,1}$ for eigenvalue λ to eigenvectors of $A_{G,0}$ with the same eigenvalue. Furthermore, for any vector v, it holds that Pv and v have the same norm. Since the diagonal entries of the matrix D in Eq. (7) is the same for $A_{G,1}$ and $A_{G,0}$, there are weights $\alpha_e \in \{\pm 1\}$ for every $e \in E(G)$ such that

$$X_{G,1}((v,S),e) = \alpha_e \cdot X_{G,0}(\pi(v,S),e)$$

for all $(v,S) \in V(G_1)$ and $e \in E(G)$. We claim that π is an isomorphism $G_1 \to G_0$. Let $(v,S),(u,T) \in V(G_U)$ be adjacent, i.e., $vu \in E(G)$ and $vu \notin S \triangle T$.

Let $e \coloneqq uv$. By Eq. (2), it holds that $X_{G,1}((v,S),e) = X_{G,1}((u,T),e) \in \{\pm 1\}$. By assumption:

$$X_{G,0}(\pi(v,S),e) = \alpha_e \cdot X_{G,1}((v,S),e) = \alpha_e \cdot X_{G,U}((u,T),e) = X_{G,U'}(\pi(u,T),e) \in \{\pm 1\}.$$

It follows, again by Eq. (2), that $\pi(v,S)$ and $\pi(u,T)$ are adjacent. □

A.2.5 Weisfeiler–Leman indistinguishability

We show that the Weisfeiler–Leman algorithm on $A_{G,0}$ and $A_{G,1}$ can be simulated by running it on the CFI graphs G_0 and G_1. The CFI graphs are known not to be distinguished by Weisfeiler–Leman.

For two vertices $(v,S),(u,T) \in V(G_U)$, the entry $A_{G,U}((v,S),(u,T))$ is

$$\begin{aligned} A_{G,U}((v,S),(u,T)) = &\sum_{e\in E(G)} D_e \cdot X_{G,U}((v,S),e) \cdot X_{G,U}((u,T),e) \\ &+ \sum_{e'\in E(G)} D'_{e'} \cdot X'_{G,U}((v,S),e') \cdot X'_{G,U}((u,T),e') \\ &+ \sum_{w\in V(G)} D''_w \cdot I_{G,U}((v,S),w) \cdot I_{G,U}((u,T),w). \end{aligned} \tag{8}$$

Here, D, D', D'' denote the values of the diagonal matrix from Eq. (7). As a first step, we compute the products in each of the sums explicitly.

Lemma 7. *Let G be a graph and $u,v,w \in V(G)$ with $u \neq v$. For $U \subseteq V(G)$ and $e \in E(G)$, it holds that*

$$X_{G,U}((u,S),e)^2 = X'_{G,U}((u,S),e)^2 = \begin{cases} 1, & \text{if } e \in E(u), \\ 0, & \text{otherwise}. \end{cases} \tag{9}$$

$$X_{G,U}((u,S),e) \cdot X_{G,U}((v,T),e) = \begin{cases} 1, & \text{if } e = uv \text{ and } (u,S) \sim (v,T) \\ -1, & \text{if } e = uv \text{ and } (u,S) \not\sim (v,T) \\ 0, & \text{otherwise}. \end{cases} \tag{10}$$

$$X'_{G,U}((u,S),e) \cdot X'_{G,U}((v,T),e) = \begin{cases} -1, & \text{if } e = uv \text{ and } (u,S) \sim (v,T) \\ 1, & \text{if } e = uv \text{ and } (u,S) \not\sim (v,T) \\ 0, & \text{otherwise}. \end{cases} \tag{11}$$

$$\begin{aligned} X_{G,U}((u,S),e) \cdot X_{G,U}((u,T),e) &= X'_{G,U}((u,S),e) \cdot X'_{G,U}((u,T),e) \\ &= \begin{cases} 1, & \text{if } e \in E(u) \text{ and } e \notin S\triangle T, \\ -1, & \text{if } e \in E(u) \text{ and } e \in S\triangle T, \\ 0, & \text{otherwise}. \end{cases} \end{aligned} \tag{12}$$

Proof. Immediate from Eqs. (2) and (3). Equation (11) differs from Eq. (10) only by the sign flip. In Eq. (12) the sign flip is squared and hence neutralized. □

For a CFI graph G_U, write G_U^* for the vertex-colored graph obtained from G_U by coloring (v, S) by v. That is, every vertex fiber gets a unique color. Formally, G_U^* is the adjacency matrix of G_U with diagonal entries $G_U^*((v,S),(v,S)) \coloneqq c_v$ for some injective map $v \mapsto c_v$, fixed for all U; cf. Definition 4. Since we are comparing WL colors of multiple graphs, we write $C^{(t)}(G, \bar{v})$ for the round-t k-WL color of the tuple $\bar{v} \in V(G)^k$, and atp_G for the atomic type in G.

Lemma 8. *Let $k \geq 2$ and $r \geq 0$. Let G be a graph of minimum degree ≥ 3 and $U, U' \subseteq V(G)$. For all $\bar{x} \in V(G_U)^k$ and $\bar{x}' \in V(G_{U'})^k$,*

$$C^{(r+1)}(G_U^*, \bar{x}) = C^{(r+1)}(G_{U'}^*, \bar{x}') \implies C^{(r)}(A_{G,U}, \bar{x}) = C^{(r)}(A_{G,U'}, \bar{x}').$$

In particular, if $G_U^* \equiv_k G_{U'}^*$, then $A_{G,U} \equiv_k A_{G,U'}$.

Proof. The "in particular" follows since the implication, applied also with $U = U'$, shows that $C^{(r)}(A_{G,U}, \cdot)$ and $C^{(r)}(A_{G,U'}, \cdot)$ are given by one function of $C^{(r+1)}(G_U^*, \cdot)$ and $C^{(r+1)}(G_{U'}^*, \cdot)$, respectively. We prove the implication by induction on r.

Base case $r = 0$. Suppose $C^{(1)}(G_U^*, \bar{x}) = C^{(1)}(G_{U'}^*, \bar{x}')$. We need to show $\mathrm{atp}_{A_{G,U}}(\bar{x}) = \mathrm{atp}_{A_{G,U'}}(\bar{x}')$. The equality patterns of $\bar{x}$ and $\bar{x}'$ agree since they are part of $\mathrm{atp}_{G_U^*}(\bar{x}) = \mathrm{atp}_{G_{U'}^*}(\bar{x}')$. It remains to show that $A_{G,U}(x_i, x_j) = A_{G,U'}(x_i', x_j')$ for all $i, j \in [k]$. Fix i, j and write $x_i = (u, S)$, $x_j = (v, T)$, $x_i' = (u', S')$, and $x_j' = (v', T')$. Since the diagonal entries of G_U^* and $G_{U'}^*$ encode the fibers, $u = u'$ and $v = v'$.

We shall show that, for every $e \in E(G)$ and $w \in V(G)$,

$$X_{G,U}((u,S),e) \cdot X_{G,U}((v,T),e) = X_{G,U'}((u,S'),e) \cdot X_{G,U'}((v,T'),e), \tag{13}$$

$$X'_{G,U}((u,S),e) \cdot X'_{G,U}((v,T),e) = X'_{G,U'}((u,S'),e) \cdot X'_{G,U'}((v,T'),e), \text{ and} \tag{14}$$

$$I_{G,U}((u,S),w) \cdot I_{G,U}((v,T),w) = I_{G,U'}((u,S'),w) \cdot I_{G,U'}((v,T'),w). \tag{15}$$

By Eq. (8), these identities imply that

$$A_{G,U}((u,S),(v,T)) = A_{G,U'}((u,S'),(v,T')),$$

as desired.

If $u \neq v$, Eqs. (10) and (11) imply Eqs. (13) and (14) since $\mathrm{atp}_{G_U^*}(\bar{x}) = \mathrm{atp}_{G_{U'}^*}(\bar{x}')$ encodes whether $(u,S) \sim (v,T)$ and whether $(u,S') \sim (v,T')$. If $x_i = x_j$, then $x_i' = x_j'$, and Eqs. (13) and (14) follow from Eq. (9).

The remaining case $u = v$ and $S \neq T$ is more subtle since the condition Eq. (12) is not encoded in the atomic type. By Eq. (12), it suffices to consider the case $e = uw$ for some $w \in V(G)$.

Claim 1. It holds that $e = uw \notin S \triangle T$ if, and only if, there exists a shared neighbor (w, R) of (u, S) and (u, T).

Proof of Claim. For the forward direction, by the assumption that every vertex in G has at least three neighbors, there exist two vertices $(w, R_0), (w, R_1) \in V(G_U)$ such that $uw \notin R_0$ and $uw \in R_1$. Depending on whether $e \in S$ or not, it follows that (u, S) and (w, R_0) or (w, R_1) are adjacent. Hence, if $e \notin S \triangle T$, that is, S and T "agree" about e, then (u, S) and (u, T) are both adjacent to one of (w, R_0), (w, R_1).

Conversely, if (u, S) and (u, T) are both adjacent to (w, R), then $uw \notin S \triangle R$ and $uw \notin T \triangle R$ by Definition 2. Hence, $uw \notin (S \triangle R) \triangle (T \triangle R) = S \triangle T$, as desired. ⌟

By Definition 4, $C^{(1)}(G_U^*, \bar{x})$ determines the multiset $\{\!\{\mathrm{atp}_{G_U^*}(\bar{x}y) \mid y \in V(G_U)\}\!\}$. For each $y = (w, R)$, the atomic type $\mathrm{atp}_{G_U^*}(\bar{x}y)$ records the fiber w of y and whether y is adjacent to x_i and to x_j. Hence, $C^{(1)}(G_U^*, \bar{x}) = C^{(1)}(G_{U'}^*, \bar{x}')$ implies that, for every $w \in V(G)$, the vertices (u, S) and (u, T) share a neighbor in fiber w if, and only if, (u, S') and (u, T') do. By Claim 1, $E(u) \cap (S \triangle T) = E(u) \cap (S' \triangle T')$, and Eq. (12) implies Eqs. (13) and (14).

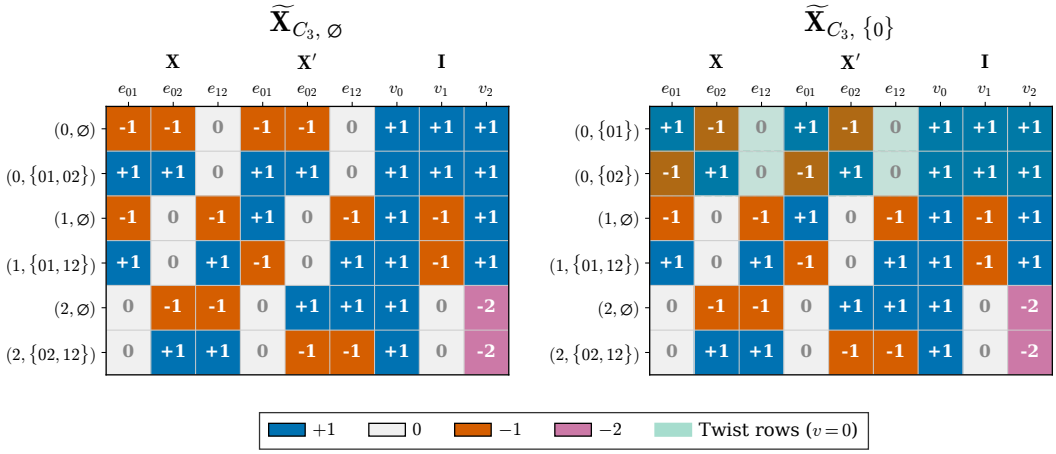

Figure 2: Integral encoding matrices for the CFI construction on C_3. **Left:** $\tilde{X}_{C_3,\varnothing}$ (trivial lift): the vertex-0 rows have *coherent* signs in the X-block. **Right:** $\tilde{X}_{C_3,\{0\}}$ (non-trivial lift, highlighted rows): the parity twist at vertex 0 *splits* the signs, making the two matrices non-isomorphic under $S_6 \times \{\pm 1\}^9$.

It remains to consider Eq. (15). By Eq. (5),

$$I_{G,U}((u,S),w) \cdot I_{G,U}((v,T),w) = W(u,w) \cdot W(v,w) = I_{G,U'}((u,S'),w) \cdot I_{G,U'}((v,T'),w).$$

In other words, the value of $I_{G,U}((u,S),w)$ does not depend on S.

Inductive step. Suppose the implication holds for r and that $C^{(r+2)}(G^*_U, \bar{x}) = C^{(r+2)}(G^*_{U'}, \bar{x}')$. Then $C^{(r+1)}(G^*_U, \bar{x}) = C^{(r+1)}(G^*_{U'}, \bar{x}')$, so $C^{(r)}(A_{G,U}, \bar{x}) = C^{(r)}(A_{G,U'}, \bar{x}')$ by the induction hypothesis. Moreover, by Definition 4, there is a bijection $\pi\colon V(G_U) \to V(G_{U'})$ such that, for all $y \in V(G_U)$ and $j \in [k]$,

$$\mathrm{atp}_{G^*_U}(\bar{x}y) = \mathrm{atp}_{G^*_{U'}}(\bar{x}'\pi(y)) \quad \text{and} \quad C^{(r+1)}(G^*_U, \bar{x}[y/j]) = C^{(r+1)}(G^*_{U'}, \bar{x}'[\pi(y)/j]).$$

By the induction hypothesis, $C^{(r)}(A_{G,U}, \bar{x}[y/j]) = C^{(r)}(A_{G,U'}, \bar{x}'[\pi(y)/j])$ for all $j \in [k]$. It remains to show that $\mathrm{atp}_{A_{G,U}}(\bar{x}y) = \mathrm{atp}_{A_{G,U'}}(\bar{x}'\pi(y))$. The equality patterns agree since they are part of the atomic types in G^*_U and $G^*_{U'}$. The entries among $x_1, \ldots, x_k$ agree since $C^{(r)}(A_{G,U}, \bar{x}) = C^{(r)}(A_{G,U'}, \bar{x}')$ determines $\mathrm{atp}_{A_{G,U}}(\bar{x})$. For $i \in [k]$, the entries $A_{G,U}(x_i, y)$ and $A_{G,U}(y, y)$ occur in $\mathrm{atp}_{A_{G,U}}(\bar{x}[y/j])$ for any $j \neq i$, which exists since $k \geq 2$. This atomic type is determined by $C^{(r)}(A_{G,U}, \bar{x}[y/j])$, and analogously for U'. Hence, $C^{(r+1)}(A_{G,U}, \bar{x}) = C^{(r+1)}(A_{G,U'}, \bar{x}')$. □

A.2.6 Summary

This concludes the preparations for the main theorem.

Theorem 1. *For every $k \in \mathbb{N}$, there exist non-isomorphic simple-spectrum multigraphs with $\Theta(k)$ nodes that are not distinguished by the k-dimensional Weisfeiler–Leman algorithm.*

Proof. By [43] and [42, Lemma 5 and Corollary 7], there exists, for every sufficiently large $n \in \mathbb{N}$, a connected 3-regular $2n$-vertex graph G_n of treewidth $\geq \frac{\alpha}{3(\alpha+1)} \cdot 2n - 1 = \frac{n}{12} - 1$ for $\alpha = \frac{1}{7}$. By [33, Lemma 12, Theorems 10 and 14], the vertex-colored CFI graphs $G^*_{n,0}$ and $G^*_{n,1}$ of G_n are not distinguished by the $\left(\left\lceil \frac{n}{12} \right\rceil - 2\right)$-dimensional Weisfeiler–Leman algorithm. By Lemma 8, so are multigraphs $A_{G_n,0}$ and $A_{G_n,1}$. By Lemma 6, the multigraphs $A_{G_n,0}$ and $A_{G_n,1}$ are non-isomorphic. They have simple spectrum by construction in Section A.2.3. □

A.3 CFI Construction on C_3: Detailed Example

We provide the full details of the CFI construction on C_3 (Example 1). Let $G = C_3$, the triangle on vertex set $\{0,1,2\}$ with edges $\{0,1\}$, $\{1,2\}$, $\{0,2\}$. Each vertex i has degree 2, so $|E(i)| = 2$ and

the even-cardinality subsets of $E(i)$ are $\varnothing$ and $E(i)$ itself, yielding two fiber vertices per base vertex and six CFI vertices in total.

Trivial lift ($U = \varnothing$). The fiber of each vertex i is $\{(i, \varnothing),\ (i, E(i))\}$. Concretely:

$$\begin{aligned}
\text{vertex } 0: &\quad (0, \varnothing), \quad (0, \{\{0,1\},\{0,2\}\}),\\
\text{vertex } 1: &\quad (1, \varnothing), \quad (1, \{\{0,1\},\{1,2\}\}),\\
\text{vertex } 2: &\quad (2, \varnothing), \quad (2, \{\{0,2\},\{1,2\}\}).
\end{aligned}$$

For each base edge $e = \{i, j\}$, the adjacency condition $e \notin S \triangle T$ forces S and T to agree on e. Since each fiber has exactly one vertex containing e and one not containing e, each base edge contributes two edges, one connecting the "contains e" pair and one connecting the "does not contain e" pair. The $\varnothing$-type vertices $\{(i, \varnothing)\}_{i=0}^{2}$ therefore form a triangle among themselves, as do the $(i, E(i))$-type vertices, with no edges between the two groups. Hence $G_0 \cong 2\,C_3$.

Non-trivial lift ($U = \{0\}$). The parity constraint at vertex 0 shifts to odd-cardinality subsets: $(0, \{\{0,1\}\})$ and $(0, \{\{0,2\}\})$. Vertices 1 and 2 retain their even-parity fibers. The adjacency rewiring at base edges $\{0,1\}$ and $\{0,2\}$ produces:

$$\begin{aligned}
&(0, \{\{0,1\}\}) \sim (1, \{\{0,1\},\{1,2\}\}), \qquad && (0, \{\{0,1\}\}) \sim (2, \varnothing),\\
&(0, \{\{0,2\}\}) \sim (1, \varnothing), && (0, \{\{0,2\}\}) \sim (2, \{\{0,2\},\{1,2\}\}).
\end{aligned}$$

Together with the unchanged edges $(1, \varnothing) \sim (2, \varnothing)$ and $(1, \{\{0,1\},\{1,2\}\}) \sim (2, \{\{0,2\},\{1,2\}\})$ from base edge $\{1,2\}$, the six vertices form a single cycle:

$$\begin{aligned}
(0, \{\{0,1\}\}) &\to (1, \{\{0,1\},\{1,2\}\}) \to (2, \{\{0,2\},\{1,2\}\}) \to (0, \{\{0,2\}\})\\
&\to (1, \varnothing) \to (2, \varnothing) \to (0, \{\{0,1\}\}).
\end{aligned}$$

Hence $G_{\{0\}} \cong C_6$.

A.4 Integral Encoding on C_3: Detailed Example

We provide the full integral encoding matrices for the CFI construction on C_3 (Section 4). Let $G = C_3$ on vertices $\{0,1,2\}$ with oriented edges $e_{01} = (0,1)$, $e_{02} = (0,2)$, $e_{12} = (1,2)$, where the smaller vertex is always the first endpoint. Each vertex has degree 2, so each fiber contains two CFI vertices and $|V(\mathrm{CFI}(G,U))| = 6$ for every U. The integral encoding $\mathbf{X}^*_{G,U} = [\,\mathbf{X} \mid \mathbf{X}' \mid \mathbf{I}\,] \in \mathbb{Z}^{6\times 9}$.

Trivial lift ($U = \varnothing$).

$$\mathbf{X}^*_{C_3,\varnothing} = \begin{array}{c} \\ (0,\varnothing)\\ (0,\{01,02\})\\ (1,\varnothing)\\ (1,\{01,12\})\\ (2,\varnothing)\\ (2,\{02,12\}) \end{array} \begin{array}{c} \begin{array}{ccccccccc} e_{01} & e_{02} & e_{12} & e_{01} & e_{02} & e_{12} & v_0 & v_1 & v_2 \end{array}\\ \begin{pmatrix} -1 & -1 & 0 & -1 & -1 & 0 & 1 & 1 & 1\\ +1 & +1 & 0 & +1 & +1 & 0 & 1 & 1 & 1\\ -1 & 0 & -1 & +1 & 0 & -1 & 1 & -1 & 1\\ +1 & 0 & +1 & -1 & 0 & +1 & 1 & -1 & 1\\ 0 & -1 & -1 & 0 & +1 & +1 & 1 & 0 & -2\\ 0 & +1 & +1 & 0 & -1 & -1 & 1 & 0 & -2 \end{pmatrix} \end{array}$$

The first three columns form $\mathbf{X}$, the next three form $\mathbf{X}'$ (which flips the sign whenever v is the *second* endpoint of e), and the last three form the indicator $\mathbf{I}$. Note that in the $\mathbf{X}$-block the vertex-0 rows have *coherent* signs: $(0,\varnothing)$ has $(-1,-1)$ and $(0,\{01,02\})$ has $(+1,+1)$ on its incident edges, that is both entries share the same sign.

Non-trivial lift ($U = \{0\}$). The parity twist at vertex 0 replaces the even-cardinality subsets $\{\varnothing, \{01,02\}\}$ with the odd-cardinality subsets $\{\{01\},\{02\}\}$. Only the first two rows change:

$$\mathbf{X}^*_{C_3,\{0\}} = \begin{array}{c} \\ (0,\{01\})\\ (0,\{02\})\\ (1,\varnothing)\\ (1,\{01,12\})\\ (2,\varnothing)\\ (2,\{02,12\}) \end{array} \begin{array}{c} \begin{array}{ccccccccc} e_{01} & e_{02} & e_{12} & e_{01} & e_{02} & e_{12} & v_0 & v_1 & v_2 \end{array}\\ \begin{pmatrix} +1 & -1 & 0 & +1 & -1 & 0 & 1 & 1 & 1\\ -1 & +1 & 0 & -1 & +1 & 0 & 1 & 1 & 1\\ -1 & 0 & -1 & +1 & 0 & -1 & 1 & -1 & 1\\ +1 & 0 & +1 & -1 & 0 & +1 & 1 & -1 & 1\\ 0 & -1 & -1 & 0 & +1 & +1 & 1 & 0 & -2\\ 0 & +1 & +1 & 0 & -1 & -1 & 1 & 0 & -2 \end{pmatrix} \end{array}$$

The critical difference is in the vertex-0 rows of the $\mathbf{X}$-block: the signs now *split*, $(0, \{01\})$ has $(+1, -1)$ and $(0, \{02\})$ has $(-1, +1)$, encoding the parity twist. Since no simultaneous sign flip and row permutation can convert coherent pairs to split pairs, the two matrices are non-isomorphic under the action of $\{\pm 1\}^9 \times S_6$.

A.5 Spectrum of CFI Graphs

As a justification for the construction above, we show that the CFI graphs from Definition 2 do not have simple spectrum.

Theorem 4. *Let G be a 3-regular graph with $2n$ vertices and spectrum $\lambda_1, \ldots, \lambda_{2n}$. Then the spectrum of the even CFI graph G_0 of G is*

$$\underbrace{2\lambda_1, \ldots, 2\lambda_{2n}}_{2n}, \quad \underbrace{+2, \ldots, +2}_{3n}, \quad \underbrace{-2, \ldots, -2}_{3n}.$$

In particular, G_0 does not have simple spectrum.

If G has treewidth at least 3, e.g. when G is taken to be the complete graph on 4 vertices, then the CFI graphs G_0 and G_1 have the same spectrum [33], [51, Theorem 3.3.2]. In this case, neither G_0 nor G_1 have simple spectrum regardless of whether the spectrum of G is simple. The proof of Theorem 4 shows that the matrices $\tilde{X}_{G,\emptyset}$ constructed in Section A.2 roughly correspond to the eigenvectors of the CFI graph G_0.

Proof. The first $2n$ eigenvalues of G_0 correspond to the eigenvalues of G. The other $6n$ eigenvalues stem from the edges of G.

Eigenvectors constant on fibers. Let $y \in \mathbb{R}^{V(G)}$ be an eigenvector of G for eigenvalue λ. Define $x \in \mathbb{R}^{V(G_0)}$ by $x_{(v,S)} = y_v$. We claim that x is an eigenvector of G_0 with eigenvalue 2λ. Indeed, for every $(v, S) \in V(G_0)$,

$$\sum_{(u,T)\sim(v,S)} x_{(u,T)} = \sum_{(u,T)\sim(v,S)} y_u = \sum_{u\sim v} 2 \cdot y_u = 2\lambda \cdot y_v = 2\lambda \cdot x_{(v,S)}.$$

Here, the second equality holds since for every (v, S) and $u \sim v$ there are precisely 2 sets T such that $(u, T) \sim (v, S)$. As argued in Section A.2, the sets T correspond to solutions of a system of linear equations over $\mathbb{Z}_2$. The system has 3 variables and 2 equations: one enforcing that $|T| \equiv 0 \pmod 2$ and one guaranteeing that $uv \notin S \triangle T$. This system has 2 solutions. Note that these eigenvectors are constant on fibers, i.e. $x_{(v,S)} = x_{(v,T)}$ for all S, T.

Eigenvectors corresponding to edges. Next we construct eigenvectors corresponding to the $3n$ edges of G. Recall Eqs. (2) and (3). Fix an ordering $V(G) = \{1, \ldots, 2n\}$ of the vertices of G. For an edge $e \in E(G)$, define the vector $y^e \in \mathbb{R}^{V(G_0)}$ via

$$y^e_{(v,S)} \coloneqq \begin{cases} 1, & \text{if } e \in E(v) \text{ and } e \in S, \\ -1, & \text{if } e \in E(v) \text{ and } e \notin S, \\ 0, & \text{otherwise.} \end{cases}$$

Towards verifying that y^e is an eigenvector of the adjacency matrix of G_0, let $(v, S) \in V(G_0)$. Distinguish cases:

- The edge e is not incident to v or one of its neighbors. We have
$$\sum_{(u,T)\sim(v,S)} y^e_{(u,T)} = 0 = y^e_{(v,S)} = 2y^e_{(v,S)}.$$
- The edge e is incident to two neighbors w_1, w_2 of v. For each $i \in \{1, 2\}$, the number of even subsets $T \subseteq E(w_i)$ such that $vw_i \notin S \triangle T$ and $e \in T$ is one, and is thus equal to the number of even subsets $T' \subseteq E(w_i)$ such that $vw_i \notin S \triangle T'$ and $e \notin T'$. Hence each of w_1 and w_2 contributes $+1$ from the $e \in T$ term and -1 from the $e \notin T$ term, giving
$$\sum_{(u,T)\sim(v,S)} y^e_{(u,T)} = \underbrace{(1-1)}_{w_1} + \underbrace{(1-1)}_{w_2} = 0 = y^e_{(v,S)} = 2y^e_{(v,S)}.$$

- The edge e is incident to precisely one neighbor w of v and not to v. This case is analogous to the previous case.
- The edge e is incident to v, i.e. $e = vw$ for some vertex w. The number of even subsets $T \subseteq E(w)$ such that $e \notin S \triangle T$ and $e \in T$ is either zero or two, depending on whether $e \in S$. In both cases,
$$\sum_{(u,T)\sim(v,S)} y^e_{(u,T)} = 2y^e_{(v,S)}.$$

Hence, y^e is an eigenvector of the adjacency matrix of G_0 with eigenvalue 2.

For an edge $e \in E(G)$ and $(v, S) \in V(G_0)$, the remaining eigenvalues are given by

$$y^{-e}_{(v,S)} \coloneqq \begin{cases} y^e_{(v,S)} & \text{if } v \text{ is the first vertex in } e, \\ -y^e_{(v,S)} & \text{if } v \text{ is the second vertex in } e, \\ 0, & \text{if } e \notin E(v), \end{cases}$$

see Eq. (3). In order to argue that y^{-e} is an eigenvector to eigenvalue -2, we conduct a case distinction as above. The first three cases are as for y^e. In the final case, only the sign is flipped, i.e.

$$\sum_{(u,T)\sim(v,S)} y^{-e}_{(u,T)} = -2 \cdot y^{-e}_{(v,S)},$$

as desired.

By Lemma 5, the eigenvectors constructed above form an orthogonal basis. Hence, we have mapped out the entire spectrum of the adjacency matrix of G_0. □

B Canonicalization of Simple Spectrum Graphs

This appendix accompanies Section 5 and gives the formal proof of Theorem 2, working with the partition $\{C_1, \ldots, C_L\}$ produced by Algorithms 3 and 4.

B.1 Proof of Theorem 2

In this proof we identify $\{-1, 1\}^k$ with the $\mathbb{Z}_2^k = \{0, 1\}^k$. The action of $a \in \mathbb{Z}_2^k$ on $U \in \mathbb{R}^{n\times k}$, denoted by $a.U$, is given by multiplying, for each $j \in \{1, \ldots, k\}$, the j-th column of U by a corresponding sign $(-1)^{a_j}$. We denote the summation of two vectors $s, t \in \mathbb{Z}_2^k$ by $s \oplus t$.

We now prove that the map PRiSM from Section 5 is a valid canonicalization in the sense of Definition 3. The proof follows the standard reduction strategy: we first define a *sign canonicalization* $s : \mathbb{R}^{n\times k} \to \mathbb{Z}_2^k$ (Definition 7) and show that any such s induces a canonicalization c via lexicographic row sorting (Lemma 9). The technical core of the proof, in Section B.1.2, then constructs an explicit sign canonicalization on top of the partition $\{C_1, \ldots, C_L\}$ output by Algorithms 3 and 4.

B.1.1 Reduction to sign canonicalization

We begin by introducing a definition of choosing unique representative of simple-spectrum eigendecompositions only up to sign ambiguities, or actions of the group $\mathbb{Z}_2^k$ for a k-eigendecomposition. We will then show that once we obtain this canonicalization it naturally extends to the full ambiguity group $S_n \times \mathbb{Z}_2^k$ using lexicographic sorting.

Definition 6. We say that $a \in \mathbb{Z}_2^k$ is a sign automorphism of $U \in \mathbb{R}^{n\times k}$, if $a.U$ is equal to U, up to permutation.

Definition 7. We say that $s : \mathbb{R}^{n\times k} \to \mathbb{Z}_2^k$ is a sign-canonicalization, if

1. **Permutation invariance:** s is invariant to permutations.
2. **Sign equivariance:** $s(t.U) \oplus (t \oplus s(U))$ is a sign automorphism of U, for all U and $t \in \mathbb{Z}_2^k$.

We note that when U has no non-trivial automorphisms, the second condition coincides with the standard notion of equivariance of s, namely $s(t.U) = t \oplus s(U)$ for all U and $t \in \mathbb{Z}_2^k$.

Lemma 9. *If s is a sign canonicalization, then a canonicalization $c(U)$ can be defined by ordering $s(U).U$ in lexicographical ordering.*

Proof. For a given matrix U, permutation τ and sign vector t, we need to show that $c((\tau,t).U) = c(U)$. Equivalently, we need to show that $s(t.U).(t.U) \cong_{S_n} s(U).U$, where we use $\cong_{S_n}$ to denote equivalence up to permutation. Indeed

$$s(U).U \cong_{S_n} s(U).(s(t.U) \oplus t \oplus s(U)).U = (s(t.U) \oplus t).U = s(t.U).(t.U) \qquad \square$$

By Lemma 9, it suffices to construct a sign canonicalization. The remainder of the proof does this on top of the partition output by Algorithms 3 and 4.

B.1.2 Spectral refinement and the affine subspaces A_ℓ

Setup. Throughout this subsection we denote the rows of U by U_i and write $\odot$ for the elementwise product of vectors. Let $\{C_1, \dots, C_L\}$ be the partition of the vertex set $[n]$ produced by Stage 1 (PARTITION, Algorithm 3) followed by Stage 2 (REFINE, Algorithm 4) of Section 5. Two vertices i, j lie in the same class C_ℓ only if they share the absolute-value signature $\sigma(v) = (|U_{v,1}|, \dots, |U_{v,k}|)$, so within each class

$$|U_i| = |U_j| \quad \text{for all } i, j \in C_\ell.$$

In particular, the support set $K_\ell = \{j \in [k] : |U_{v,j}| > 0\}$ is well-defined for each C_ℓ (it does not depend on the choice of $v \in C_\ell$). For every nonzero entry $U_{i,k}$ let $s_{i,k}$ denote its sign bit, namely $\operatorname{sign}(U_{i,k}) = (-1)^{s_{i,k}}$, and let $s_i = (s_{i,k})_{U_{i,k}\neq 0} \in \mathbb{Z}_2^{K_\ell}$ for $i \in C_\ell$.

Spectral WL refinement. We record the spectral Weisfeiler–Leman iteration that Algorithm 4 runs, since the proof uses its termination property directly. Let $c_0(i)$ be the index of the class containing i in the ordered input partition $\mathcal{P}_0$. Initialize a coloring for each node $i \in [n]$ by

$$h_i^{(0)} = (c_0(i), U_i \odot U_i), \tag{16}$$

and refine via

$$h_i^{(t+1)} = \mathrm{UPDATE}_{(t)}\Big(h_i^{(t)}, \big\{(h_j^{(t)}, U_i \odot U_j) \mid j = 1, \dots, n\big\}\Big), \tag{17}$$

where $\{\cdot\}$ denotes a multiset and $\mathrm{UPDATE}_{(t)}$ is invariant to the order of its multiset argument. The colors $h^{(t)}$ define a sequence of partitions $C_1^{(t)}, \dots, C_{k_t}^{(t)}$ where i, j are in the same class iff $h_i^{(t)} = h_j^{(t)}$. Each successive partition refines the previous one, so the iteration stabilizes at some $t \le n$; this fixed-point partition $\{C_1, \dots, C_L\}$ is the output of Algorithm 4. By definition of the WL fixed point, classes at termination have a stable multiset of edge features:

$$\{U_i \odot U_k \mid k \in C_\ell\} = \{U_j \odot U_k \mid k \in C_\ell\} \quad \text{for all } i, j \in C_\ell,\ \ell \in [L]. \tag{18}$$

This is the structural fact the proof relies on. The iteration (17) is precisely Algorithm 4 (defined in Section B); we describe it again here only to make explicit that the partition $\{C_\ell\}$ used by the proof is the spectral-WL fixed point. Remark 1 gives an equivalent $\mathbb{Z}_2$ parity-check realization used in our implementation. The same partition arises in the classical bounded-eigenvalue-multiplicity isomorphism algorithms of Babai et al. [34] and Spielman [52], which use related sign-product invariants to refine an initial partition; the novelty of our approach is to take this stable partition as the starting point for an *explicit linear sign canonicalization* (Algorithm 5), as opposed to using it for isomorphism testing alone.

The affine subspaces A_ℓ. For $i \in C_\ell$ define $A_\ell = \{s_i : i \in C_\ell\} \subseteq \mathbb{Z}_2^{K_\ell}$.

Lemma 10. *For every $\ell \in [L]$, we have that the set*

$$A_\ell = \{s_i, i \in C_\ell\}$$

is an affine subspace of $\mathbb{Z}_2^{K_\ell}$.

Proof. By the multiset stability (18), for every $i, j \in C_\ell$,

$$\{U_i \odot U_k \mid k \in C_\ell\} = \{U_j \odot U_k \mid k \in C_\ell\},$$

which under the sign-bit projection becomes

$$\{s_i \oplus s_k \mid k \in C_\ell\} = \{s_j \oplus s_k \mid k \in C_\ell\}. \tag{19}$$

To show A_ℓ is an affine subspace, it suffices to show the set $\{s_i \oplus s_k \mid k \in C_\ell\}$ is a linear subspace (translating by s_i then yields A_ℓ). It contains the zero vector $s_i \oplus s_i$. For closure under addition, given $k, j \in C_\ell$ we have $(s_i \oplus s_k) \oplus (s_i \oplus s_j) = s_j \oplus s_k$, which by (19) equals $s_i \oplus s_q$ for some $q \in C_\ell$. We note that in this proof we treated the rows $\{U_i\}_{i \in C_\ell}$ as pairwise distinct so that $\{s_i \oplus s_k \mid k \in C_\ell\}$ is a set; it is possible in general that some of the rows coincide. However, in this case, since the coloring is stable all rows will have equal multiplicity, and so the argument carries through verbatim with sets replaced by multisets. Note also that under a full simple-spectrum eigendecomposition, that is n-eigendecomposition, the rows are pairwise distinct since they are pairwise orthogonal. □

B.1.3 Sign automorphisms of U

Let $W_\ell = A_\ell \oplus A_\ell$. This is the subspace associated with the affine space A_ℓ, and for every $i \in C_\ell$ we have $W_\ell = \{s_i \oplus s_k \mid k \in C_\ell\}$. The space W_ℓ has $|K_\ell|$ coordinates; we extend it to a subspace $\hat{W}_\ell \subseteq \mathbb{Z}_2^k$ consisting of all vectors whose restriction to K_ℓ lies in W_ℓ.

The next lemma identifies the sign-automorphism group $\hat{W}$ that the canonicalization must respect.

Lemma 11. *The set of sign automorphisms of U is precisely*

$$\hat{W} := \cap_{\ell=1}^{L} \hat{W}_\ell.$$

Proof. Let A be the set of sign automorphisms of U. By Definition 6, $A = \{a \in \mathbb{Z}_2^k \mid \exists \tau \in S_n \text{ s.t. } (\tau, a) \cdot U = U\}$. We must prove $A = \hat{W}$.

1. **Proof of $A \subseteq \hat{W}$:** Let $a \in A$. By definition, there exists a $\tau \in S_n$ such that $g = (\tau, a) \in \mathrm{Aut}(U)$. By construction of the partition $\{C_\ell\}_{\ell=1}^{L}$ from absolute-value signatures (Algorithms 3 and 4), any $g \in \mathrm{Aut}(U)$ has its associated permutation τ mapping each partition class to itself. (Permutations preserve absolute-value signatures, so $\tau(v) \in C_\ell$ whenever $v \in C_\ell$.) This means that for every $\ell \in [L]$, $\tau(C_\ell) = C_\ell$. This implies the action of g stabilizes the set of rows $\{U_i\}_{i \in C_\ell}$ for each ℓ. The group $\hat{W}_\ell$ is constructed from W_ℓ, which is derived from the affine space $A_\ell = \{s_i \mid i \in C_\ell\}$. $\hat{W}_\ell$ is, by construction, the group of sign vectors that stabilizes the class C_ℓ. Any other element $g \in \{\pm 1\}^k \setminus \hat{W}_\ell$ that is not in this group will not stabilize C_ℓ, as $g.A_\ell \neq A_\ell$, since the identity element will not appear in the coset $g.A_\ell$. Since the global automorphism a must stabilize *every* class C_ℓ (for $\ell = 1, \ldots, L$), a must be an element of $\hat{W}_\ell$ for all ℓ. Therefore, $a \in \cap_{\ell=1}^{L} \hat{W}_\ell$, which means $A \subseteq \hat{W}$.
2. **Proof of $\hat{W} \subseteq A$:** Let $a \in \hat{W} = \cap_{\ell=1}^{L} \hat{W}_\ell$. By definition, $a \in \hat{W}_\ell$ for every $\ell \in [L]$. The fact that $a \in \hat{W}_\ell$ means that a stabilizes the set of rows $\{U_i\}_{i \in C_\ell}$. This implies that for each ℓ, the set $\{a \cdot U_i\}_{i \in C_\ell}$ is a permutation of $\{U_i\}_{i \in C_\ell}$. Therefore, for each class ℓ, there exists a permutation $\tau_\ell : C_\ell \to C_\ell$ such that $a \cdot U_i = U_{\tau_\ell(i)}$ for all $i \in C_\ell$. Since the classes $\{C_\ell\}$ form a partition of $[n]$, we can construct a global permutation $\tau \in S_n$ as the disjoint union of these permutations: $\tau = \cup_{\ell=1}^{L} \tau_\ell$. By construction, this τ satisfies $(\tau, a) \cdot U = U$. By Definition 6, this means a is a sign automorphism. Therefore, $a \in A$, which means $\hat{W} \subseteq A$. □

It will be useful to define $\pi_\ell : \mathbb{Z}_2^k \to \mathbb{Z}_2^{K_\ell}$ to be the natural projection from $\hat{W}_\ell$ to W_ℓ, and choose for each ℓ a *parity-check matrix* T_ℓ of W_ℓ: a linear map $T_\ell : \mathbb{Z}_2^{K_\ell} \to \mathbb{Z}_2^{K_\ell - \dim W_\ell}$ whose kernel is exactly W_ℓ, equivalently a matrix whose rows form a basis of the orthogonal complement $W_\ell^\perp \subseteq \mathbb{Z}_2^{K_\ell}$. We choose T_ℓ canonically as the nonzero rows of the reduced row-echelon form of a basis matrix of $W_\ell^\perp$, with coordinates in increasing order and rows ordered by pivot position. Lemma 11 implies that s is an automorphism of U, if and only if it satisfies the homogeneous linear equations

$$T_\ell \circ \pi_\ell(s) = 0, \quad \forall \ell = 1, \ldots, L. \tag{20}$$

B.1.4 Iterative construction of the sign canonicalization

We now redescribe the step of the canonicalization that iteratively adds linear constraints to determine $s = s(U)$ in an equivariant fashion. The iterations are indexed by the spaces $\hat{W}_1, \ldots, \hat{W}_L$. We recall that the basic intuition is that we choose the sign vector $s \in \mathbb{Z}_2^k$ so that for $s.U$, each equivalence class C_ℓ will contain a vector whose entries are all non-negative. However, since the order inside C_ℓ is arbitrary, we must do this in a way that does not depend on the choice in C_ℓ. To do this, for every given $\hat{W}_\ell$ we pick an arbitrary $v_\ell \in A_\ell$, and require that $\pi_\ell(s) \oplus v_\ell$ is in the subspace W_ℓ, or equivalently, that $T_\ell(\pi_\ell(s) \oplus v_\ell) = 0$. This in turn implies that the restriction of $s.U$ to C_ℓ will contain a non-negative row. Thus, the condition that $s.U$ will have a non-negative row in each equivalence class translates to the non-homogeneous linear equations

$$T_\ell \circ \pi_\ell(s) = T_\ell(v_\ell), \quad \forall \ell = 1, \ldots, L$$

In general, these linear equations may be linearly dependent, and in this case the non-homogenous equations may not have a solution. To deal with this, we recursively go over the equations, ordered according to the index ℓ and the order of the rows inside each T_ℓ, and remove an equation if it is linearly dependent on the equations that preceded it. This procedure is described in Algorithm 5, and can be performed in $O(k^3)$ time [14]. As a result, we get a reduced system of non-homogeneous equations

$$\eta_\ell \circ T_\ell \circ \pi_\ell(s) = \eta_\ell \circ T_\ell(v_\ell), \quad \forall \ell = 1, \ldots, L$$

where η_ℓ is the projection that removes the dependent equations found in the ℓ-th stage.[1] The solution $s = s(U)$ does not in general satisfy $\pi_\ell(s) \in A_\ell$ for every ℓ: when η_ℓ drops a row of T_ℓ that lies in the row span of earlier rows of E, the kept rows define a strict relaxation of $\pi_\ell(s) \in A_\ell$, so $s.U$ need not have a uniformly non-negative row in C_ℓ on the coordinates spanned by the dropped rows, since the non-negativity is preserved only on the linearly-independent block of constraints. Nevertheless, s is uniquely determined modulo $\ker(E) = \hat{W}$, the sign-automorphism group of U (Lemma 11), and depends equivariantly on U. We lex-minimize over the coset $s_0 \oplus \hat{W}$ to fix a canonical representative, leading to a valid canonicalization as we now prove:

Theorem 5. *The function $s(U)$ defined above is a sign canonicalization.*

Proof. To show s is a sign canonicalization, we need to prove:

1. s is permutation invariant.
2. $s(t.U) \oplus (t \oplus s(U))$ is a sign automorphism of U, for all U and $t \in \mathbb{Z}_2^k$.

The permutation invariance follows from the invariant construction of the partitions C_ℓ, and the fact that the linear equations constructed do not depend on the choice of representative in C_ℓ.

To prove (2), we note that the sign canonicalization $s = s(U)$ is defined as the solution to the system of equations over $\mathbb{Z}_2$:

$$\eta_\ell \circ T_\ell \circ \pi_\ell(s) = \eta_\ell \circ T_\ell(v_\ell), \quad \forall \ell = 1, \ldots, L$$

and for the sign canonicalization $s' = s(t.U)$, the analogous system reads

$$\eta_\ell \circ T_\ell \circ \pi_\ell(s') = \eta_\ell \circ T_\ell(\pi_\ell(t) \oplus v_\ell), \quad \forall \ell = 1, \ldots, L$$

This implies that

$$\eta_\ell \circ T_\ell \circ \pi_\ell(s \oplus s' \oplus t) = 0, \quad \forall \ell = 1, \ldots, L.$$

Since this homogenous linear equation is equivalent to the homogenous linear equation (20), we deduce that $s \oplus s' \oplus t$ is a sign automorphism, as required. □

Conclusion. Theorem 5 produces a sign canonicalization s. Composing with the lexicographic row sort (Lemma 9) yields the canonicalization PRiSM of Algorithm 1 (stated in Section C), completing the proof of Theorem 2.

[1] This system is exactly the $E \cdot s = f$ of Algorithm 5: stacking the surviving rows of $\eta_\ell \circ T_\ell \circ \pi_\ell$ as rows of a single matrix E over $\mathbb{Z}_2^k$ (zero-padding the columns outside K_ℓ), and the surviving right-hand sides $\eta_\ell \circ T_\ell(v_\ell)$ as entries of f, gives the assembled global system used by the algorithm.

Algorithm 1 PRiSM on simple-spectrum eigenvector matrices (base procedure)

Require: Eigenvector matrix $U \in \mathbb{R}^{n\times k}$ (simple spectrum), precision p
Ensure: Canonicalized matrix $\tilde{U} \in \mathbb{R}^{n\times k}$
Step 1: Initial partition by absolute-value signature
$\mathcal{P}_0 \leftarrow$ PARTITION(U, p) ▷ Algorithm 3
Step 2: $\mathbb{Z}_2$-balanced refinement of the partition
$\{C_1, \dots, C_L\} \leftarrow$ REFINE$(\mathcal{P}_0, U)$ ▷ Algorithm 4
Step 3: Global sign resolution via $\mathbb{Z}_2$ linear system
$s(U) \leftarrow$ SOLVE$(\{C_\ell\}_{\ell=1}^{L}, U)$ ▷ Algorithm 5
Step 4: Canonical row ordering
$\tilde{U} \leftarrow$ rows of $U\,\mathrm{diag}((-1)^{s(U)})$ sorted lexicographically
return $\tilde{U}$
Fast path: when the signature σ produced in Step 1 is injective, Steps 2 and 3 reduce to a single per-column scan, and the shortcut is FASTSIGN *(Algorithm 2).*

Algorithm 2 FASTSIGN, Fast-path sign canonicalization (injective σ)

Require: Eigenvector matrix $U \in \mathbb{R}^{n\times k}$ (simple spectrum), precision p
Ensure: Canonicalized matrix $\tilde{U} \in \mathbb{R}^{n\times k}$; valid whenever the absolute-value signature σ is injective
Step 1: Absolute-value signature
for each node $v \in [n]$ **do**
  $\sigma(v) \leftarrow \big(|U_{v,1}|, \dots, |U_{v,k}|\big)$ rounded to precision p
end for
Step 2: Canonical vertex ordering
$\pi \leftarrow$ lexicographic argsort$(\{\sigma(v)\}_{v\in[n]})$ ▷ $O(nk \log n)$
Step 3: Sign resolution
for $j = 1, \dots, k$ **do**
  $v^* \leftarrow$ first vertex in π with $|U_{v^*,j}| > 0$
  if $U_{v^*,j} < 0$ **then**
    $U_{\cdot,j} \leftarrow -U_{\cdot,j}$ ▷ flip column j
  end if
end for
$\tilde{U} \leftarrow$ rows of U sorted lexicographically ▷ canonical row ordering
return $\tilde{U}$

C Canonicalization Details

This section collects the five algorithms used by PRiSM. The master procedure PRiSM (Algorithm 1) chains three subroutines: PARTITION (Algorithm 3) groups vertices by their absolute-value signature; REFINE (Algorithm 4) refines that partition via spectral Weisfeiler–Leman to enforce the multiset stability (18); and SOLVE (Algorithm 5) assembles a $\mathbb{Z}_2$ linear system from the resulting partition and solves it for the canonical sign vector. Algorithm 2 is a fast-path shortcut, FASTSIGN, that applies whenever the absolute-value signature is already injective. Complexity bounds are given in Section C.1.

PRiSM**, master procedure.** Algorithm 1 ties the three subroutines together: it produces the partition $\{C_1, \dots, C_L\}$, calls SOLVE to obtain the canonical sign-bit vector $s(U)$, and lex-sorts the rows of $U\ \mathrm{diag}((-1)^{s(U)})$ to fix the canonical row ordering, where exponentiation is componentwise.

FASTSIGN, shortcut for injective signatures. When PARTITION already separates every vertex (i.e. σ is injective), the partition is trivial, REFINE is a no-op, and SOLVE reduces to a per-column sign scan. Algorithm 2 runs this shortcut directly.

PARTITION, subroutine 1. Algorithm 3 computes the absolute-value signature $\sigma(v) = (|U_{v,1}|, \dots, |U_{v,k}|)$ for every vertex and groups vertices with identical signatures into classes; the classes are then ordered lexicographically by σ.

Algorithm 3 PARTITION, Initial partition from absolute-value signature

Require: Eigenvector matrix $U \in \mathbb{R}^{n\times k}$, precision p
Ensure: Partition $\mathcal{P}_0 = (P_1, \dots, P_r)$ of $[n]$ with lexicographically ordered signatures
for each node $v \in [n]$ **do**
  $\sigma(v) \leftarrow \big(|U_{v,1}|, \dots, |U_{v,k}|\big)$ rounded to precision p
end for
Group nodes into classes $P_i \leftarrow \{v \in [n] : \sigma(v) = \sigma_i\}$ ▷ one class per distinct signature value
Order the classes $(P_1, \dots, P_r)$ by lexicographic σ_i
return $\mathcal{P}_0 = (P_1, \dots, P_r)$

Algorithm 4 REFINE, spectral Weisfeiler–Leman refinement

Require: Ordered partition $\mathcal{P}_0$, eigenvector matrix $U \in \mathbb{R}^{n\times k}$
Ensure: Refined partition $\{C_1, \dots, C_L\}$ on which the multiset stability (18) holds
$c_0(v) \leftarrow$ index of the class containing v in $\mathcal{P}_0$, for each $v \in [n]$
$h_v^{(0)} \leftarrow (c_0(v), U_v \odot U_v)$ for each $v \in [n]$ ▷ preserve the input partition
$t \leftarrow 0$
repeat
  for each $v \in [n]$ **do**
    $h_v^{(t+1)} \leftarrow \textsc{Hash}\Big(h_v^{(t)},\ \{\!\{(h_u^{(t)},\ U_v \odot U_u) : u \in [n]\}\!\}\Big)$ ▷ multiset of edge features
  end for
  $t \leftarrow t + 1$
until the partition induced by $\{h_v^{(t)}\}$ equals the partition induced by $\{h_v^{(t-1)}\}$
$\{C_1, \dots, C_L\} \leftarrow$ classes of $\sim_h$, ordered lexicographically by representative color
return $\{C_1, \dots, C_L\}$
Implementation note: a concrete realization with sufficient fixed-point partition uses, for each class C, the $\mathbb{Z}_2$-independent sign-bit columns $Q \subseteq [k]$ and splits C whenever some product $p_S(v) = \prod_{\ell\in S} U_{v,\ell}$ ($S \subseteq Q$) has imbalanced $\pm$ counts. See Remark 1 below.

REFINE, subroutine 2. Algorithm 4 refines $\mathcal{P}_0$ by spectral Weisfeiler–Leman: each vertex's color is updated using a multiset of edge features $U_v \odot U_u$, and iteration continues until the partition stabilizes. The fixed point satisfies the multiset stability (18) on which the proof of Theorem 2 relies. Remark 1 below gives an equivalent $\mathbb{Z}_2$ parity-check realization used in our implementation.

Remark 1 ($\mathbb{Z}_2$ realization of REFINE)**.** The companion implementation realizes the spectral-WL fixed point above by a $\mathbb{Z}_2$ parity-check scheme. For each working class C, maintain the set $Q \subseteq [k]$ of sign-bit columns of $U|_C$ that are $\mathbb{Z}_2$-independent, and for every newly added column $j \in Q$ test all subsets $S \subseteq Q$ containing j: if the product $p_S(v) = \prod_{\ell\in S} U_{v,\ell}$ has unequal counts of $+x$ and $-x$ over $v \in C$ for some $|x| > 0$, split C along $\mathrm{round}(p_S)$ and recurse. This produces the same partition as the iteration in Algorithm 4 but avoids explicit color hashing.

SOLVE, subroutine 3. Given the balanced partition $\{C_1, \dots, C_L\}$, Algorithm 5 assembles the $\mathbb{Z}_2$ linear system $E \cdot s = f$ described in the proof, dropping linearly dependent rows on the fly, and returns the lex-minimal element of the solution coset $s_0 \oplus \ker(E)$ together with the sign-automorphism group $\hat{W} = \ker(E)$.

C.1 Complexity

The total complexity of simple-spectrum canonicalization PRiSM is $O(nk \log n + k^3 + nk^2)$. We account for each stage separately.

Partition. Computing the absolute-value signature $\sigma(v) = (|U_{v,1}|, \dots, |U_{v,k}|)$ for all n vertices costs $O(nk)$. Sorting the n signatures lexicographically costs $O(nk \log n)$. The stage total is $O(nk \log n)$.

Refine and Solve. The Algorithm 5 processes at most n balanced blocks (one per node in the worst case). For each block it builds the parity-check matrix T_ℓ and embeds its rows into the global system E, which has k columns. Each block contributes at most k rows (by linear dependence), and checking

Algorithm 5 SOLVE, Solve $\mathbb{Z}_2$-linear system for canonical signs

Require: Balanced partition $\{C_1, \ldots, C_L\}$ from Algorithm 4, eigenvector matrix U
Ensure: Canonical sign vector $s(U) \in \mathbb{Z}_2^k$; sign automorphism group $\hat{W} \leq \mathbb{Z}_2^k$

$E \leftarrow$ empty matrix over $\mathbb{Z}_2$ (k columns); $f \leftarrow$ empty vector
for $\ell = 1, \ldots, L$ **do**
  $K_\ell \leftarrow \{j \in [k] : |U_{v,j}| > 0,\ v \in C_\ell\}$
  if $K_\ell = \emptyset$ **then continue**
  end if
  $A_\ell \leftarrow \{\mathbf{1}[U_{v,K_\ell} < 0] : v \in C_\ell\}$; fix $\mathbf{a}_0 \in A_\ell$
  $W_\ell \leftarrow \mathrm{rowspace}_{\mathbb{Z}_2}(\{\mathbf{a} \oplus \mathbf{a}_0 : \mathbf{a} \in A_\ell\})$
  $T_\ell \leftarrow$ canonical RREF basis matrix of $W_\ell^\perp$ ▷ $\ker(T_\ell) = W_\ell$
  $\mathbf{b}_\ell \leftarrow T_\ell \cdot \mathbf{a}_0$ ▷ RHS; independent of choice of $\mathbf{a}_0 \in A_\ell$
  for each row $\mathbf{r}$ of T_ℓ in pivot order, with entry b of $\mathbf{b}_\ell$ **do**
    Embed: $\hat{\mathbf{r}}_j \leftarrow \mathbf{r}_j$ if $j \in K_\ell$, else 0
    if $\hat{\mathbf{r}} \notin \mathrm{rowspace}(E)$ **then** Append $\hat{\mathbf{r}}$ to E; append b to f
    end if
  end for
end for
Solve $E \cdot s = f$ over $\mathbb{Z}_2$; $s_0 \leftarrow$ particular solution
$s(U) \leftarrow \text{lex-min}(s_0 \oplus \ker(E))$ ▷ $\ker(E) = \hat{W}$
return $s(U), \hat{W} = \ker(E)$

independence against the existing rowspace of E costs $O(k)$ per row, giving $O(k^2)$ per block and $O(nk^2)$ total across all n blocks.

The $O(k^3)$ term comes from two operations on the assembled k-column $\mathbb{Z}_2$ system:

1. Solving $E \cdot s = f$ over $\mathbb{Z}_2$ via Gaussian elimination on a matrix with at most k rows and k columns: $O(k^3)$.
2. Lex-minimizing over the coset $s_0 \oplus \ker(E)$: finding the lex-min element of an affine subspace of $\mathbb{Z}_2^k$ by a greedy pass over the basis of $\ker(E)$ costs $O(k^2)$, dominated by the $O(k^3)$ solve.

Combining, the stage total is $O(nk^2 + k^3)$.

Total. Summing over all stages, the overall complexity is $O(nk \log n + k^3 + nk^2)$, matching the bound stated in the main text.

Complexity of FASTSIGN When the absolute-value signature σ is injective, the FASTSIGN fast path (Algorithm 2) suffices: reading nk absolute values is $O(nk)$; the lexicographic argsort costs $O(nk \log n)$; the per-column sign scan is $O(nk)$. The total complexity is therefore $O(nk \log n)$.

The cubic factor is a small constant in practice. The $O(k^3)$ term comes from Gaussian elimination on the $\mathbb{Z}_2$ parity-check matrix E assembled by SOLVE. The row count of E is upper-bounded by $r = n - L$, where L is the number of equivalence classes returned by PARTITION. To see the bound, recall that a non-singleton class C_ℓ of size m contributes to E only constraints spanned by the pairwise sign-bit differences $\{s_v \oplus s_w : v, w \in C_\ell\}$, since the absolute positions of the per-vertex sign-bit vectors $\{s_v\}_{v \in C_\ell} \subseteq \mathbb{Z}_2^{K_\ell}$ flip under a column-wise sign change of U while their differences do not. Fix any anchor $v_0 \in C_\ell$: the $m-1$ anchored differences $\{s_v \oplus s_{v_0}\}_{v \in C_\ell \setminus \{v_0\}}$ already span this hull, since every other difference telescopes as $s_v \oplus s_w = (s_v \oplus s_{v_0}) \oplus (s_w \oplus s_{v_0})$. Hence C_ℓ contributes at most $m-1$ independent relative-sign relations, as singleton classes ($m = 1$) contribute none, summing over all classes gives $\sum_\ell (m_\ell - 1) = n - L$. We measured r on the entire validation splits of ZINC [28] and the eleven OGB-MOL datasets [27] (about 62k graphs total), using full eigendecompositions ($k = n$) of the normalized Laplacian with tolerance 10^{-6}, and we report the statistics in Table 4. The median r is 1–3 across every dataset, and $r \leq 4$ on 64–88% of graphs, so the cubic stage costs at most $4^3 = 64$ operations on the median graph, a constant independent of n and k. On 3.3–45.3% of graphs the partition is already injective ($r = 0$) and PRiSM degenerates to

FASTSIGN. On a further 3–17% the partition produces exactly one non-singleton block. In that case the linear system collapses to a single sign convention, for instance forcing the lex-smallest row of the block to be non-negative, so no $\mathbb{Z}_2$ solve is performed.

Table 4: Distribution of the $\mathbb{Z}_2$ row-count bound $r = n - L$ on entire validation splits, full eigendecomposition ($k = n$) of the normalized Laplacian, tolerance 10^{-6}. Columns are the number of validation graphs |val|, the mean number of nodes per graph $\bar{n}$, the fraction $r{=}0$ on which FASTSIGN applies, the fraction "1 block" with exactly one non-singleton block (no $\mathbb{Z}_2$ solve needed), and the median r together with the fraction satisfying $r \leq 4$ which summarise the residual cubic cost ($\leq 4^3 = 64$).

Dataset	|val|	$\bar{n}$	$r{=}0$	1 block	median r	$r \leq 4$	max r
ZINC-12K	1,000	23.1	22.8%	10.9%	2.0	88.0%	11
MOLHIV	4,113	27.8	21.2%	6.4%	3.0	63.9%	53
MOLTOX21	783	26.8	17.5%	8.6%	3.0	67.4%	32
MOLCLINTOX	148	32.8	27.0%	9.5%	2.0	70.3%	37
MOLBBBP	204	33.2	14.2%	17.2%	2.0	81.4%	32
MOLBACE	151	37.2	3.3%	10.6%	3.0	81.5%	6
MOLESOL	113	17.7	38.1%	14.2%	1.0	77.0%	15
MOLFREESOLV	64	12.0	45.3%	4.7%	1.5	78.1%	12
MOLLIPO	420	27.3	14.8%	7.9%	3.0	74.3%	25
MOLMUV	9,309	25.3	12.8%	3.3%	3.0	73.3%	17
MOLTOXCAST	858	26.2	20.5%	10.7%	2.5	69.4%	37
MOLPCBA	43,793	27.1	13.9%	4.7%	3.0	74.3%	71

D Proof of Universal Approximation (Theorem 3)

Theorem 3 (Universal Approximation over Graphs with a Simple Spectrum)**.** *Fix $M > 0$ and $\delta > 0$. Let $\mathcal{K}(N, M, \delta)$ denote the set of simple-spectrum weighted graphs with at most N vertices, edge weights bounded by M, and minimum eigenvalue gap at least δ (i.e. $|\lambda_i - \lambda_j| \geq \delta$ for $i \neq j$). Let $f : \mathcal{K}(N, M, \delta) \to \mathbb{R}$ be a continuous, isomorphism-invariant function and let $\epsilon > 0$. Denote the eigendecomposition of a graph g by $(U_g, \vec{\lambda}_g)$. Then for any universal set model there exists a model instantiation ψ such that*

$$\forall g \in \mathcal{K}(N, M, \delta), \quad |f(g) - \psi \circ \mathrm{PRiSM}(U_g, \vec{\lambda}_g)| < \epsilon.$$

Before applying the proof, we add some clarifications on the meaning of a universal set model. Recall that a graph $g \in \mathcal{K}(N, M, \delta)$ has simple spectrum, $n \leq N$ nodes and edge weights bounded by M. Its corresponding eigendecomposition is comprised by an $n \times n$ matrix of eigenvectors, and n eigenvalues. We note that $(U_g, \vec{\lambda}_g)$ reside in a bounded set $\mathcal{C}_n \subseteq \mathbb{R}^{n \times n} \oplus \mathbb{R}^n$: the eigenvector matrix is orthogonal so its rows have unit norm, and the eigenvalues satisfy

$$\|\vec{\lambda}_g\|_2 = \|A_g\|_F \leq M\, n,$$

where A_g is the adjacency matrix of g.

Regarding the eigenvalue gap constraint, the constraint $|\lambda_i - \lambda_j| \geq \delta$ ensures that $\mathcal{C}_n$ stays bounded away from the repeating-eigenvalue region, where the simple-spectrum hypothesis would fail and ensures the domain is compact, which must hold for the universality results to be correct.

We now show explicitly that $\mathcal{C}_n$ is a closed set; since we have already shown previously it is bounded, consequently it is compact. Weyl [53] proves that for any pair of symmetric matrices $A, B \in \mathbb{R}^{n \times n}$ with eigenvalues $\lambda_1(A) \geq \cdots \geq \lambda_n(A)$ and $\lambda_1(B) \geq \cdots \geq \lambda_n(B)$, one has

$$|\lambda_i(A) - \lambda_i(B)| \leq \|A - B\|_{\mathrm{op}} \leq \|A - B\|_F,$$

for every i , so the ordered eigenvalue map $A \mapsto \vec{\lambda}(A)$ is 1-Lipschitz, hence continuous.

Take a sequence $(U_k, \vec{\lambda}_k) \in \mathcal{C}_n$ converging to some $(U_*, \vec{\lambda}_*)$. The associated adjacency matrices $A_k = U_k \,\mathrm{diag}(\vec{\lambda}_k) U_k^\top$ converge in Frobenius norm to $A_* = U_* \,\mathrm{diag}(\vec{\lambda}_*) U_*^\top$. By Weyl, $\vec{\lambda}(A_k) \to \vec{\lambda}(A_*)$, and by uniqueness of the eigendecomposition we identify $\vec{\lambda}(A_k) = \vec{\lambda}_k$ and $\vec{\lambda}(A_*) = \vec{\lambda}_*$. Hence the closed conditions $\{\|A\|_\infty \leq M\}$ and $\{|\lambda_i - \lambda_j| \geq \delta\}$ both pass to the limit point, so A_*

is the adjacency of a graph in $\mathcal{K}(N, M, \delta)$ and $(U_*, \vec{\lambda}_*) \in \mathcal{C}_n$. Combining with the boundedness above, $\mathcal{C}_n$ is compact, and so is the finite union $\mathcal{C}_N = \cup_{n \leq N} \mathcal{C}_n$, as we set out to prove.

Accordingly, when we discuss universal set models, we mean model classes which can uniformly approximate to arbitrary accuracy on $\mathcal{C}_N$ all continuous functions which are invariant to permutation of the rows of U. Examples of such models are DeepSets [23] or Transformers [24, 25], where care needs to be taken to adapt these models to process the additional eigenvalue input. For example, a DeepSets model which would be appropriate for this theorem is

$$\psi(U, \vec{\lambda}) = \mathrm{MLP}^{(2)}\left(\vec{\lambda}, \sum_{i=1}^{n} \mathrm{MLP}^{(1)}(U(i,:))\right).$$

To make sure the model is defined and universal for all $n \leq N$ simultaneously we can apply padding to make sure the input is always of the same dimension.

Proof of Theorem 3. on $\mathcal{C}_N$ define the function $\tilde{f}(U, \vec{\lambda}) = f(U \operatorname{diag}(\vec{\lambda}) U^T)$. Then $\tilde{f}$ is

1. **Continuous** as a composition of continuous functions,
2. **Invariant** For any graph $g \in \mathcal{K}(N, M, \delta)$ with $n \leq N$ nodes, any $n \times n$ permutation matrix P and diagonal matrix S with diagonal entries in $\{-1, 1\}$, we have that

   $$\tilde{f}(PUS, \vec{\lambda}) = f(PUS \operatorname{diag}(\vec{\lambda}) SU^T P^T) \overset{(*)}{=} f(U \operatorname{diag}(\vec{\lambda}) U^T) = \tilde{f}(U, \vec{\lambda}),$$

   where $(*)$ is because f is isomorphism-invariant, diagonal matrices commute, and $S^2 = I_n$.
3. **Consistent** By which we mean that $\tilde{f}(U_g, \vec{\lambda}_g) = f(g)$ for any simple-spectrum graph g.

For given $\epsilon > 0$, since $\tilde{f}$ is permutation invariant, we can approximate $\tilde{f}$ with a universal set model ψ to ϵ accuracy on $\mathcal{C}_N$. It follows that for all $g \in \mathcal{K}(N, M, \delta)$

$$\begin{aligned} |f(g) - \psi \circ \mathrm{PRiSM}(U_g, \vec{\lambda}_g)| &\overset{\text{consistency}}{=} |\tilde{f}(U_g, \vec{\lambda}_g) - \psi \circ \mathrm{PRiSM}(U_g, \vec{\lambda}_g)| \\ &\overset{\text{invariance}}{=} |\tilde{f} \circ \mathrm{PRiSM}(U_g, \vec{\lambda}_g) - \psi \circ \mathrm{PRiSM}(U_g, \vec{\lambda}_g)| < \epsilon. \end{aligned}$$

□

Remark 2. There is a fundamental trade-off between continuity and completeness on continuous-edge-weight simple-spectrum graphs. Existing approaches sacrifice completeness in different ways: SignNet [20] retains continuity via sign-equivariant architectures but collapses distinct eigendecompositions, trading injectivity for continuity. MAP [21] and OAP [22] are heuristic and fail on both axes, being neither injective nor continuous. Our scheme achieves completeness on simple-spectrum graphs but sacrifices continuity on $\mathcal{K}(N, M, \delta)$.

E Experimental Details

Experimental protocol. All eigendecompositions are computed using `torch.linalg.eigh` in float64 precision. Eigenvalue multiplicity statistics for our molecular benchmarks are reported in Section E.6. Two eigenvalues are considered equal when their absolute difference falls below 10^{-8}. This threshold is shared across all canonicalization methods. We verified that varying this threshold between 10^{-6} and 10^{-10} does not alter the reported scores. All methods use an identical GNN backbone with a DeepSets [23] eigenvector embedding and matched parameter budget ($\sim$12K parameters), isolating the effect of the canonicalization method.

PRiSM **on graphs with higher eigenvalue multiplicities.** The BREC and molecular experiments apply the hybrid extension of PRiSM described in Algorithm 6. This extension is heuristic at multiplicities $m_i > 1$ and is not complete on $\mathcal{G}_n^M$ for $M > 1$; on a simple-spectrum graph every eigenspace has multiplicity 1, so the extension reduces to the base PRiSM of Algorithm 1 and inherits its completeness. The method proceeds in two stages:

Stage 1: Canonical vertex ordering via the norm-value signature. We compute the eigendecomposition of the Laplacian, grouping eigenvalues into K eigenspaces of multiplicity $m_1, \ldots, m_K$ (within tolerance τ). Let π_i denote orthogonal projection onto the i-th eigenspace. The norm-value signature $\sigma(v) = (\|\pi_1 v\|, \ldots, \|\pi_K v\|)$ is a permutation-equivariant function of the vertex v: relabeling the graph permutes σ rows accordingly, so the lexicographic argsort of σ produces a canonical vertex ordering up to ties between equal-signature vertices. Ties (when present) are broken sequentially in Stage 2 using canonical eigenvector values from already-processed eigenspaces.

Stage 2a: Joint sign canonicalization on the simple-spectrum block. Let $\mathcal{I}_1 = \{i : m_i = 1\}$ and let $U^{(1)} \in \mathbb{R}^{n \times k}$ be the matrix formed by stacking the eigenvectors of all $m = 1$ eigenspaces, where $k = |\mathcal{I}_1|$. We run REFINE (Algorithm 4) on $U^{(1)}$ starting from the σ-induced partition $\mathcal{P}_0$, obtaining a refined partition $\{C_1, \ldots, C_L\}$. We then run SOLVE (Algorithm 5) on this partition to obtain $s \in \mathbb{Z}_2^k$, and replace $U^{(1)}$ by $U^{(1)}\ \mathrm{diag}((-1)^s)$, writing the sign corrections back into the corresponding U_i for $i \in \mathcal{I}_1$. The refined partition $\{C_\ell\}$ together with the canonical values of $U^{(1)}$ are also used to break σ-ties in the vertex ordering before Stage 2b.

Stage 2b: Per-eigenspace $O(m_i)$ resolution for $m_i \geq 2$, in increasing multiplicity. For each remaining eigenspace the eigenvectors are defined up to an $O(m_i)$ rotation, which we resolve according to the multiplicity:

- $m_i = 2$ (rotation + reflection): Align the first non-trivial pivot to the positive x-axis via an $SO(2)$ rotation; resolve the residual $\mathbb{Z}_2$ reflection from the next pivot.
- $m_i = 3$ (Householder + rotation + reflection): Householder-reflect the first pivot to the x-axis; x-axis-rotate the second pivot's yz-component to the positive y-axis; flip column sign for the residual $\mathbb{Z}_2$.
- $m_i \geq 4$ (generalized QR): Select m_i pivot rows via Gram-Schmidt in canonical order (preferring unique-σ vertices); apply QR with the Stewart sign convention (positive diagonal of R).

After each higher-multiplicity eigenspace is canonicalized, when σ-ties remain we refine the vertex ordering using the canonical values produced so far before moving on to the next eigenspace. The resulting eigenvectors form the input features to the downstream model. Algorithm 6 collects the full procedure in pseudocode.

E.1 ZINC 12K Dataset and Protocol

ZINC 12K [19] is a subset of the ZINC molecular database consisting of approximately 12,000 drug-like molecules. The task is regression on constrained solubility (logP score), evaluated by mean absolute error (MAE). Molecules have a mean of approximately 23 nodes and 25 edges; the graph structure is derived from the molecular bond graph. The dataset is split into 10,000 / 1,000 / 1,000 train/val/test molecules following the standard split of Dwivedi et al. [19].

We evaluate two backbone architectures. The GatedGCN [48] backbone uses gated graph convolution with residual connections and edge features; the PNA [49] backbone uses principal neighbourhood aggregation with degree-based scalers. Both backbones process positional encodings produced by the canonicalization and fuse them with node features additively, following the GPS framework [54]. The sign-equivariant embedding network (sign-inv-net) uses a DeepSets architecture [23] applied independently to the positive and negative parts of each eigenvector column, with their outputs summed to produce a sign-invariant embedding.

Hyperparameter configuration.

Training protocol. Models are trained for 200 epochs with the OneCycleLR scheduler and early stopping based on validation MAE with patience 50. Performance is reported as mean $\pm$ std over four independent seeds with different random initializations. The evaluation metric is MAE on the held-out test set (lower is better). Eigendecompositions use all $n - 1$ non-trivial eigenvectors of the normalized graph Laplacian, computed in float64 precision via `torch.linalg.eigh`.

Algorithm 6 PRiSM on graphs with bounded eigenvalue multiplicity (hybrid extension of Algorithm 1)

Require: Adjacency matrix $A \in \{0,1\}^{n \times n}$, multiplicity tolerance τ
Ensure: Canonical eigenvector matrix $[U_1 \,\|\, \cdots \,\|\, U_K]$

Stage 1: Canonical vertex ordering via norm-value signature.

$L \leftarrow D - A$; eigendecompose L into eigenspaces $\{(\lambda_i, U_i)\}_{i=1}^{K}$ with multiplicities m_i (grouping within tolerance τ)

$\sigma(v) \leftarrow (\|\pi_1 v\|, \ldots, \|\pi_K v\|)$ for each vertex v ▷ norm-value signature; π_i is projection onto U_i

order $\leftarrow$ lex-argsort(σ) ▷ canonical vertex ordering up to σ-ties

Stage 2a: Joint sign canonicalization on the simple-spectrum block.

$\mathcal{I}_1 \leftarrow \{i : m_i = 1\}$; $U^{(1)} \leftarrow [U_i]_{i \in \mathcal{I}_1} \in \mathbb{R}^{n \times k}$ ▷ $k = |\mathcal{I}_1|$, the number of $m = 1$ eigenspaces

$\{C_1, \ldots, C_L\} \leftarrow$ REFINE$(\mathcal{P}_0, U^{(1)})$, with $\mathcal{P}_0$ induced by σ ▷ spectral WL refinement on the $m = 1$ block; Algorithm 4

$s \leftarrow$ SOLVE$(\{C_\ell\}_{\ell=1}^{L}, U^{(1)}) \in \mathbb{Z}_2^k$ ▷ $\mathbb{Z}_2$ system over the $m = 1$ columns; Algorithm 5

$U^{(1)} \leftarrow U^{(1)}\,\mathrm{diag}((-1)^s)$; write $\mathrm{diag}((-1)^s)$ back into the corresponding U_i for $i \in \mathcal{I}_1$

Refine order using $\{C_\ell\}$ together with the canonical values of $U^{(1)}$ ▷ spectral WL + absolute-norm signature ordering

Stage 2b: Per-eigenspace $O(m_i)$ resolution for $m_i \geq 2$, in increasing multiplicity.

for eigenspace i with $m_i \geq 2$, taken in order $m_i = 2, 3, \ldots$ **do**
  if σ-ties remain **and** prior eigenspaces canonicalized **then**
    Refine order using canonical values from already-processed eigenspaces
  end if
  if $m_i = 2$ **then**
    Align the first non-trivial pivot to the positive x-axis via an $SO(2)$ rotation; resolve the residual $\mathbb{Z}_2$ reflection from the next pivot
  else if $m_i = 3$ **then**
    Householder-reflect the first pivot to the x-axis; x-axis-rotate the second pivot's yz-component to the positive y-axis; flip column sign for the residual $\mathbb{Z}_2$
  else ▷ $m_i \geq 4$, generalized QR
    Pick m_i pivot rows via Gram–Schmidt in order (preferring unique-σ vertices); apply QR with the Stewart sign convention (positive diagonal of R)
  end if
end for
return $[U_1 \,\|\, \cdots \,\|\, U_K]$

E.2 OGB Molecular Datasets and Protocol

We evaluate on four datasets from the Open Graph Benchmark (OGB) molecular suite [27]. All are drawn from MoleculeNet and use scaffold-based train/val/test splits provided by OGB.

- **MolTox21**: 7,831 molecules (6,264 / 783 / 784 train/val/test). Multi-task binary classification across 12 toxicology assays. Mean ≈ 18 nodes, ≈ 19 edges per molecule. Metric: mean ROC-AUC (higher is better).
- **MolHIV**: 41,127 molecules (32,901 / 4,113 / 4,113 train/val/test). Single binary classification task (HIV inhibition). Mean ≈ 25 nodes, ≈ 27 edges. Metric: ROC-AUC.
- **MolClinTox**: 1,477 molecules (1,181 / 148 / 148 train/val/test). Binary classification across 2 clinical toxicology assays. Mean ≈ 26 nodes, ≈ 28 edges. Metric: mean ROC-AUC.
- **MolBBBP**: 2,039 molecules (1,631 / 204 / 204 train/val/test). Binary classification for blood-brain barrier penetration. Mean ≈ 24 nodes, ≈ 26 edges. Metric: ROC-AUC. MOLBBBP appears in the Transformer-backbone main results (Table 3) using the no-rank-embedding PRiSM configuration (sequential refinement only); the RoPE-augmented configuration degraded on this small dataset and is reported in the supplementary archive at `exps/`. We additionally report eigenvalue multiplicity statistics in Table 9.

The GNN experiments on MolTox21 use GatedGCN [48] and PNA [49] backbones with the same GPS-style fusion used for ZINC (Section E.1).

Table 5: Hyperparameter configuration for ZINC 12K experiments. Values marked (default) were not exhaustively tuned and follow the conventions of Dwivedi et al. [19].

Hyperparameter	GatedGCN	PNA
Learning rate	1×10^{-3}	5×10^{-4}
Batch size	128	128
Hidden dimension	128	80
Number of layers	4	4
Dropout	0.0	0.05
Optimizer	AdamW	AdamW
LR scheduler	OneCycleLR	OneCycleLR
Training epochs	200	200
Weight decay	0.0 (default)	0.0 (default)
Number of seeds	4	4
Parameter budget ($\approx$)	472K	473K

Table 6: Hyperparameter configuration for OGB MolTox21 GNN experiments. Values marked (default) follow OGB benchmark conventions.

Hyperparameter	GatedGCN	PNA
Learning rate	5×10^{-4}	5×10^{-4}
Batch size	512	512
Hidden dimension	128	128
Number of layers	4	4
Dropout	0.1	0.1
Optimizer	AdamW	AdamW
LR scheduler	ReduceLROnPlateau	ReduceLROnPlateau
Training epochs	100 (default)	100 (default)
Weight decay	0.0 (default)	0.0 (default)
Number of seeds	4	4
Parameter budget ($\approx$)	1.54M	n/a

Hyperparameter configuration.

Training protocol. Models are trained with early stopping based on validation ROC-AUC with patience 20. Performance is reported as mean $\pm$ std over four seeds. The sign-inv-net uses the same DeepSets [23] architecture as in the ZINC experiments. SignNet, MAP, and OAP baselines were not evaluated on the PNA backbone in our pipeline, and these entries are marked "n/a" in Table 2.

E.3 Transformer Backbone Configuration

The Transformer backbone experiments on OGB molecular datasets and ALCHEMY (Section 6.3) use the graph Transformer architecture of Kim et al. [25] with Laplacian positional encodings. Canonical eigenvectors (CanonLPE) replace the standard random-sign Laplacian PE; the canonical vertex rank produced by Algorithm 6 is then encoded into token positions in one of three ways, with the choice fixed per dataset: (i) Rotary Position Embedding (RoPE) [55] for relative positions on MOLHIV and ALCHEMY; (ii) sinusoidal absolute encoding combined with sequential ordering refinement on MOLCLINTOX; (iii) sequential ordering refinement only, with no rank embedding, on MOLBBBP. "Sequential refinement" uses canonical eigenvector values from already-processed eigenspaces to break ties in the vertex ordering for subsequent eigenspaces. The Transformer encoder uses multi-head self-attention with pre-norm layer normalization.

Hyperparameter configuration.

Architecture description. Node tokens are formed by projecting atom features and positional encodings into the hidden dimension using separate linear layers, then summing. Global readout uses mean pooling over node tokens followed by an MLP classification head. MAP baseline [21] uses the

Table 7: Hyperparameter configuration for Transformer backbone experiments on OGB molecular datasets. Values marked (default) follow the conventions of Kim et al. [25].

Hyperparameter	**Value**
Hidden dimension	128
Number of attention heads	8
Number of Transformer layers	3
Learning rate	1×10^{-4}
Batch size	128
Dropout	0.1
Optimizer	AdamW
LR scheduler	Warmup + cosine decay
Warmup steps	1,000 (default)
Training epochs	100 (default)
Weight decay	1×10^{-5} (default)
Number of seeds	3 on MOLHIV, MOLCLINTOX, and MOLBBBP, and 3 on ALCHEMY (seeds $\{0, 1, 3\}$)
Positional encoding	CanonLPE with RoPE
PE dimension	16

same architecture with MAP-canonicalized eigenvectors in place of CanonLPE; LPE baseline uses the standard random-sign Laplacian PE without any canonicalization.

Training protocol. Models are trained with early stopping based on validation ROC-AUC with patience 20. Performance is reported as mean $\pm$ std over three seeds for all OGB datasets and over $\{0, 1, 3\}$ for ALCHEMY (seed 2 excluded as a $\sim 5\sigma$ outlier). The OAP baseline was run for MOLHIV (single seed) but is omitted from Table 3 as a single-seed result is not sufficiently reliable for comparison.

Why PRiSM outperforms MAP and OAP on transformers. A canonicalization usable as a transformer positional encoding must supply three pieces of information: (a) deterministic eigenvectors (sign and basis canonicalized); (b) a deterministic vertex ordering; and (c) an embedding of that ordering into token positions. MAP [21] and OAP [22], implemented in our experiments using code imported from the authors' original implementations, supply only (a): each resolves sign and basis ambiguity but does not produce a vertex ordering, leaving the transformer without anchor positions. PRiSM supplies all three: PARTITION together with sequential refinement (Algorithm 6) yields a deterministic node permutation from canonical eigenvector signatures, and the chosen rank embedding (RoPE for relative or sinusoidal for absolute) feeds that ordering to the attention layers. The 0.6–6.8 pp gaps to MAP across MOLHIV, MOLCLINTOX, and MOLBBBP, together with the 8.4% relative MAE reduction on ALCHEMY, in Table 3 are consistent with this mechanism.

E.4 Empirical Invariance Test

Incompleteness in canonicalization leads to the breaking of desired permutation equivariance of graph eigendecomposition composed with the canonicalization. This often occurs when permuting the entries results in a different choice of 'anchor' the canonicalization uses to resolve an ambiguity. To empirically exemplify this, we measure permutation-equivariance failure rates of each canonicalization (200 graphs $\times$ 5 relabelings = 1,000 trials per method; Table 8) on ZINC and MOLTOX21 datasets.

Despite PRiSM's incompleteness on general graphs (it is complete on simple-spectrum graphs), PRiSM is exactly equivariant on ZINC (0/1,000) and near-exact on MOLTOX21 (10/1,000), while MAP and OAP fail on 68–77% of trials, possibly a reason for their degraded downstream performance. We attribute this to PRiSM's completeness on simple-spectrum eigendecompositions and its principled canonicalization procedure.

Table 8: Canonicalization permutation-equivariance failure counts under random vertex permutations with independent full eigendecomposition. 200 training graphs per dataset $\times$ 5 random permutations $= 1{,}000$ trials per cell. Each entry is the number of trials in which the canonical positional encoding of the permuted graph is not a row-permutation of the canonical positional encoding of the original (row-multiset equality, $\mathrm{atol} = 10^{-6}$). Lower is better.

Dataset	**Method**	**Failures /** $1{,}000\downarrow$
ZINC	PRiSM **(ours)**	**0**
	MAP	692
	OAP	684
MolTox21	PRiSM **(ours)**	**10**
	MAP	772
	OAP	753

E.5 Compute Resources

Hardware. Canonicalization (PRiSM, MAP, OAP) is a CPU-only preprocessing step: every operation is real-arithmetic linear algebra on float64 numpy arrays, with no GPU dependency. Downstream training of the GNN backbones (Section E.1, Section E.2) and Transformer backbone (Section E.3) was carried out on a pool of 8 NVIDIA A40 GPUs (48 GB memory each), with a single GPU assigned per training run; no multi-node or cluster compute beyond this 8-GPU pool was used. CPU memory required for the eigendecomposition step is negligible at the molecular scale used throughout.

Per-run compute budget. Canonicalization wall-clock per graph is sub-millisecond to a few milliseconds at molecular scale (a timing micro-benchmark of PRiSM, MAP, and OAP on connected Erdős–Rényi graphs across $n \in \{16, \ldots, 512\}$ is included in the supplementary archive). A full pass over each molecular dataset (ZINC-12K, OGB-Mol, Alchemy) takes minutes of CPU wall-clock and is run once per dataset, then cached. Per-seed GPU wall-clock varies considerably by backbone, dataset, and configuration; ranges read directly from `TOTAL TIME TAKEN` lines across the retained training logs are: $\sim$1.0–4.3 h for GatedGCN-ZINC, $\sim$4.8–18.0 h for PNA-ZINC, $\sim$0.4–1.5 h for GatedGCN/PNA-MolTox21, $\sim$3.8–44.6 h for MolPCBA (PNA-MolPCBA at the upper end), and $\sim$3–9 h per seed for Transformer training (Section E.3) depending on dataset. BREC distinguishability runs (Table 1) use the matched parameter budget recorded in the table caption on a single GPU.

Total project compute. Aggregating GPU wall-clock across all GNN-backbone runs whose logs we retained (158 runs covering ZINC, MolTox21, and MolPCBA on the GatedGCN and PNA backbones, including ablations and unreported configurations), the molecular GNN track consumed approximately 1,500 GPU-hours. The Transformer-backbone track (Table 3, four datasets $\times$ three seeds, plus the discarded RoPE configuration on MolBBBP and other unreported configurations whose logs we retained) consumed approximately an additional 500 GPU-hours. The reported tables are a strict subset of these runs; the remainder is preliminary and exploratory work that did not make it into the final paper. We did not run any single experiment requiring more than a single GPU.

E.6 Eigenvalue Multiplicity Statistics

Per-dataset distributions of eigenvalue multiplicities are summarized in Table 9. We sampled 500 graphs uniformly at random from each dataset and computed the eigenvalues of the normalized adjacency $D^{-1/2}AD^{-1/2}$ in float64 precision via `numpy.linalg.eigvalsh`, grouping eigenvalues into multiplicity classes with absolute tolerance 10^{-6} (matching the threshold used throughout our pipeline). For each graph we count the number of *distinct* eigenvalues per multiplicity class. Across all five molecular benchmarks, on average more than 85% of distinct eigenvalues per graph have multiplicity one, and the mean count of eigenvalues with multiplicity $m \geq 4$ is below 0.3 per graph. Strict simple-spectrum graphs (every eigenvalue distinct) occur in 29%–38% of sampled graphs; the remainder typically have one or two repeated eigenvalues induced by small graph symmetries. This empirical profile motivates the bounded-multiplicity hybrid (Algorithm 6), which combines

the rigorous simple-spectrum canonicalization (Algorithm 1) on the dominant $m{=}1$ block with a per-eigenspace $O(m_i)$ resolution for the rare higher-multiplicity blocks.

Table 9: Eigenvalue multiplicity statistics per dataset. Each row summarizes multiplicities over 500 random graphs from the respective dataset. "Mean #$m{=}k$" is the average per-graph count of *distinct* eigenvalues whose multiplicity class equals k (so a single eigenvalue with multiplicity 3 contributes 1 to the $m{=}3$ column). "% simple" is the fraction of sampled graphs whose eigenvalues are all distinct.

Dataset	**Sampled graphs**	**Mean nodes**	**Mean #$m{=}1$**	**Mean #$m{=}2$**	**Mean #$m{=}3$**	**Mean #$m{\geq}4$**	**% simple spectrum**
ZINC	500	23.0	20.86	0.59	0.20	0.09	38.0%
MolHIV	500	25.7	21.54	0.88	0.30	0.28	32.8%
MolTox21	500	17.9	14.96	0.72	0.21	0.17	38.4%
MolClinTox	500	26.6	23.34	0.59	0.25	0.24	28.6%
MolBBBP	500	23.5	20.93	0.49	0.26	0.16	34.4%

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The three main claims of the abstract and introduction (incompleteness of the WL hierarchy on simple-spectrum graphs, the first complete sign canonicalization PRiSM, and universal approximation under composition with DeepSets/Transformer) correspond exactly to Theorem 1, Theorem 2, and Theorem 3; the empirical claims correspond to the comparisons in Section 6.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The closing section "Conclusion, Limitations and Future Work" states that we do not address completeness for higher eigenvalue multiplicities and that our incompleteness result is for multigraphs, leaving the simple-graph extension as future work. The discontinuity of our canonicalization is discussed in Section D.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Each theorem (Theorem 1, Theorem 2, Theorem 3) states its assumptions in the theorem statement, and full proofs appear in the appendix (Section A, Section B.1, Section D). Lemmas the proofs rely on are stated and proved in the same appendix sections and are cross-referenced from the proof bodies.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section E and the per-experiment appendices (Section E.1, Section E.2, Section E.3) document data splits, hyperparameter configurations, optimizer choices, parameter budgets, and seed sets. Pseudocode for PRiSM is given in Section C, and the supplementary code archive contains runnable environments and scripts for each experimental track.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The supplementary archive contains the code used for the BREC, molecular-GNN, and Transformer-OGB experiments, with separate conda environment files (`env_brec.yml`, `env_transformer.yml`, `env_molecular.yml`) and per-track READMEs. All datasets used (ZINC [28, 19], OGB-Mol [27], BREC [26], Alchemy [29]) are publicly available through their official loaders, cited at first use.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section E specifies the shared experimental protocol; the per-experiment appendices Section E.1, Section E.2, and Section E.3 list the dataset splits, hyperparameter ranges and selected configurations, optimizer settings, training schedule, and the early-stopping criterion used for each backbone-dataset cell.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: All molecular property prediction tables report mean $\pm$ *sample* standard deviation over multiple seeds (closed-form formula, normality assumed) in Table 2, Table 3 and Section E.3; standard deviations are reported in the same cell as the mean.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section E.5 documents the compute resources used for the experiments: hardware (CPU-only canonicalization, training runs on a pool of 8 NVIDIA A40 GPUs (48 GB each), one GPU per run, no cluster beyond this pool), per-run wall-clock ranges read directly from training logs (1.0–4.3 h GatedGCN-ZINC, 4.8–18.0 h PNA-ZINC, 0.4–1.5 h GatedGCN/PNA-MolTox21, 3.8–44.6 h MolPCBA, 3–9 h per seed Transformer per dataset), and total project compute including preliminary / unreported experiments ($\sim$1,500 A40-GPU-hours on the molecular GNN track plus $\sim$500 A40-GPU-hours on the Transformer track, of which the reported tables are a strict subset). The worst-case complexity of PRiSM is in Section C.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

Justification: The work is theoretical and algorithmic, addressing graph expressivity and isomorphism testing on standard public benchmarks; it does not involve human subjects, sensitive data, or applications with foreseeable misuse potential.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: This is foundational research on the expressivity of graph neural networks; we are not aware of a direct societal-impact pathway, positive or negative, that is specific to this work as opposed to the broader graph-learning literature.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper releases an algorithmic preprocessing method (PRiSM) and reports results on standard public benchmarks; no high-risk dataset or model is released.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets (ZINC [28, 19], OGB-Mol [27], BREC [26], Alchemy [29]) and baseline methods (MAP [21], OAP [22], SignNet [20], GatedGCN [48], PNA [49], graph Transformer [25]) are cited at first use. The OGB datasets are released under the MIT license, BREC under the MIT license, ZINC under a research-use license, and Alchemy under CC-BY 4.0; code components imported from prior work retain their original licenses and attribution headers in the supplementary archive.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: PRiSM is documented at three levels of detail: pseudocode in Section C (Algorithms 3–6), formal proof in Section B.1, and runnable code in the supplementary archive with per-track READMEs and conda environment files.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The work does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The work does not involve human subjects research, so IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as an important, original, or non-standard component of the core methods. Any usage was limited to writing, editing, or formatting assistance, which the policy explicitly excludes from the declaration requirement.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.