Graph Signal Processing

Methods and Applications

James Sharpnack

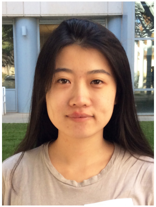
JSM 2018

UC Davis, Statistics Department
Work supported by NSF DMS-12-23137

Table of contents

Collaborators

Oscar Hernan
Madrid Padilla (Postdoc)



Shitong Wei
(UCD Grad Student)

Veeranjaneyulu Sadhanala
(CMU Grad Student)

And many others... Ryan Tibshirani, Robert Bassett, Yu-Xiang Wang, Alex Smola, Aarti Singh, Akshay Krishnamurthy, Daniela Witten, Yanzhen Chen

Introduction

Graph Signal Processing

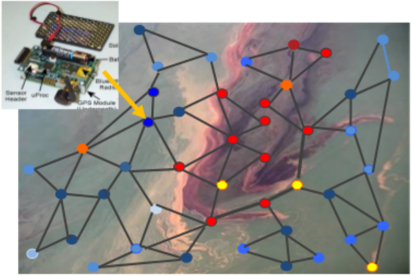


Figure 1: Water contamination sensors

Figure 2: Air traffic graph for flu counts

Graph signal processing: noisy measurement y_v on vertex v in graph G

Social Media Applications

Anomalous Behavior in Groups

Best Ad Placement

Detecting Viral Content

Image from http://williamjturkel.net

Applications in this Talk

GSP is a *flexible* framework that admits *efficient algorithms*. In this talk we will see...

- non-parametric function estimation,
- road network segmentation,
- network modeling with graphons

Methods for GSP

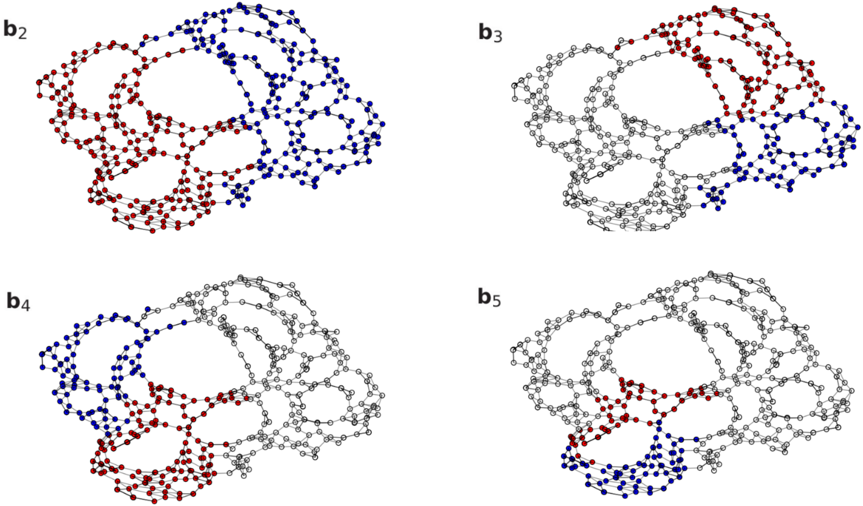
Methodologies for GSP...

- graph kernels [Smola and Kondor, 2003],
- graph wavelets [Gavish et al., 2010, Crovella and Kolaczyk, 2003],
- graph trend filtering [Wang et al., 2016].

—— representative publications above

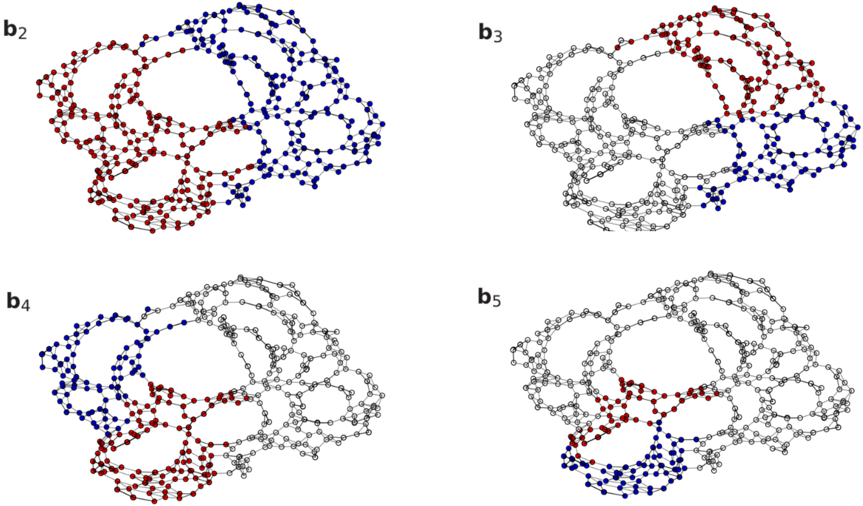
Methods for Graph Signal Processing

Graph Wavelets



Haar wavelet basis based on random spanning trees
[Sharpnack et al., 2013]

Graph Wavelets



Challenges: determining a good hierarchical decomposition of graph to produce wavelets.

Graph Derivatives

Let j be an edge between vertices i_0, i_1 then $\nabla \in \mathbb{R}^{m \times n}$ (the graph has m edges, n vertices)

$$\nabla_{j,i_0} = \sqrt{W_{i_0,i_1}}, \quad \nabla_{j,i_1} = -\sqrt{W_{i_0,i_1}}.$$

Then $\Delta^{(1)} := \nabla$ is the graph derivative and it turns out that $\Delta^{(2)} := \Delta = \nabla^\top \nabla$ is the combinatorial Laplacian. We can define higher order graph derivatives

$$\Delta^{(k+1)} = \nabla^\top \Delta^{(k)}, \quad k \text{ is odd}$$
$$\Delta^{(k+1)} = \nabla \Delta^{(k)}, \quad k \text{ is even.}$$

k is called the 'order' of the derivative.

- Many graph kernels are functions of these operators.
- Spectrum of derivatives yield graph Fourier transform.

Graph Trend Filtering

Define the kth order **Graph Trend Filter** to be

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \|\boldsymbol{y} - \boldsymbol{x}\|_2^2 + \lambda \|\Delta^{(k+1)} \boldsymbol{x}\|_1$$

which evaluates to (for unweighted graphs)

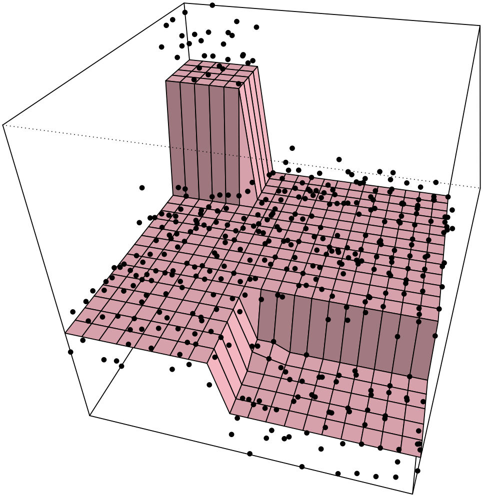
$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \|\boldsymbol{y} - \boldsymbol{x}\|_2^2 + \lambda \sum_{i,j: i \sim j} |x_i - x_j|, \quad k = 0 \quad \text{(Fused Lasso)}$$

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} \|\boldsymbol{y} - \boldsymbol{x}\|_2^2 + \lambda \sum_{i} \left| \sum_{j: j \sim i} (x_i - x_j) \right|, \quad k = 1 \quad \text{(2nd order)}$$

Graph Sobolev norm regularization with $p = 1$. [Wang et al., 2016]

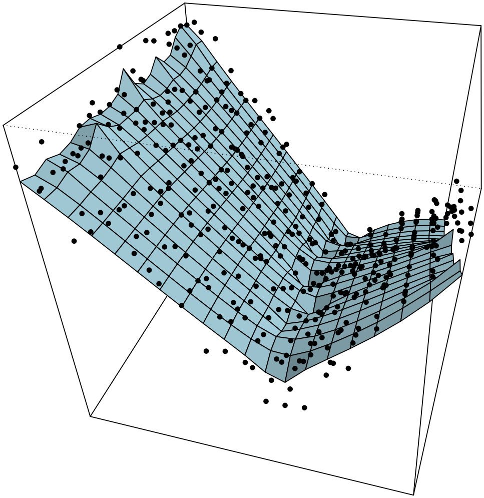
Piecewise Polynomials on Graphs

GTF with $k = 0$



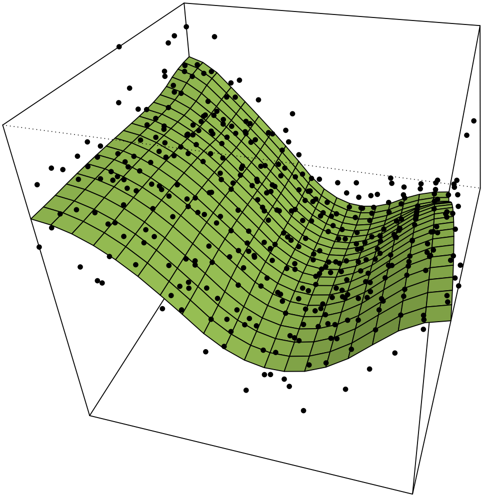
Piecewise Polynomials on Graphs

GTF with $k = 1$

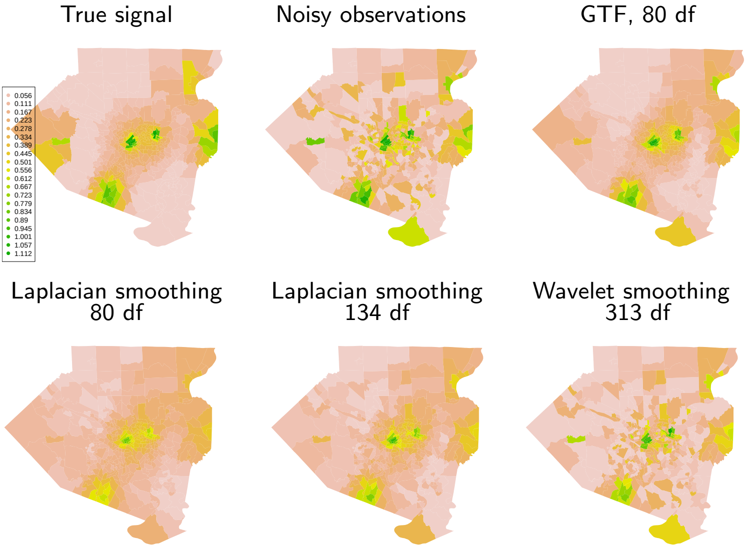


Piecewise Polynomials on Graphs

GTF with $k = 2$



Local Adaptivity



Non-parametric estimation with K-nearest neighbors GTF

KNN GTF

Given samples $\{x_i, y_i\}_{i=1}^n$ over a metric space $(\mathcal{X}, d_{\mathcal{X}}) \times \mathbb{R}$ do:

1. Form the K-nearest neighbor graph G_K over $\{x_i\}_{i=1}^n$ with an undirected edge for every K-NN.
2. Run the graph trend filtering over $\mathbf{y}, G_K$ yielding $\hat{\mathbf{y}}$.
3. Predict for new points $x^\star$ by standard KNN prediction with $\{x_i, \hat{y}_i\}_{i=1}^n$.

KNN GTF

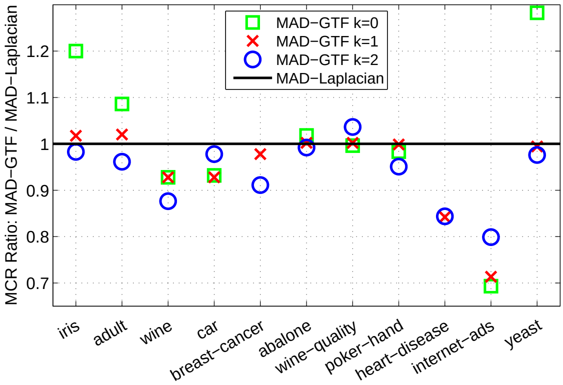


Figure 3: Ratio of the misclassification rate of 5NN-GTF to 5NN Laplacian regularization, for graph-based semi-supervised learning, on the 11 most popular UCI classification data sets (MAD refers to modified absorption problem with prior). [Wang et al., 2016]

KNN Fused Lasso

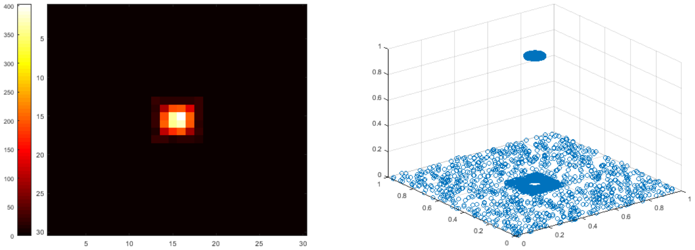


Figure 4: A heatmap of $n = 5000$ draws from a density (left) that is higher around where there is a change in the regression function (right) [Padilla et al., 2018].

KNN Fused Lasso

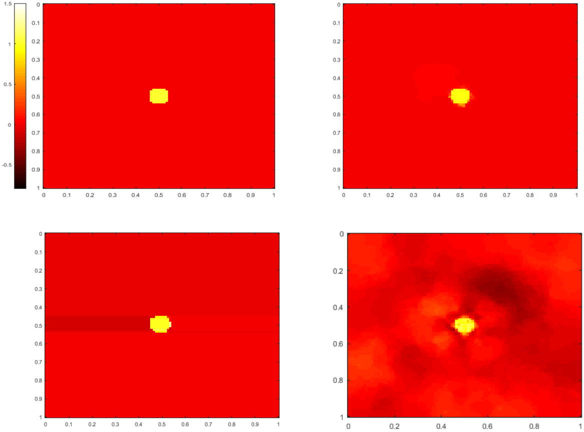


Figure 5: *Top Left:* The regression function evaluated at an evenly-spaced grid of size 100×100 in $[0,1]^2$. *Top Right:* The estimate obtained via K-NN-FL. *Bottom Left:* The estimate of obtained via CART. *Bottom Right:* The estimate of obtained via K-NN regression [Padilla et al., 2018].

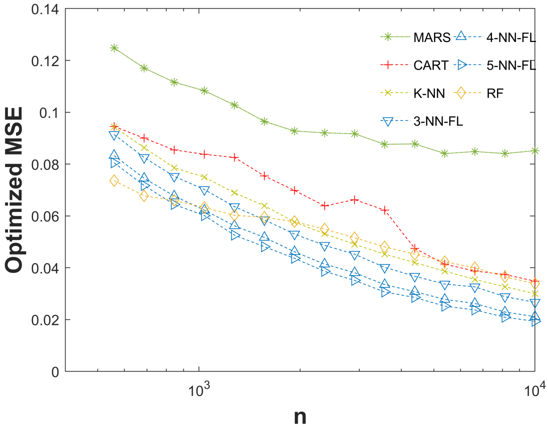


Figure 6: The MSE comparisons for the previous slide simulation.

KNN Fused Lasso

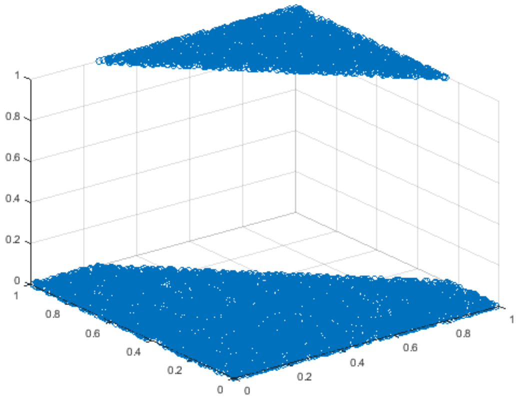


Figure 7: A new simulation with uniform X density, but boundary that is not axis aligned.

KNN Fused Lasso

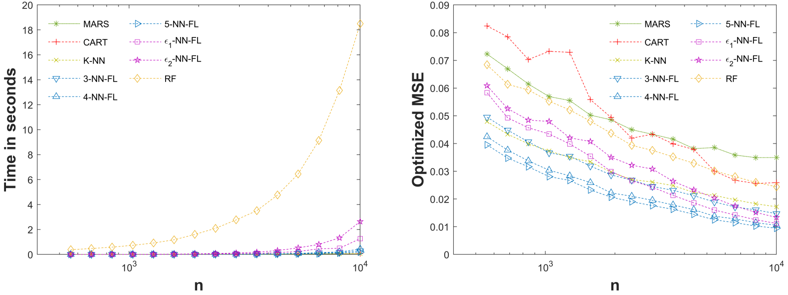
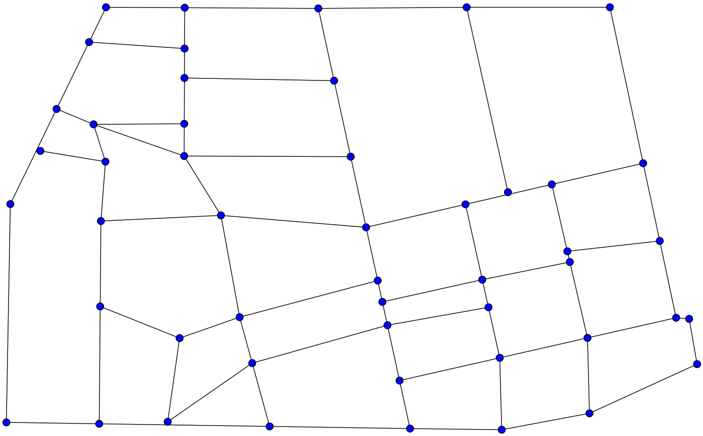


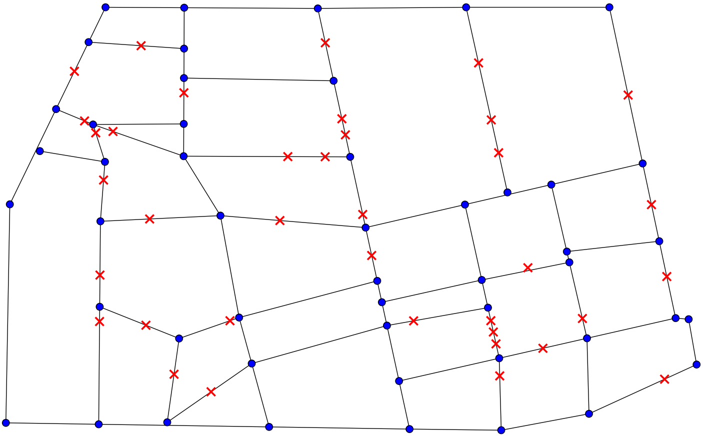
Figure 8: Computational time in seconds (left), averaged over 150 Monte Carlo simulations and optimized MSE (right).

Density estimation over transportation networks

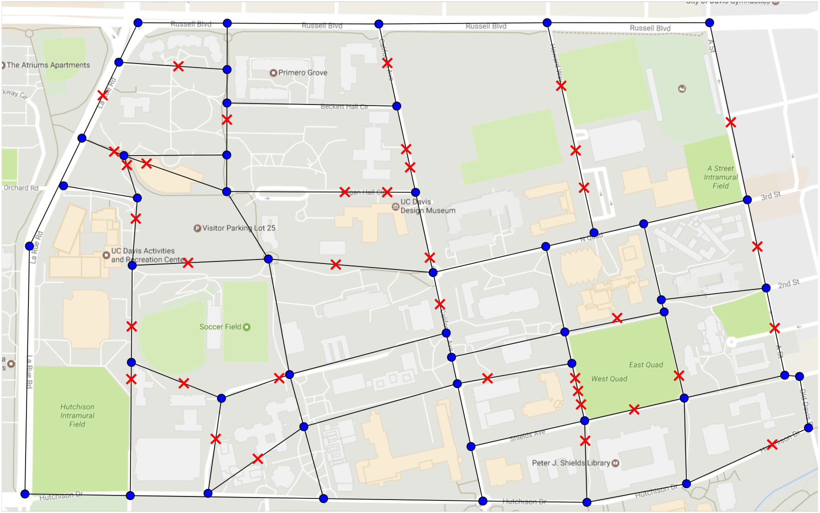
Processes on Geometric Networks



Processes on Geometric Networks



Processes on Geometric Networks



Log-density Estimation on Geometric Networks

$$\min_{g \in BV} \frac{-1}{n} \sum_{i=1}^{n} g(x_i) + \lambda \mathrm{TV}(g)$$
$$\text{s.t.} \int_G e^g = 1$$

- g is log-density
- TV is the total variation (if g is differentiable then this is $\int |g'|$) over the geometric network
- Geometric network is a collection of 1D manifolds that end and start at vertices.

Representer Theorem

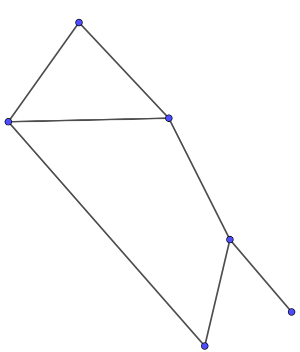
Proposition [Bassett and Sharpnack, 2018]

Total variation penalized density estimation can be solved with the following program,

$$\min_{f \in \mathbb{R}^n} \frac{1}{2} f^T S f + v^T f + \|\nabla f\|_1$$

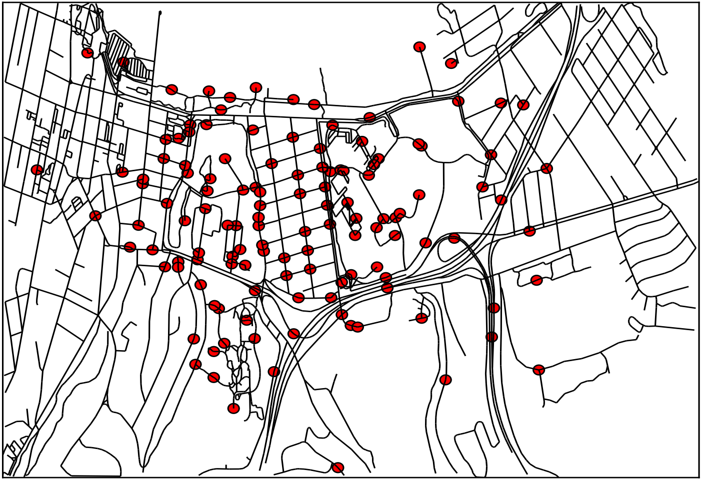
has the same optimality conditions e^g where S is a diagonal matrix and v is a vector.

Augmented graph

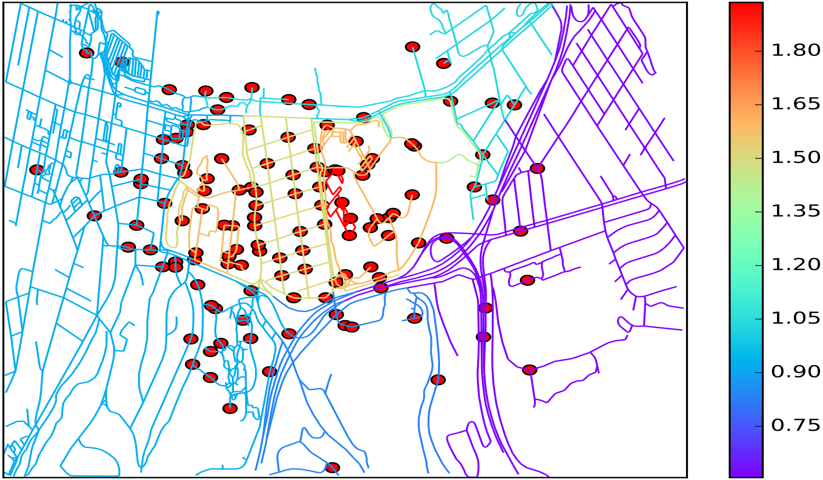


∇ is the graph derivative for an expanded graph over the observed points and the original vertices.

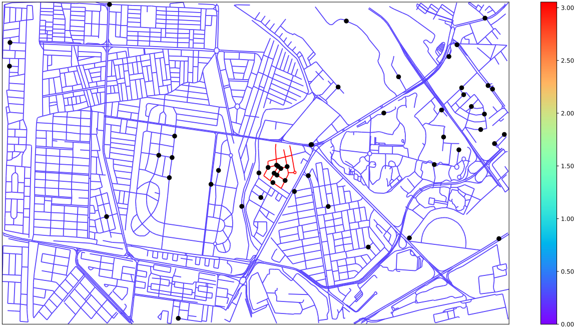
Simulated crash data



Simulated crash data



Terrorist Incident Localization



Terrorist incidents from 2013 to 2016, according to the Global Terrorism Database within a road network in Baghdad.

Matrix denoising via Cartesian product GTF

Cartesian Power Graphs

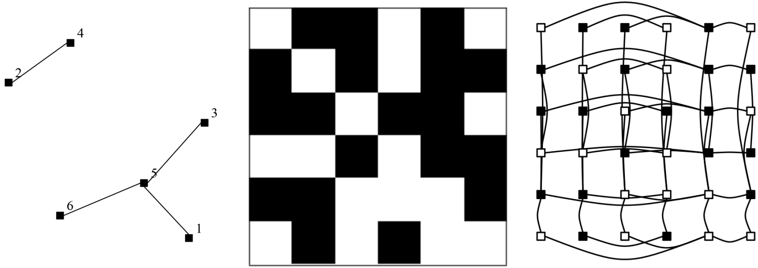


Figure 9: KNN-PGFL method: the 2-NN graph G_2 (left) is learned from the adjacency matrix A (middle) of the network H, then fused lasso is applied to Cartesian power graph, $G_2^{\square 2}$, with the A_{ij} dyadic labels (right).

Power Graph Fused Lasso

(PGFL) is the solution to

$$\min_{P \in \mathbb{R}^{n \times n}} \|A - P\|_F^2 + \lambda \left(\|\nabla P\|_1 + \|\nabla P^\top\|_1 \right). \tag{1}$$

To see that the RHS of (1) is the TV penalty over the Cartesian power graph, notice that

$$\begin{aligned} \|\nabla P\|_1 + \|\nabla P^\top\|_1 &= \sum_{k=1}^{n} \|\nabla P_k\|_1 + \|\nabla (P^\top)_k\|_1 \\ &= \sum_{(i,j) \in E, k \in V} \left(|P_{k,i} - P_{k,j}| + |P_{i,k} - P_{j,k}| \right), \end{aligned}$$

where P_k is the kth column of P.

Graphon Models

Graphon model

- Incidence matrix A_{ij} has independent Bernoulli entries with probability matrix P_{ij}.
- Uniform$[0,1]$ latent variables ξ_i for each vertex i
- Graphon f is bivariate function such that $P_{ij} = f(\xi_i, \xi_j)$

[Zhang et al., 2015] estimated a smooth graphon f by first estimating a distance metric $\hat{d}$ between vertices and smoothing the matrix A wrt $\hat{d}$.

Fused Graphon Estimation

Input: network H with adjacency matrix A, tuning parameter $\lambda > 0$

Output: partition of $V^{\times 2}$, $\mathcal{S}$, and an estimate $\hat{P}$ of $\mathbb{E}A$.

1. Calculate the $\hat{d}_1$ distance matrix $\hat{D}_1 = (\hat{d}_1(i,j))_{i,j}$ (or any distance);
2. Generate the undirected KNN graph G_K: $(i,j) \in E$ if i is a KNN of j or vice versa;
3. Calculate $\hat{P}$ with Distributed PGFL on G_K, A, λ.

——see details in [Wei et al., 2018]

Graphon Estimation

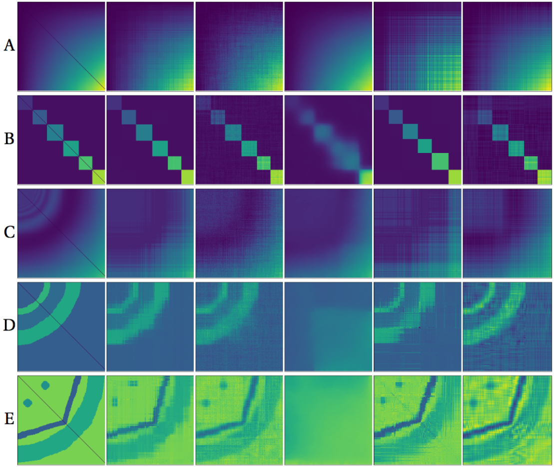


Figure 10: Plots left to right are the true probability matrix, the estimates using PGFL, Neigh Smoothing, Sorting-and-smoothing, SBM, and USVT.

Graphon Estimation

Table 1: Mean-square error comparisons

Method	Graphon A	B	C	D	E
KNN-PGFL	7.39	**3.10**	17.54	**34.91**	**61.08**
Neigh. Smooth	13.68	9.55	17.16	45.18	66.76
SAS	**6.29**	9.20	23.68	97.90	190.38
SBM	37.65	6.60	35.77	44.45	62.68
USVT	7.05	9.61	**12.24**	50.34	71.94

Thanks for your time

http://jsharpna.github.io

Bassett, R. and Sharpnack, J. (2018).
Fused density estimation: Theory and methods.
arXiv preprint arXiv:1805.03288.

Crovella, M. and Kolaczyk, E. (2003).
Graph wavelets for spatial traffic analysis.
In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1848–1857. IEEE.

Gavish, M., Nadler, B., and Coifman, R. R. (2010).
Multiscale wavelets on trees, graphs and high dimensional data: Theory and applications to semi supervised learning.
In *ICML*, pages 367–374.

References ii

Padilla, O. H. M., Sharpnack, J., Chen, Y., and Witten, D. (2018).
Adaptive non-parametric regression with the k-nn fused lasso.
ArXiV preprint.

Sharpnack, J., Singh, A., and Krishnamurthy, A. (2013).
Detecting activations over graphs using spanning tree wavelet bases.
In *Artificial Intelligence and Statistics*, pages 536–544.

Smola, A. J. and Kondor, R. (2003).
Kernels and regularization on graphs.
In *Learning theory and kernel machines*, pages 144–158. Springer.

Wang, Y.-X., Sharpnack, J., Smola, A. J., and Tibshirani, R. J. (2016).
Trend filtering on graphs.
The Journal of Machine Learning Research, 17(1):3651–3691.

Wei, S., Madrid-Padilla, O. H., and Sharpnack, J. (2018).
Distributed cartesian power graph segmentation for graphon estimation.
arXiv preprint arXiv:1805.09978.

Zhang, Y., Levina, E., and Zhu, J. (2015).
Estimating network edge probabilities by neighborhood smoothing.
arXiv preprint arXiv:1509.08588.