Intro to Network Sheaves and Laplacians in Neural Networks

Geometry and Topology in Machine Learning Seminar

Aug 11th, 2025

Outline

Graph Laplacian: definition

Let $G = (V, E)$ be a finite (*optional: weighted/directed*) graph with $|V| = n$.

Graph Laplacian

The graph laplacian is defined as $L = D - A$, where A is the adjacency matrix

$$A_{ij} = \begin{cases} w_{ij} & \text{if } \{v_i, v_j\} \in E \text{ and weighted} \\ 1 & \text{if } \{v_i, v_j\} \in E \text{ and unweighted} \\ 0 & \text{otherwise} \end{cases}$$

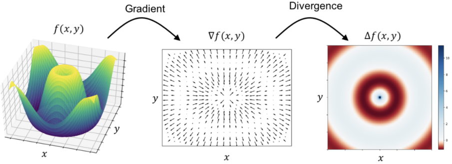
$w_{ij} = -w_{ji}$ if directed; otherwise $w_{ij} = w_{ji}$
and D is the degree matrix $D_{ii} = \sum_{j=1}^{n} A_{ij}, \quad D_{ij} = 0$ for $i \neq j$

In practice, we directly use the normalized Laplacian

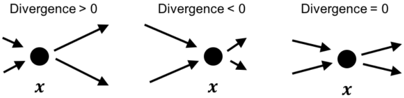
$$L_{\text{sym}} = I - D^{-1/2} A D^{-1/2}$$

Graph Laplacian: a discrete Laplacian

- The continuous Laplacian $\nabla^2 f$ measures how the gradient of f diverges capturing curvature.[1]



- And the divergence measures the net rate at which a quantity flows in and out



- The graph Laplacian plays the same role for discrete data structured as nodes and edges.

[1] Pictures from the blog article *The graph Laplacian*

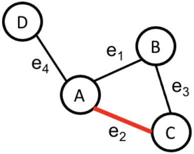
Graph Laplacian: incidence matrix

- On a graph, the analogue of a gradient is given by its edges: for an edge $e_k = (v_i, v_j) \in E$, the "gradient" is the difference in function values $g(e_k) = f(v_i) - f(v_j)$
- For a graph $G = (V, E)$, the incidence matrix $B \in \mathbb{R}^{|E| \times |V|}$ encodes edge–vertex relationships.
- Choose an arbitrary orientation for each edge $e = (u, v)$:

$$B_{e,w} = \begin{cases} +1 & \text{if } w \text{ is the head of } e, \\ -1 & \text{if } w \text{ is the tail of } e, \\ 0 & \text{otherwise} \end{cases}$$

Graph Laplacian: incidence matrix example

Graph G



Incidence Matrix K

$$
\begin{array}{c}
 \\ e_1 \\ e_2 \\ e_3 \\ e_4
\end{array}
\begin{array}{c}
\begin{array}{cccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} \end{array} \\
\begin{bmatrix}
1 & -1 & 0 & 0 \\
1 & 0 & -1 & 0 \\
0 & 1 & -1 & 0 \\
1 & 0 & 0 & -1
\end{bmatrix}
\end{array}
$$

$$
\underbrace{\begin{bmatrix}
1 & -1 & 0 & 0 \\
1 & 0 & -1 & 0 \\
0 & 1 & -1 & 0 \\
1 & 0 & 0 & -1
\end{bmatrix}}_{K}
\underbrace{\begin{bmatrix} f(A) \\ f(B) \\ f(C) \\ f(D) \end{bmatrix}}_{\boldsymbol{f}}
=
\underbrace{\begin{bmatrix} f(A)-f(B) \\ f(A)-f(C) \\ f(B)-f(C) \\ f(A)-f(D) \end{bmatrix}}_{\boldsymbol{g}}
$$

Rows correspond to edges

Columns correspond to vertices

Entries correspond to edges

Entries correspond to edges

[2]

[2] Pictures from the blog article *The graph Laplacian*

Graph Laplacian: net rate as divergence

How can one get the total sum of flow-in and out at a certain vertex? How can matrix perform this summation? Consider the product of K and the transpose

$$\underset{K^T}{\begin{bmatrix} 1 & 1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}} \underset{K}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}} = \underset{L}{\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 2 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}}$$

The graph Laplacian $L = D - A = K^T K$ captures the net flow of each vertex, just like the continuous Laplacian measures divergence of gradient at a point. However, the expressivity of the graph Laplacian is inherently limited. To capture richer and more nuanced structure, we can instead consider the sheaf Laplacian. [3]

[3] Pictures from the blog article *The graph Laplacian*

Outline

Motivation

Graph Laplacian assumes a single scalar space: all fibers $\mathcal{F}(v) = \mathbb{R}$ and restrictions are identities, so diffusion is "average your neighbors". But many tasks have local coordinates/types, then we can use nontrivial sheaves for transporting [4]

- **3D meshes / geometry processing:** node fibers are tangent planes; restrictions are parallel transport. Tasks: smoothing/denoising vector fields, texture/UV stitching.
- **Multi-view vision / image stitching:** fibers carry image features; restrictions are homographies $H \in \mathrm{GL}(3)$. Tasks: panorama stitching, object reconstruction.
- **Heterophily graphs:** common fiber $\mathbb{R}^d$ with relation-specific linear maps. Tasks: link prediction, typed message passing.

[4] the trivial sheaf case reduces to the standard graph Laplacian

Network Sheaf: definition

Setup: Let $X = (V, E)$ be a graph with oriented edges E. Let $\mathcal{C}$ be a category (e.g., sets, groups, vector spaces).

Definition (Network Sheaf)

A network sheaf $\mathcal{F}$ on X with values in $\mathcal{C}$ consists of:

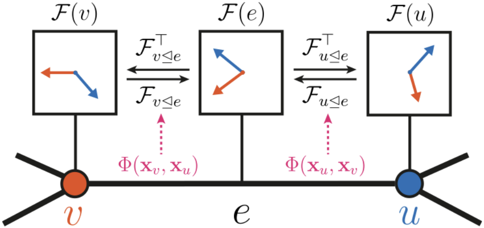
1. For each vertex $v \in V$, an object $\mathcal{F}(v) \in \mathcal{C}$, called the vertex stalk.
2. For each edge $e \in E$, an object $\mathcal{F}(e) \in \mathcal{C}$, called the edge stalk.
3. For each incidence $v \to e$, a restriction morphism $\mathcal{F}_{v\to e} : \mathcal{F}(v) \to \mathcal{F}(e)$ in $\mathcal{C}$.

Global Sections. A global section is an assignment $s_x \in \mathcal{F}(x)$ for all $x \in V \cup E$ such that for every edge $e = (u, v)$:

$$\mathcal{F}_{u\to e}(s_u) = \mathcal{F}_{v\to e}(s_v)$$

The set of global sections is denoted $\Gamma(X, \mathcal{F})$ or $H^0(X; \mathcal{F})$.

Network Sheaf: illustration



[5]

[5] Picture from *Neural Sheaf Diffusion*

Sheaf Laplacian: definition

Setup: Let $\mathcal{F}$ be a network sheaf of vector spaces[6] on $X = (V, E)$ with inner product on each stalk. The cochain spaces are

$$C^0(X;\mathcal{F}) = \prod_{v\in V} \mathcal{F}(v), \;\; C^1(X;\mathcal{F}) = \prod_{e\in E} \mathcal{F}(e)$$

Coboundary operator: For a 0-cochain $f = \{f_v\}$ and oriented edge $e = (u, v)$,

$$(\delta f)_e = \mathcal{F}_{v\to e}(f_v) - \mathcal{F}_{u\to e}(f_u)$$

Definition (Sheaf Laplacian):

Given the adjoint δ^* (with respect to the induced inner products),

$$\Delta = \delta^*\delta \; : \; C^0(X;\mathcal{F}) \longrightarrow C^0(X;\mathcal{F})$$

[6] for convenient, not a strict restriction

Sheaf Laplacian: example

$$\overset{K^T}{\begin{bmatrix} 1 & 1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}} \overset{K}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}} = \overset{L}{\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 2 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}}$$

Sheaf Laplacian is a generalization of the graph Laplacian. We can represent it as a symmetric block matrix with blocks indexed by the vertices of the complex. The entries on the diagonal are

$$\Delta^0_{v,v} = \sum_{v \leq e} \mathcal{F}^*_{v \leq e} \mathcal{F}_{v \leq e}$$

and the entries on the off-diagonal are

$$\Delta^0_{u,v} = -\mathcal{F}^*_{u \leq e} \mathcal{F}_{v \leq e}$$

Sheaf Diffusion Operator: Motivation

The sheaf Laplacian is just the generator, but using $\Delta_{\mathcal{F}}$ alone only yields a fixed one–step update $I - \alpha\Delta_{\mathcal{F}}$ or a minimization that easily degenerates. What we need for controlled propagation is a diffusion operator built from $\Delta_{\mathcal{F}}$. This provides explicit control of scale/time (t), an stability, and it can also be made learnable.

Sheaf Diffusion Operator: definition

Setup: Let $(G, \mathcal{F})$ be a cellular sheaf on a graph $G = (V, E)$ with vector-space stalks $\mathcal{F}(v)$ and $\mathcal{F}(e)$, and restriction maps defining the co-boundary $\delta : C^0(G, \mathcal{F}) \to C^1(G, \mathcal{F})$.

Diffusion PDE:

$$\frac{\partial X(t)}{\partial t} = L_{\mathcal{F}} X(t)$$

where $L_{\mathcal{F}} = \delta^\top \delta$ is the sheaf Laplacian.
The corresponding Euler discrete update is

$$X(t+1) = X(t) - L_{\mathcal{F}} X(t) = (I - L_{\mathcal{F}}) X(t)$$

Outline

Sheaf Neural Networks (SNNs)

Setup: Given a cellular sheaf $\mathcal{F}$ on a graph $X = (V, E)$ with vector-space stalks and restriction maps $\mathcal{F}_{v \to e} : \mathcal{F}(v) \to \mathcal{F}(e)$, the sheaf Laplacian is

$$L_{\mathcal{F}} = \delta^* \delta$$

where δ is the sheaf coboundary.

Core idea: Generalize graph neural networks (GNNs) by replacing the standard graph Laplacian with the sheaf Laplacian, allowing learnable restriction maps that encode heterogeneous local relations. It works for any message-passing models with the Laplacian included.

SNN layer (GCN as an example)

Let $p, q \in \mathbb{N}$. Stack p input features as columns: $X \in C^0 \otimes \mathbb{R}^p \cong \mathbb{R}^{nk \times p}$. Let $A \in \mathbb{R}^{p \times q}$ (feature mixing across channels) and $B \in \mathbb{R}^{k \times k}$ (within-stalk mixing). Write $I_n \otimes B$ for the block-diagonal map applying B at each vertex stalk, so $I_n \otimes B : C^0 \to C^0$. Extend a nonlinearity $\rho : \mathbb{R}^k \to \mathbb{R}^k$ stalkwise to $\rho : C^0 \to C^0$.

$$\mathrm{SheafConv}_{A,B}(X) := \rho\Big(D_{\mathcal{F}}\,(I_n \otimes B)XA\Big) \in C^0 \otimes \mathbb{R}^q$$

where $D_{\mathcal{F}}(h) = I + hL_{\mathcal{F}} \approx e^{-hL_{\mathcal{F}}}$ is a discrete approximation of SDO when h is small. In practice, we can take $h = \frac{1}{d_{max}}$ [7]

Note: the trivial sheaf case recovers a vanilla GCN layer

[7] Reference: *Sheaf Neural Networks*

Neural Sheaf Diffusion (NSD)

Motivation: SNN directly uses a fixed kernel $D_{\mathcal{F}}$ that defined by the sheaf Laplacian $I - \frac{1}{d_{max}} L_{\mathcal{F}}$ in the learning progress. It's an approximation of a specific diffusion. But the actual propagation path could be much more complicated. To solve this question, we can replace fixed diffusion with learnable spectral filters. [8]

Diffusion operator:

$$P_{\mathcal{F}} = I - \alpha \Delta_{\mathcal{F}}, \quad \alpha \in (0,1) \text{ learnable}$$

[8] Reference: *Neural Sheaf Diffusion*

Outline

Ubiquity: the same operator across physics

The (negative) Laplacian on $\mathbb{R}^n$ is $\mathrm{Lap} = -\sum_{i=1}^{n} \partial_{x_i}^2$; on a Riemannian manifold (M, g) it is the Laplace–Beltrami $= -\mathrm{div}_g \nabla_g$.

- **Steady potential flows / electrostatics:** harmonic potential u solves $\mathrm{Lap}(u) = 0$
- **Heat diffusion:** $\partial_t(u) = \kappa \, \mathrm{Lap}(u)$
- **Waves (fixed boundary)**: $\partial_t^2(u) = c^2 \, \mathrm{Lap}(u)$
- **Quantum (free particle):** $i\bar{h}\, \partial_t(u) = -\frac{\bar{h}^2}{2m} \mathrm{Lap}(u)$

Same operator, different physics $\Rightarrow$ geometry of space controls flow, heat, waves, and probability. [9]

[9] Reference of the section: *The Laplacian on a Riemannian Manifold*

Symmetry & characterization

Euclidean invariances

commutes with translations and rotations: for any rigid motion T,

$$(\mathrm{Lap}\, f) \circ T = \mathrm{Lap}\,(f \circ T).$$

- Among linear, local, second-order operators with rotational invariance and no preferred location, the Laplacian is (up to scale) the canonical choice.
- On (M, g), the natural, metric-compatible choice is the Laplace–Beltrami $= -\mathrm{div}_g \nabla_g$, invariant under isometries
- Analytic characterizations: mean value property and maximum principle single out harmonic functions ($\mathrm{Lap} u = 0$).

Separation of variables leads to an eigenproblem

For heat/wave/Schrödinger, try $u(x,t) = \alpha(t)\,\phi(x)$; we obtain the spatial Helmholtz problem

$$\mathrm{Lap}\phi = \lambda\,\phi \quad \text{(with boundary conditions).}$$

Time factors

$$\begin{aligned}
\text{Heat:}\quad \alpha(t) &= e^{-\lambda t}\\
\text{Wave:}\quad \alpha(t) &= e^{\pm i\sqrt{\lambda}\,t}\\
\text{Schr.:}\quad \alpha(t) &= e^{\pm i\lambda t}
\end{aligned}$$

Modal expansions

$$u(x,t) = \sum_k c_k\,\alpha_k(t)\,\phi_k(x),$$

where (λ_k, ϕ_k) are Laplacian eigenpairs adapted to the domain/metric.

Conclusion: dynamics reduce to *geometry via the spectrum* of Lap.

A fun inverse questions: "Can one hear the shape?"

- The spectrum determines many global quantities (dimension, volume; with boundary: area, perimeter, Euler characteristic via heat coefficients).
- But in general one cannot uniquely recover the exact shape: there exist non-isometric isospectral manifolds/domains.
- Still, spectral data is a powerful geometric probe in analysis, PDE, graphics, and data science.

Why special?

The Laplacian is the unique symmetry-respecting, metric-native second-order operator whose spectrum bridges geometry and dynamics.

References I

- Jakob Hansen and Thomas Gebhart.
  Sheaf Neural Networks.
  arXiv preprint arXiv:2012.06333, 2020.

- Cristian Bodnar, Francesco Di Giovanni, Benjamin Paul Chamberlain, Pietro Liò, and Michael M. Bronstein.
  Neural Sheaf Diffusion: A Topological Perspective on Heterophily and Oversmoothing in GNNs.
  arXiv preprint arXiv:2202.04579, 2022.
  https://arxiv.org/abs/2202.04579

- Matthew N. Bernstein.
  The graph Laplacian.
  Blog post, published November 11, 2020. https://mbernste.github.io/posts/laplacian_matrix/

References II

Steven Rosenberg.
The Laplacian on a Riemannian Manifold: An Introduction to Analysis on Manifolds.
London Mathematical Society Student Texts, vol. 31.
Cambridge University Press, 1997.