Supplement to "Principled Simplicial Neural Networks for Trajectory Prediction"

T. Mitchell Roddenberry [*1] **Nicholas Glaze** [*1] **Santiago Segarra** [1]

In this supplement, we discuss practical concerns for the implementation of simplicial neural networks. In Section 1, we discuss how to represent chains as real vectors and boundary maps as matrices, as well as how to implement activation functions using this representation. Then, using these convenient representations, in Section 2 we redefine SCoNe using real vectors and matrices, rather than vectors and linear maps in the chain complex $\mathcal{C}$, and then specify the hyperparameters used in our experiments. We briefly discuss the implementation of the nullspace projection methods in Section 3, followed by a computational complexity analysis of SCoNe in Section 4. The necessity of odd, nonlinear activation functions as stated in Proposition 1 is proven in Section 5. The use of odd activation functions is contrasted with existing work in Section 6. We provide more details on Theorem 1 as it pertains to $\mathcal{C}_1$ in Section 7, before finally discussing an implementation of SCoNe for cubical complexes in Section 8.

1. Representing Chains and Boundary Maps as Vectors and Matrices

In Algorithm 1, we specify SCoNe in terms of boundary maps acting on chains interleaved with matrix multiplication from the right and activation functions. Here, we describe simple procedures for constructing vector representations of $k-$chains, as well as matrix representations of boundary maps that act on said representations.

Let $\mathcal{X}$ be a simplicial complex over a set of nodes $\mathcal{X}_0$, with edges $\mathcal{X}_1$ and triangles $\mathcal{X}_2$. Begin by labeling the vertices $\mathcal{X}_0$ with the integers $\{1, \ldots, n\}$. Letting $m = |\mathcal{X}_1|$, sort the edges lexicographically by their constituent nodes and label them accordingly with the integers $\{1, \ldots, m\}$. Similarly, letting $p = |\mathcal{X}_2|$, sort the triangles lexicographically by their constituent nodes and label them with the integers $\{1, \ldots, p\}$. For each edge $e = \{i, j\}$, for $i, j \in \mathcal{X}_0$, assign to e the orientation $[i, j], i < j$. Similarly, for each triangle $t = \{i, j, k\}$, assign to t the orientation $[i, j, k], i < j < k$.

1.1. Chains as Real Vectors

We represent a $1-$chain as a vector in $\mathbb{R}^m$. Let $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ be the set of labeled, oriented edges, and suppose for real coefficients α_k we have a $1-$chain $\mathbf{c}_1 = \sum_{i=1}^m \alpha_i e_i$. We identify $\mathbf{c}_1$ with a vector in $\mathbb{R}^m$, so that in this representation $[\mathbf{c}_1]_i = \alpha_i$ for $1 \leq i \leq m$. Similar representations as vectors of real numbers can be derived for other $k-$chains.

1.2. Boundary Maps as Matrices

With chains admitting natural representations as real vectors, we represent the boundary operators as matrices. First, recall that $\partial_1 : \mathcal{C}_1 \to \mathcal{C}_0$, and $\mathcal{C}_1, \mathcal{C}_0$ are $m, n-$dimensional vector spaces, respectively. A matrix representation of ∂_1, denoted by the matrix $\mathbf{B}_1$, then, must match these dimensions, so that $\mathbf{B}_1 \in \mathbb{R}^{n \times m}$. The entries of $\mathbf{B}_1$ are defined as follows. Again, let $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$ be the set of labeled, oriented edges, and let $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ be the set of labeled nodes. Then, the entries of $\mathbf{B}_1$ are given by

$$[\mathbf{B}_1]_{ij} = \begin{cases} -1 & e_j = [v_i, \cdot] \\ 1 & e_j = [\cdot, v_i] \\ 0 & \text{otherwise.} \end{cases} \tag{S-1}$$

[*]Equal contribution [1]Department of Electrical and Computer Engineering, Rice University, Houston, Texas, USA. Correspondence to: TMR <mitch@rice.edu>, NG <nkg2@rice.edu>, SS <segarra@rice.edu>.

Table S-1. Hyperparameters for synthetic and real data experiments. The hyperparameters for the synthetic experiments were identical between the standard and reversed settings, and changed slightly in the transfer learning setting. Hyperparameters were selected by hand, which was reasonable due to the simple nature of the proposed architecture.

Parameter	Synthetic (Std./Rev.)	Synthetic (Trans.)	Drifters
Learning rate	0.001	0.001	0.0025
Training samples	800	333	160
Test samples	200	333	40
Epochs	500	1000	425

Observe that this aligns with the definition of the boundary map in Eq. (1). Under this definition, $\mathbf{B}_1$ is precisely the *signed incidence matrix* of the graph $(\mathcal{X}_0, \mathcal{X}_1)$. Moreover, the adjoint of ∂_1 is represented as the transpose of this matrix, *i.e.* $\mathbf{B}_1^\top$.

The definition of the matrix representation of ∂_2 is slightly more complex. Let $\{t_1, t_2, \dots, t_p\}$ be the set of labeled, oriented triangles. Since $\partial_2 : \mathcal{C}_2 \to \mathcal{C}_1$ is a map from a $p-$dimensional vector space to an $m-$dimensional vector space, the matrix representation $\mathbf{B}_2$ must also match this, so that $\mathbf{B}_2 \in \mathbb{R}^{m \times p}$. For each $1 \le j \le m$ and $1 \le k \le p$, let $e_j = [i_0, i_1]$ be the j^{th} oriented edge. Then, the entries of $\mathbf{B}_2$ are defined as

$$[\mathbf{B}_2]_{jk} = \begin{cases} -1 & t_k = [i_0, \cdot, i_2] \\ 1 & t_k = [\cdot, i_0, i_1] \\ 1 & t_k = [i_0, i_1, \cdot] \\ 0 & \text{otherwise.} \end{cases} \tag{S-2}$$

Again, one can check that this coincides with the definition of the boundary map in Eq. (1), and that $\mathbf{B}_1\mathbf{B}_2 = 0$ in accordance with Lemma 1. Observe that the first Hodge Laplacian Δ_1 can be written as the matrix $\mathbf{L}_1 = \mathbf{B}_1^\top\mathbf{B}_1 + \mathbf{B}_2\mathbf{B}_2^\top$.

1.3. Application of Elementwise Activation Functions

We construct SCoNe using an odd, elementwise activation function ϕ applied to vectors in $\mathcal{C}_1$. Based on the representation of chains as vectors of real numbers defined before, the application of the activation function is fairly obvious: simply apply ϕ to each real-valued coordinate in the vector. For the sake of completeness, we outline here how activation functions can be applied to $1-$chains without using the intermediate representation as real-vectors.

Let $\{e_1, e_2, \dots, e_m\}$ denote the set of $m = |\mathcal{X}_1|$ oriented edges of a simplicial complex, so that any $1-$chain $\mathbf{c}_1 \in \mathcal{C}_1$ is a unique linear combination of these oriented edges. We extend the activation function $\phi : \mathbb{R} \to \mathbb{R}$ to act on $\mathcal{C}_1$ as follows:

$$\phi(\mathbf{c}_1) = \sum_{\ell=1}^{m} \phi(\langle \mathbf{c}_1, e_\ell \rangle) \cdot e_\ell. \tag{S-3}$$

Observe that this can be equivalently computed by representing $\mathbf{c}_1$ as a real vector and applying ϕ elementwise, as desired.

2. Implementation of SCoNe

With the representation of $1-$chains as real vectors and boundary maps as matrices defined, we redefine Algorithm 1 in terms of these real vectors and matrices, detailed in Algorithm S-1. For each experiment, we trained SCoNe using $L = 3$ layers, with hidden dimensions $F_\ell = 16$. It was trained using the Adam optimizer (Kingma & Ba, 2015), using the default parameters of $\beta_1 = 0.9, \beta_2 = 0.99$, for the cross-entropy loss between the output of SCoNe and the true labeled successor node, with an additional weight decay term with coefficient $5\text{e}{-5}$ (Krogh & Hertz, 1992). The remaining hyperparameters varied across experiments, and are listed in Table S-1.

Algorithm S-1 SCoNe for Trajectory Prediction Defined in Terms of Real Vectors and Matrices

```
1: Input: partial trajectory $[i_0, i_1, \dots, i_{m-1}]$
2: Parameters:
   boundary matrices $\{\mathbf{B}_k\}_{k=0}^2$ for oriented edges $\mathcal{E} = \{e_1, e_2, \dots, e_m\}$
   number of layers $L$
   hidden dimensions $\{F_\ell\}_{\ell=0}^{L+1}, F_0 = F_{L+1} = 1$
   weight matrices $\{\{\mathbf{W}_k^\ell \in \mathbb{R}^{F_\ell \times F_{\ell+1}}\}_{\ell=0}^L\}_{k=0}^2$
   activation function $\phi$
3: Initialize: $\mathbf{c}_1^0 \in \mathbb{R}^{|\mathcal{E}|}, \mathbf{c}_1^0 = 0.$
4: for $j = 0$ to $m-2$ do
5:   if $[i_j, i_{j+1}] \in \mathcal{E}$ then
6:     Choose $\ell$ such that $e_\ell = [i_j, i_{j+1}]$
7:     $[\mathbf{c}_1^0]_\ell \leftarrow 1$
8:   else
9:     Choose $\ell$ such that $e_\ell = [i_{j+1}, i_j]$
10:    $[\mathbf{c}_1^0]_\ell \leftarrow -1$
11:  end if
12: end for
13: for $\ell = 0$ to $L-1$ do
14:   $$\begin{aligned}\mathbf{c}_1^{\ell+1} \leftarrow \phi(&\mathbf{B}_2\mathbf{B}_2^\top \mathbf{c}_1^\ell \mathbf{W}_2^\ell \\ &+ \mathbf{c}_1^\ell \mathbf{W}_1^\ell \\ &+ \mathbf{B}_1^\top \mathbf{B}_1 \mathbf{c}_1^\ell \mathbf{W}_0^\ell)\end{aligned} \tag{S-4}$$
15: end for
16: $\mathbf{c}_0^{L+1} \leftarrow \mathbf{B}_1 \mathbf{c}_1^L \mathbf{W}_0^L$
17: $\mathbf{z} \leftarrow \mathrm{softmax}(\{[\mathbf{c}_0^{L+1}]_j : j \in \mathcal{N}(i_{m-1})\})$
18: Return: $\widehat{i}_m \leftarrow \arg\max_j z_j$
```

3. Implementation of Nullspace Projection Methods

Examining the implementation of SCoNe, we see that the architecture consists of a map from 1-chains to 1-chains, followed by the application of the boundary map ∂_1 to obtain a 0-chain. In Section 6, we compared SCoNe to methods based on projecting 1-chains into the kernel of ∂_1 or Δ_1, following the work of Schaub et al. (2020). Of course, by Lemma 1, applying ∂_1 to this would always yield the zero vector, which is useless for prediction tasks. Therefore, we adjusted this by constructing a version of the boundary map restricted to the edges adjacent to the terminal node. In practice, this amounts to choosing the edge with the largest outgoing flow from i_{m-1}, then predicting the next node as the endpoint of that edge.

4. Computational Complexity Analysis

We establish the $\mathcal{O}(|\mathcal{X}_1|F_\ell F_{\ell+1} + |\mathcal{X}_2|\min\{F_\ell, F_{\ell+1})$ runtime of the ℓ^{th} layer of SCoNe here. Observe in Algorithm S-1 that the ℓ^{th} layer of SCoNe maps a matrix in $\mathbb{R}^{|\mathcal{X}_1|\times F_\ell}$ to a matrix in $\mathbb{R}^{|\mathcal{X}_1|\times F_\ell}$ by multiplying the argument by $F_\ell \times F_{\ell+1}$ matrices from the right, and combinations of boundary maps from the left, followed by an aggregation and activation step. One such way to compute this is by considering intermediate representations $\mathbf{d}^\ell$, defined as follows.

$$\mathbf{d}_0^\ell = \mathbf{B}_1^\top \mathbf{B}_1 \mathbf{c}_1^\ell \mathbf{W}_0^\ell \tag{S-5}$$
$$\mathbf{d}_1^\ell = \mathbf{c}_1^\ell \mathbf{W}_1^\ell \tag{S-6}$$
$$\mathbf{d}_2^\ell = \mathbf{B}_2 \mathbf{B}_2^\top \mathbf{c}_1^\ell \mathbf{W}_2^\ell. \tag{S-7}$$

Then, the output is computed by

$$\mathbf{c}_1^{\ell+1} = \phi\left(\mathbf{d}_0^\ell + \mathbf{d}_1^\ell + \mathbf{d}_2^\ell\right). \tag{S-8}$$

The complexity of computing $\mathbf{c}_1^{\ell+1}$, then, is equal to the sum of the complexities of computing $\mathbf{d}_0^\ell, \mathbf{d}_1^\ell, \mathbf{d}_2^\ell$ and the complexity of taking their sum and applying ϕ.

Table S-2. Computational complexities for computing intermediate representations of a single SCoNe layer. We indicate the order of operations in the "Expression" column using parentheses, which yields different complexities based on the size of each level of the simplicial complex $\mathcal{X}$. The best conditions for each expression in terms of the relative sizes of the levels of $\mathcal{X}$ are listed in the column "Best use case."

QUANTITY	EXPRESSION	COMPLEXITY	BEST USE CASE
$\mathbf{d}_0^\ell$	$\mathbf{B}_1^\top(\mathbf{B}_1(\mathbf{c}_1^\ell\mathbf{W}_0^\ell))$	$\mathcal{O}(\lvert\mathcal{X}_1\rvert F_\ell F_{\ell+1})$	$\lvert\mathcal{X}_1\rvert \approx \lvert\mathcal{X}_0\rvert$
	$\mathbf{B}_1^\top((\mathbf{B}_1\mathbf{c}_1^\ell)\mathbf{W}_0^\ell)$	$\mathcal{O}(\lvert\mathcal{X}_0\rvert F_\ell F_{\ell+1} + \lvert\mathcal{X}_1\rvert(F_\ell + F_{\ell+1}))$	$\lvert\mathcal{X}_1\rvert \gg \lvert\mathcal{X}_0\rvert$
	$(\mathbf{B}_1^\top(\mathbf{B}_1\mathbf{c}_1^\ell))\mathbf{W}_0^\ell$	$\mathcal{O}(\lvert\mathcal{X}_1\rvert F_\ell F_{\ell+1})$	$\lvert\mathcal{X}_1\rvert \approx \lvert\mathcal{X}_0\rvert$
$\mathbf{d}_1^\ell$	$\mathbf{c}_1^\ell\mathbf{W}_1^\ell$	$\mathcal{O}(\lvert\mathcal{X}_1\rvert F_\ell F_{\ell+1})$	
$\mathbf{d}_2^\ell$	$\mathbf{B}_2(\mathbf{B}_2^\top(\mathbf{c}_1^\ell\mathbf{W}_2^\ell))$	$\mathcal{O}(\lvert\mathcal{X}_1\rvert F_\ell F_{\ell+1} + \lvert\mathcal{X}_2\rvert F_{\ell+1})$	$\lvert\mathcal{X}_2\rvert \gg \lvert\mathcal{X}_1\rvert$
	$\mathbf{B}_2((\mathbf{B}_2^\top\mathbf{c}_1^\ell)\mathbf{W}_2^\ell)$	$\mathcal{O}(\lvert\mathcal{X}_2\rvert(F_\ell F_{\ell+1}))$	$\lvert\mathcal{X}_2\rvert \leq \lvert\mathcal{X}_1\rvert$
	$(\mathbf{B}_2(\mathbf{B}_2^\top\mathbf{c}_1^\ell))\mathbf{W}_2^\ell$	$\mathcal{O}(\lvert\mathcal{X}_1\rvert F_\ell F_{\ell+1} + \lvert\mathcal{X}_2\rvert F_\ell)$	$\lvert\mathcal{X}_2\rvert \gg \lvert\mathcal{X}_1\rvert$

Observing that the matrix $\mathbf{B}_1$ is typically a sparse matrix with $\mathcal{O}(|\mathcal{X}_1|)$ nonzero entries, and $\mathbf{B}_2$ is typically a sparse matrix with $\mathcal{O}(|\mathcal{X}_2|)$ nonzero entries, complexities for computing $\mathbf{d}_0^\ell, \mathbf{d}_1^\ell, \mathbf{d}_2^\ell$ can be derived in a straightforward manner. We remark that the order of operations has an impact on complexity, as applying the boundary map to an input may increase or decrease the dimension, which has an impact on the application of the weight matrix $\mathbf{W}_k^\ell$. For instance, if a simplicial complex has very few triangles, one should multiply $\mathbf{c}_1^\ell$ by $\mathbf{B}_2^\top$ first when computing $\mathbf{d}_2^\ell$, since the lower-dimensional space $\mathcal{C}_2$ will have lower complexity when multiplying from the right by $\mathbf{W}_2^\ell$. We gather the complexities of all possible ways to evaluate these expressions in Table S-2. Observing that each $\mathbf{d}_k^\ell$ can be evaluated in $\mathcal{O}(|\mathcal{X}_1|F_\ell F_{\ell+1} + |\mathcal{X}_2|\min\{F_\ell, F_{\ell+1}\})$ time yields the desired complexity, since the application of ϕ is only $\mathcal{O}(|\mathcal{X}_1|F_{\ell+1})$.

5. Proof of Proposition 1

We provide a proof of necessary and sufficient conditions for SCoNe to be an admissible architecture, as stated in Proposition 1. We first establish the following auxiliary result.

Proposition S-1. *Let $\phi : \mathbb{R} \to \mathbb{R}$ be a function defined on the reals. Suppose that for some pair of real numbers a, b such that $a \neq -b$ and $a \neq b$, we have*

$$\begin{aligned} \phi(a) &= -\phi(b) \\ \phi(a) &\neq 0, \end{aligned} \tag{S-9}$$

and for all $\gamma \in \mathbb{R}$

$$\phi(\gamma a) = -\phi(\gamma b). \tag{S-10}$$

Under these conditions, ϕ is not continuous.

Proof. Since $a \neq -b$ and $a \neq b$, $|a| \neq |b|$. Without loss of generality, assume $|a| > |b|$, so that putting $\gamma = \frac{b}{a}$ satisfies $|\gamma| < 1$. We then have

$$\phi(\gamma a) = \phi(b) \tag{S-11}$$

$$\gamma^j b = \gamma^{j+1} a \tag{S-12}$$

for all nonnegative integers j. An immediate consequence of this is that for all nonnegative integers j, we have

$$\phi(a) = (-1)^j \phi\left(\gamma^j a\right). \tag{S-13}$$

Consider the sequence $\{c_j\}_{j=1}^\infty$ where $c_j = \gamma^j b$ for each j. Observe that as $j \to \infty$, $c_j \to 0$ due to the fact that $|\gamma| < 1$. Moreover, by (S-13), we have that $\phi(c_j) = -\phi(c_{j+1})$ for each j. Therefore, for all $j > 0$,

$$|\phi(c_j) - \phi(c_{j+1})| = 2\,|\phi(a)| > 0. \tag{S-14}$$

That is, although the sequence $\{c_j\}_{j=1}^{\infty}$ converges to 0, the sequence $\{\phi(c_j)\}_{j=1}^{\infty}$ does not converge. Therefore, ϕ is not continuous at the point 0, so that ϕ is not a continuous function, as desired. $\square$

We now prove Proposition 1 (restated below).

Proposition S-2 (Restatement of Proposition 1)**.** *Assume that the activation function ϕ is continuous and applied elementwise. If* SCoNe *(as defined in Algorithm 1) is admissible, ϕ must be an odd and nonlinear function.*

Proof. We prove this statement by first establishing that SCoNe is permutation equivariant by virtue of ϕ being applied elementwise, and then show that orientation equivariance and simplicial awareness are satisfied only if ϕ is odd and nonlinear, respectively.

Permutation equivariance.

Let $\{e_1, e_2, \dots, e_m\}$ be a chosen orientation of edges for a simplicial complex $\mathcal{X}$, where each $\{e_\ell\}_{j=1}^{m}$ is an oriented 1-simplex (and thus is a 1-chain). The activation function ϕ is applied to arbitrary $\mathbf{c}_1 \in \mathcal{C}_1$ as follows:

$$\phi(\mathbf{c}_1) = \sum_{j=1}^{m} \phi(\langle \mathbf{c}_1, e_\ell \rangle) e_\ell. \tag{S-15}$$

Observe that ϕ commutes with permutation matrices, *i.e.* $\phi(\mathbf{P}_1\mathbf{c}_1) = \mathbf{P}_1\phi(\mathbf{c}_1)$. In particular, we take

$$\mathbf{c}_1^{\ell+1} \leftarrow \phi(\partial_2\partial_2^\top \mathbf{c}_1^\ell \mathbf{W}_2^\ell + \mathbf{c}_1^\ell \mathbf{W}_1^\ell + \partial_1^\top \partial_1 \mathbf{c}_1^\ell \mathbf{W}_0^\ell). \tag{S-16}$$

Letting $\mathcal{P} = \{\mathbf{P}_k\}_{k=0}^{2}$ as in Property 1, we consider a version of (S-16) where the input $\mathbf{c}_1^\ell$ and the boundary maps are permuted by $\mathcal{P}$:

$$\begin{aligned}
\phi(\mathbf{P}_1\partial_2\mathbf{P}_2^\top\mathbf{P}_2\partial_2^\top\mathbf{P}_1^\top\mathbf{P}_1\mathbf{c}_1^\ell\mathbf{W}_2^\ell + \mathbf{P}_1\mathbf{c}_1^\ell\mathbf{W}_1^\ell + \mathbf{P}_1\partial_1^\top\mathbf{P}_0^\top\mathbf{P}_0\partial_1\mathbf{P}_1^\top\mathbf{P}_1\mathbf{c}_1^\ell\mathbf{W}_0^\ell) &= \\
\phi(\mathbf{P}_1\partial_2\partial_2^\top\mathbf{c}_1^\ell\mathbf{W}_2^\ell + \mathbf{P}_1\mathbf{c}_1^\ell\mathbf{W}_1^\ell + \mathbf{P}_1\partial_1^\top\partial_1\mathbf{c}_1^\ell\mathbf{W}_0^\ell) &= \\
\phi(\mathbf{P}_1(\partial_2\partial_2^\top\mathbf{c}_1^\ell\mathbf{W}_2^\ell + \mathbf{c}_1^\ell\mathbf{W}_1^\ell + \partial_1^\top\partial_1\mathbf{c}_1^\ell\mathbf{W}_0^\ell)) &= \\
\mathbf{P}_1\phi(\partial_2\partial_2^\top\mathbf{c}_1^\ell\mathbf{W}_2^\ell + \mathbf{c}_1^\ell\mathbf{W}_1^\ell + \partial_1^\top\partial_1\mathbf{c}_1^\ell\mathbf{W}_0^\ell) &= \\
\mathbf{P}_1\mathbf{c}_1^{\ell+1}&.
\end{aligned} \tag{S-17}$$

That is to say, each layer of SCoNe is equivariant to joint permutations of the input and the boundary maps, so that the entirety of SCoNe is permutation equivariant, as desired.

Orientation equivariance.

To show that ϕ being odd is a necessary condition for orientation equivariance, suppose that $\phi : \mathbb{R} \to \mathbb{R}$ is not odd and is continuous. Then, there exists $x \in \mathbb{R}$ such that $\phi(x) \neq -\phi(-x)$. This implies that either $\phi(x)$ or $\phi(-x)$ is nonzero. Without loss of generality, then, suppose $\phi(x) \neq 0$.

Take $\mathcal{X}$ to be a simplicial complex with two nodes and a single edge connecting them: $\mathcal{X} = \{\{i_0\}, \{i_1\}, \{i_0, i_1\}\}$. Finding it convenient to represent 1-chains in this case as real numbers, let $\mathbf{c}_1^0 = x$ be a 1-chain, and let $\{\{\mathbf{W}_k^\ell\}_{\ell=0}^{L}\}_{k=0}^{2}$ all be contained in $\mathbb{R}^{1\times1}$, so that their application is equivalent to scalar multiplication. Set $\mathbf{W}_1^\ell = \mathbf{W}_2^\ell = 0$ for all ℓ, and denote by w^ℓ the scalar component of $\mathbf{W}_0^\ell$ for each ℓ. We now show inductively that for any nonnegative integer L, there exists an L-layer SCoNe architecture with coefficients $\{w^\ell\}_{\ell=0}^{L-1}$ that does not satisfy orientation equivariance.

For the base case, let $L = 1$, and consider a 1-layer SCoNe architecture:

$$\mathbf{c}_1^1 = \phi(w^0\mathbf{c}_1^0). \tag{S-18}$$

Setting $w^0 = 1$ yields $\mathbf{c}_1^1 = \phi(x)$. One can easily see that this is not orientation equivariant, since $\phi(x) \neq -\phi(-x)$.

For the inductive step, suppose that $L > 1$, and that a SCoNe architecture with $L-1$ layers and coefficients $\{w^\ell\}_{\ell=0}^{L-2}$ is not orientation equivariant for the input $\mathbf{c}_1^0$: denote the differently oriented outputs as $\mathbf{c}_{1+}^{L-1}$ and $\mathbf{c}_{1-}^{L-1}$, so that $\mathbf{c}_{1+}^{L-1} \neq -\mathbf{c}_{1-}^{L-1}$. If $\mathbf{c}_{1+}^{L-1} = \mathbf{c}_{1-}^{L-1}$, implying that $\mathbf{c}_{1+}^{L-1} \neq 0$, take $w^{L-1} = x/\mathbf{c}_{1+}^{L-1}$, so that

$$\phi(w^{L-1}\mathbf{c}_{1+}^{L-1}) = \phi(x) \neq -\phi(x) = -(w^{L-1}\mathbf{c}_{1-}^{L-1}), \tag{S-19}$$

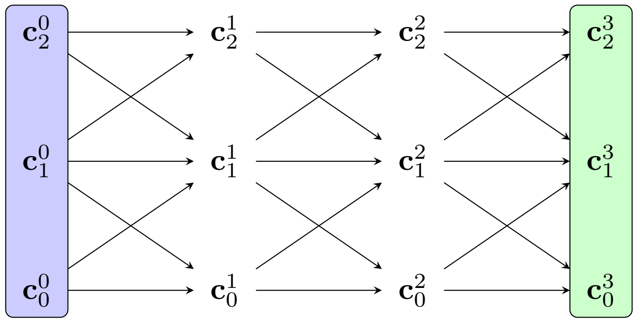


Figure S-1. General structure of the simplicial 2-complex convolutional neural network of Bunch et al. (2020). Each horizontal arrow corresponds to the application of a normalized Hodge Laplacian, and diagonal arrows correspond to the application of normalized boundary or coboundary maps. At each node, the inputs are summed then passed through the activation function ϕ.

yielding an architecture that is not orientation equivariant. Otherwise, suppose for the sake of contradiction that $\mathbf{c}_{1+}^{L-1} \neq \mathbf{c}_{1-}^{L-1}$ and for all $w^{L-1} \in \mathbb{R}$ we have orientation equivariance for the input $\mathbf{c}_1^0$. That is,

$$\phi(w^{L-1}\mathbf{c}_{1+}^{L-1}) = -\phi(w^{L-1}\mathbf{c}_{1-}^{L-1}). \tag{S-20}$$

Since at least one of the 1-chains $\mathbf{c}_{1+}^{L-1}, \mathbf{c}_{1-}^{L-1}$ is nonzero, one can always choose w^{L-1} such that $\phi(w^{L-1}\mathbf{c}_{1?}^{L-1})$ is nonzero, where $\mathbf{c}_{1?}^{L-1}$ denotes said nonzero 1-chain. By Proposition S-1, this implies that ϕ is not a continuous function, yielding a contradiction, as desired. Thus, under these conditions, there exists a SCoNe architecture that is not orientation equivariant.

Simplicial awareness.

Finally, we consider simplicial awareness of order 2. Suppose ϕ is an odd, linear function: it is sufficient to assume that ϕ is the identity map. Let a 1-chain $\mathbf{c}_1^0 \in \mathcal{C}_1$ be given arbitrarily. By Theorem 1, there exists $\mathbf{w} \in \mathcal{C}_0, \mathbf{x} \in \ker(\Delta_1), \mathbf{y} \in \mathcal{C}_2$ such that $\mathbf{c}_1^0 = \partial_1^\top \mathbf{w} + \mathbf{x} + \partial_2 \mathbf{y}$. Some simple algebra, coupled with Lemma 1, shows that when ϕ is the identity map, the 0-chain at the output of SCoNe is given by

$$\mathbf{c}_0^{L+1} = \partial_1 \sum_{j=0}^{L} (\partial_1^\top \partial_1)^L \partial_1^\top \mathbf{w}\omega(j), \tag{S-21}$$

where each $\omega(j)$ can be written as a polynomial of the weight matrices (always yielding a 1×1 matrix, due to the constraint $F_0 = F_{L+1} = 1$). That is, the output of SCoNe does not depend on ∂_2, and thus fails to fulfill simplicial awareness of order 2. However, if ϕ is nonlinear, Lemma 1 does not come in to effect, since $\partial_1 \circ \phi \circ \partial_2 \neq 0$, allowing for simplicial awareness, as desired. □

6. Admissibility of Previous Simplicial Neural Networks

Simplicial neural networks similar to SCoNe were previously proposed by Ebli et al. (2020); Bunch et al. (2020). The convolutional layer for k-(co)chains of Ebli et al. (2020) takes the following form:

$$\mathbf{c}_k^{\ell+1} \leftarrow \phi \left(\sum_{j=0}^{N} \Delta_k^j \mathbf{c}_k^\ell \mathbf{W}_j^\ell \right), \tag{S-22}$$

where ϕ is an activation function applied according to the standard coordinate basis as in (S-3), and $\{\mathbf{W}_j^\ell\}$ are trainable weight matrices. Observe that (S-22) takes a form similar to that of SCoNe, but using polynomials of the entire Hodge Laplacian instead of considering the components $\partial_k^\top \partial_k$ and $\partial_{k+1}\partial_{k+1}^\top$ separately. A similar argument to the proof of Proposition 1 shows that the simplicial neural network architecture of Ebli et al. (2020) is orientation equivariant for elementwise nonlinearity ϕ if and only if ϕ is an odd function (when $k > 0$), and simplicial awareness of orders $k-1$ and $k+1$ is satisfied as well. In particular, for a 2-dimensional simplicial complex, simplicial awareness is satisfied when $k = 1$, as in SCoNe. Notably, since their architecture does not involve a “readout” mapping a 1-chain to a 0-chain, ϕ does not

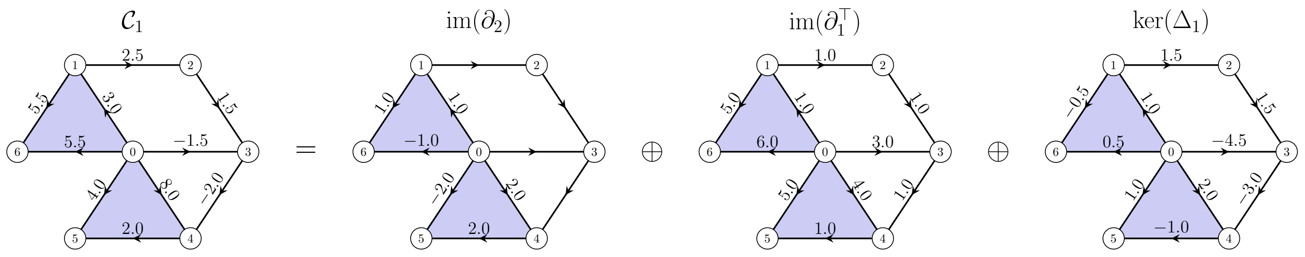


Figure S-2. Illustration of the Hodge Decomposition for an example 1-chain. The subspace $\mathrm{im}(\partial_2)$ consists of 1-chains that are curly around 2-simplices, while $\mathrm{im}(\partial_1^\top)$ is determined by the differences between nodal coefficients in a 0-chain. $\ker(\Delta_1)$ corresponds to 1-chains that are not curly, and not determined by coefficients in a 0-chain, unlike $\mathrm{im}(\partial_1^\top)$. Figure adapted from (Schaub et al., 2020).

have to be nonlinear, unlike the case of SCoNe. We would like to remark that the original authors used the "Leaky ReLU" activation function in their empirical studies, which is not odd, thus failing to satisfy orientation equivariance.

The simplicial 2-complex convolutional neural network of Bunch et al. (2020) also employs convolutional layers based on the boundary maps, albeit using normalized versions of said maps following Schaub et al. (2020). Rather than restricting their internal states $\mathbf{c}^\ell$ to be supported on a single level of the simplicial complex as in Ebli et al. (2020) and SCoNe, they propose an architecture that maintains representations on all levels of the simplicial complex. Ignoring the details of normalizing the boundary maps and Hodge Laplacians, their convolutional layer takes the form

$$\mathbf{c}_0^{\ell+1} \leftarrow \phi\left(\partial_1 \mathbf{c}_1^\ell \mathbf{W}_\ell^{0,1} + \Delta_0 \mathbf{c}_0^\ell \mathbf{W}_\ell^{0,0}\right) \tag{S-23}$$

$$\mathbf{c}_1^{\ell+1} \leftarrow \phi\left(\partial_2 \mathbf{c}_2^\ell \mathbf{W}_\ell^{1,2} + \Delta_1 \mathbf{c}_1^\ell \mathbf{W}_\ell^{1,1} + \partial_1^\top \mathbf{c}_0^\ell \mathbf{W}_\ell^{1,0}\right) \tag{S-24}$$

$$\mathbf{c}_2^{\ell+1} \leftarrow \phi\left(\Delta_2 \mathbf{c}_2^\ell \mathbf{W}_\ell^{2,2} + \partial_2^\top \mathbf{c}_1^\ell \mathbf{W}_\ell^{2,1}\right), \tag{S-25}$$

where $\mathbf{c}_0^\ell, \mathbf{c}_1^\ell, \mathbf{c}_2^\ell$ are the input $0, 1, 2$-chains, respectively, and ϕ is an elementwise activation function. We illustrate the structure of this architecture in Fig. S-1. Again, a convolutional neural network composed of such layers follows a result similar to Proposition 1, where it is admissible if and only if ϕ is odd. If one considers the chains stored at all levels of the simplicial complex, ϕ does not need to be nonlinear, unlike SCoNe. However, if one only takes *one* of the levels of the chain complex as output, similar to the output of SCoNe or the architecture of Ebli et al. (2020), the requirement of ϕ being nonlinear comes into effect.

7. Chains, Flows, and the Hodge Decomposition

We elaborate here on using 1-chains to model flows on the edges of a simplicial complex, viewed through the lens of the Hodge Decomposition [*cf.* Theorem 1]. Suppose we have a simplicial complex $\mathcal{X}$, on which there is a *flow* over the edges. We interpret a flow as having a magnitude and an orientation. Analogously to an electrical circuit, the magnitude of the flow is the absolute current flowing through a wire, and the orientation is determined by the sign of the measurement as well as the direction it is being measured in: that is, if the measurement direction is reversed, the sign of the measurement will change. This skew-symmetric property is reflected by the vector space of 1-chains over the real numbers, since for any oriented edge $[i, j]$, we have $[i, j] = -[j, i]$. Indeed, this behavior mirrors that of a directional derivative on a surface: reversing the direction of the derivative of a function merely changes the sign. Thus, we will use the term "1-chain" and "flow" interchangeably.

With this model in mind, we now provide an interpretation of Theorem 1. First, it will be useful to understand what *integrating over a path* means in this context. Let $S = \{i_\ell\}_{\ell=1}^m$ be a sequence of nodes, such that any two nodes that are adjacent in the sequence are also adjacent in $\mathcal{X}$: we call such a sequence a *path*. If S is such that the final node in the sequence is the same as the first node, we call S a *closed path*. Now, let $\mathbf{c}_1 \in \mathcal{C}_1$ be a 1-chain, and let $\mathbf{c}_S \in \mathcal{C}_1$ represent S as a 1-chain in the following way:

$$\mathbf{c}_S = \sum_{\ell=1}^{m-1} [i_j, i_{j+1}]. \tag{S-26}$$

Then, we say that the integral of $\mathbf{c}_1$ over the path S is the inner product

$$\langle \mathbf{c}_1, \mathbf{c}_S \rangle = \sum_{\ell=1}^{m-1} \langle \mathbf{c}_1, [i_\ell, i_{\ell+1}] \rangle. \tag{S-27}$$

We call the subspace $\operatorname{im}(\partial_2)$ *curly*, since it corresponds to flows around 2-simplices. As pictured in Fig. S-2, these flows are supported strictly on the boundary of 2-simplices, dictated by a curl about each 2-simplex. In particular, the pictured flow has a curl of 1.0 about the 2-simplex $[0, 1, 6]$, and a curl of 2.0 about $[0, 4, 5]$. Elsewhere, the flow takes value 0.

The subspace $\operatorname{im}(\partial_1^\top)$ is referred to as *gradient*, since it corresponds to flows induced by differences between so-called "potentials" at each node. That is, for each $\mathbf{c}_{grad} \in \operatorname{im}(\partial_1^\top)$ and oriented simplex $[i, j]$, there exists a $\mathbf{c}_0 \in \mathcal{C}_0$ such that

$$\langle \mathbf{c}_{grad}, [i, j] \rangle = \langle \mathbf{c}_0, [j] \rangle - \langle \mathbf{c}_0, [i] \rangle. \tag{S-28}$$

In Fig. S-2, each node has a potential corresponding to its label, *e.g.* node 3 satisfies $\langle \mathbf{c}_0, [3] \rangle = 3.0$, so that $\langle \mathbf{c}_{grad}, [0, 3] \rangle = 3.0$. The integral of a gradient flow, analogous to the line integral of a vector field that is the gradient of a scalar field, is *path independent*, in that the integral over a path of a gradient flow only depends on the starting and end points. That is, if S and S' are both paths whose initial and terminal points are the same, then for any gradient flow $\mathbf{c}_{grad} \in \operatorname{im}(\partial_1^\top)$,

$$\langle \mathbf{c}_{grad}, \mathbf{c}_S \rangle = \langle \mathbf{c}_{grad}, \mathbf{c}_{S'} \rangle. \tag{S-29}$$

In particular, the integral of a gradient flow over a closed loop S'' is null: $\langle \mathbf{c}_{grad}, \mathbf{c}_{S''} \rangle = 0$.

Finally, the subspace $\ker(\Delta_1)$ consists of *harmonic* flows. Harmonic flows satisfy two properties: the integral of a harmonic flow around a 2-simplex is null, and the sum of the flows incident to any node is null, as exemplified in Fig. S-2. More precisely, for any $\mathbf{c}_{harm} \in \ker(\Delta_1)$, $[i] \in \mathcal{C}_0$, $[j_0, j_1, j_2] \in \mathcal{C}_2$,

$$\langle \mathbf{c}_{harm}, \partial_2 [j_0, j_1, j_2] \rangle = 0 \tag{S-30}$$

$$\langle \partial_1 \mathbf{c}_{harm}, [i] \rangle = 0. \tag{S-31}$$

8. Hodge Laplacians for Cubical 2-Complexes

In the same way that an abstract simplicial complex can be thought of as a set of points, edges, triangles, tetrahedra, etc., with the property of being closed under restriction, a cubical complex is a natural analog constructed from points, edges, squares, cubes, hypercubes, and so on. Cubical complexes arise when considering grid-structured domains (Wagner et al., 2012), and particularly in Section 6 when considering the Berlin map data. Defining appropriate boundary maps for cubical complexes of high dimension can be tedious, so we restrict our discussion to cubical complexes of dimension 2, or cubical 2-complexes.

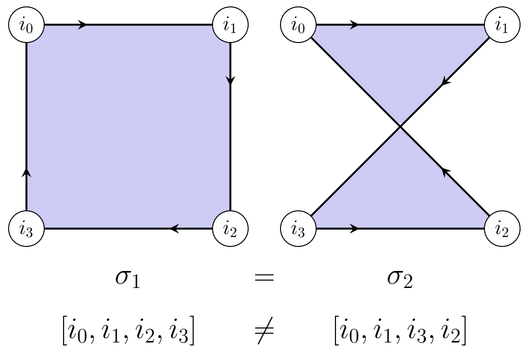
We adapt the definition of Farley (2003) to more naturally capture the notion of orientation and boundary. The *standard abstract k-cube* is the set $\{0, 1\}^k$. By convention, we say $\{0, 1\}^0 = \{0\}$. The *faces* of the standard abstract k-cube are the sets taking the form $\prod_{j=1}^k A_j$, where each A_j is a nonempty subset of $\{0, 1\}^k$ with $|A_j| = 1$ for exactly one index j. By convention, we say that $\{0\}$ is a face of itself. For a finite set σ paired with a bijection $\psi_\sigma : \sigma \to \{0, 1\}^k$, we say that a subset $\sigma' \subseteq \sigma$ is a *face of σ with respect to ψ_σ* if the image of σ' under ψ_σ is a face of $\{0, 1\}^k$.

A *cubical* 2*-complex* $\mathcal{X}$ over a set $\mathcal{V}$ is a multiset of subsets of $\mathcal{V}$ paired with a set of bijections $\{\psi_\sigma : \sigma \to \{0, 1\}^{k_\sigma}\}_{\sigma \in \mathcal{X}}$, where $k_\sigma \in \{0, 1, 2\}$ for each $\sigma \in \mathcal{X}$, with the following properties:

1. $\mathcal{X}$ covers $\mathcal{V}$.
2. For each $\sigma \in \mathcal{X}$, if $\sigma' \subseteq \sigma$, then $\sigma' \in \mathcal{X}$ if and only if σ' is a face of σ with respect to ψ_σ.

We denote the set of elements of $\mathcal{X}$ with 2^k elements as $\mathcal{X}_k$: the elements of $\mathcal{X}_k$ are naturally referred to as k-cubes, due to the existence of a bijection with the standard abstract k-cube.

Note the similarities and differences with the case of an abstract simplicial complex: for a simplex in an abstract simplicial complex, all of its subsets are faces, so that all of its faces are contained in the abstract simplicial complex due to closure



	ψ_1	ψ_2
i_0	$(0,0)$	$(0,0)$
i_1	$(0,1)$	$(0,1)$
i_2	$(1,1)$	$(1,0)$
i_3	$(1,0)$	$(1,1)$

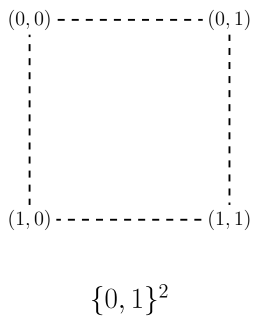


Figure S-3. A set does not a 2-cube make. The defined bijection between an element of a cubical 2-complex and the standard 2-cube dictates what its faces are. Pictured are two 2-cubes over the same set of nodes, but with different bijections ψ_1, ψ_2 between the standard abstract 2-cube. Although each 2-cube is defined over the same set of vertices, the faces induced by the bijections ψ_1, ψ_2 differ.

under restriction. In the case of an abstract cubical complex, we only require closure under restriction *to faces* with respect to the bijections ψ_σ. This is more general than closure under restriction, and in fact subsumes closure under restriction for abstract simplicial complexes. We illustrate this in Fig. S-3.

In order to define a boundary operator, we first define an appropriate notion of orientation. For 0-cubes and 1-cubes, an orientation is the exact same as in the case of a simplicial complex. That is, if $\sigma = \{i_0\}$ for some $i_0 \in \mathcal{V}$, the only orientation of σ is $[i_0]$. Similarly, if $\sigma = \{i_0, i_1\}$, there are two orientations of σ: $[i_0, i_1]$ and $[i_1, i_0]$. When $\sigma \in \mathcal{X}_2$, extra restrictions on the notion of orientations are needed. Suppose $\sigma = \{i_0, i_1, i_2, i_3\}$ is an element of $\mathcal{X}_2$. An ordered sequence of the elements of σ is said to be an orientation with respect to ψ_σ if each pair of cyclically adjacent elements in the sequence forms a face of σ with respect to ψ_σ. Based on adjacency being considered in a cyclic fashion, we take these orientations modulo cyclic permutations. By convention, we say that "reversals" of an orientation change the sign, so that $[i_0, i_1, i_2, i_3] = -[i_3, i_2, i_1, i_0]$. One can check that modulo cyclic permutations, these are the only two orientations of a 2-cube.

As before, denote by $\mathcal{C}_k$ the vector space with the oriented k-simplices of $\mathcal{X}$ as a canonical orthonormal basis, defined over the field of real numbers. The boundary map $\partial_2 : \mathcal{C}_2 \to \mathcal{C}_1$ of an oriented 2-cube is defined as follows:

$$\partial_2([i_0, i_1, i_2, i_3]) = [i_0, i_1] + [i_1, i_2] + [i_2, i_3] - [i_0, i_3]. \tag{S-32}$$

The boundary map ∂_1 is defined as before:

$$\partial_1([i_0, i_1]) = [i_1] - [i_0]. \tag{S-33}$$

In this setting, Lemma 1 holds: $\partial_1 \partial_2 = 0$. Moreover, taking the adjoint of the operators ∂_1 and ∂_2, we can construct the k^{th} cubical Hodge Laplacian in the expected way:

$$\Delta_k = \partial_k^\top \partial_k + \partial_{k+1} \partial_{k+1}^\top, \tag{S-34}$$

with the corresponding cubical analog to the Hodge Decomposition:

$$\mathcal{C}_k = \operatorname{im}(\partial_{k+1}) \oplus \operatorname{im}(\partial_k^\top) \oplus \ker(\Delta_k). \tag{S-35}$$

Given the representation of boundary maps acting on formal sums of oriented cubes in an abstract cubical complex, architectures analogous to SCoNe follow naturally via simple substitutions of the boundary maps.

References

Bunch, E., You, Q., Fung, G., and Singh, V. Simplicial 2-complex convolutional neural networks. In *NeurIPS Workshop on Topological Data Analysis and Beyond*, 2020.

Ebli, S., Defferrard, M., and Spreemann, G. Simplicial neural networks. In *NeurIPS Workshop on Topological Data Analysis and Beyond*, 2020.

Farley, D. S. Finiteness and $\mathrm{CAT}(0)$ properties of diagram groups. *Topology*, 42(5):1065–1082, 2003. doi:10.1016/S0040-9383(02)00029-0.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Krogh, A. and Hertz, J. A. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, pp. 950–957, 1992.

Schaub, M. T., Benson, A. R., Horn, P., Lippner, G., and Jadbabaie, A. Random walks on simplicial complexes and the normalized Hodge 1-Laplacian. *SIAM Review*, 62(2):353–391, 2020. doi:10.1137/18M1201019.

Wagner, H., Chen, C., and Vuçini, E. Efficient computation of persistent homology for cubical data. In *Topological methods in data analysis and visualization II*, pp. 91–106. Springer, 2012.