Weisfeiler and Lehman Go Topological: Message Passing Simplicial Networks Appendix

A. Proofs of SWL Theory Results

We first introduce the required notions and notation. We note that even though these results mainly refer to simplicial complexes, they also apply to graphs because any graph is also a simplicial complex.

Definition 20 (Simplicial Colouring)**.** *A simplicial colouring is a map c that maps a simplicial complex $\mathcal{K}$ and one of its simplices σ to a colour from a fixed colour palette. We denote this colour by $c_\sigma^{\mathcal{K}}$.*

To unload the notation, we will often drop $\mathcal{K}$ from the superscript when the underlying $\mathcal{K}$ is known from the context.

Definition 21. *Let $\mathcal{K}_1, \mathcal{K}_2$ be two simplicial complexes and c a simplicial colouring. We say that $\mathcal{K}_1, \mathcal{K}_2$ are c-similar, denoted by $c^{\mathcal{K}_1} = c^{\mathcal{K}_2}$, if the number of simplices of dimension n in $\mathcal{K}_1$ coloured with a given colour equals the number of simplices of dimension n in $\mathcal{K}_2$ with the same colour. Otherwise, we have $c^{\mathcal{K}_1} \neq c^{\mathcal{K}_2}$.*

Although not explicitly stated in the definition of a colouring, we are only interested in colourings c for which all isomorphic simplicial complex pairs are c-similar or, more formally, if $\mathcal{K}_1$ is isomorphic to $\mathcal{K}_2$, then $c^{\mathcal{K}_1} = c^{\mathcal{K}_2}$.

Definition 22. *A simplicial colouring c refines a simplicial colouring d, denoted by $c \sqsubseteq d$, if for all simplicial complexes $\mathcal{K}_1$ and $\mathcal{K}_2$ and all $\sigma \in \mathcal{K}_1$ and $\tau \in \mathcal{K}_2$ with $\dim(\sigma) = \dim(\tau)$, $c_\sigma^{\mathcal{K}_1} = c_\tau^{\mathcal{K}_2}$ implies $d_\sigma^{\mathcal{K}_1} = d_\tau^{\mathcal{K}_2}$. Additionally, if $d \sqsubseteq c$, we say the two colourings are equivalent and we represent it by $c \equiv d$.*

We now prove a lemma that will be used repeatedly in our proofs in this section.

Lemma 23. *Let $\mathcal{K}_1, \mathcal{K}_2$ be any simplicial complexes with $A \subseteq \mathcal{K}_1$ and $B \subseteq \mathcal{K}_2$, two subsets containing simplices of the same dimension. Consider two simplicial colourings c, d such that $c \sqsubseteq d$. If $\{\!\{d_\sigma^{\mathcal{K}_1} \mid \sigma \in A\}\!\} \neq \{\!\{d_\tau^{\mathcal{K}_2} \mid \tau \in B\}\!\}$, then $\{\!\{c_\sigma^{\mathcal{K}_1} \mid \sigma \in A\}\!\} \neq \{\!\{c_\tau^{\mathcal{K}_2} \mid \tau \in B\}\!\}$.*

Proof. If $\{\!\{d_\sigma^{\mathcal{K}_1} \mid \sigma \in A\}\!\} \neq \{\!\{d_\tau^{\mathcal{K}_2} \mid \tau \in B\}\!\}$, there exists a colour $\mathbb{C}$ that shows up (without loss of generality) more times in the first multi-set than in the second multi-set. Define by

$$A^* = \{\sigma \in A \mid d_\sigma^{\mathcal{K}_1} = \mathbb{C}\}$$
$$B^* = \{\tau \in B \mid d_\tau^{\mathcal{K}_2} = \mathbb{C}\}$$

the sets of those simplices in A and B, respectively, that have been assigned colour $\mathbb{C}$. Note that because $\mathbb{C}$ shows up more times in the first multi-set, $|A^*| > |B^*|$.

Since $c \sqsubseteq d$, if $d_\sigma \neq d_\tau$, then $c_\sigma \neq c_\tau$, for all σ and τ. Therefore, the c colouring of all the simplices in A^* and B^* are different from the colours assigned to the other simplices. More formally,

$$\{\!\{c_\sigma \mid \sigma \in A^* \cup B^*\}\!\} \cap \{\!\{c_\tau \mid \tau \in (A \cup B) \setminus (A^* \cup B^*)\}\!\} = \emptyset. \quad (15)$$

Suppose for the sake of contradiction that

$$\{\!\{c_\sigma^{\mathcal{K}_1} \mid \sigma \in A\}\!\} = \{\!\{c_\tau^{\mathcal{K}_2} \mid \tau \in B\}\!\}.$$

Together with Equation 15, this implies

$$\{\!\{c_\sigma^{\mathcal{K}_1} \mid \sigma \in A^*\}\!\} = \{\!\{c_\tau^{\mathcal{K}_2} \mid \tau \in B^*\}\!\}.$$

Then, $|A^*| = |B^*|$. However, $|A^*| > |B^*|$. □

This result leads to an important corollary.

Corollary 24. *Consider two simplicial colourings c, d such that $c \sqsubseteq d$. If $d^{\mathcal{K}_1} \neq d^{\mathcal{K}_2}$, then $c^{\mathcal{K}_1} \neq c^{\mathcal{K}_2}$*

Proof. This follows immediately by substituting the subsets A, B from the proof above with all the simplices of a given dimension in $\mathcal{K}_1$ and $\mathcal{K}_2$, respectively. □

An equivalent way to think about this corollary is that if c refines d, then it is able to distinguish all the non-isomorphic simplicial complex pairs that d can distinguish (and potentially others). In that sense, we say c is *at least as powerful as* d. This will be our main vehicle to prove the results.

Equipped with this notation and preliminary results, we proceed to prove the results from the main text.

Lemma 25. *SWL with $\mathrm{HASH}\big(c_\sigma^t, c_\mathcal{B}^t(\sigma), c_\downarrow^t(\sigma), c_\uparrow^t(\sigma)\big)$ is as powerful as SWL with the generalised update rule $\mathrm{HASH}\big(c_\sigma^t, c_\mathcal{B}^t(\sigma), c_\mathcal{C}^t(\sigma), c_\downarrow^t(\sigma), c_\uparrow^t(\sigma)\big)$.*

Proof. Let a^t denote the colouring of the general update rule at iteration t and b^t the colouring of the restricted update rule at the same iteration. Then, $a^t \sqsubseteq b^t$ because it considers the additional colours of the co-boundaries $c_\mathcal{C}^t(\sigma)$ in the colour updating rule. We will now prove by induction $b^t \sqsubseteq a^t$, which implies $a^t \equiv b^t$.

The base case trivially holds since all simplices have the same colour at initialisation. Let $\sigma \in \mathcal{K}_1$ and $\tau \in \mathcal{K}_2$ be two simplices of the same dimension from two arbitrary complexes. Suppose $b_\sigma^{t+1} = b_\tau^{t+1}$. Then, we have that the arguments of the hash function are equal. Thus, $b_\sigma^t =$

$b^t_\tau, b^t_\downarrow(\sigma) = b^t_\downarrow(\tau), b^t_\uparrow(\sigma) = b^t_\uparrow(\tau)$, and $b^t_\mathcal{B}(\sigma) = b^t_\mathcal{B}(\tau)$. The aim is to show that these also imply that $b^t_\mathcal{C}(\sigma) = b^t_\mathcal{C}(\tau)$.

Because $b^t_\uparrow(\sigma) = b^t_\uparrow(\tau)$, by substituting their definition we have the following equality of multi-sets.

$$\{\!\{b^t_{\delta_\sigma} \mid (\cdot, b^t_{\delta_\sigma}) \in b^t_\uparrow(\sigma)\}\!\} = \{\!\{b^t_{\delta_\tau} \mid (\cdot, b^t_{\delta_\tau}) \in b^t_\uparrow(\tau)\}\!\}.$$

Because σ and τ have the same dimension n, the colour of each $\delta_\sigma \in \mathcal{C}(\sigma)$ and each $\delta_\tau \in \mathcal{C}(\tau)$ shows up in exactly $n+1$ tuples in $b^t_\uparrow(\sigma)$ and $b^t_\uparrow(\tau)$, respectively. By removing the duplicate colours for each such δ_σ and δ_τ we obtain the desired equality:

$$\{\!\{b^t_{\delta_\sigma} \mid \delta_\sigma \in \mathcal{C}(\sigma)\}\!\} = \{\!\{b^t_{\delta_\tau} \mid \delta_\tau \in \mathcal{C}(\tau)\}\!\}.$$

By the induction hypothesis, we also have $a^t_\sigma = a^t_\tau, a^t_\downarrow(\sigma) = a^t_\downarrow(\tau), a^t_\uparrow(\sigma) = a^t_\uparrow(\tau)$, $a^t_\mathcal{B}(\sigma) = a^t_\mathcal{B}(\tau)$, and $a^t_\mathcal{C}(\sigma) = a^t_\mathcal{C}(\tau)$. Thus, $a^{t+1}_v = a^{t+1}_w$. □

Proof of Theorem 6. Let b^t denote the colouring of CWL using $\text{HASH}\big(b^t_\sigma, b^t_\mathcal{B}(\sigma), b^t_\uparrow(\sigma)\big)$ and a^t the colouring of CWL using the rule $\text{HASH}\big(a^t_\sigma, a^t_\mathcal{B}(\sigma), a^t_\downarrow(\sigma), a^t_\uparrow(\sigma)\big)$ from Lemma 25. As before, it is trivial to show $a^t \sqsubseteq b^t$ because of the additional argument $a^t_\downarrow(\sigma)$ used in the update rule. We now prove that $b^{2t} \sqsubseteq a^t$ by induction. The reason we consider $2t$ is because the information from the lower adjacencies propagates two times slower through the boundary adjacencies.

As before, the base case holds since all the colours are equal at initialisation. Again, consider $\sigma \in \mathcal{K}_1$ and $\tau \in \mathcal{K}_2$, two simplices of the same dimension from two arbitrary complexes. Suppose $b^{2t+2}_\sigma = b^{2t+2}_\tau$. By unwrapping the hash function two steps back in time, we obtain $b^{2t}_\sigma = b^{2t}_\tau, b^{2t}_\mathcal{B}(\sigma) = b^{2t}_\mathcal{B}(\tau), b^{2t}_\uparrow(\sigma) = b^{2t}_\uparrow(\tau)$. The goal is to show that $b^{2t}_\downarrow(\sigma) = b^{2t}_\downarrow(\tau)$ also holds.

Suppose for the sake of contradiction that $b^{2t}_\downarrow(\sigma) \neq b^{2t}_\downarrow(\tau)$. This means that there exists a pair of colours $(\mathbb{C}_0, \mathbb{C}_1)$ that shows up (without loss of generality) more times in $b^{2t}_\downarrow(\sigma)$ than in $b^{2t}_\downarrow(\tau)$. For simplicity, we assume that $b^{2t}_\sigma \neq \mathbb{C}_0 \neq b^{2t}_\tau$, since this edge case can be trivially treated separately.

First, we split the apparitions of $(\mathbb{C}_0, \mathbb{C}_1)$ by the boundary simplices of σ and τ where they appear. Consider the collection of multi-sets A indexed by simplices δ of a fixed dimension:

$$A(\delta) = \{\!\{(b^{2t}_\psi = \mathbb{C}_0, b^{2t}_\delta = \mathbb{C}_1) \mid \psi \in \mathcal{C}(\delta)\}\!\}.$$

We are interested in counting the size of these multi-sets. For this purpose, for each simplex γ, we define a multi-set C_γ:

$$C_\gamma = \{\!\{|A(\delta)| \mid \delta \in \mathcal{B}(\gamma)\}\!\}.$$

Clearly, $C_\sigma \neq C_\tau$ because the sum of the elements in C_σ (the number of tuples $(\mathbb{C}_0, \mathbb{C}_1)$ in $b^{2t}_\downarrow(\sigma)$) is greater than the sum of the elements of C_τ (the number of tuples $(\mathbb{C}_0, \mathbb{C}_1)$ in $b^{2t}_\downarrow(\tau)$). The next proposition, shows this leads to a contradiction with our original assumption that $b^{2t+2}_\sigma = b^{2t+2}_\tau$

Proposition 26. *If $C_\sigma \neq C_\tau$, then $b^{2t+2}_\sigma \neq b^{2t+2}_\tau$.*

Proof. Consider the simplicial colouring $c(\delta) = |A(\delta)|$. We will show that $b^{2t+1} \sqsubseteq c$. Let δ_1, δ_2 be two simplices of equal dimension with $c(\delta_1) \neq c(\delta_2)$. We assume without loss of generality $|A(\delta_1)| > |A(\delta_2)|$. Then $\mathbb{C}_0$ shows up more times in $b^{2t}_\uparrow(\delta_1)$ than in $b^{2t}_\uparrow(\delta_2)$, which implies $b^{2t}_\uparrow(\delta_1) \neq b^{2t}_\uparrow(\delta_2)$. Therefore, $b^{2t+1}_{\delta_1} \neq b^{2t+1}_{\delta_2}$, which proves $b^{2t+1} \sqsubseteq c$.

Applying Lemma 23 for the multi-sets C_σ and C_τ, we obtain two non-equal multi-sets:

$$\{\!\{b^{2t+1}_{\delta_1} \mid \delta_1 \in \mathcal{B}(\sigma)\}\!\} \neq \{\!\{b^{2t+1}_{\delta_2} \mid \delta_2 \in \mathcal{B}(\tau)\}\!\}$$

Which are exactly the multi-sets of colours corresponding to the boundary simplices of σ and τ. So the relation above is equivalent to $b^{2t+1}_\mathcal{B}(\sigma) \neq b^{2t+1}_\mathcal{B}(\tau)$. Finally, this implies that $b^{2t+2}_\sigma \neq b^{2t+2}_\tau$. □

This contradiction proves $b^{2t}_\downarrow(\sigma) = b^{2t}_\downarrow(\tau)$. Finally, applying the induction hypothesis, we have that $a^t_v = a^t_w, a^t_\mathcal{B}(v) = a^t_\mathcal{B}(w), a^t_\uparrow(v) = a^t_\uparrow(w)$ and $a^t_\downarrow(v) = a^t_\downarrow(w)$. Then, $b^{2t} \sqsubseteq a^t$. □

Next, we show a slightly weaker version of Theorem 7.

Lemma 27. *SWL is at least as powerful as WL in distinguishing non-isomorphic simplicial complexes.*

Proof. Let $\mathcal{K}$ be a simplicial complex. Let a^t be the colouring of the vertices of $\mathcal{K}$ at iteration t of WL and b^t the colouring of the same vertices in $\mathcal{K}$ at iteration t of SWL. To prove the lemma, we will show by induction that $b^t \sqsubseteq a^t$.

For the base case, the implication holds at initialisation since all nodes are assigned the same colour. For the induction step, suppose $b^{t+1}_v = b^{t+1}_w$, for two 0-simplices v and w in two arbitrary complexes $\mathcal{K}_1, \mathcal{K}_2$. As vertices are only upper adjacent and have no boundary simplices, $b^t_v = b^t_w$ and $b^t_\uparrow(v) = b^t_\uparrow(w)$. Using the definition of the latter multi-set equality:

$$\{\!\{b^t_z \mid (b^t_z, \cdot) \in b^t_\uparrow(v)\}\!\} = \{\!\{b^t_u \mid (b^t_u, \cdot) \in b^t_\uparrow(w)\}\!\}.$$

Equivalently, this can be rewritten in terms of the upper-neighbours of the vertices as

$$\{\!\{b^t_z \mid z \in \mathcal{N}_\uparrow(v)\}\!\} = \{\!\{b^t_u \mid u \in \mathcal{N}_\uparrow(w)\}\!\}.$$

By the induction hypothesis, $a^t_v = a^t_w$ and $a^t_\uparrow(v) = a^t_\uparrow(w)$. Since these are the arguments the WL hash function uses to compute the colours of v and w at the next step, we obtain $a^{t+1}(v) = a^{t+1}(w)$. □

Informally, this proof shows that the information coming from the higher-dimensional simplices of the complex will refine the colouring of the vertices. This means that SWL will be able to distinguish just through its vertex-level colour histogram at least the same set of simplicial complexes (and graphs) that WL can distinguish. However, this proof ignores the histograms of the higher-levels and these can indeed be used to show that SWL is strictly more powerful than WL when using a clique complex lifting. This is done in Theorem 7.

Proof of Theorem 7. Based on Lemma 27, it is sufficient to present a pair of graphs that cannot be distinguished by WL, but whose clique complexes can be distinguished by SWL. Such a pair is included in Figure 2. While WL produces the same colouring for both graphs, one clique complex contains two triangles, while the other has no triangles. □

Proof of Lemma 9. Let c^t and h^t be the colouring at iteration t of SWL and the t-th layer of an MPSN, respectively. We consider an MPSN model with L layers. For $t > L$, we assume $h^t = h^L$. We will show by induction that c^t refines the colouring of h^t. For this proof, it is convenient to use the most general version of SWL, containing the complete set of adjacencies.

The base case trivially holds. For the induction step, suppose we have two simplicies σ and τ such that $c^{t+1}(\sigma) = c^{t+1}(\tau)$. Because the SWL colouring is an injective mapping, the arguments to the HASH must also be equal. This means that $c^t_\sigma = c^t_\tau$ and the multi-sets of colours formed by their neighbours are identical: $c^t_\downarrow(\sigma) = c^t_\downarrow(\tau), c^t_\uparrow(\sigma) = c^t_\uparrow(\tau), c^t_\mathcal{B}(\sigma) = c^t_\mathcal{B}(\tau), c^t_\mathcal{C}(\sigma) = c^t_\mathcal{C}(\tau)$. By the induction hypothesis, these multi-sets will also be equal under the colouring h^t. Enumerating all, $h^t(\sigma) = h^t(\tau), h^t_\downarrow(\sigma) = h^t_\downarrow(\tau), h^t_\uparrow(\sigma) = h^t_\uparrow(\tau), h^t_\mathcal{B}(\sigma) = h^t_\mathcal{B}(\tau), h^t_\mathcal{C}(\sigma) = h^t_\mathcal{C}(\tau)$. Because the exact same multi-sets are supplied as input to the message, aggregate and update functions, their output will also be the same for σ and τ. Thus, $h^{t+1}_\sigma = h^{t+1}_\tau$. □

Proof of Theorem 10. By Lemma 9, we only need to show that for an MPSN model satisfying the conditions in the theorem, we have that $h^t \sqsubseteq c^t$.

The base case can be proved by definition. For the step case, given that the update, aggregate and message functions are injective, their composition is also injective. Therefore, for any two simplicies σ, τ with $h^{t+1}_\sigma = h^{t+1}_\tau$, the multi-sets of colours in their neighbourhoods are also the same. As in our previous proofs, by applying the induction hypothesis, the inputs to the SWL HASH function at iteration t for σ and τ are also equal and $c^{t+1}_\sigma = c^{t+1}_\tau$. It follows $h^t \sqsubseteq c^t$, $c^t \sqsubseteq h^t$ (Lemma 9) and, consequently, $c^t \equiv h^t$. □

A.1. Higher-Order WL and Strongly Regular Graphs

Higher-order variants of the standards WL procedure operate on node tuples rather than single nodes and iteratively apply color refinement steps thereon.

k-WL The k-WL is one such higher-order variants. It specifically operates on node k-tuples by refining their colors based on the generalized notion of j-neighborhood. The j-neighborhood ($j \in \{1, \ldots, k\}$) for node k-tuple $\mathbf{v} = (v_1, v_2, \ldots, v_k)$ is defined as $\mathcal{N}_j(\mathbf{v}) = \{(v_1, \ldots, v_{j-1}, w, v_{j+1}, \ldots, v_k) | w \in \mathcal{V}_G\}$. The algorithm first initialises node tuples based on their isomorphism type: two k-tuples $\mathbf{v}^a = (v^a_1, v^a_2, \ldots, v^a_k)$, $\mathbf{v}^b = (v^b_1, v^b_2, \ldots, v^b_k)$ have the same isomorphism type (and are thus assigned the same initial colour $c_{\mathbf{v}^a} = c_{\mathbf{v}^b}$) iff (i) $\forall\, i, j \in \{1, \ldots, k\}, v^a_i = v^a_j \Leftrightarrow v^b_i = v^b_j$, (ii) $\forall\, i, j \in \{1, \ldots, k\}, v^a_i \sim v^a_j \Leftrightarrow v^b_i \sim v^b_j$, where $\sim$ indicates adjacency. Given this initial colouring, the procedure iteratively applies the following color refinement step

$$c^{t+1}_\mathbf{v} = \text{HASH}\Big(c^t_\mathbf{v}, M^t(\mathbf{v})\Big), \tag{16}$$

$$M^t(\mathbf{v}) = \big(\{\!\{c^t_\mathbf{u} | \mathbf{u} \in \mathcal{N}_j(\mathbf{v})\}\!\} \big| j = 1, 2, \ldots, k\big) \tag{17}$$

until the colouring does not change further. The k-WL procedure can be employed to *test* the isomorphism between graphs in the same way as the standard WL one is. For any $k \geq 2$, it is known that $(k+1)$-WL test is strictly stronger than k-WL one, i.e. there exist exemplary pairs of non-isomorphic graphs that k-WL cannot distinguish while $(k+1)$-WL can, but not vice-versa. Local variants of k-WL have recently been introduced in Morris et al. (2020b).

k-FWL The k-Folklore WL procedure (k-FWL) is another higher-order variant of the standard WL. Similarly to k-WL, it operates by refining the colors of node k-tuples, initialised based on their isomorphism type. However, it employs a different notion of neighborhood and refinement step. The Folklore j-neighborhood for node k-tuple $\mathbf{v}$ is defined as $\mathcal{N}^F_j(\mathbf{v}) = \big((j, v_2, \ldots, v_k), (v_1, j, \ldots, v_k), \ldots, (v_1, \ldots, v_{k-1}, j)\big)$, with $j \in \mathcal{V}_G$. The algorithm iteratively applies the steps

$$c^{t+1}_\mathbf{v} = \text{HASH}\Big(c^t_\mathbf{v}, M^{F,t}(\mathbf{v})\Big), \tag{18}$$

$$M^{F,t}(\mathbf{v}) = \{\!\{\big(c^t_\mathbf{u} | \mathbf{u} \in \mathcal{N}^F_j(\mathbf{v})\big) \big| j \in \mathcal{V}_G\}\!\} \tag{19}$$

until the colouring does not change anymore. It is known that k-FWL is equivalent to $(k+1)$-WL for $k \geq 2$.

Strongly Regular Graphs A Strongly Regular graph in the family *SR(n,d,λ,μ)* is a regular graph with n nodes and degree d, for which every two adjacent nodes always have λ mutual neighbours and every two non-adjacent nodes always have μ mutual neighbours. This class of graphs is of particular interest due to the following lemma.

Lemma 28. *No pair of Strongly Regular graphs in family* SR(n,d,λ,μ) *can be distinguished by the* 2*-FWL test.*

Proof. Let us denote by $\mathcal{V}^2_G$ the set of all node 2-tuples in graph G. We note that three isomorphism types are induced by considering node 2-tuples:

(1) *node type*: $\mathbf{v} = (v_1, v_1)$

(2) *edge type*: $\mathbf{v} = (v_1, v_2)$ with $v_1 \sim v_2$ (the two nodes are adjacent in the original graph)

(3) *non-edge type*: $\mathbf{v} = (v_1, v_2)$ with $v_1 \not\sim v_2$ (the two nodes are *not* adjacent in the original graph).

These three isomorphism types partition the tuple set $\mathcal{V}^2_G$ into the three subsets $\mathcal{V}^2_{G(1)}, \mathcal{V}^2_{G(2)}, \mathcal{V}^2_{G(3)}$ such that any tuple $\mathbf{v} \in \mathcal{V}^2_{G(i)}$ is of isomorphism type i. We write $\mathbf{v}^{(i)}$ to indicate $\mathbf{v} \in \mathcal{V}^2_{G(i)}$ for simplicity.

At initialisation, the 2-FWL algorithm assigns a colour to each tuple $\mathbf{v} \in \mathcal{V}^2_G$ based on its isomorphism type, that is $\forall \mathbf{v} \in \mathcal{V}^2_{G(i)}, c_{\mathbf{v}} = c^0_i$. The colouring is therefore constant within partitions. Then, we notice that the colouring is kept constant within partitions through the application of the refinement steps described by Equation 18. In other words, the 2-FWL procedure cannot produce a colour partitioning of the set of node 2-tuples that is finer than the one at initialisation. This is shown by induction on the step t of color refinement.

The base case evidently holds for $t = 0$ since all tuples in the same partition are assigned the same colour by the 2-FWL initialisation procedure.

For the induction step we assume that the colouring is constant within each partition at t and show that it maintains constant at $t+1$, that is, after the application of one colour refinement step. This is proved by showing that all node tuples within the same partition have their colour refined identically. We will show this for each of the three partitions separately, leveraging on the induction hypothesis and the properties of Strongly Regular graphs.

A node tuple $\mathbf{v} = (v_1, v_1) \in \mathcal{V}^2_{G(1)}$ has $\mathcal{N}^F_j(\mathbf{v}) = \big((j, v_1), (v_1, j)\big), j \in \mathcal{V}_G$. Therefore, *any* $\mathbf{v} \in \mathcal{V}^2_{G(1)}$ has exactly:

- ($j = v_1$) 1 neighborhood of the form $\big(\mathbf{w}^{(1)}, \mathbf{w}^{(1)}\big)$, associated with color tuple (c^t_1, c^t_1);
- ($j \sim v_1$) d neighborhoods of the form $\big(\mathbf{w}^{(2)}, \mathbf{u}^{(2)}\big)$, associated with color tuple (c^t_2, c^t_2);
- ($j \not\sim v_1$) $n - d - 1$ neighborhoods of the form $\big(\mathbf{w}^{(3)}, \mathbf{u}^{(3)}\big)$, associated with color tuple (c^t_3, c^t_3).

For *any* $\mathbf{v} \in \mathcal{V}^2_{G(1)}$ we thus have

$$c^{t+1}_{\mathbf{v}} = \text{HASH}\Big(c^t_1, M^{F,t}(\mathbf{v})\Big)$$
$$M^{F,t}(\mathbf{v}) = \{\!\{\underbrace{(c^t_1, c^t_1)}_{1 \text{ time}}, \underbrace{(c^t_2, c^t_2)}_{d \text{ times}}, \underbrace{(c^t_3, c^t_3)}_{n-d-1 \text{ times}}\}\!\}.$$

A node tuple $\mathbf{v} = (v_1, v_2) \in \mathcal{V}^2_{G(2)}$ has $\mathcal{N}^F_j(\mathbf{v}) = \big((j, v_2), (v_1, j)\big), j \in \mathcal{V}_G$. Therefore, *any* $\mathbf{v} \in \mathcal{V}^2_{G(2)}$ has exactly:

- ($j = v_2$) 1 neighborhood of the form $\big(\mathbf{w}^{(1)}, \mathbf{u}^{(2)}\big)$, associated with color tuple (c^t_1, c^t_2);
- ($j = v_1$) 1 neighborhood of the form $\big(\mathbf{w}^{(2)}, \mathbf{u}^{(1)}\big)$, associated with color tuple (c^t_2, c^t_1);
- ($j \sim v_2, j \sim v_1$) λ neighborhoods of the form $\big(\mathbf{w}^{(2)}, \mathbf{u}^{(2)}\big)$, associated with color tuple (c^t_2, c^t_2);
- ($j \sim v_2, j \not\sim v_1$) $d - \lambda$ neighborhoods of the form $\big(\mathbf{w}^{(2)}, \mathbf{u}^{(3)}\big)$, associated with color tuple (c^t_2, c^t_3);
- ($j \not\sim v_2, j \sim v_1$) $d - \lambda$ neighborhoods of the form $\big(\mathbf{w}^{(3)}, \mathbf{u}^{(2)}\big)$, associated with color tuple (c^t_3, c^t_2);
- ($j \not\sim v_2, j \not\sim v_1$) $k = n - 2 - 2d + \lambda$ neighborhoods of the form $\big(\mathbf{w}^{(3)}, \mathbf{u}^{(3)}\big)$, associated with color tuple (c^t_3, c^t_3).

For *any* $\mathbf{v} \in \mathcal{V}^2_{G(2)}$ we have

$$c^{t+1}_{\mathbf{v}} = \text{HASH}\Big(c^t_2, M^{F,t}(\mathbf{v})\Big)$$
$$M^{F,t}(\mathbf{v}) = \{\!\{\underbrace{(c^t_1, c^t_2)}_{1 \text{ time}}, \underbrace{(c^t_2, c^t_1)}_{1 \text{ time}}, \underbrace{(c^t_2, c^t_2)}_{\lambda \text{ times}}, \underbrace{(c^t_2, c^t_3)}_{d-\lambda \text{ times}}, \underbrace{(c^t_3, c^t_2)}_{d-\lambda \text{ times}}, \underbrace{(c^t_3, c^t_3)}_{k \text{ times}}\}\!\}.$$

A node tuple $\mathbf{v} = (v_1, v_2) \in \mathcal{V}^2_{G(3)}$ has $\mathcal{N}^F_j(\mathbf{v}) = \big((j, v_2), (v_1, j)\big), j \in \mathcal{V}_G$. Therefore, *any* $\mathbf{v} \in \mathcal{V}^2_{G(3)}$ has exactly:

- ($j = v_2$) 1 neighborhood of the form $\big(\mathbf{w}^{(1)}, \mathbf{u}^{(3)}\big)$, associated with color tuple (c^t_1, c^t_3);
- ($j = v_1$) 1 neighborhood of the form $\big(\mathbf{w}^{(3)}, \mathbf{u}^{(1)}\big)$, associated with color tuple (c^t_3, c^t_1);
- ($j \sim v_2, j \sim v_1$) μ neighborhoods of the form $\big(\mathbf{w}^{(2)}, \mathbf{u}^{(2)}\big)$, associated with color tuple (c^t_2, c^t_2);

- $(j \sim v_2, j \not\sim v_1)$ $d-\mu$ neighborhoods of the form $\big(\mathbf{w}^{(2)}, \mathbf{u}^{(3)}\big)$, associated with color tuple (c^t_2, c^t_3);
- $(j \not\sim v_2, j \sim v_1)$ $d-\mu$ neighborhoods of the form $\big(\mathbf{w}^{(3)}, \mathbf{u}^{(2)}\big)$, associated with color tuple (c^t_3, c^t_2);
- $(j \not\sim v_2, j \not\sim v_1)$ $k = n-2-2d+\mu$ neighborhoods of the form $\big(\mathbf{w}^{(3)}, \mathbf{u}^{(3)}\big)$, associated with color tuple (c^t_3, c^t_3).

For *any* $\mathbf{v} \in \mathcal{V}^2_{G(3)}$, we then obtain

$$\begin{aligned} c^{t+1}_{\mathbf{v}} &= \mathrm{HASH}\Big(c^t_3, M^{F,t}(\mathbf{v})\Big) \\ M^{F,t}(\mathbf{v}) &= \{\!\!\{\underbrace{(c^t_1, c^t_3)}_{\text{1 time}}, \underbrace{(c^t_3, c^t_1)}_{\text{1 time}}, \underbrace{(c^t_2, c^t_2)}_{\mu \text{ times}}, \\ &\quad \underbrace{(c^t_2, c^t_3)}_{d-\mu \text{ times}}, \underbrace{(c^t_3, c^t_2)}_{d-\mu \text{ times}}, \underbrace{(c^t_3, c^t_3)}_{k \text{ times}}\}\!\!\}. \end{aligned}$$

This proves the induction and confirms that all tuples in the same partition have the same colour at any colour refinement time step t.

If the colouring is constant within partitions at any 2-FWL step, then the colour histogram associated with a graph at step t purely depends on the cardinality of each of the three. We notice that, for *any* $G \in$ *SR(n,d,λ,μ)*, they are completely determined by the first two parameters with

- $|\mathcal{V}^2_{G(1)}| = n$
- $|\mathcal{V}^2_{G(2)}| = nd$
- $|\mathcal{V}^2_{G(3)}| = |\mathcal{V}^2_{G}| - (n + nd)$.

Given the above, any two graphs $G_1, G_2 \in$ *SR(n,d,λ,μ)* are associated with the same colour histograms at any step of the 2-FWL procedure and, therefore, cannot possibly deemed non-isomorphic by the last. □

We leverage on Lemma 28 to prove Theorem 8.

Proof of Theorem 8. In virtue of Lemma 28 and the fact that 2-FWL is as powerful as 3-WL, Theorem 8 is proved by exhibiting a pair of Strongly Regular graphs in the same family that are distinguished by the SWL test. This pair is given by the two graphs in Figure 3: Rook's 4x4 graph (G_1) and the Shrikhande graph (G_2), (the only) members of the *SR(16,6,2,2)* family of Strongly Regular graphs. The SWL test which considers their clique 3-complexes distinguish them due to the fact that, differently from G_1, G_2 possesses no 4-cliques, thus its associated complex has no 3-simplices. □

B. Proofs of Linear Regions Results

Background on Hyperplane Arrangements A function $f\colon \mathbb{R}^N \to \mathbb{R}^M$ is a *piecewise linear function* if its graph $\{(x, f(x))\colon x \in \mathbb{R}^N\} \subseteq \mathbb{R}^N \times \mathbb{R}^M$ consists of a finite number of polyhedral pieces. Projecting these polyhedra back onto $\mathbb{R}^N$ by $(x, y) \mapsto x$ defines a polyhedral subdivision of $\mathbb{R}^N$. The *linear regions* of the function are the N-dimensional pieces in this subdivision. These are the (inclusion maximal) connected regions of the input space where the function is affine linear.

Let $\psi\colon \mathbb{R} \to \mathbb{R}$; $s \mapsto \max\{0, s\}$ be the linear rectification. A ReLU with N inputs defines a function $y\colon x \mapsto \psi(w^\top x)$, which for any fixed value of the weight vector $w \in \mathbb{R}^N$, $w \neq 0$, has gradient with respect to the input vector $x \in \mathbb{R}^N$ equal to 0 on the open halfspace $\{x\colon w^\top x < 0\}$ and equal to w on the open halfspace $\{x\colon w^\top x > 0\}$. Hence a ReLU defines a piecewise linear function with two linear regions. A layer of ReLUs $\psi(w_i^\top x)$, $i = 1, \ldots, M$ has linear regions given by the intersection of linear regions of the individual ReLUs. The number of linear regions of the function represented by the layer is equal to the number of connected components that are left behind once we remove $\cup_{i=1}^M A_i$ from $\mathbb{R}^N$, where $A_i = \{x \in \mathbb{R}^N\colon w_i^\top x = 0\}$ is the hyperplane dividing the two linear regions of the ith ReLU. Hence the linear regions of a layer of ReLUs can be described in terms of a *hyperplane arrangement*, i.e. a collection $\mathcal{A} = \{A_i\colon i = 1, \ldots, M\}$ of hyperplanes.

An arrangement of hyperplanes in $\mathbb{R}^N$ is *in general position* if the intersection of any k hyperplanes in the arrangement has the expected co-dimension, k. We will focus on *central* arrangements, where each hyperplane contains the origin. A central arrangement is in general position when the normal vectors $w_{i_1}, \ldots, w_{i_k}$ of any $k \leq N$ of the hyperplanes are linearly independent. The following well-known result from the theory of hyperplane arrangements will be particularly important in our discussion.

Theorem 29. *Let $\mathcal{A}$ be a central arrangement of M hyperplanes in $\mathbb{R}^N$ in general position. Then the number of regions of the arrangement, denoted $r(\mathcal{A})$, is equal to $2\sum_{j=0}^{N-1}\binom{M-1}{j}$. This is also the maximum number of regions of any central arrangement of M hyperplanes in $\mathbb{R}^N$.*

This result can be derived from Zaslavsky's theorem (Zaslavsky, 1975), which expresses the number of regions of an arbitrary arrangement, not necessarily in general position, in terms of properties of a partially ordered set, namely the collection of intersections of the hyperplanes in the arrangement partially ordered by reverse inclusion.

We will focus on central arrangements, but we point out that similar results to Theorem 29 can be derived for the case of non-central hyperplane arrangements. An non-central arrangement of (affine) hyperplanes in $\mathbb{R}^N$ is in general position when any $k \leq N$ of the hyperplanes intersect in a set of dimension $N-k$, and any $k > N$ of the hyperplanes have an empty intersection. For such an arrangement, the number of regions is $\sum_{j=0}^{N} \binom{M}{j}$.

The main challenges in computing the number of regions defined hyperplane arrangements happen when the hyperplanes satisfy some type of constraints and are not in general position. The type of layers that we discuss in the following correspond to central arrangements subject to certain constraints, namely that the normal vectors are the rows of a matrix with a particular block Kronecker product structure.

B.1. GNNs

Proof of Theorem 15. For simplicity, we write $Y = \mathcal{H}(A,X)^T \in \mathbb{R}^{d \times S_0}$ and $V = W^T \in \mathbb{R}^{m \times d}$. We denote $Y_{:j}$ the $j$the column of Y, and $V_{i:}$ the ith row of V. The GNN layer defines hyperplanes, for $i = 1, \dots, m,\ j = 1, \dots, S_0$,

$$A_{ij} := \{Y \in \mathbb{R}^{d \times S_0} : V_{i:} Y_{:j} = 0\}.$$

The arrangement $\mathcal{A} = \{A_{ij} \colon i = 1, \dots, m, j = 1, \dots, S_0\}$ is a direct sum of the arrangements $\mathcal{A}_j = \{A_{ij} \colon i = 1, \dots, m\}, j = 1, \dots, S_0$. It can be shown (see Zaslavsky, 1975) that this implies $r(\mathcal{A}) = \prod_{j=1}^{S_0} r(\mathcal{A}_j)$.

Each $\mathcal{A}_j$ is an arrangement of m hyperplanes in $\mathbb{R}^N$, $N = dS_0$, whose normals span a subspace of dimension at most d, irrespective of S_0. Counting the number of regions defined by $\mathcal{A}_j$ is equivalent to counting the number of regions defined by its *essentialization* $\mathrm{ess}(\mathcal{A}_j)$, which is the arrangement that the hyperplanes define on the span of their normal vectors. We can regard $\mathrm{ess}(\mathcal{A}_j)$ as a (central) arrangement of m hyperplanes in $\mathbb{R}^d$ with normals $V_{i:} \in \mathbb{R}^d$, $i = 1, \dots, m$. For generic choices of the weight matrix $W^\top = V$, this is a central arrangement in general position. Hence, by Theorem 29, $r(\mathcal{A}_j) = r(\mathrm{ess}(\mathcal{A}_j)) = 2\sum_{i=0}^{d-1} \binom{m-1}{i}$.

For the number of regions of the entire arrangement $\mathcal{A}$, corresponding to the number of linear regions of the function expressed by the layer, we obtain $R_{\mathrm{GNN}} = r(\mathcal{A}) = \prod_{j=1}^{S_0} r(\mathcal{A}_j) = \left(2\sum_{i=0}^{d-1}\binom{m-1}{i}\right)^{S_0}$. This then concludes the proof. □

B.2. SCNNs

Proof of Theorem 16. By the definition of the SCNN layer, for each dimension n, each of the S_n n-simplices in the simplicial complex has d_n input features. Similar to the proof of Theorem 15, the arrangement for the n-dimensional simplices corresponds to a direct sum of S_n arrangements, each of m_n hyperplanes in $\mathbb{R}^{d_n}$. Hence the number of linear regions for this part of the complex is

$$\left(2\sum_{i=0}^{d_n-1}\binom{m_n-1}{i}\right)^{S_n}. \tag{20}$$

Now for the entire layer, the arrangements for the different n are also combined as a direct sum, so that their number of regions multiply. We have n ranging from dimension 0 to p, so that

$$\prod_{n=0}^{p}\left(2\sum_{i=0}^{d_n-1}\binom{m_n-1}{i}\right)^{S_n}. \tag{21}$$

This concludes the proof. □

B.3. MPSNs

We can rewrite (11) more generality and more concisely as follows. For each n the output features on $\mathcal{S}_n$ can be written as

$$\begin{aligned} H_n^{\mathrm{out}} &= \psi(M_n H_n + U_n H_{n-1} + O_n H_{n+1}) \\ &= \psi\left([U_n H_{n-1} | M_n H_n | O_n H_{n+1}] \begin{bmatrix} W_{n-1} \\ W_n \\ W_{n+1} \end{bmatrix}\right), \end{aligned} \tag{22}$$

for some fixed matrices $U_n \in \mathbb{R}^{S_n \times S_{n-1}}$, $M_n \in \mathbb{R}^{S_n \times S_n}$ and $O_n \in \mathbb{R}^{S_n \times S_{n+1}}$ depending only on the simplicial complex. To avoid clutter, we omit the superscript "in" of the input feature matrices H_n. Concatenating (22) for all n, we can write the entire MPSN layer as

$$\begin{bmatrix} H_0^{\mathrm{out}} \\ H_1^{\mathrm{out}} \\ H_2^{\mathrm{out}} \\ \vdots \\ H_p^{\mathrm{out}} \end{bmatrix} = \psi\left(\begin{bmatrix} M_0H_0 & O_0H_1 & & & \\ U_1H_0 & M_1H_1 & O_1H_2 & & \\ & U_2H_1 & M_2H_2 & O_2H_3 & \\ & & & \ddots & \\ & & & & \end{bmatrix}\begin{bmatrix} W_0 \\ W_1 \\ W_2 \\ \vdots \\ W_p \end{bmatrix}\right). \tag{23}$$

We will use this representation (or rather 22) in the proof of the first bound in Theorem 19.

It is also useful to write the linear function in standard form. Using Roth's lemma, we can write (22) as

$$\begin{aligned} \mathrm{vec}(H_n^{\mathrm{out}}) &= \psi\Big([W_{n-1}^\top \otimes U_n | W_n^\top \otimes M_n | W_{n+1}^\top \otimes O_n] \\ &\quad \times \mathrm{vec}\big([H_{n-1}|H_n|H_{n+1}]\big)\Big). \end{aligned} \tag{24}$$

Now, concatenating over n yields the expression $\psi(WH)$ from (12) for the entire layer, with the matrix $W \in \mathbb{R}^{M \times N}$ from (13).

Proof of Proposition 18. This result is known in theory of partial orders and hyperplane arrangements. We include a proof which illustrates the evaluation of the characteristic polynomial. If there is K so that $\mathrm{rank}(W_{B:}) =$

$\min\{|B|, K\}$ for all B, then Lemma 17 can be evaluated as

$$\begin{aligned}r(\mathcal{A}) &= \sum_B (-1)^{|B|-\min\{|B|,K\}}\\ &= \sum_{j=0}^{K} \sum_{B\in\binom{\{1,\dots,M\}}{j}} 1 + \sum_{j=K+1}^{M} \sum_{B\in\binom{\{1,\dots,M\}}{j}} (-1)^{j-K}\\ &= \sum_{j=0}^{K}\binom{M}{j} + (-1)^{M-K}\sum_{j=0}^{M-(K+1)}\binom{M}{j}(-1)^j\\ &= \sum_{j=0}^{K}\binom{M}{j} + (-1)\binom{M-1}{K}\\ &= 2\sum_{j=0}^{K-1}\binom{M-1}{j},\end{aligned}$$

which is what was claimed. □

We now proceed with the proof of Theorem 19. We will use the following.

Proposition 30. *If $W, W' \in \mathbb{R}^{M\times N}$ are two matrices with $\operatorname{rank}(W_{B:}) \geq \operatorname{rank}(W'_{B:})$ for all $B \subseteq \{1,\dots,M\}$, then $r(\mathcal{A}) \geq r(\mathcal{A}')$.*

Proof of Proposition 30. Notice that if $W'_{B:}$ is not full rank, then W' solves a polynomial system. More precisely, some minors (determinants of sub-matrices) of $W'_{B:}$ vanish. Hence, increasing the rank corresponds to stepping outside of the solution set to a polynomial system. This can be accomplished by an arbitrarily small perturbation of the matrix. On the other hand, the number of regions of a central arrangement with normals W corresponds to the number of vertices of a polytope which is the convex hull of points parametrized by the entries of W. The number of vertices is a lower semi-continuous function of the considered polytope (see Grünbaum, 2003, Section 5.3), and hence the number of regions is a lower semi-continuous function of the entries of W. This means that for sufficiently small perturbations of the entries, the number of regions of the corresponding hyperplane arrangement does not decrease. □

Further, we will use an inequality for the rank of a block matrix (see, e.g. Matsaglia & Styan, 1974, Theorem 19).

Lemma 31. *Let $A \in \mathbb{R}^{k\times l}$, $B \in \mathbb{R}^{m\times l}$, $C \in \mathbb{R}^{m\times n}$ be matrices. Then* $\operatorname{rank}([\begin{smallmatrix}A & 0\\ B & C\end{smallmatrix}]) \geq \operatorname{rank}(A) + \operatorname{rank}(C)$.

Proof of Theorem 19. The proof of the first upper bound is analogous to Theorem 16. The difference is now we consider also the boundary and co-boundary simplices that interact with an n-simplex in the MPSN. We use the expression (22). The difference compared with Theorem 16 lies in the number of input features for each n, which here results in

$$\prod_{n=0}^{p}\left(2\sum_{i=0}^{d_{n-1}+d_n+d_{n+1}-1}\binom{m-1}{i}\right)^{S_n}, \qquad (25)$$

which is the first upper bound. The second upper bound is the trivial upper bound, which is the maximum possible number of regions of a central arrangement of M hyperplanes in $\mathbb{R}^N$ from Theorem 29.

The lower bound follows from Proposition 30. We verify that the hypothesis of the proposition is satisfied. Note that the matrix W from (13) is lower block triangular, of the form

$$W = \begin{bmatrix} * & * & 0 & 0 & 0\\ * & * & * & 0 & 0\\ * & * & * & * & 0\\ * & * & * & * & *\\ * & * & * & * & * \end{bmatrix}.$$

Applying Lemma 31 recursively, for the network with outputs $H_0^{\text{out}},\dots,H_{p-1}^{\text{out}}$ we find that $\operatorname{rank}(W) \geq \operatorname{rank}([W_0^\top \otimes M_0 | W_1^\top \otimes O_0]) + \sum_{n=1}^{p-1}\operatorname{rank}(W_{n+1}^\top \otimes O_n)$. A similar expression holds for any selection of rows, $W_{B:}$.

On the other hand, the corresponding matrix W' for an SCNN is block diagonal with blocks $W_n^\top \otimes M_n$ and hence $\operatorname{rank}(W') = \sum_{n=0}^{p-1}\operatorname{rank}(W_n^\top \otimes M_n)$. A similar expression holds for any selection of rows, $W'_{B:}$.

Hence, if $d_{n+1} \geq d_n$ and $\operatorname{rank}((O_n)_{C:}) \geq \operatorname{rank}((M_n)_{C:})$ for any subsets C of rows, then, choosing full rank matrices W_n, we have that the overall matrix satisfies $\operatorname{rank}(W_{B:}) \geq \operatorname{rank}(W'_{B:})$ for any subset B of rows. Hence, applying Proposition 30 gives the desired result. □

It is not difficult to obtain case by case improvements of the bounds in Theorem 19 by conducting a more careful analysis of the row independencies in matrix W for specific values of the input feature dimensions $d_0,\dots,d_p$, output feature dimension m, numbers of simplices $S_0,\dots,S_p$, and the structure of the matrices U_n, M_n, O_n, $n = 0,\dots,p$.

MPSN with Populated Higher-Features Finally, we consider the populated higher features for a situation when we are given a simplicial complex but only vertex features are available. This strategy can increase the network complexity, as proved by Proposition 32 below.

Proposition 32 (MPSN with populated higher-features)**.** *Consider an arbitrary simplicial complex and an MPSN layer mapping $\mathbb{R}^{S_0\times d_0} \to \mathbb{R}^{S_0\times m}; H_0^{\text{in}} \mapsto H_0^{\text{out}}$, whereby higher-dimensional input features are populated as linear functions of the input vertex features. Consider further the corresponding SCNN layer which computes $H_0^{\text{out}} = \psi(L_0 H_0^{\text{in}} W_0)$. Then, $R_{\text{MPSN}} \geq R_{\text{SCNN}}$. Furthermore,*

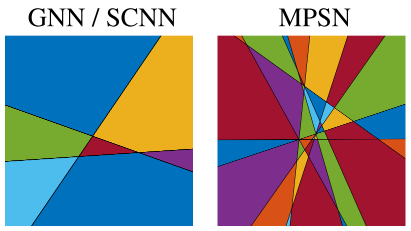

Figure 7. Shown is a 2D slice of the input vertex feature space of SCNN and MPSN layers with $S_0 = S_1 = 3$, $d_0 = d_1 = 1$, $m = 3$, computing output vertex features, with input edge features set as a random linear function of the input vertex features.

for certain simplicial complexes and feature dimensions $d_0, \dots, d_p, m$, the inequality is strict.

The regions for the two cases are illustrated in Figure 7. It shows MPSN (Right) has more regions than GNN/SCNN (Left) and has a higher complexity for populated case.

Proof of Proposition 32. Focusing on the $\mathcal{S}_0$ output features, the relevant matrices are $[W_0^\top \otimes M_0]$ for the simplicial network and $[W_0^\top \otimes M_0 | W_1^\top \otimes O_0]C$ for the MPSN with populated higher-dimensional features. Both matrices have format $mS_0 \times dS_0$ and rank $\min\{m, d_0\}\,\mathrm{rank}(M_0)$. However, subsets of rows have different ranks in both cases. For illustration, if $d = 1$ and l is the smallest number of nonzero entries of any row in M_0, then $[W_0^\top M_0]$ has an m row submatrix of rank $\min\{m, l\}$. In contrast, an m row submatrix of the augmented matrix will have rank $\min\{m, l + l'\}$, where l' is the smallest number of nonzero entries of a row in O_0. □

C. Relationship to Convolutions

Background on Hodge Laplacian The simplicial convolutions described in this section rely on the Hodge Laplacians of the simplicial complex. Such a Laplacian operator L_p is associated with each dimension p of the complex. The operator L_p has a special structure that depends on the boundary operators of the corresponding dimensions B_p and B_{p+1}. These matrices are the discrete equivalent of the boundary operators encountered in algebraic topology (see Schaub et al. (2020) for more details). They essentially describe things such as the fact that boundary of a 2-simplex is formed by its edges with some relative orientation (positive or negative). The Laplacian can be written as a function of these boundary matrices as $L_p = B_p^\top B_p + B_{p+1} B_{p+1}^\top$. We denote the first term by $L_p^\downarrow$ and the second by $L_p^\uparrow$, because they encode up and down adjacencies between the p-simplicies of the complex and the relative orientation between them. Additionally, let us recall the important relation $B_p B_{p+1} = \mathbf{0}$ (i.e. the zero map). This equation, which is fundamental in homology theory and differential geometry, formally specifies that the boundary of a simplex has no boundary (e.g. the boundary of a 2-simplex has no boundary because it forms a loop and, therefore, it has no endpoints).

We emphasise that the boundary matrices and the Hodge Laplacian depend on an arbitrary choice of orientation for the simplicial complex. Although these linear operators are orientation-equivariant, more general convolutional layers based on these operators are not guaranteed to be orientation-equivariant. When originally introduced, the orientation equivariance properties of the convolutional operators from Ebli et al. (2020) and Bunch et al. (2020) were not considered. Therefore, when the orientation of the input complex is changed, these models could produce completely different outputs. This is also emphasised by the MPSN ReLU model in Section 6. In Appendix D we analyse these aspects in more detail.

Simplicial Neural Networks Ebli et al. (2020) presented Simplicial Neural Networks (SNNs), neural network models extending convolutional layers to attributed simplicial complexes. The theoretical construction closely resembles the one by Defferrard et al. (2016), where the standard graph Laplacian is simply replaced by the more general Hodge p-Laplacian L_p, i.e. the generalization of the Laplacian operator to simplices of order p. The p-th order SNN convolutional layer is defined as

$$\mathcal{F}_p^{-1}(\phi_W) *_p c = \psi\left(\sum_{r=0}^{R} W_r L_p^r c\right), \tag{26}$$

where ϕ_W is a convolutional filter with learnable parameters W, $c \in C^p(K)$ is a p-cochain on input simplicial complex K (i.e. a real valued function over the set of p-simplices in K) and ψ accounts for the application of bias and non-linearity. In particular, the convolutional filter ϕ_W is parameterized as an R-degree polynomial of the Hodge p-Laplacian L_p. By imposing a small degree R, it is possible to guarantee spatial filter localization similarly as in graphs.

This proposed convolutional layer can easily be be rewritten in terms of our message passing scheme. Let us first conveniently introduce the following Lemma:

Lemma 33. *The r-th power of the Hodge p-Laplacian, L_p^r, is equivalent to the sum of the r-th powers of its constituent upper and lower components, that is: $L_p^r = (L_p^\downarrow + L_p^\uparrow)^r = (L_p^\downarrow)^r + (L_p^\uparrow)^r$.*

Proof. We prove the lemma by induction on the power exponent r. As the base case, we consider exponent $r = 1$, for which the equivalence clearly holds as $L_p^1 = (L_p^\downarrow + L_p^\uparrow)^1 = L_p^\downarrow + L_p^\uparrow = (L_p^\downarrow)^1 + (L_p^\uparrow)^1$.

For the induction step, we assume that $L_p^{r-1} = (L_p^\downarrow)^{r-1} +$

$(L_p^{\uparrow})^{r-1}$ holds true and prove that $L_p^r = (L_p^{\downarrow})^r + (L_p^{\uparrow})^r$. L_p^r is defined as:

$$L_p^r = L_p L_p^{r-1} = (L_p^{\downarrow} + L_p^{\uparrow})(L_p^{\downarrow} + L_p^{\uparrow})^{r-1}$$

By the induction hypothesis:

$$\begin{aligned}
&(L_p^{\downarrow} + L_p^{\uparrow})(L_p^{\downarrow} + L_p^{\uparrow})^{r-1} = \\
&\quad = (L_p^{\downarrow} + L_p^{\uparrow})\big((L_p^{\downarrow})^{r-1} + (L_p^{\uparrow})^{r-1}\big) = \\
&\quad = L_p^{\downarrow}(L_p^{\downarrow})^{r-1} + L_p^{\downarrow}(L_p^{\uparrow})^{r-1} \\
&\qquad + L_p^{\uparrow}(L_p^{\downarrow})^{r-1} + L_p^{\uparrow}(L_p^{\uparrow})^{r-1} = \\
&\quad = (L_p^{\downarrow})^r + L_p^{\uparrow}(L_p^{\downarrow})^{r-1} + L_p^{\downarrow}(L_p^{\uparrow})^{r-1} + (L_p^{\uparrow})^r
\end{aligned}$$

The second term $L_p^{\uparrow}(L_p^{\downarrow})^{r-1}$ can be rewritten as $B_{p+1}B_{p+1}^\top(B_p^\top B_p)^{r-1} = B_{p+1}B_{p+1}^\top \underbrace{B_p^\top B_p}_{(r-1)\text{ times}} = \mathbf{0}$, since $B_{p+1}^\top B_p^\top = (B_p B_{p+1})^\top$, which equals $\mathbf{0}^\top$ by definition.

Similarly, the third term $L_p^{\downarrow}(L_p^{\uparrow})^{r-1}$ can be rewritten as $B_p^\top B_p(B_{p+1}B_{p+1}^\top)^{r-1} = B_p^\top B_p \underbrace{B_{p+1}B_{p+1}^\top}_{(r-1)\text{ times}} = \mathbf{0}$, since, again, $B_p B_{p+1} = \mathbf{0}$. □

We now proceed to prove Theorem 12 that our MPSN can be reduced to SCNN of Ebli et al. (2020) or Bunch et al. (2020). We split the proofs in two individuals.

Proof of Theorem 12 in Reference to Ebli et al. (2020). We first rephrase Equation (26) for a generic layer and multi-dimensional input and output p-simplicial representations:

$$\begin{aligned}
H^{t+1} &= \psi\Big(\sum_{r=0}^{R} L_p^r H^t W_r^{t+1}\Big) \\
&= \Big(H^t W_0^{t+1} + \sum_{r=1}^{R} L_p^r H^t W_r^{t+1}\Big).
\end{aligned}$$

The convolutional operation for p-simplex σ can be rewritten as

$$\begin{aligned}
h_\sigma^{t+1} &= \psi\Big(h_\sigma^t W_0^{t+1} + \sum_{r=1}^{R} (L_p^r)_\sigma H^t W_r^{t+1}\Big) \\
&= \psi\Big(h_\sigma^t W_0^{t+1} + \sum_{r=1}^{R}\sum_{\tau\in\mathcal{S}_p} (L_p^r)_{\sigma,\tau} h_\tau^t W_r^{t+1}\Big),
\end{aligned}$$

where h_σ^{t+1} denotes the σ-th row of matrix H^{t+1}, $(L_p^r)_\sigma$ denotes the σ-th row of operator L_p^r and $(L_p^r)_{\sigma,\tau}$ its entry at position σ, τ. We now leverage on Lemma 33 to rewrite the convolution operation on simplex σ as

$$\begin{aligned}
h_\sigma^{t+1} &= \psi\Big(\sum_{r=1}^{R}\sum_{\tau\in\mathcal{S}_p} \big(((L_p^{\downarrow})^r)_{\sigma,\tau} + ((L_p^{\uparrow})^r)_{\sigma,\tau}\big) h_\tau^t W_r^{t+1} \\
&\qquad + h_\sigma^t W_0^{t+1}\Big) \\
&= \psi\Big(\sum_{\tau\in\mathcal{S}_p}\sum_{r=1}^{R} ((L_p^{\downarrow})^r)_{\sigma,\tau} h_\tau^t W_r^{t+1} \\
&\qquad + \sum_{\tau\in\mathcal{S}_p}\sum_{r=1}^{R} ((L_p^{\uparrow})^r)_{\sigma,\tau} h_\tau^t W_r^{t+1} + h_\sigma^t W_0^{t+1}\Big).
\end{aligned}$$

Considering that matrices $L_p^{\downarrow}$ and $L_p^{\uparrow}$ only convey the notions of, respectively, lower and upper simplex adjacency, the equation above is easily interpreted in terms of our message passing scheme by setting

$$\begin{aligned}
M_{\uparrow}^{t+1}\big(h_\sigma^t, h_\tau^t, h_{\sigma\cup\tau}^t\big) &= \sum_{r=1}^{R} ((L_p^{\uparrow})^r)_{\sigma,\tau} h_\tau^t W_r^{t+1} \\
M_{\downarrow}^{t+1}\big(h_\sigma^t, h_\tau^t, h_{\sigma\cap\tau}^t\big) &= \sum_{r=1}^{R} ((L_p^{\downarrow})^r)_{\sigma,\tau} h_\tau^t W_r^{t+1} \\
U^{t+1}\Big(h_\sigma^t, \{m_i^t(\sigma)\}_{i=\downarrow,\uparrow}\Big) &= \psi\Big(h_\sigma^t W_0^{t+1} \\
&\qquad + m_{\uparrow\sigma}^{t+1} + m_{\downarrow\sigma}^{t+1}\Big),
\end{aligned}$$

and by letting AGG be the summation over the extended notion of upper and lower R-neighborhoods, that is neighborhoods comprising p-simplices at a distance from σ which is at most R (τ is at distance d from σ if there exists a sequence of upper- (respectively, lower-) adjacent p-simplices $[\nu_0, \nu_1, \ldots, \nu_d]$ such that $\nu_0 = \sigma, \nu_d = \tau$). □

It is noteworthy that, contrary to our general proposed framework, the two message functions $M_{\uparrow}^{t+1}$ and $M_{\downarrow}^{t+1}$ share the same learnable parameters $\{W_r^{t+1}\}_{r=1}^{R}$, and that no signal of order lower or higher than p is involved in computation.

SC-Conv Bunch et al. (2020) proposed a convolutional operator that can be applied on 2-dimensional simplicial complexes. The construction is based on the canonical normalised Hodge Laplacians defined by Schaub et al. (2020); starting from the operators, the authors generalize the Graph Convolutional Network model proposed in Kipf & Welling (2017) by defining the corresponding adjacency matrices with added self-loops:

$$\begin{aligned}
H_0^{t+1} &= \psi(D_1^{-1}B_1 H_1^t W_{0,1}^t + \tilde{A}_0^u H_0^t W_{0,0}^t) \\
H_1^{t+1} &= \psi(B_2 D_3 H_2^t W_{1,2}^t \\
&\qquad + (\tilde{A}_1^d + \tilde{A}_1^u) H_1^t W_{1,1}^t
\end{aligned}$$

$$+ D_2 B_1^\top D_1^{-1} H_0^t W_{1,0}^t)$$
$$H_2^{t+1} = \psi(\tilde{A}_2^d H_2^t W_{2,2}^t + D_4 B_2^\top D_5^{-1} H_1^t W_{2,1}^t).$$

We defer readers to Section 2.1 of the original paper for the definitions of the $\{\tilde{A}_i^\alpha\}_{i=0}^2$ and $\{D\}_{i=1}^5$ matrices in the above equations. Differently from Ebli et al. (2020), this scheme models the interactions between signals defined at different dimensions. It can, nonetheless, be rewritten in terms of our message passing framework.

Proof of Theorem 12 in Reference to Bunch et al. (2020). We report here the derivation for message passing on 1-simplices (edges) as it is the most general. The derivation on 0- and 2-simplices can simply be obtained as a special case of this last.

First, let us conveniently denote:

$$\Delta_{1,1} = \tilde{A}_1^d + \tilde{A}_1^u,\ \ \Delta_{1,0} = D_2 B_1^\top D_1^{-1},\ \ \Delta_{1,2} = B_2 D_3.$$

We note that the convolutional operation for a generic 1-simplex δ can be rewritten as:

$$\begin{aligned}
h_{1,\delta}^{t+1} &= \\
&= \psi\Big((\Delta_{1,2})_\delta H_2^t W_{1,2}^t + (\Delta_{1,1})_\delta H_1^t W_{1,1}^t \\
&\quad + (\Delta_{1,0})_\delta H_0^t W_{1,0}^t\Big) \\
&= \psi\Big(\sum_{\tau \in \mathcal{S}_2} (\Delta_{1,2})_{\delta,\tau} h_{2,\tau}^t W_{1,2}^t + \sum_{\gamma \in \mathcal{S}_1} (\tilde{A}_1^u)_{\delta,\gamma} h_{1,\gamma}^t W_{1,1}^t \\
&\quad + \sum_{\gamma \in \mathcal{S}_1} (\tilde{A}_1^d)_{\delta,\gamma} h_{1,\gamma}^t W_{1,1}^t + \sum_{\sigma \in \mathcal{S}_0} (\Delta_{1,0})_{\delta,\sigma} h_{0,\sigma}^t W_{1,0}^t\Big) \\
&= \psi\Big(\sum_{\tau \in \mathcal{C}(\delta)} (\Delta_{1,2})_{\delta,\tau} h_{2,\tau}^t W_{1,2}^t + \big(\sum_{\gamma \in \mathcal{N}_\uparrow(\delta)} (\tilde{A}_1^u)_{\delta,\gamma} h_{1,\gamma}^t \\
&\quad + \sum_{\gamma \in \mathcal{N}_\downarrow(\delta)} (\tilde{A}_1^d)_{\delta,\gamma} h_{1,\gamma}^t + (\Delta_{1,1})_{\delta,\delta} h_{1,\delta}^t\big) W_{1,1}^t \\
&\quad + \sum_{\sigma \in \mathcal{B}(\delta)} (\Delta_{1,0})_{\delta,\sigma} h_{0,\sigma}^t W_{1,0}^t\Big).
\end{aligned}$$

This equation is interpreted in terms of our message passing scheme by setting:

$$\begin{aligned}
M_{1,\uparrow}^{t+1}\big(h_{1,\delta}^t, h_{1,\gamma}^t, h_{2,\delta\cup\gamma}^t\big) &= (\tilde{A}_1^u)_{\delta,\gamma} h_{1,\gamma}^t \\
M_{1,\downarrow}^{t+1}\big(h_{1,\delta}^t, h_{1,\gamma}^t, h_{0,\delta\cap\gamma}^t\big) &= (\tilde{A}_1^d)_{\delta,\gamma} h_{1,\gamma}^t \\
M_{1,\mathcal{C}}^{t+1}\big(h_{1,\delta}^t, h_{2,\tau}^t\big) &= (\Delta_{1,2})_{\delta,\tau} h_{2,\tau}^t \\
M_{1,\mathcal{B}}^{t+1}\big(h_{1,\delta}^t, h_{0,\sigma}^t\big) &= (\Delta_{1,0})_{\delta,\sigma} h_{0,\sigma}^t
\end{aligned}$$

$$\begin{aligned}
U_1^{t+1}&\Big(h_{1,\delta}^t, \{m_i^t(\delta)\}_{i=\mathcal{B},\mathcal{C},\downarrow,\uparrow}\Big) \\
&= \psi\Big({W_{1,1}^t}^\top \big((\Delta_{1,1})_{\delta,\delta} h_{1,\delta}^t + m_\uparrow(\delta)^{t+1} + m_\downarrow(\delta)^{t+1}\big) \\
&\quad + {W_{1,2}^t}^\top m_{\mathcal{C}}(\delta)^{t+1} \\
&\quad + {W_{1,0}^t}^\top m_{\mathcal{B}}(\delta)^{t+1}\Big),
\end{aligned}$$

and AGG $= \sum$. □

Simplicial Complex Pooling Typically, CNNs interleave convolutional layers with pooling layers, which spatially downsample the input features. This problem has proven to be much more challenging on graph domains, where no obvious graph coarsening strategy exists. While we have not employed any simplicial complex pooling operators in this work, this is likely going to be an exciting direction of future work. Some of the previously proposed graph pooling operators can readily be extended to simplicial complexes. For instance, TopK pooling (Cangea et al., 2018) can be performed over the vertices of the complex and the higher-order structures associated with the pooled vertices can be maintained at the next layer. Other operators, such as the topologically-motivated pooling proposed by Bodnar et al. (2020) can already handle simplicial complexes, but it has only been employed in a graph setting.

D. Equivariance and Invariance

One would expect MPSNs to be aware of the two symmetries of a simplicial complex: relabeling of the simplicies in the complex and, optionally, changes in the orientation of the complex if the complex is oriented. We address these two below.

Permutation Equivariance Let $\mathcal{K}$ be simplicial p-complex with a sequence $\mathbf{B}$ of boundary matrices $(B_1, \ldots, B_p)$ and a sequence $\mathbf{H}$ of feature matrices $(H_0, \ldots, H_p)$. Let $\mathbf{P}$ be a sequence of permutation matrices $(P_0, \ldots, P_p)$ with $P_i \in \mathbb{R}^{S_i \times S_i}$. Denote by $\mathbf{PH}$ the sequence of permuted features $(P_0 H_0, \ldots, P_p H_p)$ and by $\mathbf{PBP}^\top$, the sequence of permuted boundary matrices $(P_0 B_1 P_1^\top, \ldots, P_{p-1} B_p P_p^\top)$.

Definition 34 (Permutation equivariance)**.** *A function $f : (\mathbf{H}^{\text{in}}, \mathbf{B}) \mapsto \mathbf{H}^{\text{out}}$ is (simplex) permutation equivariant if $f(\mathbf{PH}^{\text{in}}, \mathbf{PBP}^\top) = \mathbf{P}f(\mathbf{H}^{\text{in}}, \mathbf{B})$ for any sequence of permutation operators $\mathbf{P}$.*

Definition 35 (Permutation invariance)**.** *Similarly, we say that a function f is (simplex) permutation invariant if $f(\mathbf{PH}^{\text{in}}, \mathbf{PBP}^\top) = f(\mathbf{H}^{\text{in}}, \mathbf{B})$ for any sequence of permutation operators $\mathbf{P}$.*

Proof of Theorem 13. We abuse the notation slightly and use $P_i(a)$ to denote the corresponding permutation function of P_i acting on indices.

We focus on a single simplex σ of an arbitrary dimension n and the corresponding $\tau = P_n(\sigma)$. Let h_σ be the output feature of simplex σ for an MPSN layer taking $(\mathbf{H}, \mathbf{B})$ as input and $\bar{h}_\tau$ the output features of simplex τ for the same MPSN layer taking $(\mathbf{PH}, \mathbf{PBP}^\top)$ as input. We will now show they are equal by showing that the multi-set of features being passed to the message, aggregate and update functions are the same for the two simplicies.

The boundary of the n-simplicies in $\mathbf{B}$ are given by the non-zero elements of B_n. Similarly, in $\mathbf{PBP}^\top$, these are given by the non-zero elements of $P_{n-1}B_nP_n^\top$. That is the same boundary matrix but with the rows and columns permuted according to P_n). This then gives

$$(B_n)_{a,b} = (P_{n-1}B_nP_n^\top)_{P_{n-1}(a),P_n(b)}.$$

In particular, this holds for $b = \sigma, P_n(b) = \tau$. Because the feature matrices for the $(n-1)$-simplices in $\mathbf{PH}$ are also accordingly permuted with $P_{n-1}H_{n-1}$, σ and τ receive the same message from their boundary simplices. The proof follows similarly for co-boundary adjacencies.

The lower adjacenices of the n-simplicies in $\mathbf{B}$ are given by the non-zero entries of $B_n^\top B_n$. Similarly, the lower adjacencies in $\mathbf{PB}$ are given by the non-zero elements of

$$\begin{aligned}(P_{n-1}B_nP_n^\top)^\top(P_{n-1}B_nP_n^\top) &= P_nB_n^\top P_{n-1}^\top P_{n-1}B_nP_n^\top \\ &= P_nB_n^\top B_nP_n^\top.\end{aligned}$$

That is, the same adjacencies as in $\mathbf{B}$, but with the rows and columns are accordingly permuted. Therefore, we obtain

$$(B_n^\top B_n)_{a,b} = (P_nB_n^\top B_nP_n^\top)_{P_n(a),P_n(b)},$$

which, in particular, holds for $a = \sigma, P_n(a) = \tau$. Since the feature matrices for the n-simplices in $\mathbf{PB}$ are also permuted with P_nH_n, σ and τ receive the same message from the lower adjacenct simplicies. This can be similarly shown for upper adjacencies. □

Permutation invariance for an MPSN model can be obtained by stacking multiple permutation equivariant layers followed by a permutation invariant readout function. The readout must be permutation invariant in each of the multi-sets of simplices of different dimensions it takes as input. Any commonly used GNN readout functions such as sum or mean could be employed.

Orientation Equivariance Another symmetry that we would like to preserve is orientation. For instance, we know that the homology of the complex is invariant to the particular orientation that was chosen. Therefore, for certain applications where orientations are of interest, we would like to design MPSN layers that are orientation equivariant and MPSN networks that are orientation invariant.

When a simplex σ changes its orientation, it changes its relative orientation (i.e. $+1$ becomes -1 and vice-versa) with respect to all the neighbours in the complex, and the signature of its own features is flipped. Overall, this can be modelled by multiplying the rows and columns of the boundary matrices where σ appears and the corresponding row of the feature matrix by -1. We formalise this intuition below.

Consider a simplicial p-complex $\mathcal{K}$. Let $\mathbf{T} = (T_0, \dots, T_p)$ be a sequence of diagonal matrices with $T_i(j,j) = \pm 1$ for all $i > 0$ and any j and $T_0 = I$. The latter constraint is due to the fact that vertices have trivial orientation and it cannot be changed. Using the same notation as for permutation equivariance, define $\mathbf{TH} = (T_0H_0, \dots, T_pH_p)$ and $\mathbf{TBT} = (T_0B_1T_1, \dots, T_{p-1}B_pT_p)$.

Definition 36 (Orientation equivariance)**.** *A function f mapping $(\mathbf{H}^{\text{in}}, \mathbf{B}) \mapsto \mathbf{H}^{\text{out}}$ is orientation equivariant if $f(\mathbf{TH}^{\text{in}}, \mathbf{TBT}) = \mathbf{T}f(\mathbf{H}^{\text{in}}, \mathbf{B})$ for any sequence of operators $\mathbf{T}$.*

Definition 37 (Orientation invariance)**.** *A function f is orientation invariant if $f(\mathbf{TH}^{\text{in}}, \mathbf{TBT}) = f(\mathbf{H}^{\text{in}}, \mathbf{B})$ for any sequence of operators $\mathbf{T}$.*

Remark 38. *Orientation invariance can be trivially achieved by considering the absolute value of the features and by treating the complex as an unoriented one.*

However, it is (in general) desirable to use equivariance at the intermediate layers and make the network invariant with a final transformation (readout). Constructing such an MPSN layer requires imposing additional constraints on the structure of the message, update and aggregate functions. We will start with a simple MPSN layer that can be expressed in the matrix form to develop an intuition, and then we will generalise this to provide a more flexible MPSN layer that is orientation equivariant.

For instance, we can consider the model from (11) used in our linear regions analysis, with a convenient vectorised form

$$\begin{aligned}H_i^{\text{out}} = \psi\big(&B_i^\top B_iH_i^{\text{in}}W_1 + H_i^{\text{in}}W_2 + B_{i+1}B_{i+1}^\top H_i^{\text{in}}W_3 \\ &+ B_i^\top H_{i-1}^{\text{in}}W_4 + B_{i+1}H_{i+1}^{\text{in}}W_5\big),\end{aligned} \tag{27}$$

where we have separated the upper and lower adjacencies in two and included the feature H_i matrix as an additional term. This corresponds to an MPSN with a message function that multiplies the feature of the neighbour by the relative orientation (± 1), sum-based aggregation and an update function that adds the incoming messages to its linearly transformed features and passes the output through ψ.

Proposition 39. *When ψ is an odd activation function, the MPSN layer from Equation (27) is orientation equivariant.*

Proof. Let $H_i^{\text{out}} = f_i(B_i, B_{i+1}, H_i^{\text{in}}, H_{i-1}^{\text{in}}, H_{i+1}^{\text{in}})$ be the application of one such MPSN layer on the i-dimensional simplices. Note that the output depends only on the simplices of dimension $i-1, i$, and $i+1$.

For this MPSN layer to be equivariant, we need to show that

$$f_i(T_{i-1}B_iT_i, T_iB_{i+1}T_{i+1}, T_iH_i^{\text{in}}, T_{i-1}H_{i-1}^{\text{in}}, T_{i+1}H_{i+1}^{\text{in}}) \\ = T_iH_i^{\text{out}} \tag{28}$$

Because $T_iT_i = I$ and $T_i^\top = T_i$ for all i, we can easily rewrite LHS as

$$\psi\Big(T_i\big(B_i^\top B_iH_i^{\text{in}}W_1 + H_i^{\text{in}}W_2 + B_{i+1}B_{i+1}^\top H_i^{\text{in}}W_3 \\ + B_i^\top H_{i-1}^{\text{in}}W_4 + B_{i+1}H_{i+1}^{\text{in}}W_5\big)\Big). \tag{29}$$

Notice that if ψ and T_i commute, then this becomes $T_iH_i^{\text{out}}$. We remark that they commute when ψ is an odd function. □

We can generalise this particular architecture to obtain a more general MPSN layer that is orientation equivariant. Based on Equation (28), we can see that for an arbitrary simplex σ, the output features are invariant with respect to changes in the orientation of the neighbours and it is only affected by changes in its own orientation.

Let us rewrite the equivalent constraint from Equation (28) for a local aggregation of the neighbourhood of a simplex σ in the complex. Let us denote by $o_{\sigma,\tau}$, the relative orientation between σ and τ. Consider an abstract local neighbourhood aggregation function A taking as input the features h_σ and a multi-set of tuples containing the features of the neighbours of σ and the relative orientations $\{\!\{(o_{\sigma,\tau}, h_\tau) \mid \tau \in \mathcal{N}(\sigma)\}\!\}$. For brevity, we consider a single generic multi-set of neighbours denoted here by $\mathcal{N}(\sigma)$, but multiple types can be easily integrated into A.

Based on Equation (28), we know A is invariant to changes in the orientations of the neighbours. This means that it is invariant in changes in the signature of the relative orientations and of the features of the neighbours:

$$A(h_\sigma, \{\!\{(o_{\sigma,\tau}, h_\tau)\}\!\}) = A(h_\sigma, \{\!\{\pm(o_{\sigma,\tau}, h_\tau)\}\!\}). \tag{30}$$

At the same time, we know that the output of A must change its sign when the orientation of σ changes. That is, when all the relative orientations change their signs together with the features of σ. This leads to a second equation:

$$A(h_\sigma, \{\!\{(o_{\sigma,\tau}, h_\tau)\}\!\}) = -A(-h_\sigma, \{\!\{(-o_{\sigma,\tau}, h_\tau)\}\!\}). \tag{31}$$

In other words, A must be even in each $(o_{\sigma,\tau}, h_\tau)$ and also odd in $(h_\sigma, \{\!\{o_{\sigma,\tau}\}\!\})$.

Consider an auxiliary function A^* taking as input the feature vector h_σ and a multi-set, such that A^* is an odd function:

$$A^*(h_\sigma, \{\!\{h_\tau\}\!\}) = -A^*(-h_\sigma, \{\!\{-h_\tau\}\!\}).$$

Lemma 40. *Let A^* be an odd local aggregator as above. Then the function A defined by:*

$$A(h_\sigma, \{\!\{(o_{\sigma,\tau}, h_\tau)\}\!\}) := A^*(h_\sigma, \{\!\{o_{\sigma,\tau} \cdot h_\tau\}\!\}),$$

where $\cdot$ denotes scalar-vector multiplication, satisfies Equations 30 and 31.

Proof. First we prove A satisfies Equation (30). Substituting the definition of A:

$$\begin{aligned} A(h_\sigma, \{\!\{\pm(o_{\sigma,\tau}, h_\tau)\}\!\}) &= A^*(h_\sigma, \{\!\{\pm o_{\sigma,\tau} \cdot \pm h_\tau\}\!\}) \\ &= A^*(h_\sigma, \{\!\{o_{\sigma,\tau} \cdot h_\tau\}\!\}) \\ &= A(h_\sigma, \{\!\{(o_{\sigma,\tau}, h_\tau)\}\!\}). \end{aligned} \tag{32}$$

For proving A satisfies Equation (31) we substitute the definition again:

$$\begin{aligned} A(-h_\sigma, \{\!\{(-o_{\sigma,\tau}, h_\tau)\}\!\}) &= A^*(-h_\sigma, \{\!\{-o_{\sigma,\tau} \cdot h_\tau\}\!\}) \\ &= -A^*(h_\sigma, \{\!\{o_{\sigma,\tau} \cdot h_\tau\}\!\}) \\ &= -A(h_\sigma, \{\!\{(o_{\sigma,\tau}, h_\tau)\}\!\}). \end{aligned} \tag{33}$$

□

Proof of Theorem 14. The proof follows immediately from Lemma 40 since the composition of odd functions is odd. Therefore, the function A^* can be implemented using a combination of odd message, aggregate and update functions. □

It is useful to note that any MLP without bias units and odd activation functions is an odd function. Additionally, all linear aggregators (mean, sum) are odd. Therefore, most commonly used GNN layers could be adopted by simply using an odd activation function.

Another important remark is that if the local aggregator ignores the relative orientations, then Equations 30 and 31 simply imply that the local aggregator of the MPSN layer must be odd in the features of σ and even in the features of the neighbours. If the features of σ are not used in the message function, then an even message function combined with an update function that is odd in h_σ also satisfies the equations.

Remark 41. *A (permutation invariant) aggregation-based readout layer first applying an element-wise even function ψ to the elements of its input multiset is orientation invariant since*

$$AGG(\{\!\{\psi(x_i)\}\!\}) = AGG(\{\!\{\psi(\pm x_i)\}\!\}). \tag{34}$$

A flexible way to implement such a layer is to consider a function $\psi(x) = f(x) + f(-x)$, where f is an arbitrary non-odd function that could be parametrised as an MLP. It is easy to see that such a function is even since summation commutes:

$$\psi(x) = f(x) + f(-x) = f(-x) + f(x) = \psi(-x).$$

To conclude this section, we note that the concurrent work of Glaze et al. (2021) has also analysed the equivariance properties of simplicial networks in the context of a specific convolutional operator that is similar to the one in Equation (27). However, our simplicial message passing scheme is more general, and effectively subsumes the convolutional operator presented in their work. Therefore, the equivariance analysis performed here also applies to a much larger family of models.

E. Cubical Complexes

A message passing approach for cell complexes, a generalisation of simplicial complexes, has been proposed by Hajij et al. (2020). Cell Complexes have a similar hierarchical structure to simplicial complexes, and the approach of Hajij et al. (2020) is very close to ours. At the same time, arbitrary cell complexes might be too general for certain applications. Here we discuss how our approach could be extended to cubical complexes, a type of cell complex that from a theoretical point of view is similar to simplicial complexes (Kaczynski et al., 2004), and which is used in applications. We plan to extend our approach to cubical complexes in future work.

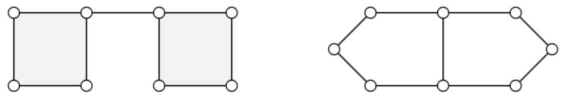

Figure 8. Example of two cubical complexes whose underlying graphs cannot be distinguished by 1-WL, but are not isomorphic as cubical complexes.

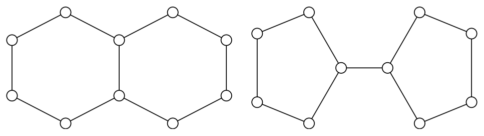

Figure 9. Decalin and *Bicyclopentyl*: Non-isomorphic molecular graphs that cannot be distinguished neither by WL nor by SWL, when based on clique complexes (here the nodes represent carbon atoms, and edges are chemical bonds).

Cubical complexes are cell complexes consisting of unions of points, edges, squares, cubes, and higher-dimensional hypercubes (Kaczynski et al., 2004). While they are less well-known than simplicial complexes, they can nevertheless be used in applications, and are better suited than simplicial complexes to the study of certain types of data sets, see, e.g. Wagner et al. (2011). Our approach to message passing can be directly implemented also for cubical complexes. One could similarly define a WL test for cubical complexes.

In Figure 8 we provide an example of two cubical complexes that are not isomorphic, while their underlying graphs are isomorphic. We note that the clique complexes of the underlying graphs are also isomorphic. The examples in Figures 2 and 8 raise the question of whether we could design tests that are better suited to take into account the topological information encoded in simplicial and cubical complexes, such as homeomorphism tests, which have already been studied, see e.g. Baik & Miller (1990). In particular, such tests should be able to distinguish the graphs in Figure 9, a pair of real world molecular graphs that cannot be distinguished by the SWL test based on clique complexes.

A natural next step in our work is to perform tests on data sets that are more naturally modelled by cubical complexes, such as digital images, for which they can provide computational speed-ups compared to simplicial complexes (Wagner et al., 2011; Kaczynski et al., 2004).

F. Additional Experimental Details

F.1. Strongly Regular Graphs

The original SR datasets can be found at `http://users.cecs.anu.edu.au/~bdm/data/graphs.html`. Families in Figure 5 are referred to as per the SR(v, k, λ, μ) notation (see Section A.1). We set $\varepsilon = 0.01$ as the Euclidean distance threshold for which graphs are deemed non-isomorphic. Graphs are embedded in a 16-dimensional space by running an untrained, 5-layer, MPSN model on the associated clique complexes, obtained by considering any $(p + 1)$-clique as a p-simplex in the complex. Nodes are initialised with a constant, scalar, unitary signal, while higher-order simplices with the sum of the features of the constituent nodes. As already mentioned in the main text, the specific MPSN architecture is dubbed 'SIN' given its resemblance to the GIN model (Xu et al., 2019b). The following message passing operations are employed to compute the $t + 1$ intermediate representation for a p-simplex σ:

$$h_\sigma^{t+1} = \text{MLP}^t_{U,p}\Big(\text{MLP}^t_{\mathcal{B},p}\big((1+\epsilon_{\mathcal{B}})h^t_\sigma + \sum_{\delta\in\mathcal{B}(\sigma)} h^t_\delta\big) \parallel \text{MLP}^t_{\uparrow,p}\big((1+\epsilon_\uparrow)h^t_\sigma + \sum_{\tau\in\mathcal{N}_\uparrow(\sigma)} M^t_{\uparrow,p}(h^t_\tau, h^t_{\sigma\cup\tau})\big)\Big)$$

$$M^t_{\uparrow,p}(h^t_\tau, h^t_{\sigma\cup\tau}) = \text{MLP}^t_{M,p}\big(h^t_\tau \parallel h^t_{\sigma\cup\tau}\big), \qquad (35)$$

where $\parallel$ indicates concatenation, $\text{MLP}^t_{\mathcal{B},p}$ and $\text{MLP}^t_{\uparrow,p}$ are 2-Layer Perceptrons and $\text{MLP}^t_{U,p}$, $\text{MLP}^t_{M,p}$ consist of a dense layer followed by a non-linearity. Parameters $\epsilon_{\mathcal{B}}$, $\epsilon_{\uparrow}$ are set to zero. Notice how, in accordance with Theorem 6, messages are only aggregated from boundary and upper-adjacent simplices (for which we also include the representations of the shared coboundary simplices). After the $L = 5$ message passing layers, for a d-complex X we perform readout operations as follows:

$$h_X = \text{MLP}^{\mathcal{K}}\Big(\sum_{p=0}^{d} \text{MLP}_p^{\mathcal{S}}\big(\sum_{\sigma \in \mathcal{S}_p} h_\sigma^L\big)\Big), \tag{36}$$

where $\mathcal{S}_p$ is the set of simplices at dimension p, and $\text{MLP}^{\mathcal{K}}$, $\text{MLP}_p^{\mathcal{S}}$ are dense layers followed by non-linearities, set to ELU (Clevert et al., 2016) throughout the whole architecture. All layer sizes are set to 16, except for the one of $\text{MLP}_p^{\mathcal{S}}$ layers, which is set to $2 \cdot 16 = 32$.

The 'MLP-sum' baseline replaces the 5 message passing layers by applying a independent dense layer, followed by non-linearity, at each complex dimension. The readout scheme replicates the one in SIN we just described above. In this model, the initial dense layers have size 256, $\text{MLP}_p^{\mathcal{S}}$ layers have size $2 \cdot 256 = 512$ and $\text{MLP}^{\mathcal{K}}$ has size 16 as in SIN. We apply ELU non-linearities in this model as well.

Finally, we report here that we ran a comparable GIN architecture and empirically verified its inability to distinguish any pair, theoretically justified by its expressive power being upper-bounded by 1-WL (Xu et al., 2019b).

F.2. Trajectory Prediction

Dataset Details To generate the synthetic complex, we uniformly sample 1,000 points in the unit square, we perform a Delaunay triangulation of these points and then remove the triangles (and points) intersecting with two predefined regions of the plane to create the two holes. To generate the trajectories, we first randomly sample a random point from the top-left corner of the complex and an end point from the bottom-right corner of the complex. We then perform a random walk on the edges of the complex. With a probability of 0.9, the neighbour closest to the end-point is chosen, and with a probability 0.1, a random neighbour is chosen. To generate the two classes, we set random points either from the bottom-left corner or the top-right corner as an intermediate checkpoint. We generate 1,000 train trajectories and 200 test trajectories.

The ocean drifter benchmark has been designed in light of a study conducted by Schaub et al. (2020) showing that clock- and counterclockwise drifter trajectories around Madagascar can be disentangled when projected on the harmonic subspace of a normalised Hodge 1-Laplacian. Such operator is the one associated with the simplicial complex representing the underlying geographical map. The original drifter measurements have been collected by the Global Drifter Program, and have been made publicly available by the U.S. National Oceanic and Atmospheric Administration (NOAA) at `http://www.aoml.noaa.gov/envids/gld/`. The simplicial complex structure around the Madagascar island is constructed in accordance with Schaub et al. (2020) and Glaze et al. (2021), i.e. by considering an underlying hexagonal tiling and by generating 2-simplices from the triangles induced by the local adjacency between tiles. Edge-flows are also determined consistently with these works, that is by drawing trajectories based on the position of measurement buoys relative to the underlying hexagonal tiles. We readapted the preprocessing script from Glaze et al. (2021), which is publicly available at `https://github.com/nglaze00/SCoNe_GCN/tree/master/ocean_drifters_data`. The final dataset we use consists of 160 train trajectories and 40 test trajectories.

For both benchmarks, the simplicial complexes in the training dataset use a fixed arbitrary orientation, while each test trajectory uses a random orientation obtained by multiplying the boundary and feature matrices with a random diagonal matrix T_1 with ± 1 entries as in Appendix D.

Architecture All the evaluated models use a similar architecture. The MPSN Id, Tanh and ReLU models use layers of the form:

$$\psi\Big(W_0 h_\sigma^t + \sum_{\tau \in \mathcal{N}_\downarrow(\sigma)} W_1 h_\tau^t o_{\sigma,\tau} + \sum_{\delta \in \mathcal{N}_\uparrow(\sigma)} W_2 h_\delta^t o_{\sigma,\delta}\Big),$$

where ψ represents the non-linearity from the model's name and $o_{\sigma,\tau} = \pm 1$ represents the relative orientation between σ and τ. Based on Theorem 14, this layer is orientation equivariant if ψ is an odd function. Thus, MPSN Id and MPSN Tanh are equivariant, whereas MPSN ReLU is not.

The L_0-inv MPSN uses a similar layer, but drops the relative orientations:

$$\psi\Big(W_0 h_\sigma^t + \sum_{\tau \in \mathcal{N}_\downarrow(\sigma)} W_1 h_\tau^t + \sum_{\delta \in \mathcal{N}_\uparrow(\sigma)} W_2 h_\delta^t\Big).$$

The GNN L_0-inv model uses the same layer, but without upper adjacencies:

$$\psi\Big(W_0 h_\sigma^t + \sum_{\tau \in \mathcal{N}_\downarrow(\sigma)} W_1 h_\tau^t\Big).$$

Additionally, the L_0-inv models are made orientation invariant by taking the absolute value of the input features before passing them through the network.

All models use a sum-readout followed by an MLP. The MPSN Id, Tanh and ReLU models use the absolute value of the features before the readout.

Table 3. Hyperparameter configurations on TUDatasets.

Hyperparameter	PROTEINS	NCI1	IMDB-B	IMDB-M	RDT-B	RDT-M5K
Batch Size	128	128	32	128	32	32
Initial LR	0.01	0.001	0.0005	0.003	0.001	0.001
LR Dec. Steps	20	50	20	50	50	20
LR Dec. Strength	0.5	0.5	0.5	0.5	0.5	0.9
Hidden Dim.	32	32	64	64	64	64
Drop. Rate	0.5	0.5	0.5	0.5	0.0	0.0
Drop. Pos.	a	a	a	a	b	b
Num. Layers	3	4	4	4	4	4

Hyperparameters & Evaluation Procedure All the models use the same hyperparameters. We use 4 layers, the hidden size is set to 64, the batch size is 64, and the initial learning rate is set to 0.001 and decayed by a factor of 0.5 with a dataset-depending frequency. On the synthetic dataset we train for 100 epochs and reduce the learning rate every 20 epochs. On the ocean drifter dataset we train for 250 epochs and reduce the learning rate every 50 epochs. We report the final training and test accuracy at the end of training for all models over five seeds.

F.3. Real-World Graph Classification

The graph classification tasks from this set of experiments are commonly used to benchmark GNNs and are available at `https://chrsmrrs.github.io/datasets/`. We lift each graph to a clique 2-complex, where 0-simplices are initialised with the original node signals as prescribed in Xu et al. (2019b). Higher dimensional simplices are initialised with the mean of the constituent nodes. We employ a SIN model which applies the following message passing scheme to compute the $t+1$ intermediate representation for p-simplex σ:

$$h_\sigma^{t+1} = \mathrm{MLP}^t_{U,p}\Big(\mathrm{MLP}^t_{\mathcal{B},p}\big((1+\epsilon_\mathcal{B})h^t_\sigma + \sum_{\delta\in\mathcal{B}(\sigma)} h^t_\delta\big) \parallel \mathrm{MLP}^t_{\uparrow,p}\big((1+\epsilon_\uparrow)h^t_\sigma + \sum_{\tau\in\mathcal{N}_\uparrow(\sigma)} h^t_\tau\big)\Big), \quad (37)$$

where $\parallel$ indicates concatenation, $\mathrm{MLP}^t_{\mathcal{B},p}$ and $\mathrm{MLP}^t_{\uparrow,p}$ are 2-Layers Perceptrons endowed with Batch Normalization (Ioffe & Szegedy, 2015) (BN) and ReLU activations, $\mathrm{MLP}^t_{U,p}$ is a dense layer followed by the application of BN and ReLU. The only exception is represented by Reddit datasets, on which BN was observed to cause instabilities in the training process and was not applied. Parameters $\epsilon_\mathcal{B}$ and $\epsilon_\uparrow$ are set to zero and are not optimised. As it is possible to notice in Equation (37), upper message $m_{\uparrow,p}(\sigma)$ is computed as $m_{\uparrow,p}(\sigma) = \sum_{\tau\in\mathcal{N}_\uparrow(\sigma)} h^t_\tau$, thus explicitly disregarding the representation of shared co-boundary simplices $h^t_{\sigma\cup\tau}$: this choice showed to yield better performance on these benchmarks. We follow Xu et al. (2019b) and apply a Jumping Knowledge (JK) scheme (Xu et al., 2018): at dimension p, readout is performed on the concatenation of the p-simplex representations obtained at each message passing iteration, projected by the non-linear dense layer $\mathrm{MLP}^\mathcal{S}_p$. Such a readout operation is dataset specific: we apply either averaging or summation as prescribed by Xu et al. (2019b). Final complex representations are obtained by summing the obtained p-embeddings at dimensions 0 ('nodes') and 2 ('triangles'). One last dense layer is applied to output class predictions. Training is performed with Adam optimiser (Kingma & Ba, 2015), starting from an initial learning rate decayed with a certain frequency.

We employ grid-search to tune batch size, hidden dimension, dropout rate, initial learning rate along with its decay steps and strengths, number of layers and the dropout position (before the final readout on the complex ('b'), or after it ('a'). We report the hyperparameter configurations in Table 3. As in experiments on SR graphs, the size of $\mathrm{MLP}^\mathcal{S}_p$ layers is doubled w.r.t. the other layers in the architecture.

F.4. Implementation & Availability

We employed PyTorch (Paszke et al., 2019) for all our experiments and we built on top of the PyTorch Geometric library (Fey & Lenssen, 2019) to implement our simplicial message passing scheme.

As for clique-lifting procedures, we relied on the simplex tree data-structure (Boissonnat & Maria, 2014) implemented in the topological data analysis library Gudhi (The GUDHI Project, 2021). Empirically, in large-scale experiments involving graphs with 10^6 nodes, simplex trees are able to generate the clique complex up to a constant desired dimension in a computational cost that is linear in the number of simplicies in the complex (Boissonnat & Maria, 2014).

We refer readers to `https://arxiv.org/abs/2103.03212` for a link to our official code repository. We provide support for SC datasets, simplicial message passing networks, oriented SCs, (higher-order) batching, clique complexes and other additional features.