Learning Sheaf Laplacians from Smooth Signals

Jakob Hansen
Robert Ghrist

Cellular Sheaves

A cellular sheaf on a graph assigns data to vertices and edges of the graph, and uses linear maps to describe the relations the data satisfies. The slogan for sheaves is “local to global”—a sheaf describes local relationships that determine a global structure for a space of signals on a graph.

Stalks: vector spaces \( F(v) \) and \( F(e) \) for each vertex and edge

Restriction maps: linear transformations \( F_{v\to u} \) for each incident vertex-edge pair

Spaces of cochains: choices of \( x_e \in F(e) \) for all vertices \( v \); similarly for edges:

\[
C^0(G; F) = \bigoplus_v F(v) \quad C^1(G; F) = \bigoplus_e F(e)
\]

Global section: choice of \( x_v \in F(v) \) for all vertices \( v \), which is consistent over edges: \( F_{v\to u}x_u = F_{u\to v}x_v \) for every edge \( e = u-v \). Global sections form a vector space, denoted \( \Gamma(G; F) \).

Sheaves can be surprising! For instance, not every partial section extends to a global section. Not every sheaf even has nontrivial global sections. Sheaves have the potential to model a number of real-world phenomena. Anywhere global properties emerge from local interactions, a sheaf is lurking in the shadows. Making sheaf structure explicit can improve our understanding of the problems associated to data on networks.

Sheaf Laplacians

Coboundary map: a linear transformation \( \delta : C^1(G; F) \to C^0(G; F) \) measuring consistency over each edge:

\[
\delta(x_e) = F_{v\to u}x_u - F_{u\to v}x_v
\]

By construction, \( \ker(\delta) = \Gamma(G; F) \).

Given an inner product structure on the stalks of \( F \), we can form the sheaf Laplacian \( \Delta_{\delta} \). If \( F \) is the constant sheaf, this is the same as the graph Laplacian. In general, it has a block structure with two types of blocks: diagonal and off-diagonal.

The Laplacian quadratic form:

\[
(q(x), \lambda) = \|x\|^2 = \sum_{v \in V} \| F_{v\to u}x_u - F_{u\to v}x_v \|^2
\]

This measures how far \( x \) is from being a global section. The smallest eigenvalue of \( \Delta_{\delta} \) is the minimal value of \( (q(x), \lambda) \) for \( \|x\| = 1 \), and so corresponds to the cochain nearest to being a global section of \( F \).

Cochains for which the Laplacian quadratic form is small are smooth. The smoothest signals are therefore the sections of the sheaf.

How do we learn network structure from data?

Suppose we have signals supported on the vertices of some unknown network. How do we determine the network? We first need a model for how the network influences the signals.

One approach is to use a model where our signals are smooth on a network. This implies low variation over edges. 

\[
\min \sum_{x \in X} \lambda \|
\sum_{v \in V} x_v L_v + a f(L) + b g(S)
\|
\]

where \( f(L) \) is a regularization term controlling sparsity and \( g(L) \) a regularization term controlling connectivity of the graph.

This is a very simple model—it assumes that edges enforce similarity in signals. But more complicated structure may be present in the signals. An edge might instead encourage signals at its endpoints to be some scalar multiple of each other. If signals on vertices are higher-dimensional, this can be extended to a linear transformation. In essence, the signals on the network might be associated with a cellular sheaf describing the structure of the data, and recovering the network will require taking the sheaf structure into account. We must learn a sheaf Laplacian, not just a graph Laplacian.

Sparse inverse covariance estimation is a similar problem, relaxing the cone of graph Laplacians to the semidefinite cone. However, the structure of a learned sheaf is easier to interpret than that of a learned inverse covariance matrix.

Types of Sheaves

Constant Sheaves: In graph signal processing, one usually implicitly works with the constant sheaf on the graph, privileging constant signals on vertices.

\[
\begin{pmatrix}
1 & & \\
& & \\
& & 1
\end{pmatrix}
\]

Matrix-Weighted Graphs: Choosing the two restriction maps on each edge to be equal ensures that the off-diagonal blocks of the Laplacian are semidefinite; we can view these as matrix-valued weights on edges. Constant functions are still sections, but there may be more.

\[
\begin{pmatrix}
W & & \\
& & \\
& & -W
\end{pmatrix}
\]

Connection Graphs: Each edge has an orthogonal linear transformation indicating the relationship between data in \( \text{R}^n \) on either end of the edge. (cf. Singer & Wu)

\[
\begin{pmatrix}
l_n & -\rho & \\
-\rho & l_n
\end{pmatrix}
\]

General Sheaves: Any choice of restriction maps is permitted. The contribution of an edge to the sheaf Laplacian can be any \( 2n \times 2n \) positive semidefinite matrix.

Each type of sheaf has an associated convex cone of Laplacians. In general, membership of a matrix \( L \) in such a cone can be checked by solving a feasibility problem: whether \( L \) can be written as a sum of block matrices of the above forms, one corresponding to each edge.

Numerical Experiments

Generate Random Sheaves: Generate Erdos-Renyi graphs for base space. Then choose restriction maps:

- General sheaves: Gaussian restriction maps \( F_{v\to u} \)
- Connection graphs: restriction maps uniform in \( \text{O(n)} \)
- Matrix-weighted graphs: Gaussian restriction maps \( F_{v\to u} = F_{u\to v}^{-1} \)

Generate Smooth Signals: Choose signals \( x \in C^2(G; F) \sim \mathcal{N}(0, 1) \) Smooth by Tikhonov filtering with Laplacian Normalize signals in \( L^2 \)

Recover Laplacians: Optimize over different cones
- Sheaf Laplacians
- Connection Laplacians (not convex)
- Positive semidefinite matrices
- Matrix-weighted Laplacians

Compute Error: Normalize matrices and compute \( L^1 \) and \( L^2 \) relative error for the Laplacian and adjacency matrices. Also compute the F-score for the sparsity pattern.