Weisfeiler and Lehman Go Topological: Message Passing Simplical Networks

Cristian Bodnar[1,*] Fabrizio Frasca[2,3,*] Yu Guang Wang[4,5*]
Nina Otter[6] Guido Montúfar[4,6,*] Pietro Liò[1] Michael Bronstein[2,3]

[1]University of Cambridge [2]Twitter [3]Imperial College London [4]MPI MIS, Leipzig
[5]University of New South Wales, Sydney [6]University of California, Los Angeles

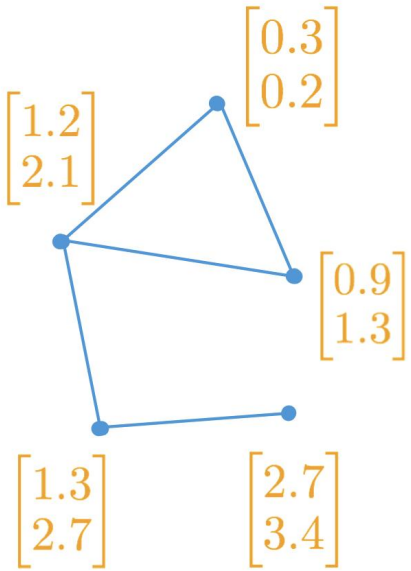
Based on **arxiv:2103.03212**

Background

Pairwise relational modeling of complex systems

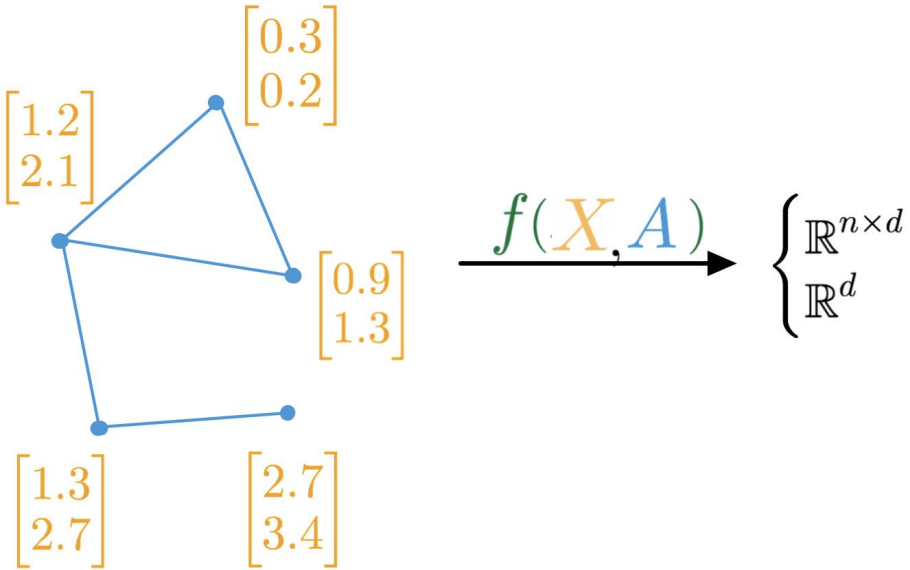
Complex systems are typically modeled as **graphs** with signals attached to the nodes and edges

- friendships in social networks
- molecule interactions
- WorldWideWeb



The surge in Graph Representation Learning

The aim is to learn **data-driven representations** of nodes or entire graphs. These can be used for node/graph regression/classification.



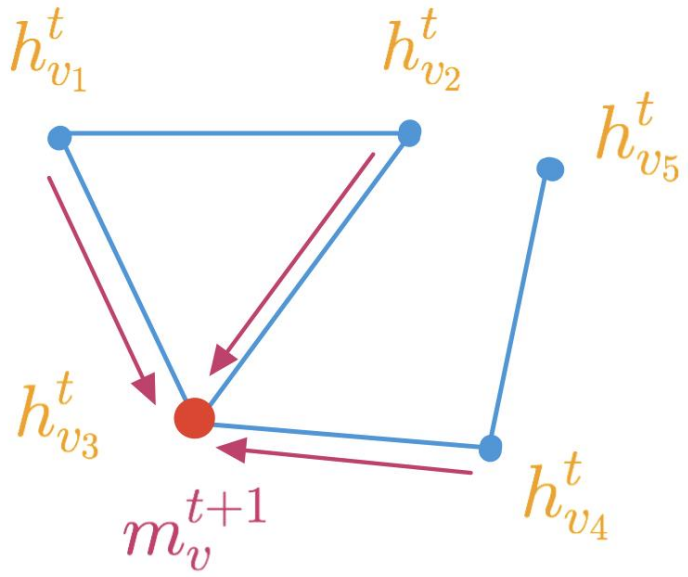
Graph Neural Networks (GNNs)

Message & Aggregate: Each node receives an aggregated message from all its neighbours.

$$m_v^{t+1} = \mathbf{AGG}_{w \in \mathcal{N}(v)} \left(M\left(h_v^t, h_w^t, h_{e(v,w)}^t\right) \right)$$

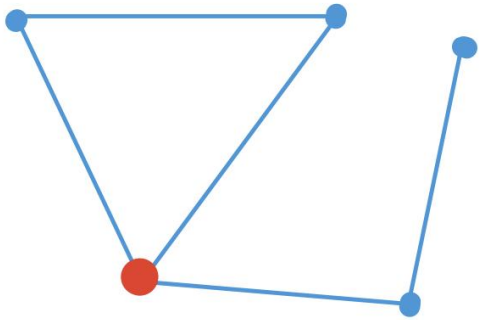
$\boldsymbol{M}$: processes messages

$\mathbf{AGG}$: aggregates processed messages



Graph Neural Networks (GNNs)

Update: Each node updates their own features as a function of the aggregated messages and its current features.



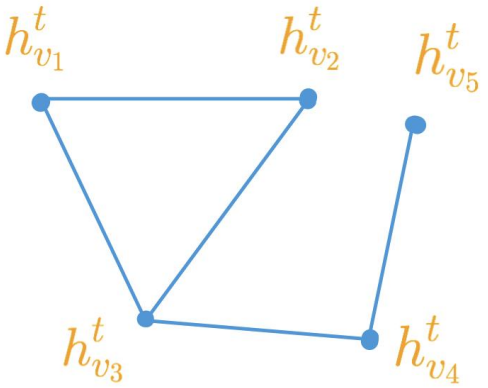
$$h_{v_3}^{t+1} := U(h_{v_3}^{t}, m_v^{t+1})$$

Graph Neural Networks (GNNs)

Readout: To obtain a global embedding, the final features of the nodes are aggregated in a single vector.

$$h_G = \text{READOUT}\left(\{\!\{h_v^L | v \in G\}\!\}\right)$$

READOUT: typically mean or sum, but can be any multiset function.



$$h_G := h_{v_1}^t + h_{v_2}^t + h_{v_3}^t + h_{v_4}^t + h_{v_5}^t$$

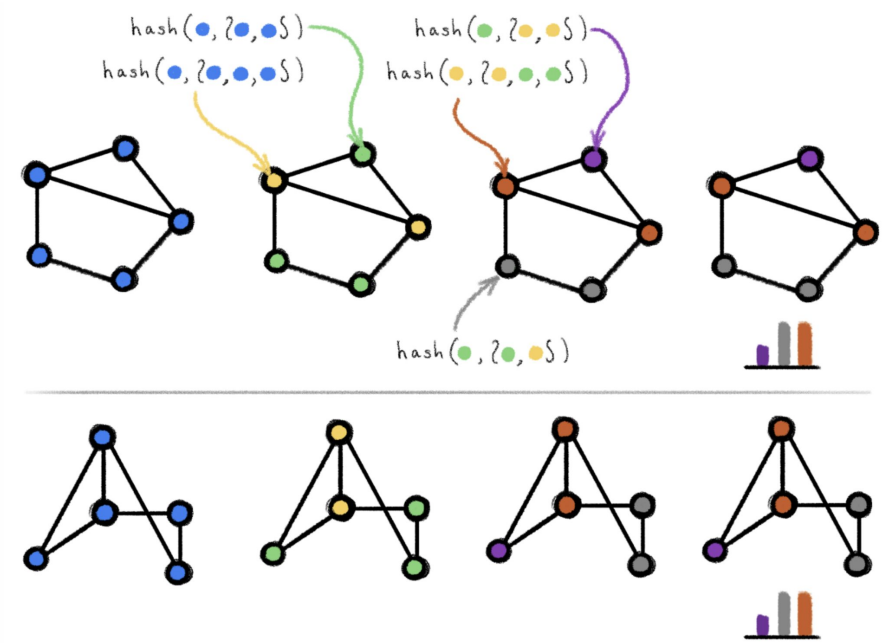
The Weisfeiler-Lehman (WL) algorithm

The WL algorithm tests isomorphism between graphs [1].

It iterates *colour refinements* via hashing of neighboring colours:

$$c_v^{t+1} = \mathrm{HASH}\left(c_v^t, \{\!\{c_u^t\}\!\}_{u \in \mathcal{N}(v)}\right)$$

The WL test works for ***almost***, but not all graphs.



Example of execution of the WL test on two isomorphic graphs [2]

[1]: The reduction of a graph to canonical form and the algebra which appears therein (Weisfeiler and Lehman, NTI Series, 1968)

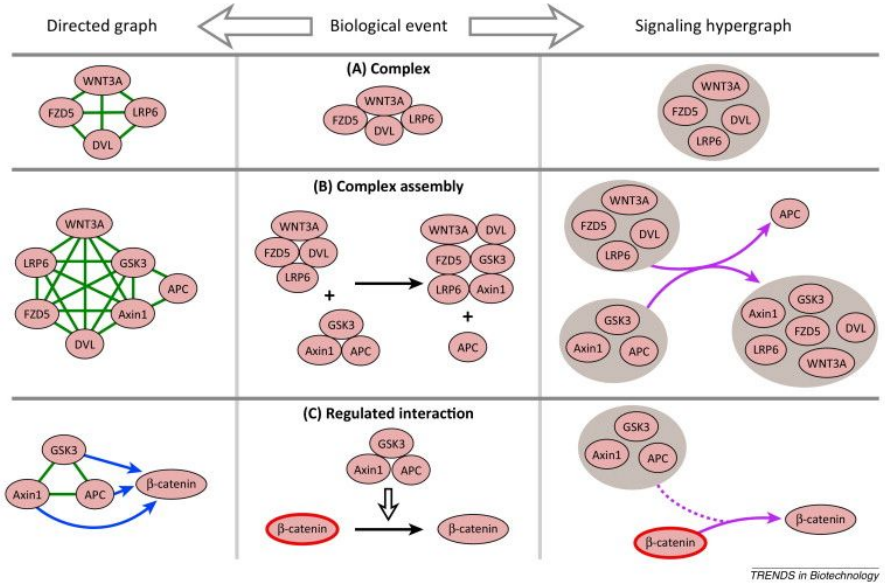
[2]: Expressive power of graph neural networks and the Weisfeiler-Lehman test, blogpost

Beyond graphs

Pairwise modeling is not enough

Higher-order structures better capture the behavior of many complex systems

- Chemical reactions [1]
- Multi-body interactions [2]
- Interaction of biomolecules [3]
- Ecological systems [4]



Comparison of representations for events that occur in signaling pathways, figure from [1].

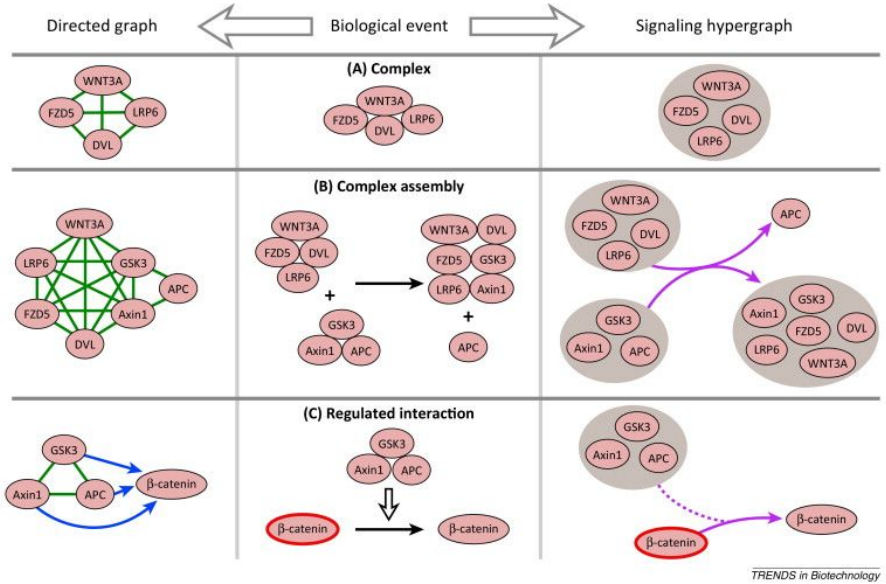
[1]: Hypergraph Laplace operators for chemical reaction networks (Jost and Mulas, 2019)

[2]: Multibody interactions and nonlinear consensus dynamics on networked systems (Neuhäuser et al., 2020)

[3]: Signaling Hypergraphs (Ritz et al., 2014)

[4]: Ecological networks over the edge: Hypergraph trait-mediated indirect interaction (TMII) structure (Golubski et al., 2016)

Pairwise modeling is not enough



Representation limits

Information loss

Deceptive Interpretation

Comparison of representations for events that occur in signaling pathways, figure from [1].

[1]: Signaling Hypergraphs (Ritz et al., 2014)

Pairwise modeling is not enough

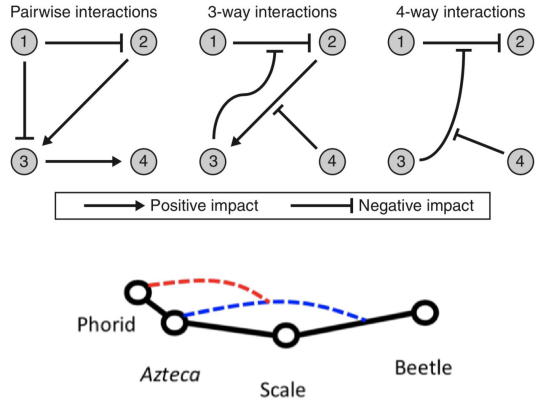
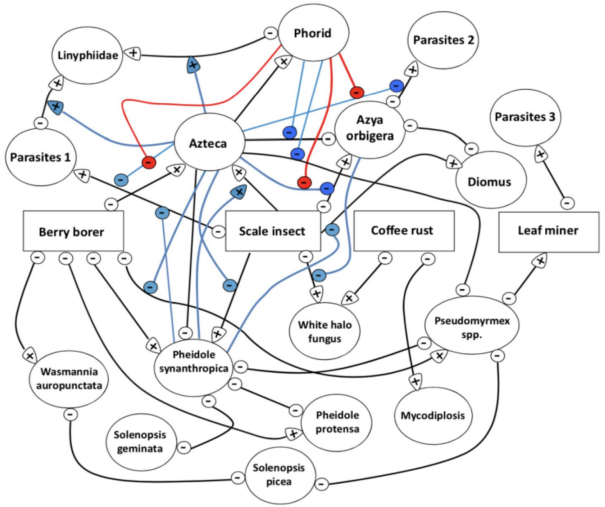


Illustration of mediated interactions. Figures from [1].



Coffee agroecosystem in southern Mexico. Figure from [2].

⇒ these forms of higher-order interactions have been studied to favour variance of species at equilibrium [3]

[1]: Networks beyond pairwise interactions (Battiston et al., 2020)

[2]: Ecological networks over the edge: Hypergraph trait-mediated indirect interaction (TMII) structure (Golubski et al., 2016)

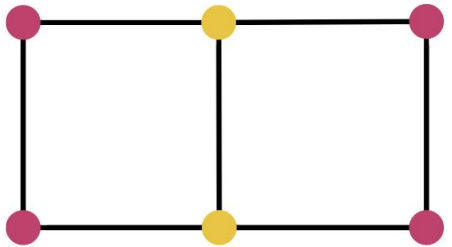
[3]: Rank abundance relations in evolutionary dynamics of random replicators (Yoshino, 2008)

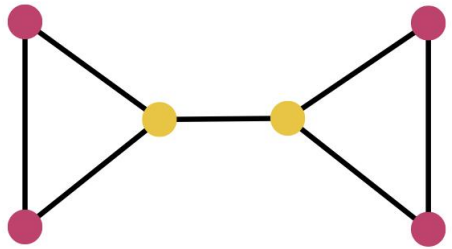
GNNs have limited expressivity power

GNNs can be made at most ***as powerful as*** the WL test with proper parameterisation [1]

If WL fails to distinguish a non-isomorphic pair, any GNN does!

“Corollary”: GNNs inherit the **WL limitations** [2] in counting graph substructures (e.g. triangles).





[1]: How Powerful are Graph Neural Networks? (Xu et al., ICLR 2019)

[2]: Can Graph Neural Networks Count Substructures? (Chen et al., NeurIPS 2020)

Higher-Order Features

How should higher-order features be processed?

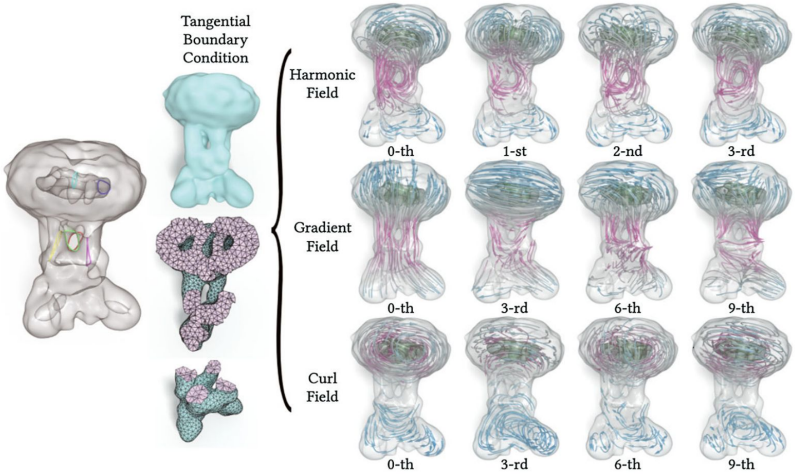
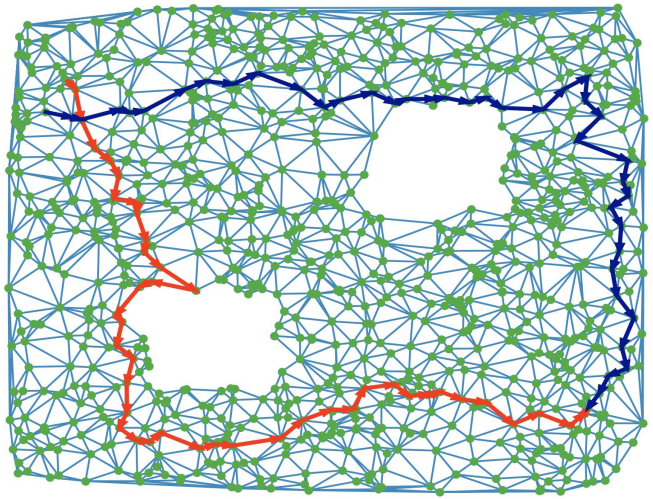


Image from [1]. Field on the surface of a macromolecule



Trajectories in 2D. Originally introduced in [2]

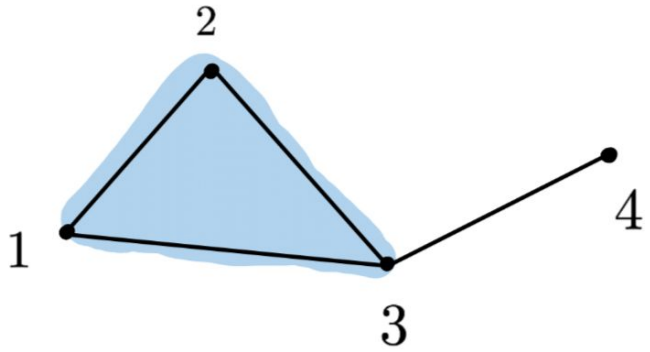
[1]: The de Rham–Hodge Analysis and Modeling of Biomolecules (Zhao et al., Bulletin of Mathematical Biology, 2020)

[2]: Random Walks on Simplicial Complexes and the normalized Hodge 1-Laplacian (Schaub et al., SIAM Review, 2020)

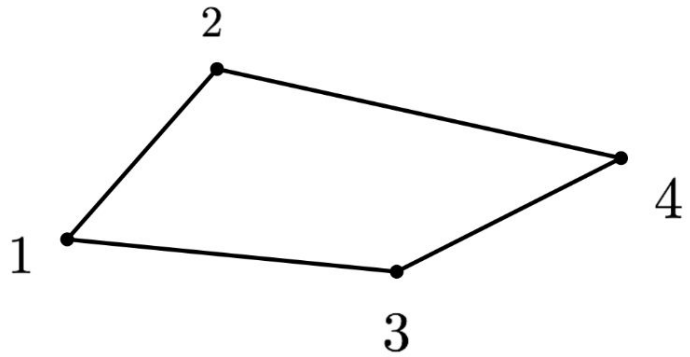
Simplicial Complexes

A **simplicial complex** on a vertex set ***V*** is a collection ***K*** of nonempty subsets of ***V*** (simplices) such that:

- it is closed under taking subsets
- contains all the singleton subsets of ***V***



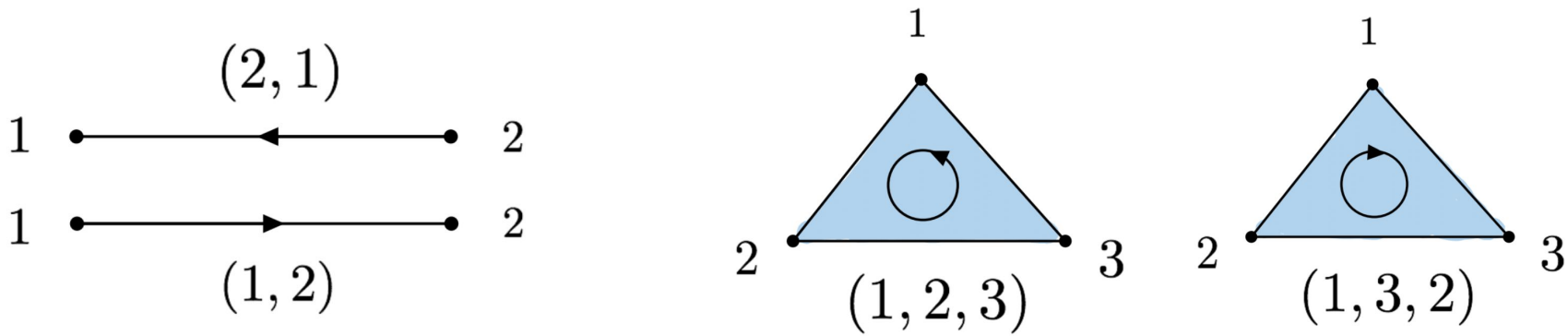
$K = \{1, 2, 3, 4, 12, 13, 23, 34, 123\}$



$K = \{1, 2, 3, 4, 12, 13, 24, 34\}$

Oriented Simplicial Complexes

For each simplex, we *can* choose an **orientation**, which can be positive or negative.



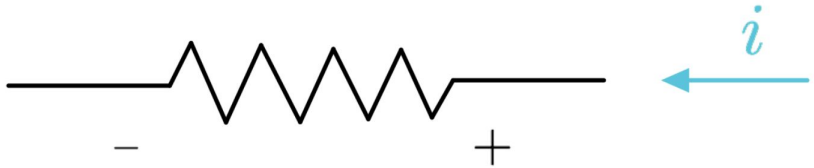
By representing simplices as ordered tuples, **even permutations** of the vertices result in a positive orientation, while **odd permutations** produce a negative orientation.

Oriented Simplicial Complexes

Why orientation? Because many signals in nature depend on an orientation:

$$\int_a^b f(x)\,dx = -\int_b^a f(x)\,dx$$

The direction of integration flips the sign

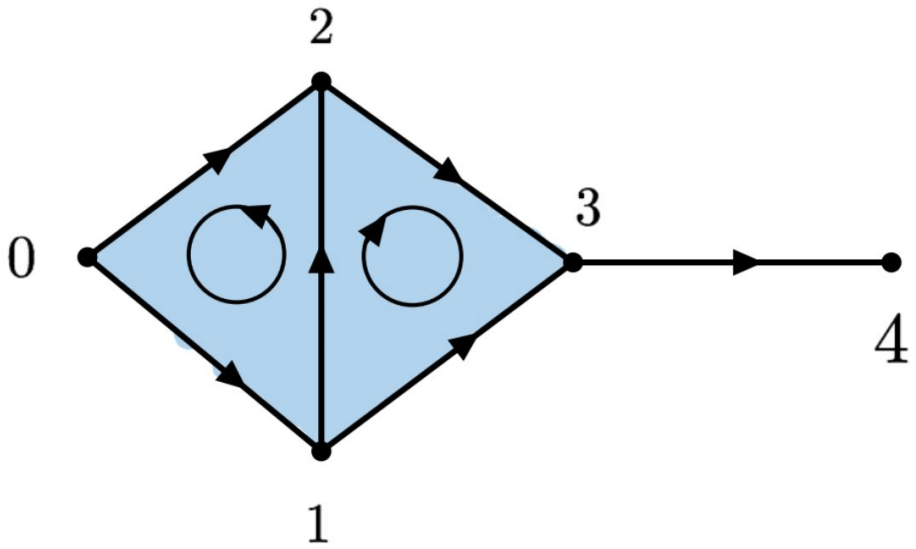


The sign convention in a circuit

Second, they are required to preserve more sophisticated connections with **differential geometry** and **algebraic topology**.

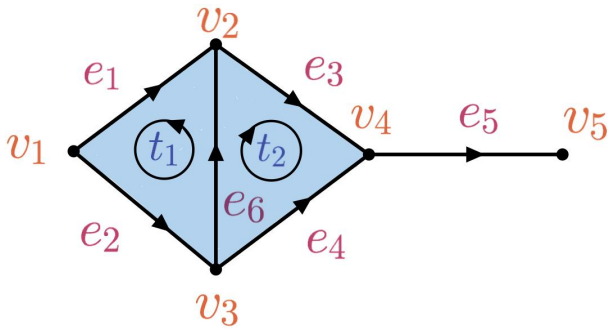
Oriented Simplicial Complexes

An **oriented simplicial complex** is a simplicial complex where each simplex has been assigned an orientation.



Boundary matrices

We can encode (signed) adjacencies with the help of (signed) **boundary matrices**.



$$B_1 = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 0 \\ +1 & 0 & -1 & 0 & 0 & +1 \\ 0 & +1 & 0 & -1 & 0 & -1 \\ 0 & 0 & +1 & +1 & -1 & 0 \\ 0 & 0 & 0 & 0 & +1 & 0 \end{bmatrix} \end{array}$$

$$B_2 = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{array} \begin{array}{c} \begin{array}{cc} t_1 & t_2 \end{array} \\ \begin{bmatrix} -1 & 0 \\ +1 & 0 \\ 0 & +1 \\ 0 & -1 \\ 0 & 0 \\ +1 & +1 \end{bmatrix} \end{array}$$

Hodge Laplacians

For each dimension of the complex, one can define a **Hodge Laplacian** [1, 2, 3, 4].

$$L_p = \underbrace{B_p^T B_p}_{L_p^{\downarrow}} + \underbrace{B_{p+1} B_{p+1}^T}_{L_p^{\uparrow}}$$

In dimension zero (i.e. for vertices), this becomes the well-known **Graph Laplacian**.

$$L_0 = 0^T 0 + B_1 B_1^T = D - A$$

[1]: Hodge Laplacian on Graphs (Lim et al., SIAM Review, 2015)

[2]: Topological Signal Processing over Simplicial Complexes (Sardellitti et al., Transactions on Signal Processing 2020)

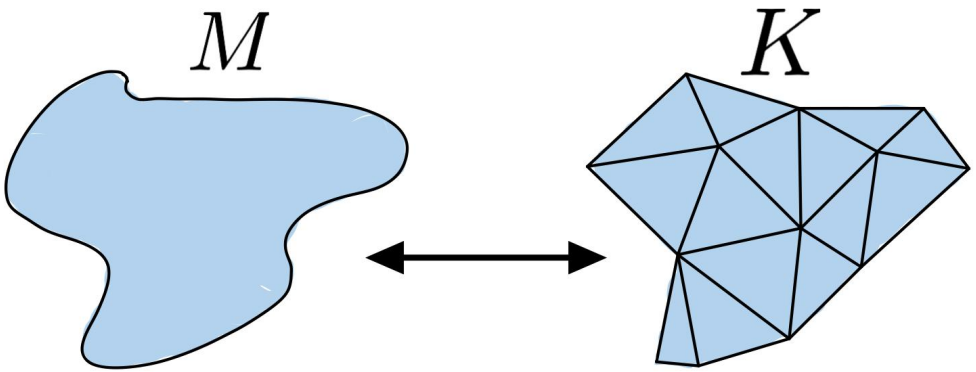
[3]: Random Walks on Simplicial Complexes and the normalized Hodge 1-Laplacian (Schaub et al., SIAM Review, 2020)

[4]: Control Using Higher Order Laplacians in Network Topologies (Muhammad et al., ISMTNS, 2006)

Extra: Differential Geometry & Algebraic Topology

Geometric view: Signals on oriented simplicial complexes can be seen as (vector-valued) **differential *k*-forms.** Transposed boundary matrices represent **discrete exterior derivatives**.

Topological view: Scalar signals are like the number of “oriented copies” of each simplex. Boundary matrices represent simplicial **boundary operators**.



CMU Course in **Discrete Differential Geometry**: https://brickisland.net/DDGSpring2021/

University of Oxford Course in **Computational Algebraic Topology**: http://people.maths.ox.ac.uk/nanda/cat/

Simplicial Complexes vs. Hypergraphs

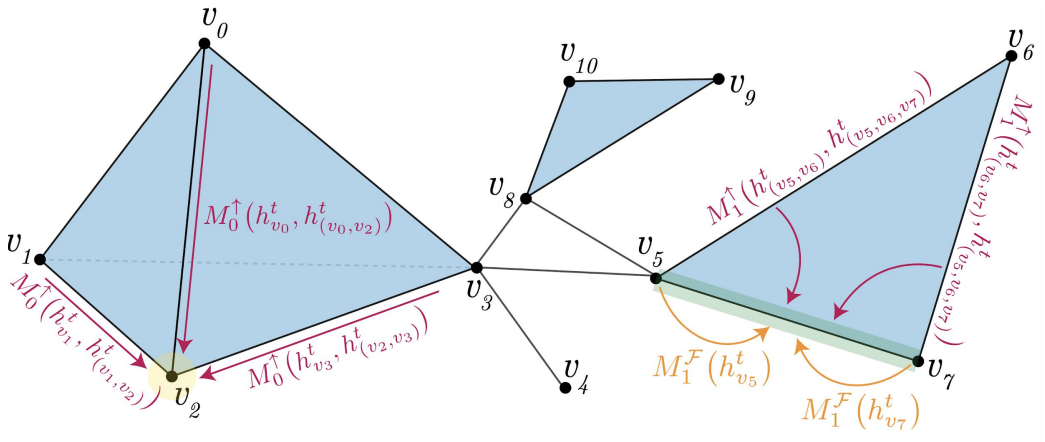
Simplicial Complexes

- Well understood spectral properties (Hodge Laplacian)
- Connections with differential geometry and algebraic topology
- Connection with physical diffusion processes
- Can easily handle interactions of different dimensions.
- Canonical representation: boundary matrices.

Hypergraphs

- The most general description of higher-order interactions
- No subset-inclusion constraint
  - suitable for certain groupwise interactions, e.g. protein complexes
- Hyperedges can (recursively) include other hyperedges

Message Passing Simplicial Networks



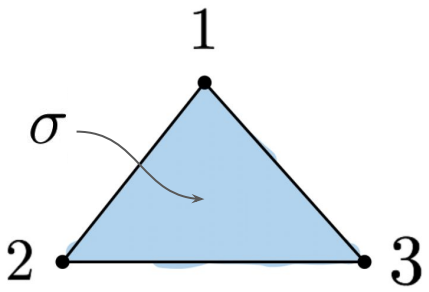
Simplicial Weisfeiler-Lehman

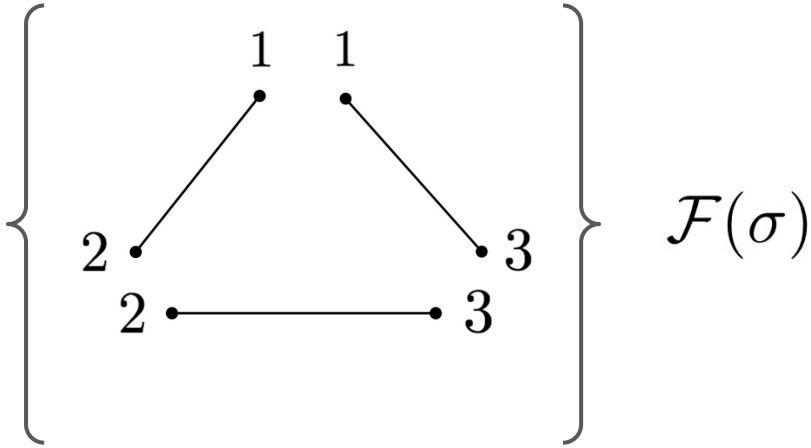
We want a theoretical tool to analyse message passing on simplicial complexes.

Let ***K*** be a simplicial complex. Simplicial WL (SWL) proceeds as follows:

1. Assign each simplex ***s*** in ***K*** an initial colour.
2. Compute the colour of ***s*** at the current time-step by hashing the colours of its neighbouring simplices.
3. Repeat for a finite number of steps or when the colours converge.

Four kinds of adjacencies: Face adjacencies



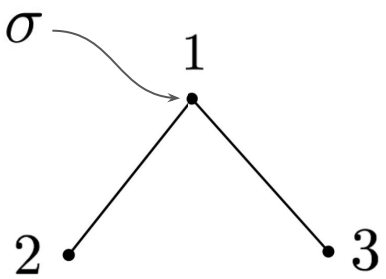


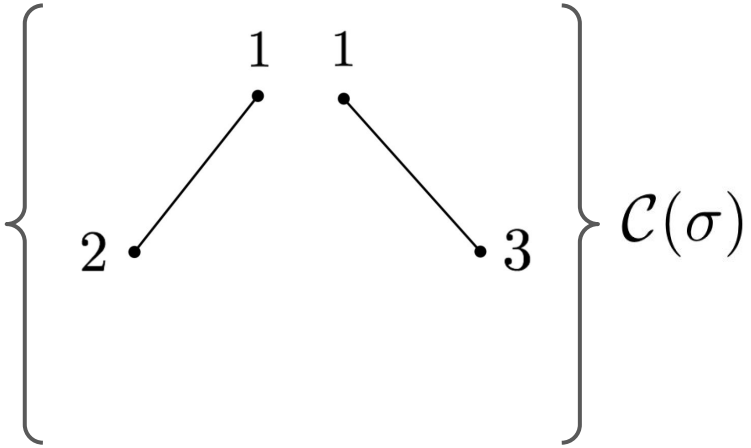
set of faces

$$c^t_{\mathcal{F}}(\sigma) = \underbrace{\{\{c^t_\omega | \omega \in \overbrace{\mathcal{F}(\sigma)}\}\}}$$

multiset of face colours

Four kinds of adjacencies: coface adjacencies

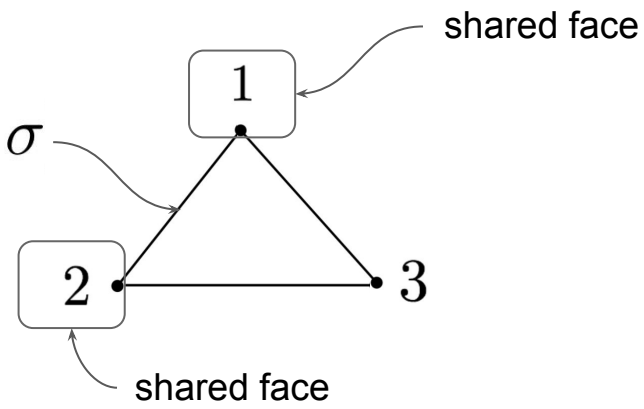


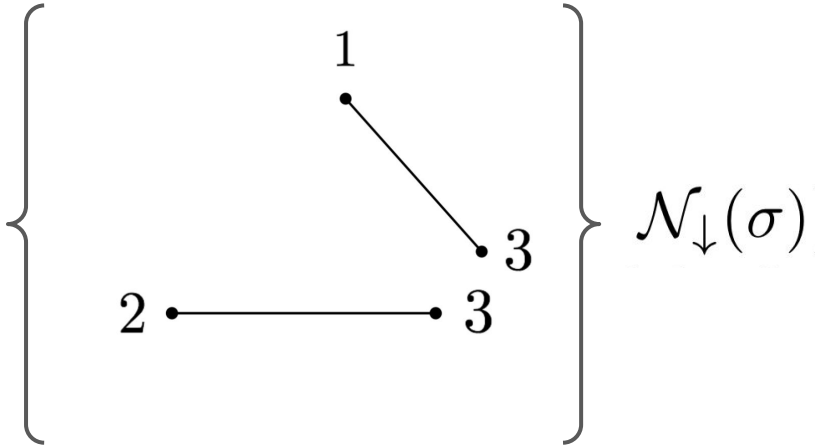


set of cofaces

$$c^t_{\mathcal{C}}(\sigma) = \underbrace{\{\!\{c^t_\omega | \omega \in \mathcal{C}(\sigma)\}\!\}}_{\text{multiset of coface colours}}$$

Four kinds of adjacencies: lower adjacencies



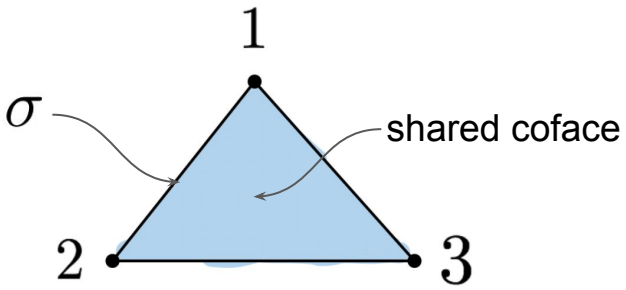


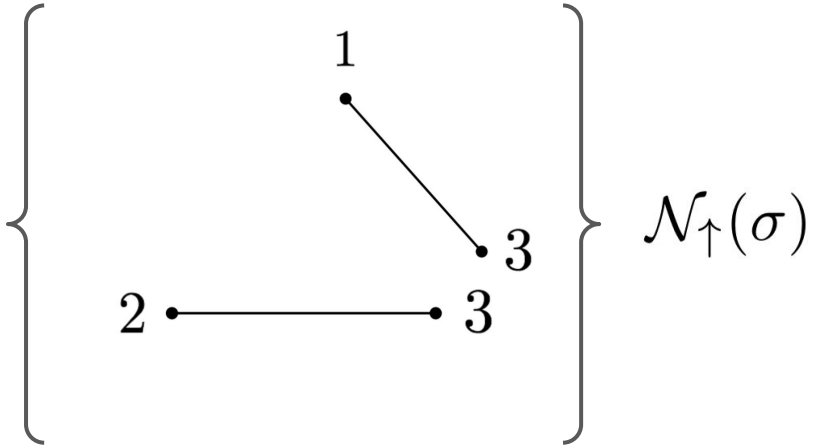
set of lower-neighbours

$$c^t_\downarrow(\sigma) = \underbrace{\{\!\{(c^t_\omega, c^t_{\sigma\cap\omega}) | \omega \in \mathcal{N}_\downarrow(\sigma)\}\!\}}_{\text{multiset of lower-neighbours colour-tuples}}$$

Two *d*-simplices are lower adjacent if they share a common face of dimension *d-1*

Four kinds of adjacencies: upper adjacencies





set of upper-neighbours

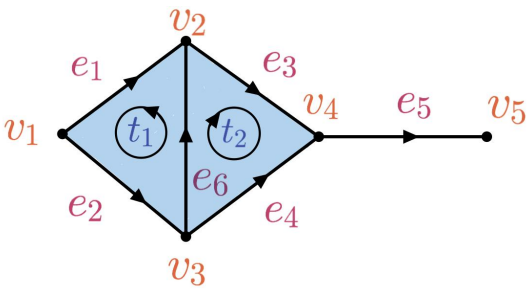
$$c^t_\uparrow(\sigma) = \underbrace{\{\{(c^t_\omega, c^t_{\sigma\cup\omega}) | \omega \in \mathcal{N}_\uparrow(\sigma)\}\}}_{\text{multiset of upper-neighbours colour-tuples}}$$

Two *d*-simplices are upper adjacent if they share a common coface of dimension *d+1*

All adjacencies

In terms of the boundary matrices, adjacencies for the ***i***-simplices are given by:

- Face & Coface adjacencies → The non zero elements of B_i, B_{i+1}
- Lower adjacencies → The non-zero entries of $B_i^T B_i$
- Upper adjacencies → The non-zero entries of $B_{i+1} B_{i+1}^T$

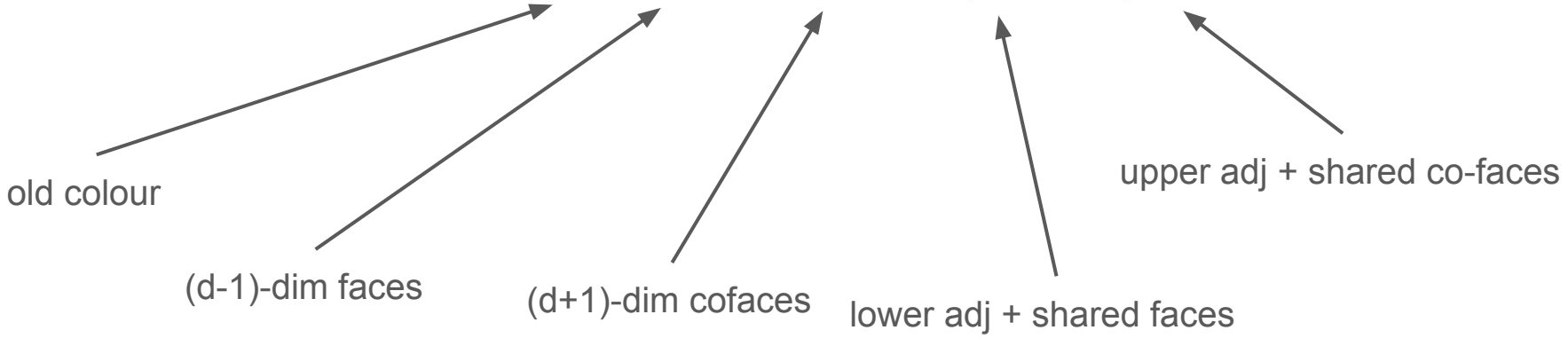


$$B_2^T B_2 = \begin{array}{c} \\ t_1 \\ t_2 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & +1 & 0 & 0 & 0 & +1 \\ 0 & 0 & +1 & -1 & 0 & +1 \end{bmatrix} \end{array} \begin{array}{c} \begin{array}{cc} t_1 & t_2 \end{array} \\ \begin{array}{c} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{array} \begin{bmatrix} -1 & 0 \\ +1 & 0 \\ 0 & +1 \\ 0 & -1 \\ 0 & 0 \\ +1 & +1 \end{bmatrix} \end{array} = \begin{array}{c} \\ t_1 \\ t_2 \end{array} \begin{array}{c} \begin{array}{cc} t_1 & t_2 \end{array} \\ \begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix} \end{array}$$

SWL: Putting everything together

For maximum expressivity, we include all adjacencies in the colour refinement rule:

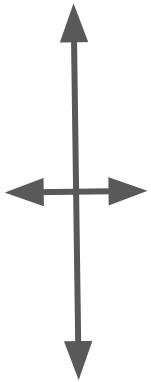
$$c_\sigma^{t+1} = \mathrm{HASH}\{c_\sigma^t, c_{\mathcal{F}}^t(\sigma), c_{\mathcal{C}}^t(\sigma), c_{\downarrow}^t(\sigma), c_{\uparrow}^t(\sigma)\}$$

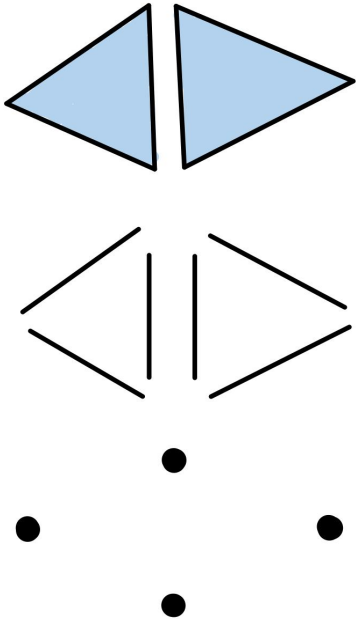


Understanding adjacencies

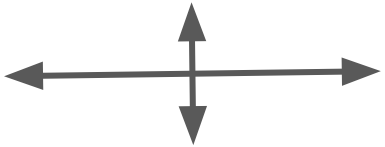
Signals propagate at different rates in the horizontal and vertical directions.

Face & Coface Adjacencies





Lower & Upper Adjacencies



Pruning adjacencies

SWL with colour refinement rule

$$c_v^{t+1} = \mathrm{HASH}\big(c_v^t, c_{\mathcal{F}}^t(v), c_{\uparrow}^t(v)\big)$$

is **as powerful as** SWL with the generalised one [Lemma1, Theorem 2].

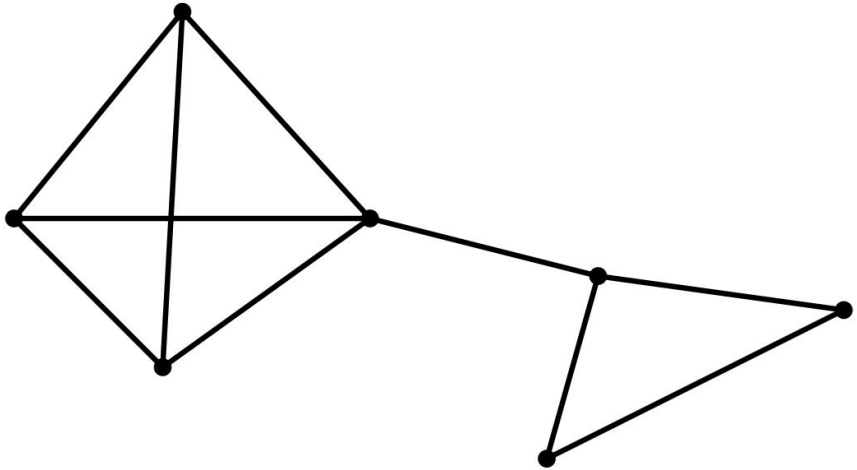
$\Rightarrow$ *we can "drop" coface- and lower-adjacencies with no impact on the expressivity of the test*

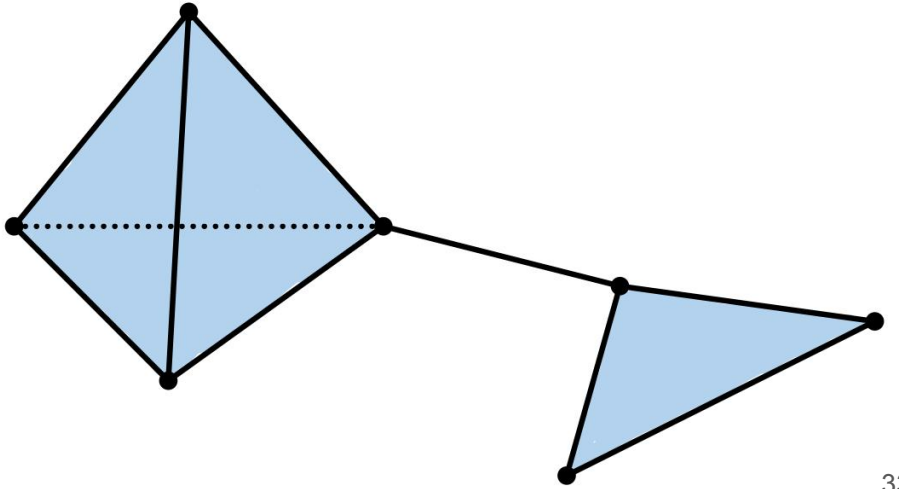
Why these specific adjacencies?

- SWL corresponds to WL when applied on 1-complexes (graphs)
- the complexity remains linear in the size of the complex (more on this later)

Clique Complexes

Given graph ***G***, its associated Clique Complex $\boldsymbol{K_G}$ is the Simplicial Complex obtained by considering each *d*-clique in ***G*** as a (d-1)-simplex in $\boldsymbol{K_G}$.





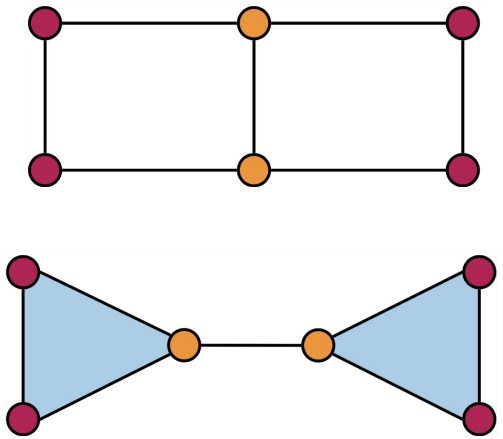
SWL is strictly more powerful than WL

1) SWL is at least as powerful as WL [Lemma 22]:

If WL distinguishes two non-isomorphic graphs, SWL does as well!

2) SWL distinguishes pairs which WL does not, when run on the corresponding clique-complexes.

⇒ SWL is **strictly more powerful** than WL [Theorem 3]



Pair of non-isomorphic graphs distinguished by SWL, but not by WL.

Higher-order WL test

The k-WL algorithm initialises *all* node-k tuples with colours based on iso-class.

It iterates *colour refinements* applied on *all* node k-tuples in the graphs:

$$c_{\mathbf{v}}^{t+1} = \mathrm{HASH}\Big(c_{\mathbf{v}}^{t}, M^{t}(\mathbf{v})\Big)$$

$$M^{t}(\mathbf{v}) = \Big(\{\!\{c_{\mathbf{u}}^{t} | \mathbf{u} \in \mathcal{N}_j(\mathbf{v})\}\!\} \Big| j = 1, 2, \ldots, k\Big)$$

$$\mathcal{N}_j(\mathbf{v}) = \{(v_1, \ldots, v_{j-1}, w, v_{j+1}, \ldots, v_k) | w \in \mathcal{V}_G\}$$

Extended notion of neighbourhood

For k≥2, (k+1)-WL is **strictly stronger** than k-WL:

There exist pairs non-isomorphic graphs that k-WL cannot distinguish while (k+1)-WL can (not vice-versa)

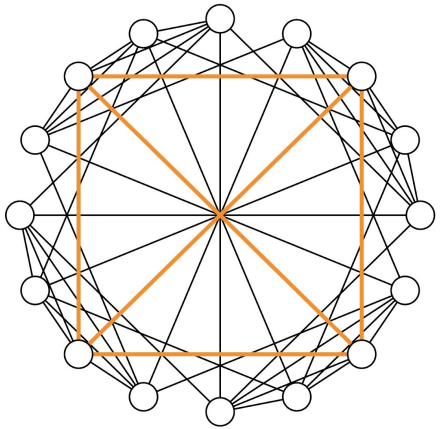
SWL is not less powerful than 3-WL

1) 3-WL provably fails to distinguish between any pair of Strongly Regular Graph in the same family [Lemma 23]

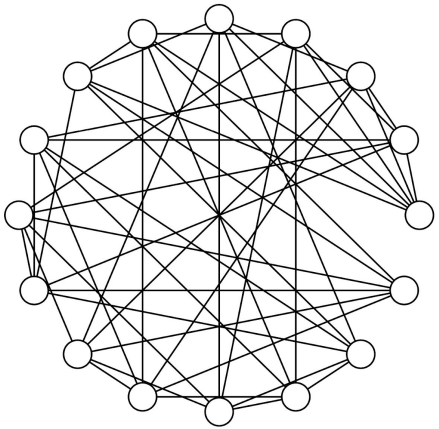
2) SWL can distinguish the two SR Graphs in family *SR(16,6,2,2)*

⇒ SWL is **not less powerful** than 3-WL [Theorem 4]

Rook's 4x4



Shrikhande



Pair of non-isomorphic SR graphs distinguished by SWL, but not by 3-WL.

Family: *SR(16,6,2,2)*; they are associated with *distinct clique complexes*.

The neural counterpart of SWL: MPSNs

$$m_{\mathcal{F}}^{t+1}(v) = \mathrm{AGG}_{w \in \mathcal{F}(v)} \left(M_{\mathcal{F}}\left(h_v^t, h_w^t\right) \right)$$

$$m_{\mathcal{C}}^{t+1}(v) = \mathrm{AGG}_{w \in \mathcal{C}(v)} \left(M_{\mathcal{C}}\left(h_v^t, h_w^t\right) \right)$$

$$m_{\downarrow}^{t+1}(v) = \mathrm{AGG}_{w \in \mathcal{N}_{\downarrow}(v)} \left(M_{\downarrow}\left(h_v^t, h_w^t, h_{v \cap w}^t\right) \right)$$

$$m_{\uparrow}^{t+1}(v) = \mathrm{AGG}_{w \in \mathcal{N}_{\uparrow}(v)} \left(M_{\uparrow}\left(h_v^t, h_w^t, h_{v \cup w}^t\right) \right)$$

} **Message & Aggregate**

$$h_v^{t+1} = U\left(h_v^t, m_{\mathcal{F}}^t(v), m_{\mathcal{C}}^t(v), m_{\downarrow}^{t+1}(v), m_{\uparrow}^{t+1}(v)\right)$$

} **Update**

$$h_G = \mathrm{READOUT}(\{\!\{h_v^L\}\!\}_{v \in \mathcal{K}_0}, \ldots, \{\!\{h_v^L\}\!\}_{v \in \mathcal{K}_p})$$

} **Readout**

MPSNs can be made as powerful as SWL

1) MPSNs are at most as powerful as SWL [Lemma 5]

2) MPSNs with a sufficient number of layers and injective *message*, *aggregate* and *update* functions are **as powerful as SWL** [Theorem 6]

⇒ There exists an MPSN **more powerful than WL** at distinguishing non-isomorphic *graphs* when using a clique-complex lifting because such parametric functions exist [1, 2] [Corollary 7]

[1]: How Powerful are Graph Neural Networks? (Xu et al., ICLR 2019)

[2]: Principal Neighbourhood Aggregation for Graph Nets (Corso et al., NeurIPS 2020)

Other (recent) simplicial neural networks

SNN [1], similar to ChebNet [3]

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

Simplicial CNN [2], similar to GCN [4]

$$\begin{aligned}
X_0^{t+1} &= \sigma(D_1^{-1} B_1 X_1^t W_{0,1}^t + \tilde{A}_0^u X_0^t W_{0,0}^t) \\
X_1^{t+1} &= \sigma(B_2 D_3 X_2^t W_{1,2}^t \\
&\quad + (\tilde{A}_1^d + \tilde{A}_1^u) X_1^t W_{1,1}^t \\
&\quad + D_2 B_1^T D_1^{-1} X_0^t W_{1,0}^t) \\
X_2^{t+1} &= \tilde{A}_2^d X_2^t W_{2,1}^t + D_4 B_2^T D_5^{-1} X_1^t W_{2,0}^t
\end{aligned}$$

The MPSN framework **generalises both** approaches [Theorem 8]

[1]: Simplicial Neural Networks (Ebli et al., TDA and Beyond NeurIPS 2020 Workshop)

[2]: Simplicial 2-Complex Convolutional Neural Networks (Bunch et al., TDA and Beyond NeurIPS 2020 Workshop)

[3]: Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering (Defferrard et al., NIPS 2016)

[4]: Semi-Supervised Classification with Graph Convolutional Networks (Kipf et al., ICLR 2017)

Permutation equivariance

GNN layers are **(node) permutation equivariant**:

$$f(\underbrace{PAP^T}_{\text{Adjacencies}}, \underbrace{PX}_{\text{Features}}) = Pf(A, X)$$

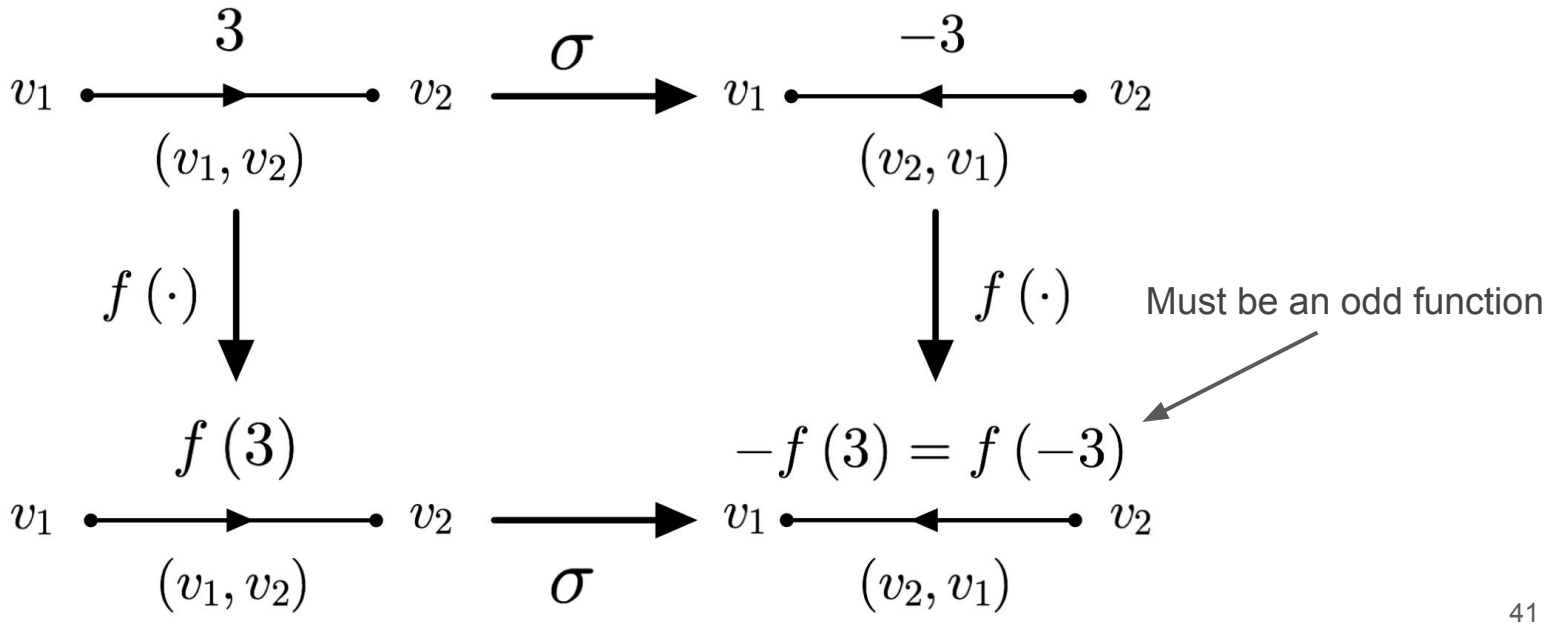
Theorem 9. MPSN layers are **(simplex) permutation equivariant**. For each dimension i:

$$f_i(\underbrace{P_{i-1}B_iP_i^T, P_iB_{i+1}P_{i+1}^T}_{\text{Adjacencies}}, \underbrace{P_{i-1}X_{i-1}, P_iX_i, P_{i+1}X_{i+1}}_{\text{Features}})$$

$$= P_i f_i(B_i, B_{i+1}, X_{i-1}, X_i, X_{i+1})$$

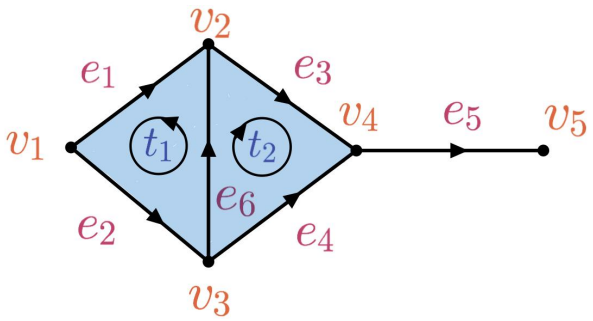
Orientation equivariance: Functions

Mathematically, **the choice of orientation is arbitrary** and, therefore, irrelevant. If orientation is changed, we would like activations to be the same up to a change in orientation.

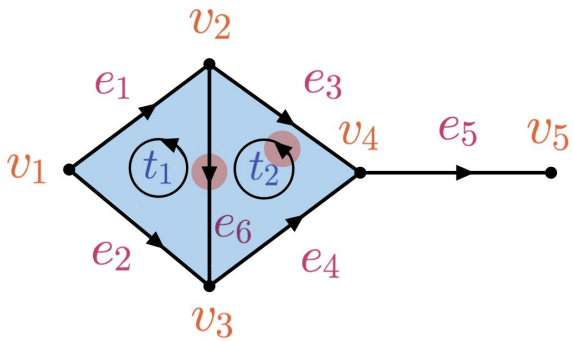


Orientation equivariance: Structure

Changes in orientation amounts to multiplying the corresponding rows and columns by +/- 1.



$$B_1 = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 0 \\ +1 & 0 & -1 & 0 & 0 & +1 \\ 0 & +1 & 0 & -1 & 0 & -1 \\ 0 & 0 & +1 & +1 & -1 & 0 \\ 0 & 0 & 0 & 0 & +1 & 0 \end{bmatrix} \end{array} \qquad B_2 = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{array} \begin{array}{c} \begin{array}{cc} t_1 & t_2 \end{array} \\ \begin{bmatrix} -1 & 0 \\ +1 & 0 \\ 0 & +1 \\ 0 & -1 \\ 0 & 0 \\ +1 & +1 \end{bmatrix} \end{array}$$



$$B_1 = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \begin{array}{c} \begin{array}{cccccc} e_1 & e_2 & e_3 & e_4 & e_5 & e_6 \end{array} \\ \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 0 \\ +1 & 0 & -1 & 0 & 0 & -1 \\ 0 & +1 & 0 & -1 & 0 & +1 \\ 0 & 0 & +1 & +1 & -1 & 0 \\ 0 & 0 & 0 & 0 & +1 & 0 \end{bmatrix} \end{array} \qquad B_2 = \begin{array}{c} \\ e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{array} \begin{array}{c} \begin{array}{cc} t_1 & t_2 \end{array} \\ \begin{bmatrix} -1 & 0 \\ +1 & 0 \\ 0 & -1 \\ 0 & +1 \\ 0 & 0 \\ -1 & +1 \end{bmatrix} \end{array}$$

Orientation equivariance

For each dimension i, we define the following matrices of size $\mathcal{S}_i \times \mathcal{S}_i$

$$\sigma_i = diag(\pm 1, \ldots, \pm 1)$$

MPSN layers are **orientation equivariant** if for each dimension i:

$$f_i(\underbrace{\sigma_{i-1} B_i \sigma_i, \sigma_i B_{i+1} \sigma_{i+1}}_{\text{Adjacencies}}, \underbrace{\sigma_{i-1} X_{i-1}, \sigma_i X_i, \sigma_{i+1} X_{i+1}}_{\text{Features}})$$

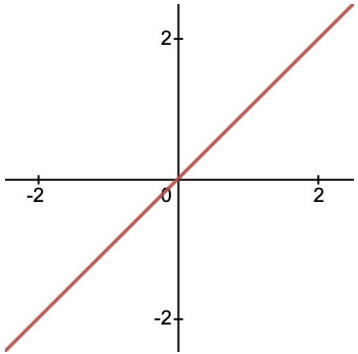
$$= \sigma_i f_i(B_i, B_{i+1}, X_{i-1}, X_i, X_{i+1})$$

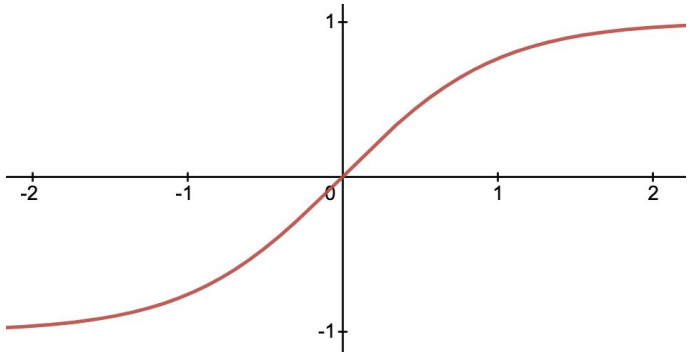
Orientation equivariance for a simple MPSN layer

Consider a simple MPSN layer using **linear transformations + a non-linear activation**.

$$\psi\Big(\underbrace{B_i^T B_i X_i W_1}_{\text{Lower adj.}} + \underbrace{X_i W_2}_{\text{Just features}} + \underbrace{B_{i+1} B_{i+1}^T X_i W_3}_{\text{Upper adj.}} + \underbrace{B_i^T X_{i-1} W_4}_{\text{Faces}} + \underbrace{B_{i+1} X_{i+1} W_5}_{\text{Co-faces}}\Big)$$

Proposition 30. This layer is orientation equivariant iff ψ is an odd function (e.g. id, tanh).





The expressivity power of MPSNs by linear regions

Linear regions of a piecewise linear function are subdivisions of the graph of the function, or the collection of all individual connected regions in the real space with hyperplanes determined by the piecewise linear functions removed.

- Number of linear regions can show the distinction between traditional shallow and deep neural networks
- Related to approximation property of the networks
- Representational power of convolutional neural networks

[1]: On the number of linear regions of deep neural networks? (Montufar et al., NIPS 2014)

[2]: On the number of linear regions of convolutional neural networks (Xiong et al., ICML 2020)

Linear regions of GNNs and SCNNs

One-layer GNN

$$X^{\text{out}} = \psi\left(\mathcal{H}(A, X^{\text{in}})W_0\right)$$

$$R_{\text{GNN}} = \left(2\sum_{i=0}^{d-1}\binom{m-1}{i}\right)^{S_0}$$

The aggregation needs linear and invertible.
S_0 is number of graph nodes
d is the dimension of feature
m is the number of output features
Activated by ReLU ψ

One-layer SCNN

$$H_n^{\text{out}} = \psi\left(M_n H_n^{\text{in}} W_n\right)$$

$$R_{\text{SCNN}} = \prod_{n=0}^{p}\left(2\sum_{i=0}^{d_n-1}\binom{m_n-1}{i}\right)^{S_n}$$

The simplicial Laplacian needs invertible.
S_n is number of n-simplices
d_n is the dimension of input feature for n-sim.
$m_n$is the number of output features
Activated by ReLU ψ

MPSN Revisit

$$h^{\text{out}}_{n,v} = \psi\Big(\sum_{w \in \mathcal{S}_n} M_{n,v,w} h^{\text{in}}_{n,w} W_n + \sum_{w \in \mathcal{S}_{n-1}} U_{n,v,w} h^{\text{in}}_{n-1,w} W_{n-1} + \sum_{w \in \mathcal{S}_{n+1}} O_{n,v,w} h^{\text{in}}_{n+1,w} W_{n+1} \Big)$$

$$H^{\text{out}} = \psi(W H^{\text{in}})$$

$$W = \begin{bmatrix} W_0^\top \otimes M_0 & W_1^\top \otimes O_0 & & \\ W_0^\top \otimes U_1 & W_1^\top \otimes M_1 & W_2^\top \otimes O_1 & \\ & W_1^\top \otimes U_2 & W_2^\top \otimes M_2 & W_3^\top \otimes O_2 \\ & & & \ddots \end{bmatrix}$$

$$H^{\text{in}} = \text{vec}([H_0^{\text{in}} | H_1^{\text{in}} | \cdots | H_p^{\text{in}}]) \in \mathbb{R}^N$$

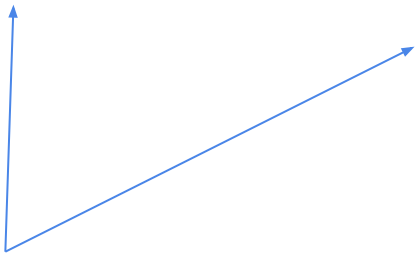
$$H^{\text{out}} = \text{vec}([H_0^{\text{out}} | H_1^{\text{out}} | \cdots | H_p^{\text{out}}]) \in \mathbb{R}^M$$

$$N = \sum_{n=0}^{p} S_n d_n \qquad M = \sum_{n=0}^{p} S_n m$$

Linear regions of MPSNs

Upper Bound

$$R_{\text{MPSN}} \leq \prod_{n=0}^{p} \left(2 \sum_{i=0}^{d_{n-1}+d_n+d_{n+1}-1} \binom{m-1}{i} \right)^{S_n}$$



$$r(\mathcal{A}) = \sum_{B \subseteq \{1,\ldots,M\}} (-1)^{|B| - \text{rank}(W_{B:})}$$

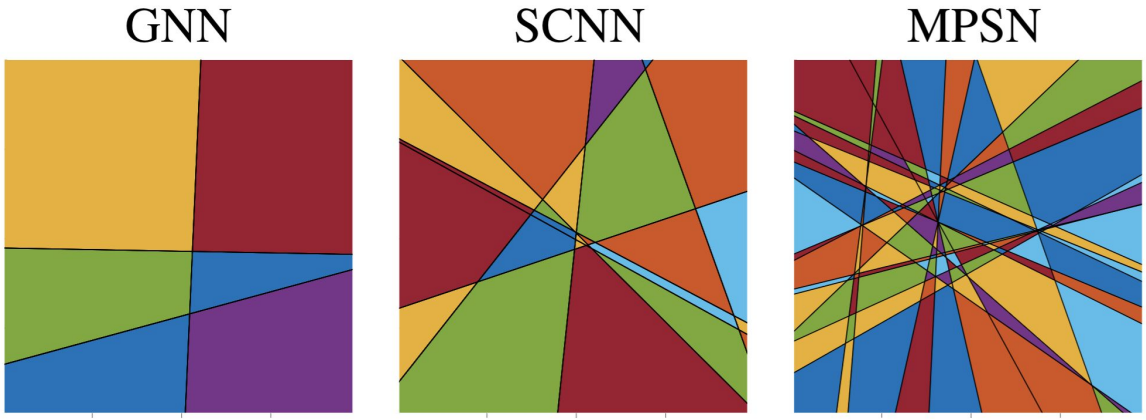
Lower Bound vs Trivial Upper Bound

$$R_{\text{MPSN}} \leq 2 \sum_{j=0}^{N-1} \binom{M-1}{j}$$

$$R_{\text{MPSN}} \geq 2 \sum_{j=0}^{N-(mS)-1} \binom{M-1}{j}$$

MPSNs attain better bounds than SNNs and GNNs

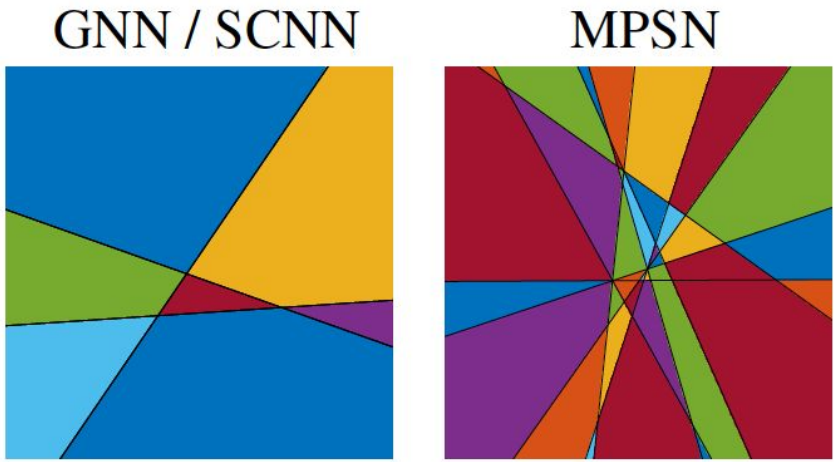
A 2D slice of the input feature spaces of GNN, SCNN, MPSN layers with S0 = S1 = 3, S2 = 1 (the complex is a triangle), d0 = d1 = d2 = 1, m = 3, colored by linear regions of the represented functions, for a random choice of the weights.



Linear regions with populated higher-order features

$$R_{\text{MPSN}} \geq R_{\text{SCNN}}$$

$$\psi(W' \operatorname{vec}(H_0^{\text{in}}))$$



MPSN complexity is linear in the size of the complex

Message passing: $\Theta\big(\sum_{d=0}^{p}\underbrace{(d+1)S_d}_{\text{Faces}}+\underbrace{\binom{d+1}{2}S_d}_{\text{Upper neighbors}}\big) \rightarrow \Theta\big(\sum_{d=0}^{p} S_d\big)$

Complex dimension

d-Simplices in the complex

(small p)

Caveat: may get *quadratic* when considering down adjacencies

The clique-lifting procedure is *empirically* tractable

- Finding all the *maximal cliques* in a graph has worst case complexity $\mathcal{O}(3^{n/3})$
- Finding cliques of a specified max size in (sparse) real-world graphs is significantly **faster in practice**.
- Simplex trees, from the GUDHI topological data analysis library (empirically) scale linearly with the number of simplices.
- Theoretical guarantees for the number of simplices exist when the max degree of the graph is known.
- Performance is expected to be significantly better than k-WL's $\Omega(n^k)$ since the number of *k*-cliques in a graph is $\mathcal{O}(n^k)$

Results

Simplicial Isomorphism Networks

The MPSN counterpart of GIN [1]:

$$h_v^{t+1} = \mathrm{MLP}_U^t\Big(\mathrm{MLP}^t\big((1+\epsilon)h_v^t + \sum_{w\in\mathcal{F}(v)} h_w^t\big) \,\|\, \mathrm{MLP}^t\big((1+\epsilon)h_v^t + \sum_{w\in\mathcal{N}(v)} M_{\uparrow}^t(h_w^t, h_{v\cup w}^t)\big)\Big)$$

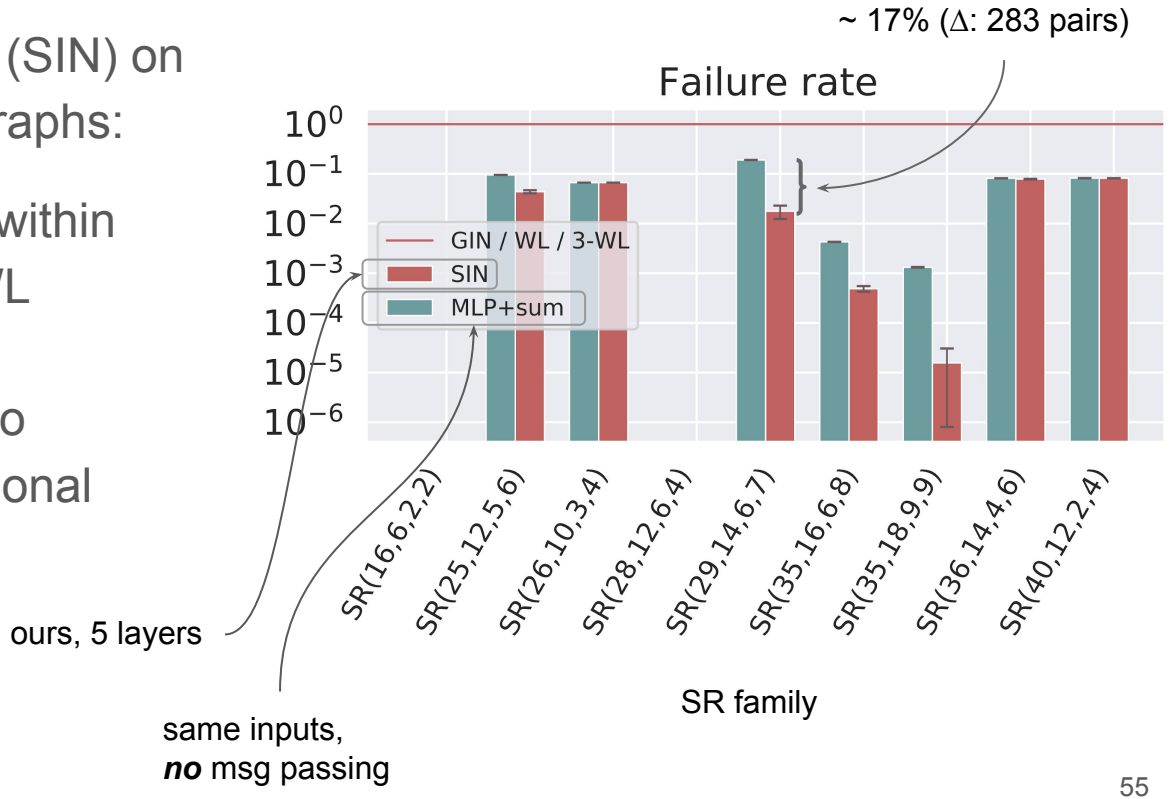
$$M_{\uparrow}^t(h_w^t, h_{v\cup w}^t) = \mathrm{MLP}_M^t\big(h_w^t \,\|\, h_{v\cup w}^t\big) \tag{18}$$

[1]: How Powerful are Graph Neural Networks? (Xu et al., ICLR 2019)

Disambiguating Strongly Regular Graphs

We run an MPSN architecture (SIN) on the clique-complexes of SR Graphs:

- SIN can distinguish pairs within the same family (WL, 3-WL provably fail on all)
- Message passing allows to distinguish between additional pairs

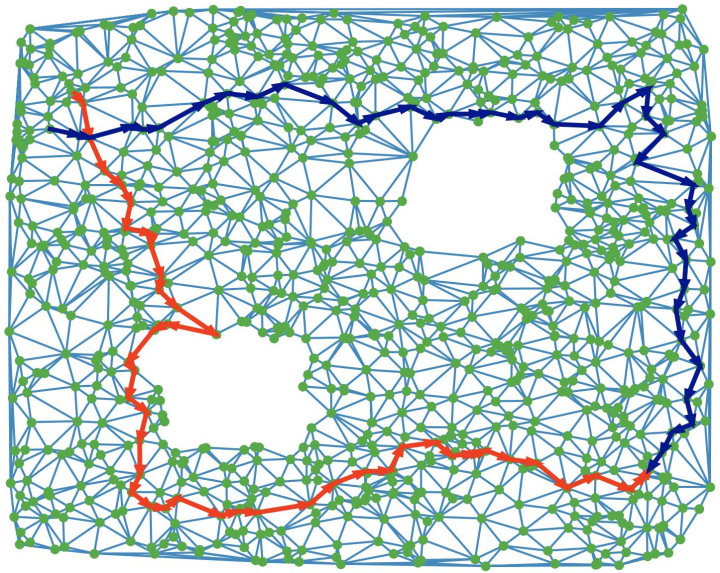


Classifying real-world graphs (TUDatasets)

Dataset	Proteins	NCI1	IMDB-B	IMDB-M	RDT-B	RDT-M5K
Graphs	1113	4110	1000	1500	2000	5000
Classes	2	2	2	3	2	5
Avg. #Nodes	39.1	29.8	19.8	13.0	429.6	508.5
Avg. #Triangles	27.4	0.05	392.0	305.9	24.8	21.8
Median #Triangles	21.0	0.0	119.5	56.0	11.0	11.0
RWK (Gärtner et al., 2003)	59.6±0.1	>3 days	N/A	N/A	N/A	N/A
GK (k=3) (Shervashidze et al., 2009)	71.4±0.31	62.5±0.3	N/A	N/A	N/A	N/A
PK (Neumann et al., 2016)	73.7±0.7	82.5±0.5	N/A	N/A	N/A	N/A
WL kernel (Shervashidze et al., 2011)	75.0±3.1	86.0±1.8	73.8±3.9	50.9±3.8	81.0±3.1	52.5±2.1
DCNN (Atwood & Towsley, 2016)	61.3±1.6	56.6±1.0	49.1±1.4	33.5±1.4	N/A	N/A
DGCNN (Zhang et al., 2018)	75.5±0.9	74.4±0.5	70.0±0.9	47.8±0.9	N/A	N/A
IGN (Maron et al., 2018)	76.6±5.5	74.3±2.7	72.0±5.5	48.7±3.4	N/A	N/A
GIN (Xu et al., 2019b)	76.2±2.8	82.7±1.7	75.1±5.1	52.3±2.8	92.4±2.5	57.5±1.5
PPGNs (Maron et al., 2019)	77.2±4.7	83.2±1.1	73.0±5.8	50.5±3.6	N/A	N/A
Natural GN (de Haan et al., 2020)	71.7±1.0	82.4±1.3	73.5±2.0	51.3±1.5	N/A	N/A
SIN (Ours)	76.4 ± 3.3	82.7 ± 2.1	75.6 ± 3.2	52.4 ± 2.9	92.2 ± 1.0	57.3 ± 1.6

Classifying synthetic trajectories

Task: distinguish between trajectories above the upper hole and below the lower hole.



The task is adapted from [1]

4x $\left\{ H_1^{\text{out}} = \left| B_1^T B_1 \right| H_i^{\text{in}} W_1 + H_1^{\text{in}} W_2 \right.$

Model	*Test Accuracy*
GNN	71.5 %
MPSN	96.5 %

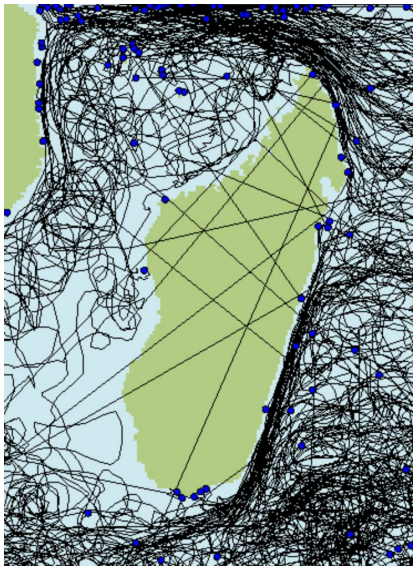
Triangle awareness

4 x $\left\{ H_1^{\text{out}} = B_1^T B_1 H_i^{\text{in}} W_1 + H_1^{\text{in}} W_2 + \boxed{B_2 B_2^T H_1^{\text{in}} W_3} \right.$

[1]: Random Walks on Simplicial Complexes and the normalized Hodge 1-Laplacian (Schaub et al., SIAM Review, 2020)

Classifying ocean drifter trajectories

Task: distinguish between clockwise and counterclockwise drifter trajectories around Madagascar (2011-2018).



The task is adapted from [1]

4x $H_1^{\text{out}} = \left|B_1^T B_1\right| H_i^{\text{in}} W_1 + H_1^{\text{in}} W_2$

Model	*Test Accuracy*
GNN	45 %
MPSN	75 %

4x $H_1^{\text{out}} = B_1^T B_1 H_i^{\text{in}} W_1 + H_1^{\text{in}} W_2 + \boxed{B_2 B_2^T H_1^{\text{in}} W_3}$

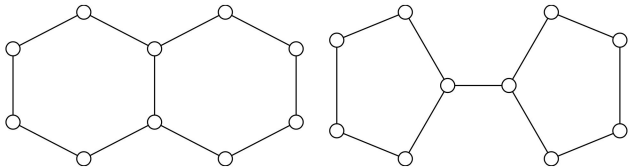
[1]: Random Walks on Simplicial Complexes and the normalized Hodge 1-Laplacian (Schaub et al., SIAM Review, 2020)

Conclusions

- We introduce a **colouring algorithm** for SC isomorphism testing: SWL
- SWL inspires a **neural message passing** framework on SCs (MPSNs)
- SWL and MPSNs on **clique complexes** are provably **more expressive** than WL
- Allowing features of different dimensions to interact increases the **number of linear regions**: GNNs → SCNNs → MPSNs
- Experiments on synthetic and real world datasets confirm:
  - expressiveness of MPSNs
  - additional advantages from higher-order interactions

Limitations

- Clique complexes do not help with certain graphs. Other graph-lifting procedures might be able to address this.
- Simplices might not perfectly reflect the notion of a “cell”. For instance, cubical complexes might be better suited for a map of Manhattan.
- Signals on higher-order objects might not exist and do not always have a physical correspondence.



Pair of non-isomorphic molecular graphs that cannot be disambiguated by either the SWL or the WL test.

Pair of non-isomorphic cubical complexes that cannot be disambiguated by the WL test.

Open Problems

- Other message passing schemes that retain the expressive power
- Better understanding of the advantages of SWL
- What is the best way to construct simplicial complexes from data?
- What is the best way to construct higher-order features?
- What are the differences between orientation-equivariant MPSNs and regular MPSNs in GNN benchmarks?
- The resurrection of edge features?
- How can MPSNs unify existent work on triangular meshes and graphs?
- More theoretical analysis using differential geometry & topology.
- What GNN results can we generalise to MPSNs?

Thank you!

Cristian Bodnar

Twitter: @CristianBodnar

Email: cb2015@cam.ac.uk

Fabrizio Frasca

Twitter: @ffabffrasca

Email: ffrasca@twitter.com

Yu Guang Wang

Twitter: @wangyg85

yuguang.wang@mis.mpg.de