Journal Club: The Emerging Field of Signal Processing on Graphs

Matt Bartos

University of Michigan

mdbartos@umich.edu

November 29, 2017

Overview

1. Other work on graph signals
2. Contribution of Shuman et al.
   - The Graph Fourier Transform
   - Signal smoothness on a graph
   - Graph signal operations
   - Graph coarsening, downsampling and reduction
   - Graph wavelets
3. Strengths and weaknesses
4. Extensions and applications

Other work on graph signals

Other work on graph signals

SNFOV Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., & Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. Signal Processing Magazine, IEEE, 30(3), 83-98.

ZFC Zhang, C., Florencio, D., & Chou, P. A. (2015). Graph signal processing: a probabilistic framework. Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2015-31.

MSLR Marques, A. G., Segarra, S., Leus, G., & Ribeiro, A. (2016). Sampling of graph signals with successive local aggregations. IEEE Transactions on Signal Processing, 64(7), 1832-1843.

PV Perraudin, N., & Vandergheynst, P. (2017). Stationary signal processing on graphs. IEEE Transactions on Signal Processing, 65(13), 3462-3477.

Contribution of Shuman et al.

Contribution of Shuman et al.

This is a tutorial review paper:

- Reviews existing work
- Provides definitions for graph signal operations and transforms (e.g. graph signal translation, graph Fourier transform)
- Provides example implementations (e.g. Tikhonov regularization, wavelet filtering)
- Summarizes open issues in the field

Topics

- Challenges of signal processing on graphs
- The Graph Laplacian
- Definition of the Graph Fourier transform
- Metrics of signal smoothness
- Generalized operators for signals on graphs
  - Filtering
  - Translation
  - Convolution
  - Modulation and Dilation
- Graph coarsening
- Graph wavelet filtering

Challenges of graph signal processing

How can we extend traditional signal processing tools to graphs?

- Translation, downsampling and modulation of signals in the graph domain
- How to implement filtering operations on graphs

Challenges:

- Graphs are irregular structures that lack a shift-invariant notion of translation.
- Modulation is nontrivial, given that the graph frequency spectrum is discrete and irregularly spaced.
- Downsampling is nontrivial, as there is no notion of "every other vertex".
- How can we capture the structure of the graph in downsampling operations?

This presentation is interactive.

Follow along at:
`github.com/mdbartos/graph-signals`

The Graph Fourier Transform

Review of the Fourier Transform

In the traditional Euclidean domain, we can find the frequency-domain representation of a time series signal using the Fourier transform:

Classical Fourier Transform

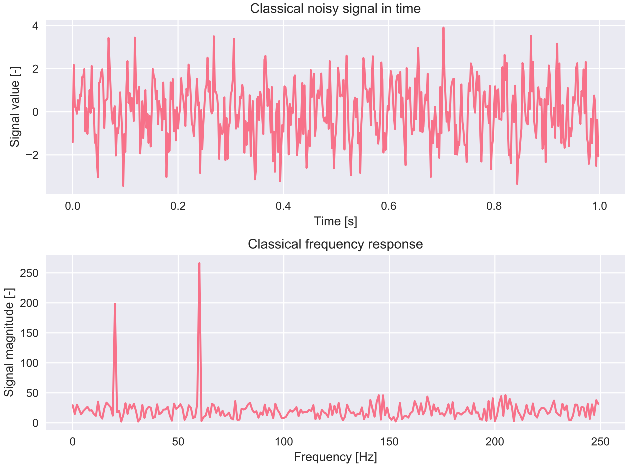
$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(t) e^{-2\pi j \xi t} dt \tag{1}$$

$\hat{f}$ is an expansion of f in terms of complex exponentials, which are the eigenfunctions of the Laplace operator:

Classical Fourier transform in terms of the Laplace operator

$$-\Delta(e^{2\pi j \xi t}) = -\frac{\partial^2}{\partial t^2} e^{2\pi j \xi t} = 4\pi^2 \xi^2 e^{2\pi j \xi t} = k e^{2\pi j \xi t} \tag{2}$$

Classical Fourier Transform



The Graph Laplacian

For each vertex i, the Laplace operator computes the weighted sum of the differences between the signal value at i and the signal value at $i's$ neighbors ($j \in N_i$).

The Laplacian is a difference operator

$$(\Delta f)(i) = (Lf)(i) = \sum_{j \in N_i} W_{i,j}[f(i) - f(j)] \tag{3}$$

The Laplace operator for an undirected graph is simply the degree matrix minus the adjacency matrix.

The Graph Laplacian

$$L = D - W \tag{4}$$

L will have a full set of orthonormal eigenvectors, and real eigenvalues. Zero will occur as an eigenvalue with multiplicity equal to the number of connected components.

The Graph Fourier Transform

The Graph Fourier transform is an expansion of f in terms of the eigenvectors u_l of the Graph Laplacian.

The Graph Fourier Transform

$$\hat{f}(\lambda_l) = \sum_{i=1}^{N} f(i) u_l^*(i) \tag{5}$$

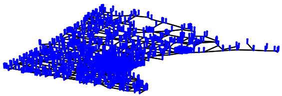
The Graph Fourier Transform (alt.)

$$\hat{\mathbf{f}} = U^* \mathbf{f} \tag{6}$$

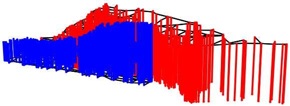
- Eigenvectors associated with the smallest eigenvalues vary slowly across the graph.
- Eigenvectors associated with the largest eigenvalues oscillate rapidly.

Example: Eigenvectors of the Laplacian of the Minnesota Road Network

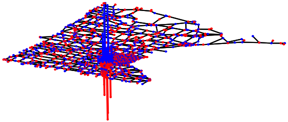
u_0



u_1



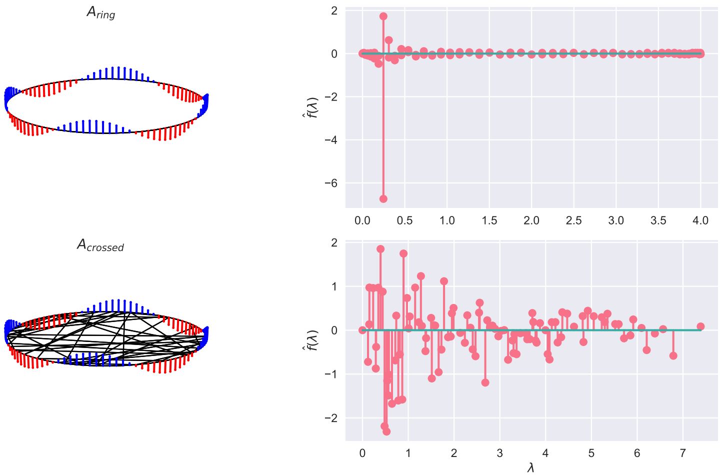
u_{N-1}



Example: Fourier transform of a graph signal on a ring graph

Signal smoothness on a graph

Signal smoothness depends on underlying structure of graph



Measuring local signal smoothness on a graph

The edge derivative of f with respect to edge $e = (i, j)$

$$\left.\frac{\partial f}{\partial e}\right|_i = \sqrt{W_{i,j}}[f(j) - f(i)] \tag{7}$$

The local variation can be measured by the square root of the sum of the squared differences between signal values at adjacent vertices.

The local variation at vertex i

$$||\nabla_i f|| = \left[\sum_{\text{e connected to i}} \left(\left.\frac{\partial f}{\partial e}\right|_i\right)^2\right]^{1/2} = \left[\sum_{j \in N_i} W_{i,j}[f(j) - f(i)]^2\right]^{1/2} \tag{8}$$

Measuring global signal smoothness on a graph

Discrete p-Dirichlet form of f

$$S_p(f) = \frac{1}{p} \sum_{i \in V} \left[\sum_{j \in N_i} W_{i,j} [f(j) - f(i)]^2 \right]^{\frac{p}{2}} \tag{9}$$

For $p = 1$, S_1 is simply the sum of local variations across all vertices.
For $p = 2$, S_2 is a quadratic function of the Laplacian:

Graph Laplacian Quadratic Form

$$\begin{aligned} S_2(f) = \frac{1}{2} \sum_{i \in V} \left[\sum_{j \in N_i} W_{i,j} [f(j) - f(i)]^2 \right]^{\frac{1}{2}} &= \sum_{(i,j) \in \epsilon} W_{i,j} [f(j) - f(i)]^2 \\ &= f^T L f \end{aligned} \tag{10}$$

S_2 is small when f has similar values at strongly-connected vertices.

Alternatives to the Graph Laplacian

Normalized Graph Laplacian

$$\tilde{L} = D^{-1/2} L D^{-1/2} \tag{11}$$

The eigenvalues of $\tilde{L}$ will be between 0 and 2. For bipartite graphs, the spectral folding phenomenon can be used.

Random Walk Matrix

$$P = D^{-1} W \tag{12}$$

Asymmetric Graph Laplacian

$$L_a = I - D^{-1} W \tag{13}$$

Graph signal operations

Filtering in the frequency/graph spectral domain

Using some transfer function $\hat{h}$, we can filter an input signal as follows:

Classical frequency filtering

$$f_{out}(t) = \mathcal{F}^{-1}\{\hat{f}_{in}(\xi)\hat{h}(\xi)\} \tag{14}$$

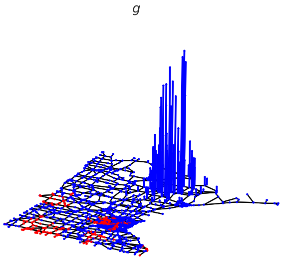
In the graph setting:

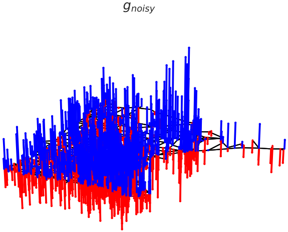
Graph filtering in the graph spectral domain

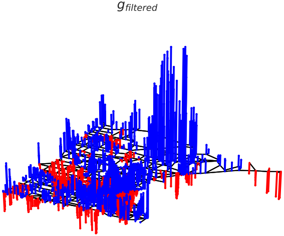
$$\hat{h}(L) = U \begin{bmatrix} \hat{h}(\lambda_0) & & 0 \\ & \ddots & \\ 0 & & \hat{h}(\lambda_{N-1}) \end{bmatrix} U^* \tag{15}$$

$$f_{out} = \hat{h}(L) f_{in}$$

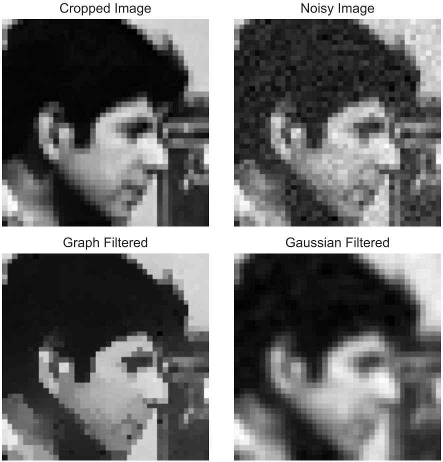
Example: Gaussian Filtering







Filtering example: Tikhonov regularization



Filtering in the time/vertex domain

We can also filter in the time domain using convolution:

Classical time-domain filtering

$$f_{out}(t) = (f_{in} * h)(t) \tag{16}$$

In the graph setting, the output at any vertex i is a linear combination of the elements of the input signal within a K-hop neighborhood (for some constants b):

Graph filtering in the vertex domain

$$f_{out}(i) = b_{i,i} f_{in}(i) + \sum_{j \in N(i,K)} b_{i,j} f_{in}(j) \tag{17}$$

Equivalence of vertex/spectral filtering

If the frequency filter is a K-order polynomial $\hat{h} = \sum_{k=0}^{K} a_k \lambda_l^k$, the frequency filtered signal at vertex i is a linear combination of the elements of the input signal at vertices within a K-hop neighborhood:

Frequency filtering when the filter is a K-order polynomial

$$f_{out}(i) = b_{i,i} f_{in}(i) + \sum_{j \in N(i,K)} \sum_{dG(i,j)}^{K} a_k (L^k)_{i,j} f_{in}(j) \tag{18}$$

Convolution

Although we cannot directly generalize a convolution product on a graph because $h(t-\tau)$ is undefined, we can use frequency filtering, as previously defined:

Convolution of a signal on a graph

$$(f * h)(i) = \sum_{l=0}^{N-1} \hat{f}(\lambda_l)\hat{h}(\lambda_l)u_l(i) \tag{19}$$

Translation

Classical translation operation

$$(T_v f)(t) = f(t - v) = (f * \delta_v)(t) \tag{20}$$

Again, we cannot directly generalize $(t - v)$ for a graph, so we consider instead the definition of translation as convolution with a Dirac delta.

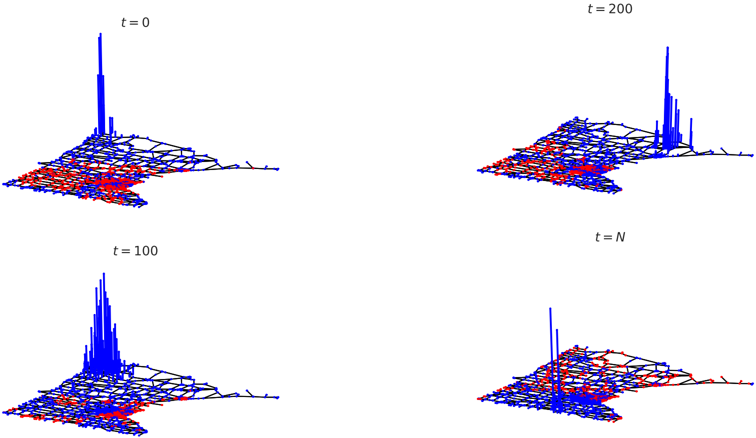
Translation of a graph signal

$$(T_n f)(i) = \sqrt{N}(f * \delta_n)(i) = \sqrt{N} \sum_{l=0}^{N-1} \hat{f}(\lambda_l) u_l^*(n) u_l(i) \tag{21}$$

Where:

$$\delta_n = \begin{cases} 1 \; if \;\; i = n \\ 0 \; otherwise \end{cases} \tag{22}$$

Graph signal translation example



Modulation

In simple terms, like a "translation" in the frequency domain.

Classical modulation

$$\begin{aligned} \text{Time domain: } (M_\omega f)(t) &= e^{2\pi j \omega t} f(t) \\ \text{Frequency domain: } \overline{M_\omega f}(\xi) &= \hat{f}(\xi - \omega) \end{aligned} \tag{23}$$

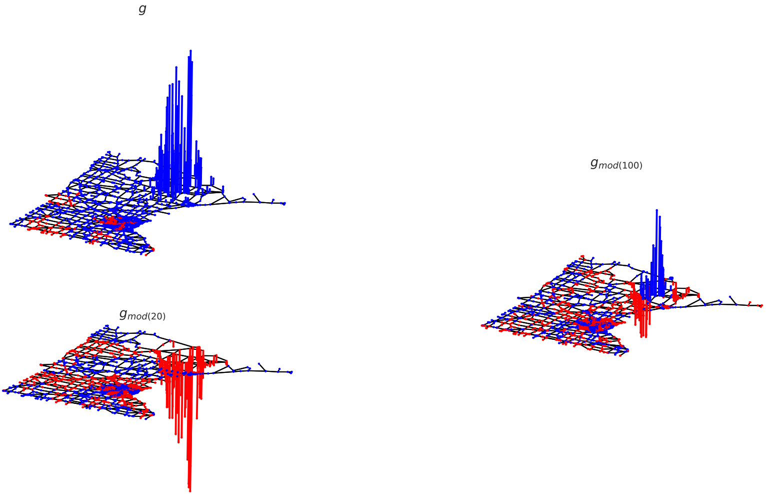
Replace complex exponential with a graph Laplacian eigenvector:

Graph modulation

$$(M_k f)(i) = \sqrt{N} u_k(i) f(i) \tag{24}$$

If a kernel f is localized around 0 in the graph spectral domain, then $\overline{M_k g}$ is localized around λ_k.

Graph signal modulation example



Dilation

Classical dilation

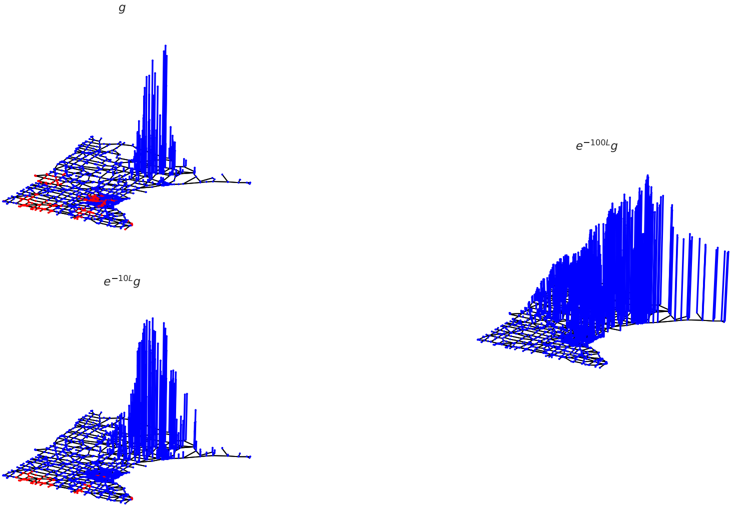
$$\text{Time domain: } (D_s f)(t) = \frac{1}{s} f\left(\frac{t}{s}\right) \qquad (25)$$
$$\text{Frequency domain: } \overline{D_s f}(\xi) = \hat{f}(s\xi)$$

Replace the frequency ξ with an eigenvalue of the Laplacian.

Graph dilation

$$(D_s f)(\lambda) = \hat{f}(s\lambda) \qquad (26)$$

Graph signal dilation example

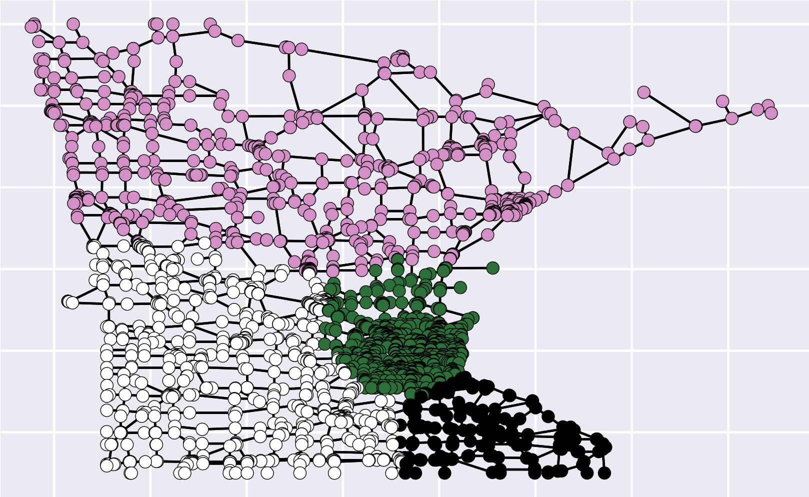


Graph coarsening, downsampling and reduction

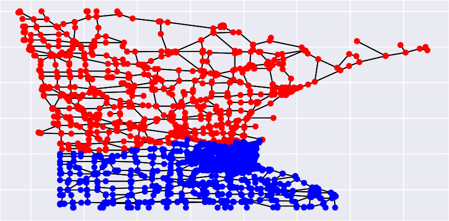
Graph coarsening, downsampling and reduction

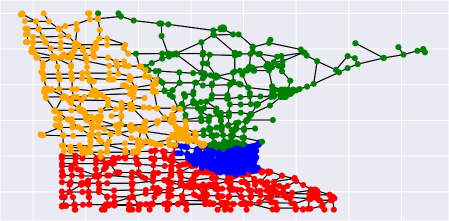
- For bipartite graphs, one can recursively downsample by a factor of two
- Downsampling based on diffusion distances
- Greedy seed selection algorithm
- Recursive spectral bisection
- Minimize number of edges connecting two vertices in a downsampled subset

Example: Spectral clustering



Example: Recursive spectral bisection





Graph wavelets

Localized multiscale transforms

Measuring the spread of graph signals in both time and frequency domains:

Spatial spread of a signal f around a center vertex i

$$\Delta^2_{G,i}(f) = \frac{1}{||f||_2^2} \sum_{j \in V} [d_G(i,j)]^2 [f(j)]^2 \qquad (27)$$

- Where $d_G(i,\cdot)$ is the geodesic distance function.
- $[f(j)]^2/||f||^2$ represents the pmf of signal f.
- $\Delta^2_{G,i}$ is the variance of the geodesic distance function at node i.

Localized multiscale transforms

The spatial and spectral spreads can thus both be characterized:

Total spatial spread of a graph signal

$$\Delta_G^2(f) = min_{i \in V}\{\Delta_{G,i}(f)\} \tag{28}$$

Total spectral spread of a graph signal

$$\Delta_\sigma^2(f) = min_{\mu \in \mathcal{R}}\left\{\frac{1}{||f||_2^2} \sum_{\lambda \in \sigma(L)} [\sqrt{\lambda} - \sqrt{\mu}]^2 [\hat{f}(\lambda)]^2\right\} \tag{29}$$

Wavelets in the vertex domain

The wavelet function Ψ at scale k and center vertex i is defined by:

Wavelet in the vertex domain around a vertex i

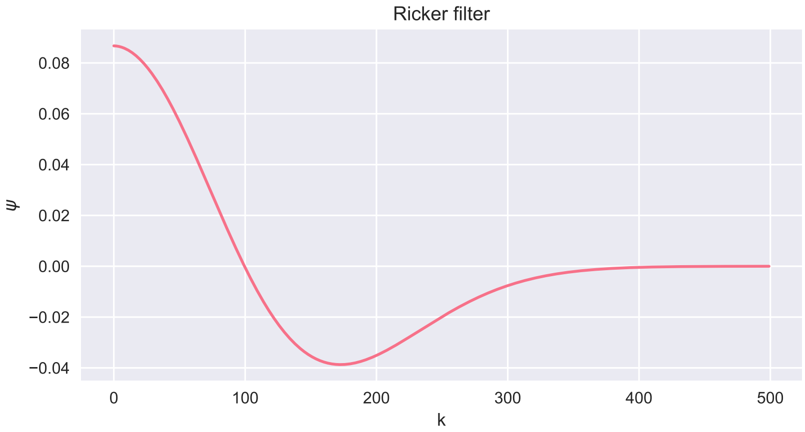
$$\Psi_{k,i}^{CKWT}(j) = \frac{a_{k,\tau}}{|\partial N_{(i,\tau)}|}, \forall j \in \partial N(i,\tau) \tag{30}$$

Where $\partial N_{(i,\tau)}$ is the set of all vertices $j \in N$ such that the geodesic distance between i and j is τ. $a_{k,\tau}$ is a set of coefficients that approximate the continuous wavelet function.

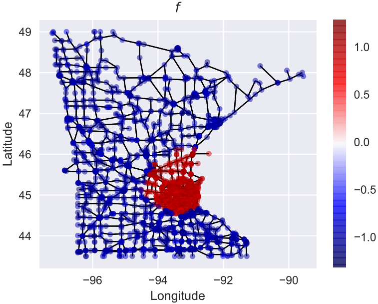
Wavelet Transform

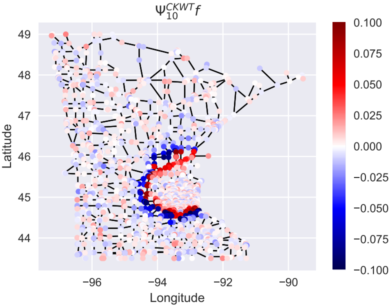
$$\boldsymbol{\Psi}_{k}^{CKWT} = [\Psi_{k,1}^{CKWT}; \Psi_{k,2}^{CKWT} \cdots \Psi_{k,N}^{CKWT}] \tag{31}$$

Example: Ricker wavelet

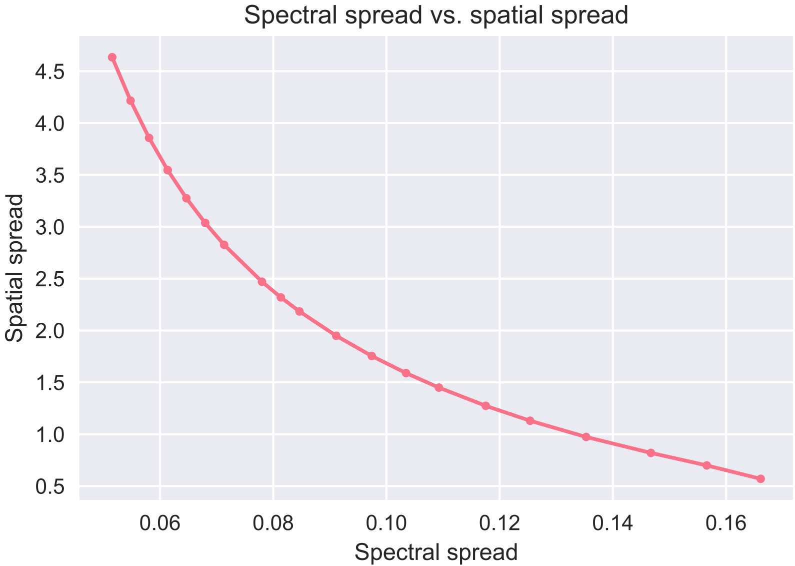


Example: Applying Ricker Wavelet Transform for edge detection





Example: Tradeoff between spectral and spatial spread



Strengths and weaknesses

Strengths and weaknesses

Strengths:

- Highly accessible
- Comprehensive introduction to signal processing on graphs
- Excellent examples and illustrations from different fields

Weaknesses:

- Doesn't emphasize computational difficulty of several proposed methods
- Some notation could be simplified
- Some assertions aren't proved or expanded upon
- Couldn't find any code

Extensions and applications

Extensions and applications

- Little is known about how the structure of the graph affects transforms
- Unclear when to use different graph matrices (Laplacian, Normalized Laplacian, etc.)
- Unclear when to use different distance metrics (geodesic, algebraic, diffusion, resistance, etc.)
- Computing the eigendecomposition of the Laplacian is **slow**

Appendix

For an infinite square lattice grid, it can be shown that the Graph Laplacian corresponds to the continuous Laplacian as $\epsilon \to 0$:

Equivalence between Continous and Graph Laplacians

$$\frac{\partial^2 F}{\partial x^2} = \lim_{\epsilon \to 0} \frac{[F(x+\epsilon) - F(x)] + [F(x-\epsilon) - F(x)]}{\epsilon^2} \tag{32}$$