Sheaf Cohomology of Linear Predictive Coding Networks

Jeffrey Seely
Sakana AI

Predictive Coding Networks ≈ Cellular Sheaves

Idea 1: **Predictive coding networks (PCNs)** can be formulated nicely as a **(cellular) sheaf**

Idea 2: The sheaf formulation helps us analyze **recurrent PCNs**:

- A neuron in a recurrent network only has a **local view.**
- It receives prediction errors from all of its incident connections.
- Some of those prediction errors may arise due to **contradictory feedback loops**
- Contradictory feedback loops create **internal tension** that PC has to unwind.
- Tools from sheaf theory help us reason about this clearly!

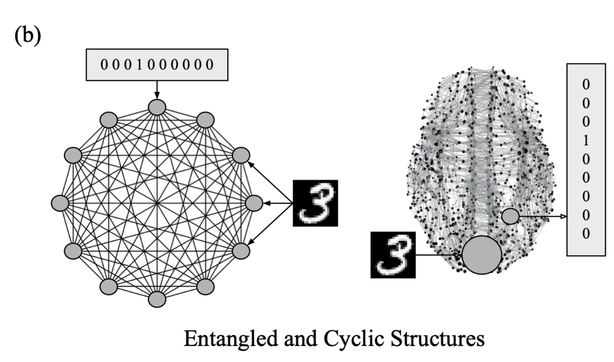
Linear(ized) Predictive Coding	Sheaf Theory
Computational graph	Base graph $G = (V, E)$
Activations s	0-cochain $s \in C^0$
Prediction errors ε	1-cochain $\varepsilon \in C^1$
Computing errors from activations	Coboundary $D : C^0 \to C^1$
PC energy E	$E = \frac{1}{2}\|Ds\|^2$
PC inference flow $\dot{s}$	Sheaf diffusion $\dot{s} = -\nabla_s E = -Ls$
Globally consistent states	$H^0 = \ker D$
Errors after optimal inference	$H^1 \cong \ker D^\top$

Predictive Coding Network

Standard DL:

$$\min_{W_1,W_2,W_3,h_1,h_2} \; \frac{1}{2}\left\|y - W_3 h_2\right\|_2^2$$

$$\text{s.t.} \quad h_1 = W_1 x, \quad h_2 = W_2 h_1,$$



Salvatori et al. 2025, https://arxiv.org/pdf/2308.07870

Predictive Coding (PC):

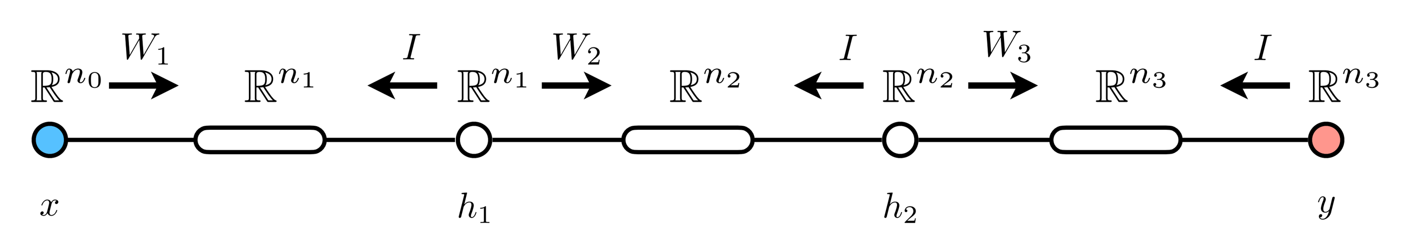
$$E_{\text{PC}}(h_1, h_2, W_1, W_2, W_3) = \tfrac{1}{2}\left(\|r_1\|_2^2 + \|r_2\|_2^2 + \|r_3\|_2^2\right),$$
$$r_1 = h_1 - W_1 x, \quad r_2 = h_2 - W_2 h_1, \quad r_3 = y - W_3 h_2.$$

PC inference: minimize E w.r.t. h
PC learning: minimize E w.r.t. W

⟵ same thing, different lingo ⟶

sheaf diffusion: minimize E w.r.t. h
sheaf learning: minimize E w.r.t. W

Cellular Sheaf



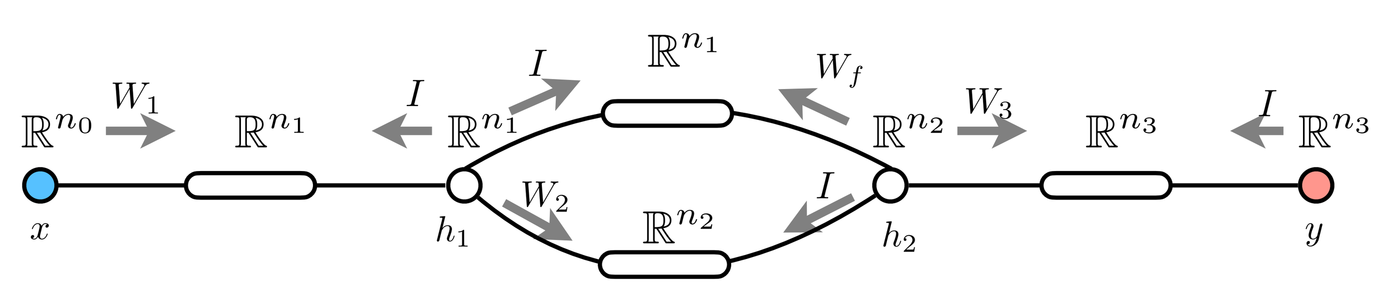
0-cochain $\quad s = (x, h_1, h_2, y) \in C^0 := \mathbb{R}^{n_0} \oplus \mathbb{R}^{n_1} \oplus \mathbb{R}^{n_2} \oplus \mathbb{R}^{n_3}.$

1-cochain $\quad r = (r_1, r_2, r_3) \in C^1 := \mathbb{R}^{n_1} \oplus \mathbb{R}^{n_2} \oplus \mathbb{R}^{n_3}.$

coboundary:
$$\delta^0 s = \begin{bmatrix} -W_1 & I_{n_1} & 0 & 0 \\ 0 & -W_2 & I_{n_2} & 0 \\ 0 & 0 & -W_3 & I_{n_3} \end{bmatrix} \begin{bmatrix} x \\ h_1 \\ h_2 \\ y \end{bmatrix} = \begin{bmatrix} h_1 - W_1 x \\ h_2 - W_2 h_1 \\ y - W_3 h_2 \end{bmatrix}$$

$$E_{\text{PC}} = \frac{1}{2}\left\|\delta^0 s\right\|_2^2$$

Clamped Networks & Hodge Decomposition



Clamp inputs and outputs to create a "target" b on edges:

$$D = \begin{bmatrix} I_{n_1} & 0 \\ -W_2 & I_{n_2} \\ 0 & -W_3 \\ I_{n_1} & -W_2^{\text{FB}} \end{bmatrix}, \quad b = \begin{bmatrix} -W_1 x \\ 0 \\ y \\ 0 \end{bmatrix}$$

$$E_{\text{rel}}(z) = \tfrac{1}{2}\|Dz + b\|^2, \quad z \in C^0_{\text{free}}.$$

Orthogonal (Hodge) decomposition of target b:

$$b = \underbrace{(-Dz^\star)}_{\in \operatorname{im} D} + \underbrace{r^\star}_{\in \ker D^\top}$$

Projection of b onto im D: part of error that **can** be explained by inference

Projection of b onto ker $D^\top$: part of error that **cannot** be reduced by inference

$$\frac{\partial E_{\text{rel}}}{\partial W_e}\bigg|_{z^\star} = \underbrace{(\mathcal{H}b)_e}_{\text{harmonic (edge)}} \; \underbrace{(\mathcal{G}b)_u^\top}_{\text{diffusive (vertex)}}$$

Learning requires harmonic-diffusive overlap on the network!

- **Harmonic** (H): projection of b onto ker Dt (edges)
- **Diffusive** (G=D†): which activations are active at optimal inference

Takeaways

✓ PC inference = sheaf diffusion

✓ H^1 captures irreducible errors

✓ Monodromy Φ diagnoses learnability

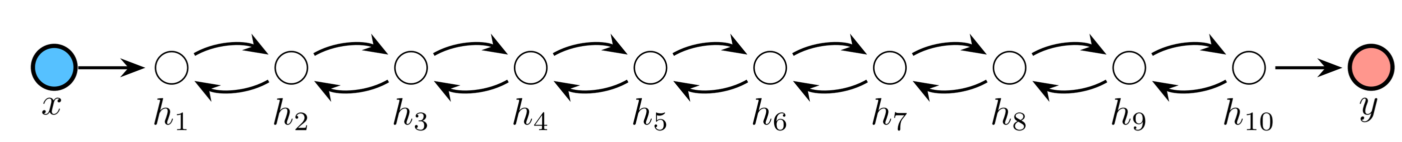
✓ Design principle: initialize toward resonance (Φ ≈ I)

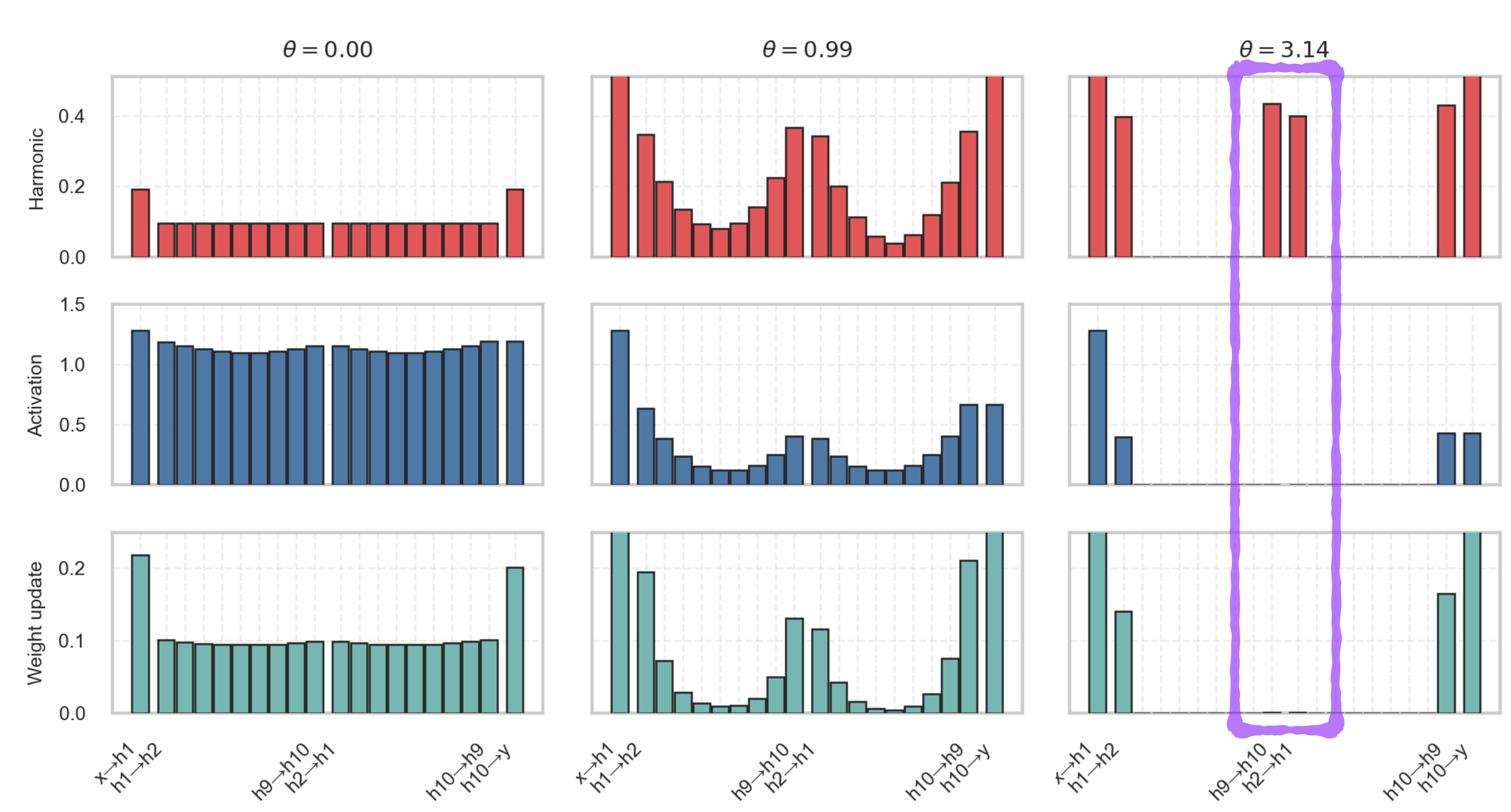
Experiments

$$\Phi = W_2^{\text{FB}} W_2 : \mathbb{R}^{n_1} \to \mathbb{R}^{n_1}$$

θ = 0 (Resonant)
Φ = I
Harmonic load spreads
Diffusion reaches all vertices
Strong updates everywhere

θ = π (Contradictory)
Φ = −I
Harmonic load trapped
Diffusion blocked at boundaries
Starved updates on internal edges





Same architecture, same weight norms -- only orientation matters