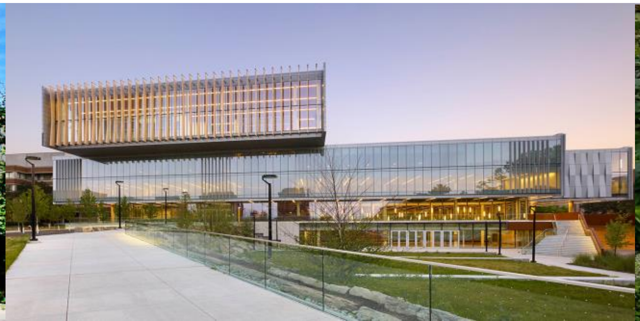
Spectral Graph Learning: Algorithm and Application to Image Coding & Graph Convolutional Nets

Gene Cheung
York University
Toronto, Canada

February 23, 2022

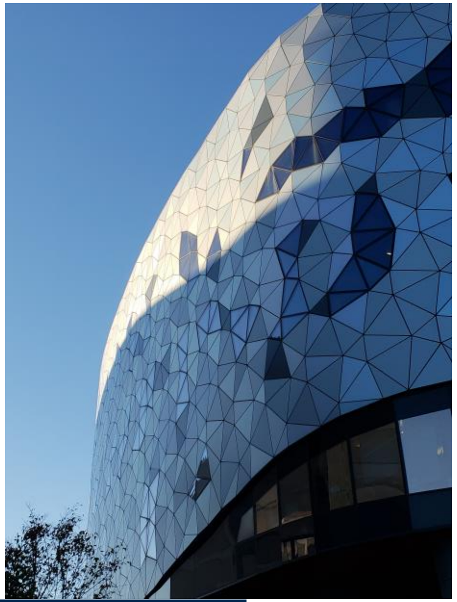



Acknowledgement

- **Graph and Image Signal Processing (GISP) Lab** (York University, Toronto, Canada)
  - Post-docs: Cheng Yang*, Xue Zhang, Chinthaka Dinesh
  - Grad students: Saghar Bagheri, Fengbo Lan, Huy Vu, Tam Thuc Do
  - Visiting researchers: Weng-tai Su, Sadid Sahami (NTHU), Fen Wang (Xidian)

- **Collaborators**
  - Richard Wildes, Michael Brown (York Univ., Canada)
  - Ivan V. Bajic (Simon Fraser Univ., Canada)
  - Antonio Ortega (Univ. of Southern California, USA)
  - Stanley Chan (Purdue Univ., USA)
  - Wai-Tian Tan (Cisco, USA)
  - Jiahao Pang, Dong Tian (InterDigital, USA)
  - Phil Chou (Google, USA)
  - Yuji Nakatsukasa (Oxford Univ., UK)
  - Thomas Maugey (Inria, France)
  - Wei Hu (Peking Univ., China), Jin Zeng (Tongji Univ., China)
  - Chia-Wen Lin (NTHU, Taiwan)
  - Kazuya Kodama (NII, Japan)
  - Yonina C. Eldar (Weizmann Inst. of Science, Israel)

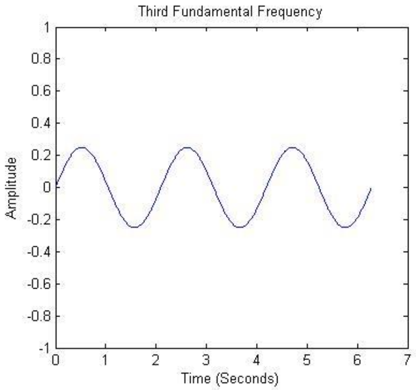
interdigital · cisco · Google · GROWERS EDGE

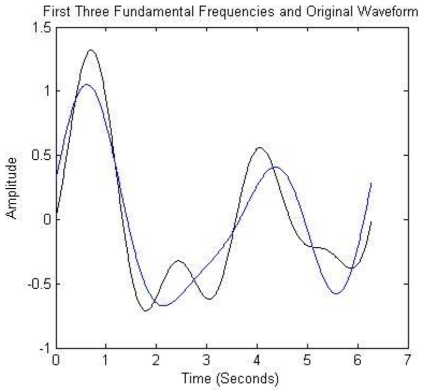


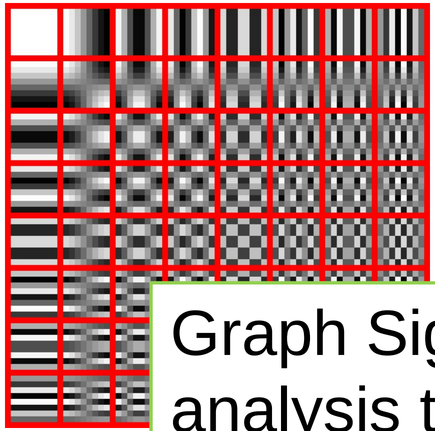
* September 2018 to June 2020.

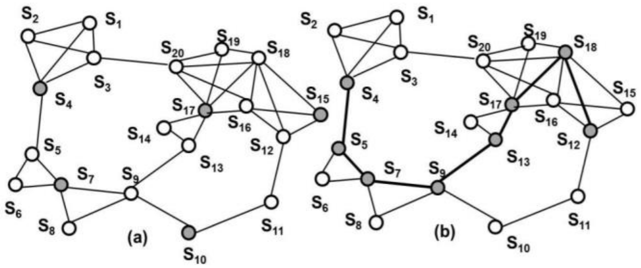
Graph Signal Processing

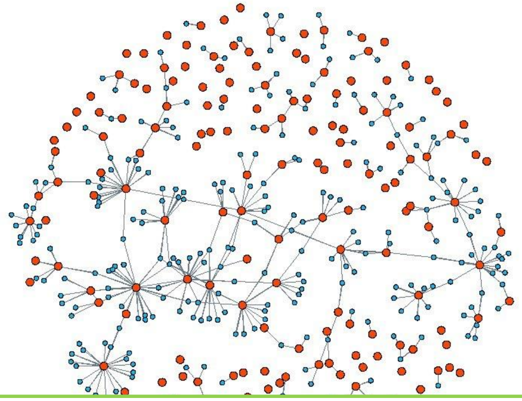
Frequency analysis + **Graph kernels** = **Graph Signal Processing**

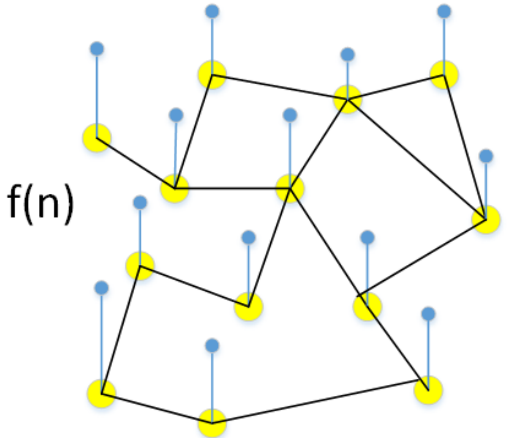




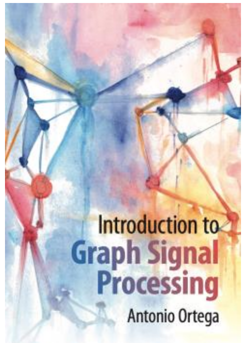


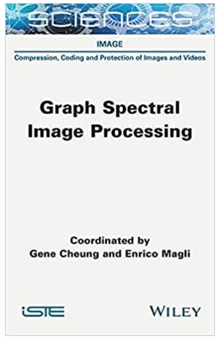






signal on graph kernel





Graph Signal Processing (GSP) studies spectral analysis tools for signals residing on graphs.

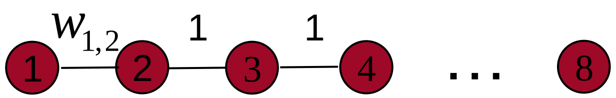
[1] A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura, and P. Vandergheynst, "**Graph signal processing: Overview, challenges, and applications**," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

Graph Spectrum

degree matrix

adjacency matrix

Graph Fourier modes: eigenvectors of *graph Laplacian matrix* **L = D - W**.



eigenvectors $\mathbf{v}_k$ in columns

eigenvalues λ_k along diagonal

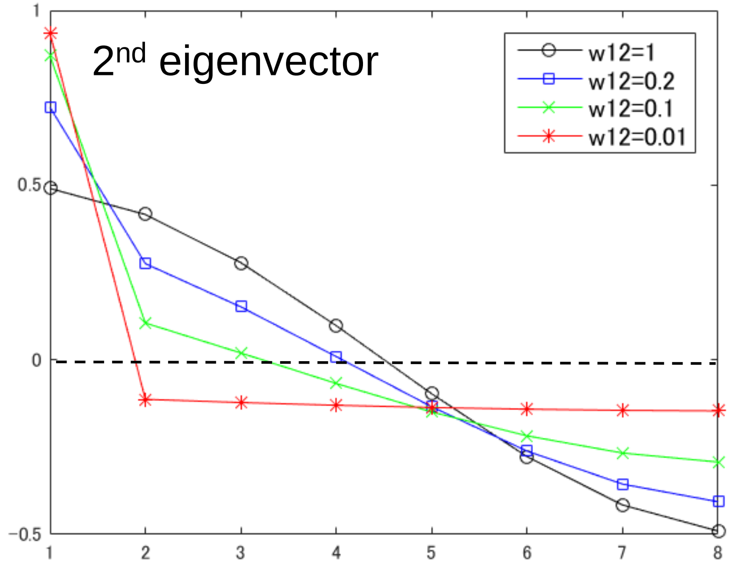
$$\mathbf{L} = \mathbf{V}\boldsymbol{\Xi}\mathbf{V}^T = \sum_k \lambda_k \mathbf{v}_k \mathbf{v}_k^T$$

sum of rank-1 matrices

Graph Fourier Transform (GFT)

GFT defaults to *DCT* for un-weighted connected line.

GFT defaults to *DFT* for un-weighted connected circle.



1. **Eigenvectors** are (***global***) aggregates of (***local***) edge weights.
   - More variations for larger eigenvalues.
2. **Eigenvalues** (≥ 0) as *graph frequencies* [1].

[1] G. Cheung, E. Magli, Y. Tanaka, M. Ng, "**Graph Spectral Image Processing**," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 907-930, May 2018. .

Graph Frequency Examples (US Temperature)

Weather stations from 100 most populated cities.

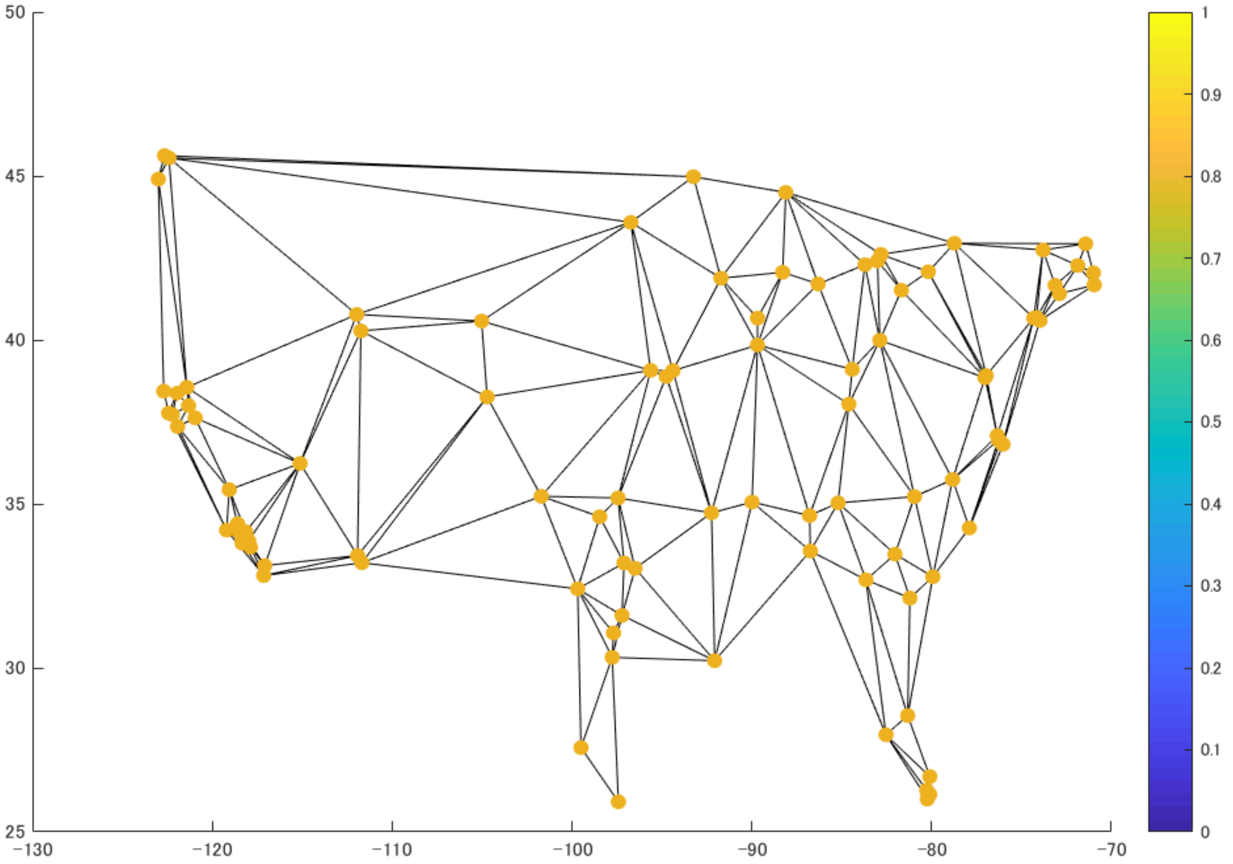
Graph connections from Delaunay Triangulation*.

Edge weights inverse proportional to distance.

location diff.

$$w_{i,j} = \exp\left(\frac{-\left\| l_i - l_j \right\|_2^2}{\sigma^2} \right)$$

Edge weights



V1: DC component

*https://en.wikipedia.org/wiki/Delaunay triangulation

Graph Frequency Examples (US Temperature)

Weather stations from 100 most populated cities.

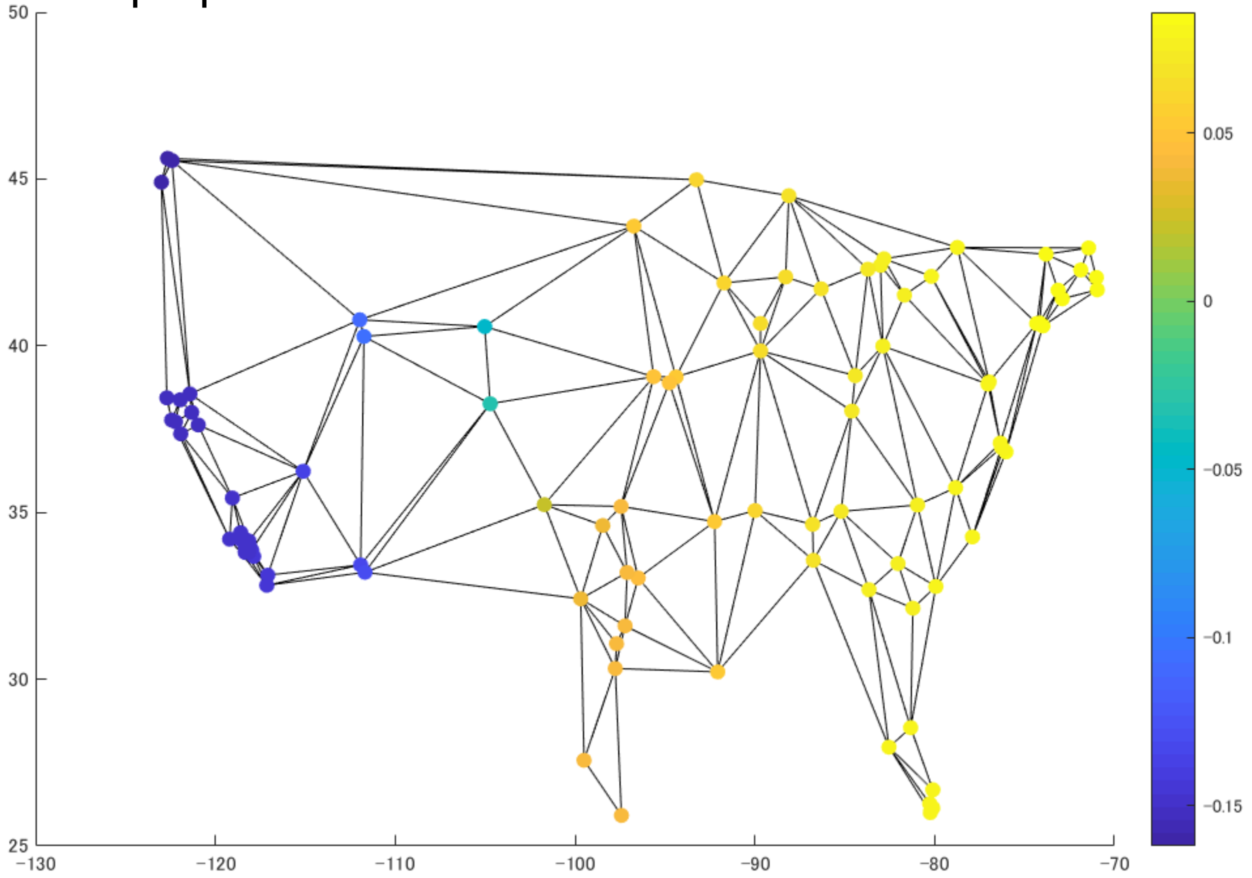
Graph connections from Delaunay Triangulation*.

Edge weights inverse proportional to distance.

location diff.

$$w_{i,j} = \exp\left(\frac{-\left\| l_i - l_j \right\|_2^2}{\sigma^2} \right)$$

Edge weights



V2: 1st AC component

Graph Frequency Examples (US Temperature)

Weather stations from 100 most populated cities.

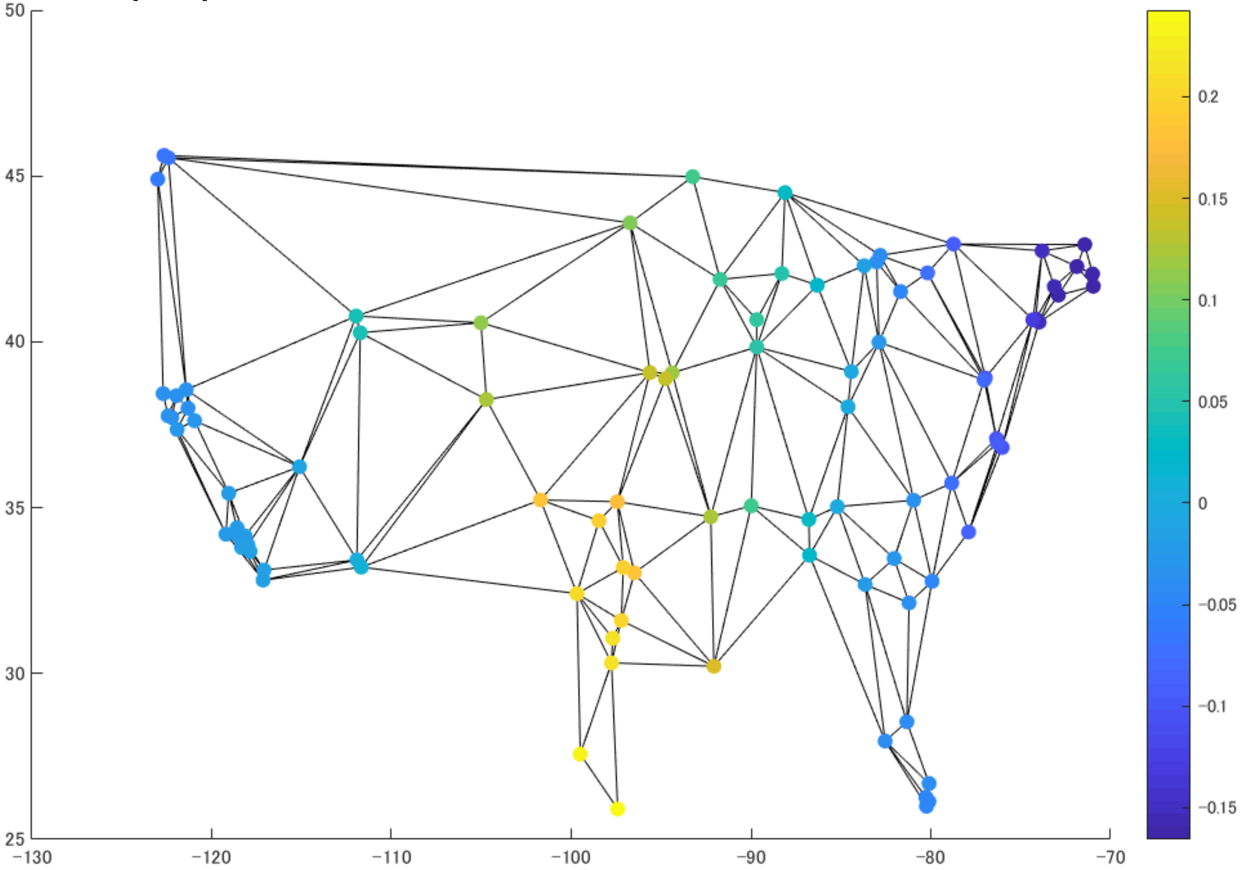
Graph connections from Delaunay Triangulation*.

Edge weights inverse proportional to distance.

location diff.

$$w_{i,j} = \exp\left(\frac{-\|l_i - l_j\|_2^2}{\sigma^2} \right)$$

Edge weights



V3: 2nd AC component

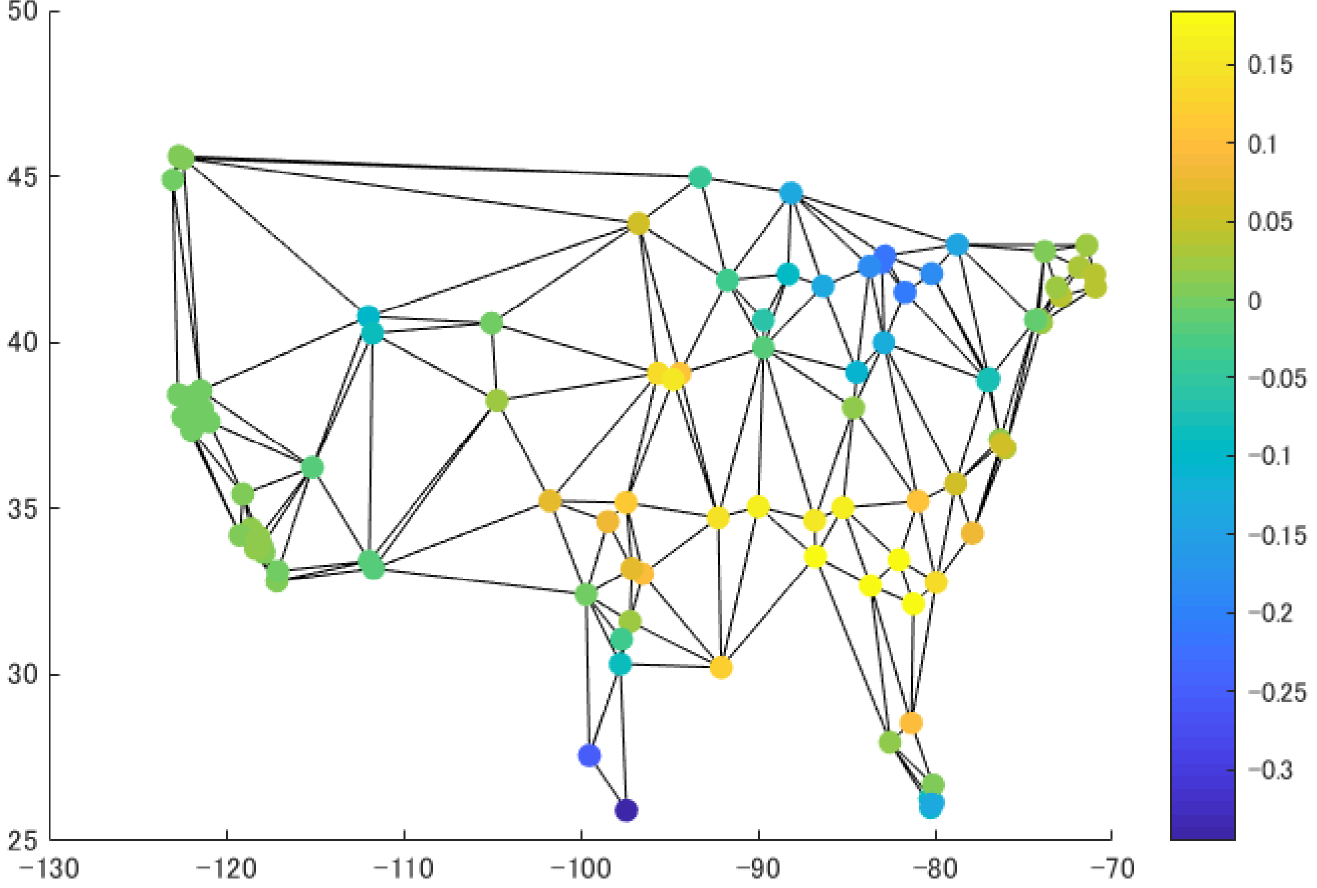
Graph Frequency Examples (US Temperature)

Weather stations from 100 most populated cities.
Graph connections from Delaunay Triangulation*.
Edge weights inverse proportional to distance.

location diff.

$$w_{i,j} = \exp\left(\frac{-\|l_i - l_j\|_2^2}{\sigma^2} \right)$$

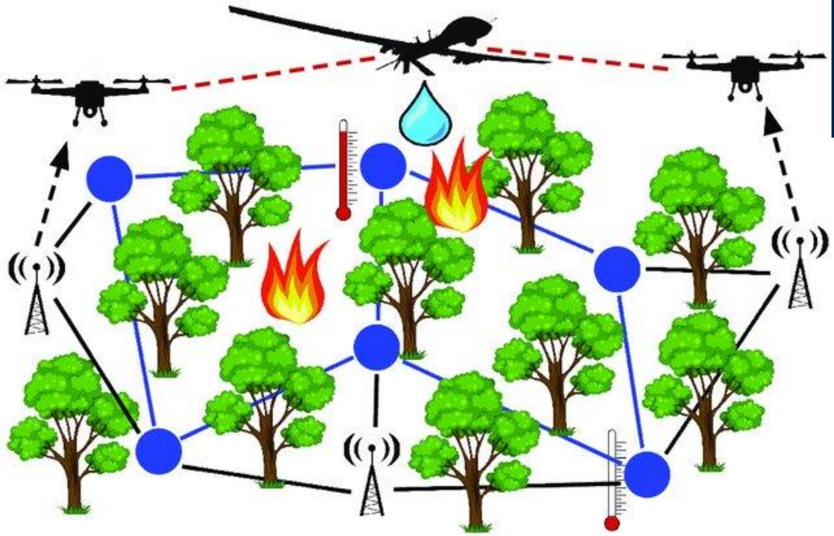
Edge weights



V4: 9th AC component

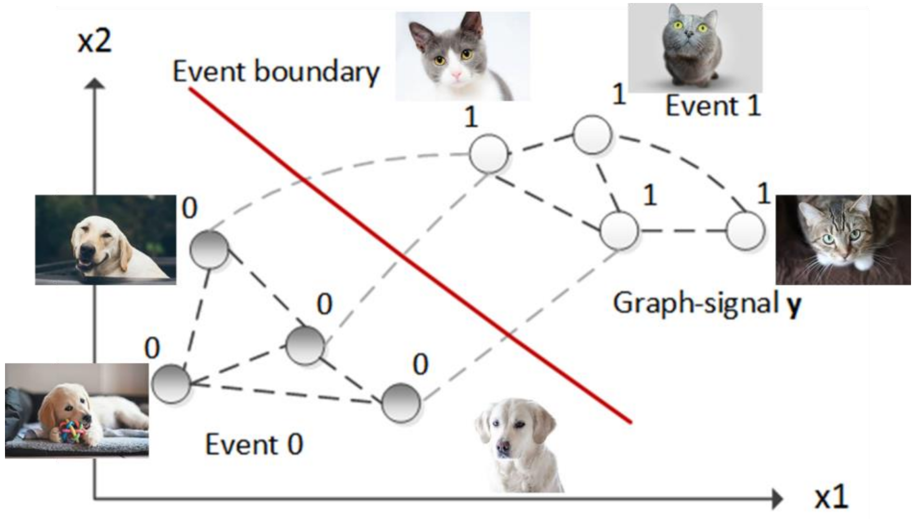
Gene Cheung (genec@yorku.ca)

What is a good graph?



- Graph captures *pairwise relationships*.
  1. Domain knowledge.
  2. Correlations.
  3. Feature distance.

- **Approaches**:
  1. Learn sparse **inverse covariance matrix** from observations [1].
     - Graphical Lasso, CLIME.
  2. Learn metric to determine **feature distance** [2].



[1] S. Bagheri, G. Cheung, A. Ortega, F. Wang, "**Learning Sparse Graph Laplacian with *K* Eigenvector Prior via Iterative GLASSO and Projection**," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Toronto, Canada, June 2021 (**best student paper finalist**).

[2] C. Yang, G. Cheung, W. Hu, "**Signed Graph Metric Learning via Gershgorin Disc Alignment**," *IEEE TPAMI*, June 2021.

Sparse Precision Matrix Estimation: GLASSO

- Given *empirical covariance matrix* Σ, **Graphical Lasso** computes positive-definite (PD) *precision matrix* Θ:

$$\max_{\Theta} \quad \log \det \Theta - \mathrm{Tr}(\Sigma\Theta) - \rho \, ||\Theta||_1$$

- 1^{st} and 2^{nd} terms are ***likelihood***.
- 3^{rd} term promotes **sparsity**.

- Solved via **block coordinate descent** (BCD) algorithm.

[1] Friedman J, Hastie T, Tibshirani R. "**Sparse inverse covariance estimation with the graphical lasso**," *Biostatistics*. 2008; 9(3): 432-441.

Graph Laplacian Estimation

- Assume **precision matrix** is:
  - **Generalized graph Laplacian** (GGLs),
  - **Diagonally dominant generalized graph Laplacian** (DDGLs), or
  - **Combinatorial graph Laplacian** (CGLs).

NOTE: Interpret precision matrix as graph Laplacian

- Given *empirical covariance matrix* S, computes *Laplacian* Θ:

$$\min_{\Theta} \quad \mathrm{Tr}(\Theta \mathbf{K}) - \log\det\Theta \quad \text{subject to} \quad \Theta \in \mathcal{L}_g(A)$$

- **K** = **S** + **H**, **H** is regularization matrix.
- $L_g(A)$ ensures Θ is GGL.
- Solved via **block coordinate descent** (BCD) algorithm.

[1] H. E. Egilmez, E. Pavez and A. Ortega, "**Graph Learning From Data Under Laplacian and Structural Constraints**," in *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 825-841, Sept. 2017

Graph Laplacian Estimation w/ Eigen-Structure Constraint

1. Side info to derive first *K* e-vectors.
2. Fast computation of first *K* e-vectors.
3. Desire eigen-structure.

- **Key Assumption**: graph Laplacian matrix **L** has *chosen first K eigenvectors*.

- Define **convex cone** $\mathcal{H}_{\mathbf{u}}^{+}$ of PSD matrices with same first *K* eigenvectors.
- Design **projection operator** to $\mathcal{H}_{\mathbf{u}}^{+}$ inspired by **Gram-Schmidt procedure**.
- Given *empirical covariance matrix* $\bar{\mathbf{C}}$, computes *graph Laplacian* $\mathbf{L}$:

$$\min_{\mathbf{L} \in \mathcal{H}_{\mathbf{u}}^{+}} \quad \text{Tr}(\mathbf{L}\bar{\mathbf{C}}) - \log\det\mathbf{L} + \rho \left\|\mathbf{L}\right\|_1$$

- Solve via alternating BCD and projection algorithm.

[1] S. Bagheri, G. Cheung, A. Ortega, F. Wang, "**Learning Sparse Graph Laplacian with *K* Eigenvector Prior via Iterative GLASSO and Projection**," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Toronto, Canada, June 2021 (**best student paper finalist**).

Preliminaries: inner product, Hilbert Space

- Define a **vector space** of real, symmetric matrices $\boldsymbol{S} = \{\mathbf{L} \in \mathrm{R}^{N\times N} | \mathbf{L}^T = \mathbf{L}\}$.
- Define **inner product** for $\mathbf{A}, \mathbf{B} \in \boldsymbol{S}$:

$$\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{tr}(\mathbf{B}^\top \mathbf{A}) = \sum_{i,j} A_{ij} B_{ij}.$$

- **Hilbert space** $\boldsymbol{H}$ is vector space $\boldsymbol{S}$ with inner product.
- Define **subspace** $\boldsymbol{H}^+$ of *positive semi-definite* (PSD) matrices:

$$\mathcal{H}^+ = \{\mathbf{A} \in \mathcal{H} \,|\, \mathbf{A} \succeq 0\}$$ ← convex cone

- Define **subspace** $\boldsymbol{H}_{\mathbf{u}}^+ \subset \boldsymbol{H}^+$ PSD matrices sharing first K eigenvectors $\{\mathbf{u}_k\}_{k=1}^K$.

$$\mathcal{H}_{\mathbf{u}}^+ = \left\{ \mathbf{L} \in \mathcal{H}^+ \,|\, \mathbf{u}_k = \arg \min_{\mathbf{x} \,|\, \mathbf{x} \perp \mathbf{u}_j, \forall j<k} \frac{\mathbf{x}^\top \mathbf{L} \mathbf{x}}{\mathbf{x}^\top \mathbf{x}}, \;\; k \in \mathcal{I}_K \right\}$$ ← convex cone

Gene Cheung (genec@yorku.ca)

Projection to Convex Cone $H_{\mathbf{u}}^{+}$: $i \leq K$

Given empirical covariance matrix $\bar{\mathbf{C}}$,

- **1st eigenvector** $\mathbf{u}_1$ of Laplacian $\mathbf{L}$:
  - Compute *last* **eigenvalue** μ_N for target $\mathbf{C} = \mathbf{L}^{-1}$ as

$$\mu_N = \langle \bar{\mathbf{C}}, \mathbf{u}_1 \mathbf{u}_1^T \rangle$$

Thus, 1st eigen-pair $\left(\frac{1}{\mu_N}, \mathbf{u}_1 \right)$ for **L**

  - Compute **residual** $\mathbf{E}_N = \bar{\mathbf{C}} - \mu_N \mathbf{u}_1 \mathbf{u}_1^T$

- **2nd eigenvector** $\mathbf{u}_2$ of $\mathbf{L}$:
  - Compute *second last* **eigenvalue** μ_{N-1} for target $\mathbf{C}$ as

$$\mu_{N-1} = \min\left(\langle \mathbf{E}_N, \mathbf{u}_2 \mathbf{u}_2^T \rangle, \mu_N \right)$$

Thus, 2nd eigen-pair $\left(\frac{1}{\mu_{N-1}}, \mathbf{u}_2 \right)$ for **L**

  - Compute **residual** $\mathbf{E}_{N-1} = \mathbf{E}_N - \mu_{N-1} \mathbf{u}_2 \mathbf{u}_2^T$

Projection to Convex Cone $H_{\mathrm{u}}^{+} : i > K$

- Compute *next* **eigenvector** $\mathbf{v}_i$ for $\mathbf{L}$:

$$\max_{\mathbf{v}_i} \langle \mathbf{E}_{N-i+2}, \mathbf{v}_i \mathbf{v}_i^T \rangle \quad \text{s.t.} \quad \begin{array}{ll} \mathbf{v}_i^T \mathbf{u}_k = 0, & k = \{1, \dots, K\} \\ \mathbf{v}_i^T \mathbf{v}_j = 0, & j = \{K+1, \dots, i-1\} \\ \|\mathbf{v}_i\| = 1 & \end{array}$$

  - **NP-hard**. See [1] for fast approx.

- Compute **eigenvalue** μ_{N-i+1}:

$$\mu_{N-i+1} = \min\left(\langle \mathbf{E}_{N-i+2}, \mathbf{v}_i \mathbf{v}_i^T \rangle, \mu_{N-i+2} \right)$$

Thus, ith eigen-pair $\left(\frac{1}{\mu_{N-i+1}}, \mathbf{v}_i \right)$ for **L**

- Compute **residual**: $\mathbf{E}_{N-i+1} = \mathbf{E}_{N-i+2} - \mu_{N-i+1} \mathbf{v}_i \mathbf{v}_i^T$

[1] S. Bagheri, G. Cheung, A. Ortega, F. Wang, "**Learning Sparse Graph Laplacian with *K* Eigenvector Prior via Iterative GLASSO and Projection**," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Toronto, Canada, June 2021 (**best student paper finalist**).

GLASSO + Projection

- Modify **GLASSO** to

$$\min_{\mathbf{L} \in \mathcal{H}_{\mathbf{u}}^{+}} \quad \text{Tr}(\mathbf{L}\bar{\mathbf{C}}) - \log\det \mathbf{L} + \rho \, \|\mathbf{L}\|_1$$

- **Dual** of GLASSO is

$$\min_{\mathbf{C}^{-1} \in \mathcal{H}^{+}} \quad -\log\det \mathbf{C}, \quad \text{s.t.} \;\; \|\mathbf{C} - \bar{\mathbf{C}}\|_\infty \leq \rho$$

- **Algorithm**:
  1. Iteratively updating one row / column of $\boldsymbol{C}$.
  2. Project $\mathbf{L} = \mathbf{C}^{-1}$ to convex cone $\boldsymbol{H}_{\mathbf{u}}^{+}$using projection operator.
  3. Repeat till convergence.

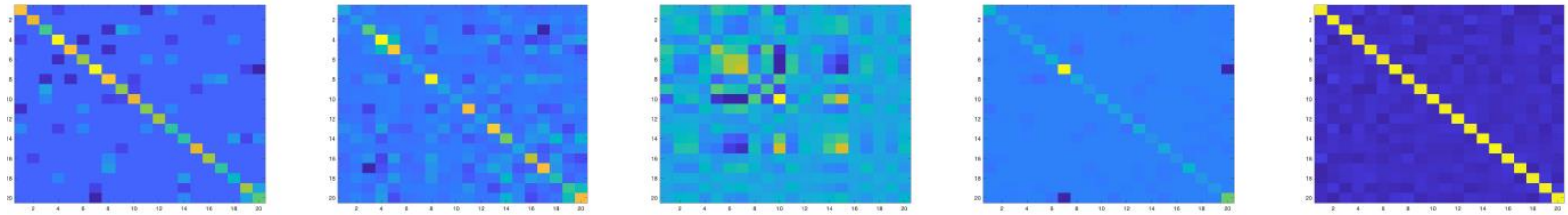
[1] Friedman J, Hastie T, Tibshirani R. "**Sparse inverse covariance estimation with the graphical lasso**," *Biostatistics*. 2008; 9(3): 432-441.

Gene Cheung (genec@yorku.ca)

Graph Laplacian Estimation: results

- Randomly located 20 nodes in 2D space. Use **Erdos-Renyi model** to determine connectivity with probability 0.6. Compute edge weights using Gaussian kernel. Remove weights <0.75. Flip sign of each edge with probability 0.5. *K*=1.

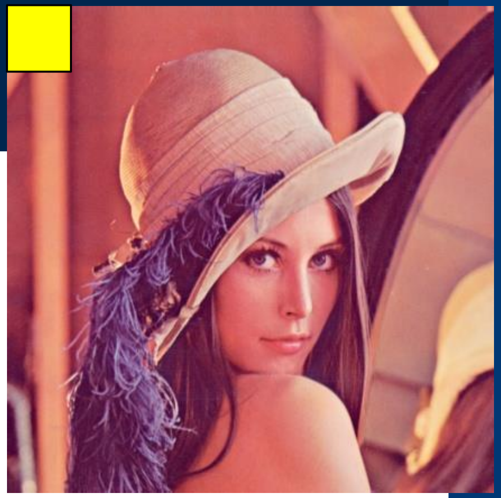
- (a) Ground Truth Laplacian L , (b) Proposed Proj-Lasso with *K* = 1, (c) GLASSO, (d) DDGL and (e) GL-SigRep .



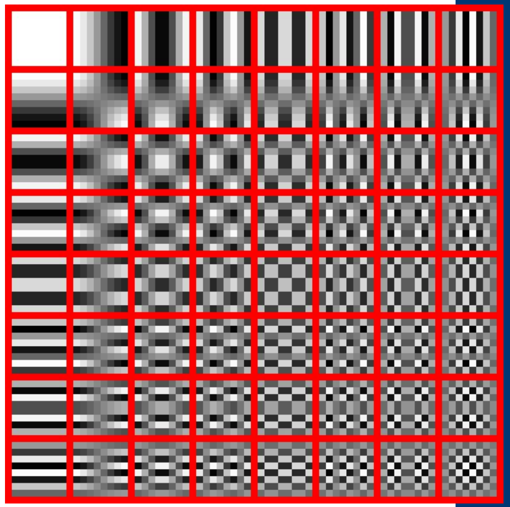
(a) (b) (c) (d) (e)

[1] S. Bagheri, G. Cheung, A. Ortega, F. Wang, "**Learning Sparse Graph Laplacian with *K* Eigenvector Prior via Iterative GLASSO and Projection**," *IEEE International Conference on Acoustics, Speech and Signal Processing*, Toronto, Canada, June 2021 (**best student paper finalist**).

Application 1: Image Coding



- **Transform Coding** is integral component in image compression.
- **Problem**: **DCT** is *fixed* transform and does not adapt locally.
- **Existing Work 1**: **Asymmetric Discrete Sine Transform** (ADST) fits better prediction residuals [1].
- **Existing Work 2**: **Karhunen-Loeve transform** (KLT) adapts well iff ∃ reliable empirical covariance matrix $\bar{\mathbf{C}}$ [2].



[1] J. Han, A. Saxena, V. Melkote, and K. Rose, "**Jointly optimized spatial prediction and block transform for video and image coding**," in *IEEE Transactions on Image Processing*, April 2012, vol. 21, no.4, pp. 1874–1884.

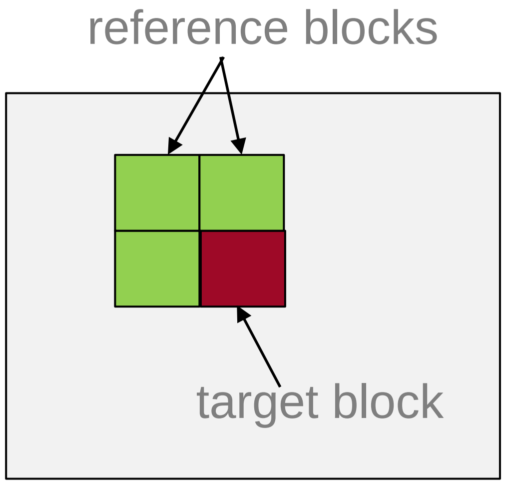
[2] Ian Blanes and Joan Serra-Sagrista, "**Pairwise orthogonal transform for spectral image coding**," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no.3, pp. 961–972, 2011.

Application 1: Image Coding

- **Key Idea**: derive first *K* e-vectors from model, compute *N-K* e-vectors from data.

- **Advantages**:
  1. Reduce degree of freedom when empirical covariance $\bar{\mathbf{C}}$ is unreliable.
  2. Parameter *K* is tunable depending on covariance reliability.
  3. Reduce computation cost for first K transform coefficients.

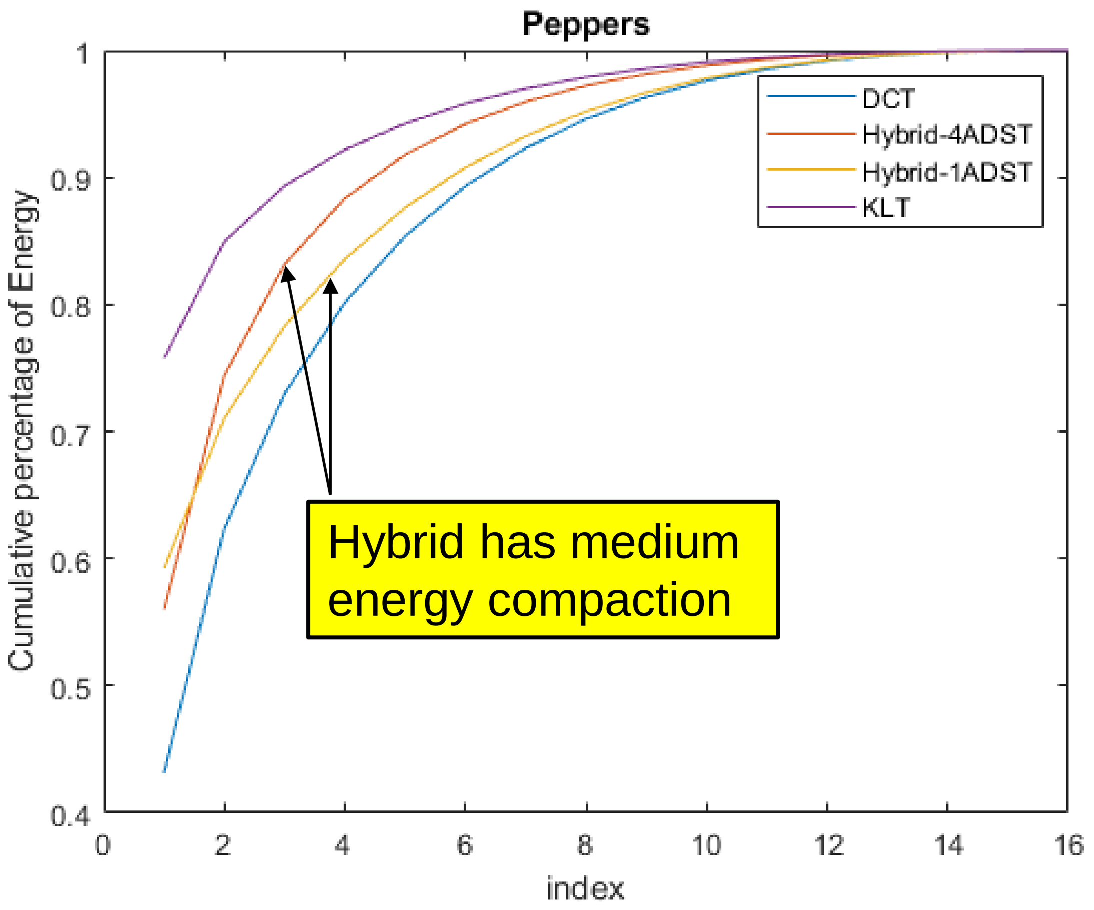
- **Disadvantages**:
  1. Larger computation cost than DCT.

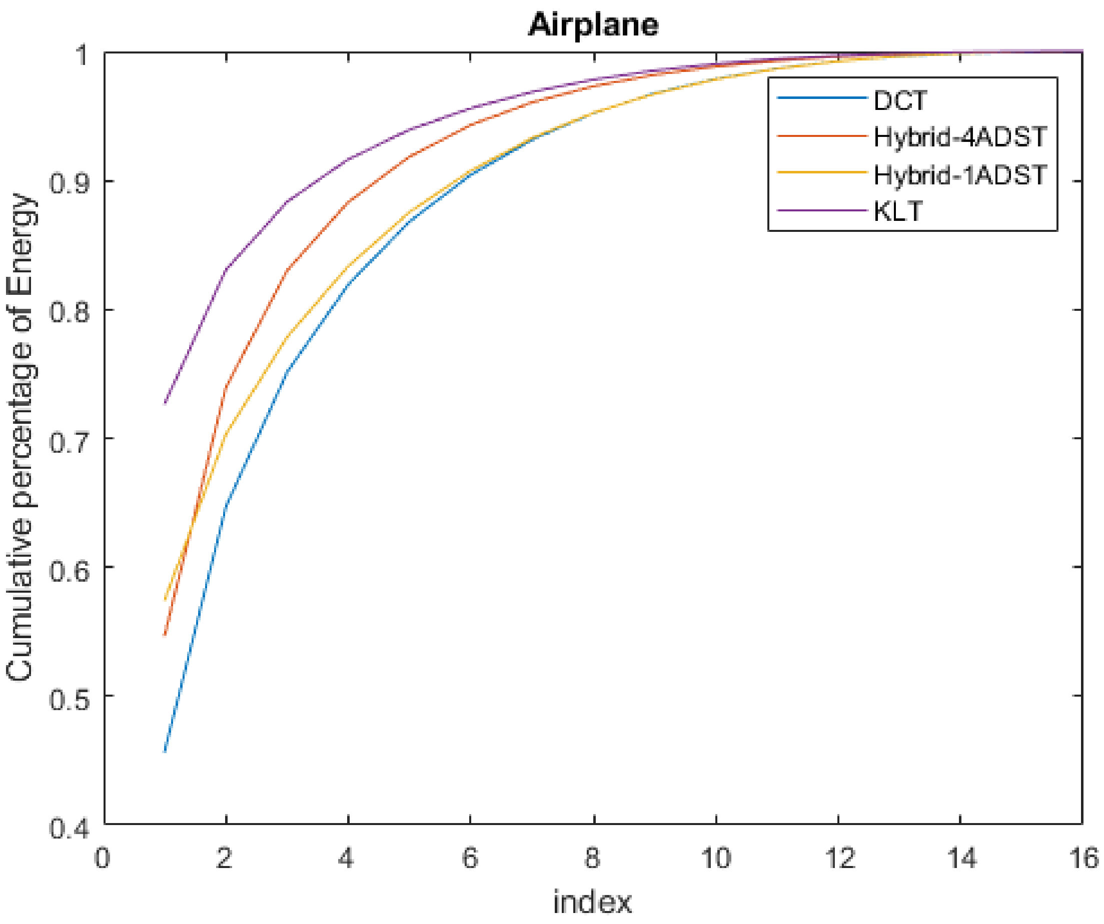


[1] Saghar Bagheri, Tam Thuc Do, Gene Cheung, Antonio Ortega, "**Hybrid Model-based / Data-driven Graph Transform for Image Coding**," submitted to *IEEE Conference on Image Processing*, 2022.

Image Coding: results 1 (energy compaction)

- **Setting**: **WebP** image codec [1]. DC4 intra-prediction mode. Improve prediction residual coding of 4x4 block over default DCT.





[1] https://developers.google.com/speed/webp

Image Coding: results 2 (error variance)

- **Setting**: **WebP** image codec [1]. DC4 intra-prediction mode. Improve prediction residual coding of 4x4 block over default DCT.

smaller error variance than KLT

smallest error variance

Image Name	DCT	KLT	Hybrid-4ADST	ADST
"Female"	"0.053911"	"0.041706"	"0.042182"	"0.037954"
"Couple"	"0.055329"	"0.048432"	"0.044757"	"0.041618"
"Peppers"	"0.060291"	"0.056722"	"0.054069"	"0.05174"
"Airplane"	"0.056519"	"0.055206"	"0.049232"	"0.04924"

[1] https://developers.google.com/speed/webp

Application 2: Filter Training in GCN

- **Graph convolutional nets** (GCN) performs graph filtering and pointwise non-linear operation (e.g., ReLU) in a sequence of neural layers.
- **Problem**: GCN starts to *oversmooth* as the number of layers grows [1].
- **Analysis**: GCN output approaches a subspace spanned by 1st eigenvector of normalized graph Laplacian $\tilde{\mathbf{L}}$ with convergence rate $\propto$ "**eigen-gap**" [2].
- **Existing Solution**: randomly drop edges at layers for sparse graph [3].

[1] G. Li, M. Muller, A. Thabet, and B. Ghanem, "**DeepGCNs: Can GCNs go as deep as CNNs?**" in *Proceedings of the IEEE/CVF ICCV*, 2019, pp. 9267–9276.

[2] K. Oono and T. Suzuki, "**Graph neural networks exponentially lose expressive power for node classification**," in *International Conf. on Learning Rep.*, 2020.

[3] Y. Rong et al., "**DropEdge: Towards deep graph convolutional networks on node classification**," in International Conf. on Learning Rep., 2020.

Spectral Graph Learning for Filter Training in GCN

- **Key Idea**: learn Laplacian $\mathbf{L}$ from empirical covariance $\bar{\mathbf{C}}$ w/ desired eigen-gap.
- **Algorithm**:

1. Compute *last* eigenvalue μ_N for target $\mathbf{C} = \mathbf{L}^{-1}$ as

$$\mu_N = \langle \bar{\mathbf{C}}, \mathbf{x}\mathbf{x}^T \rangle$$

avg. signal

Thus, 1st eigen-pair $\left(\frac{1}{\mu_N}, \mathbf{x}\right)$ for **L**

2. Compute residual $\mathbf{E}_N = \bar{\mathbf{C}} - \mu_N \mathbf{x}_1 \mathbf{x}_1^T$
3. Approximate 2nd **eigenvector** $\mathbf{v}_2$

$$\max_{\mathbf{v}_2} \langle \mathbf{E}_N, \mathbf{v}_2 \mathbf{v}_2^T \rangle \quad \text{s.t.} \quad \mathbf{v}_2^T \mathbf{x} = 0, \quad \|\mathbf{v}_2\| = 1$$

4. Compute *2nd last* **eigenvalue** μ_{N-1} for target $\mathbf{C} = \mathbf{L}^{-1}$ as

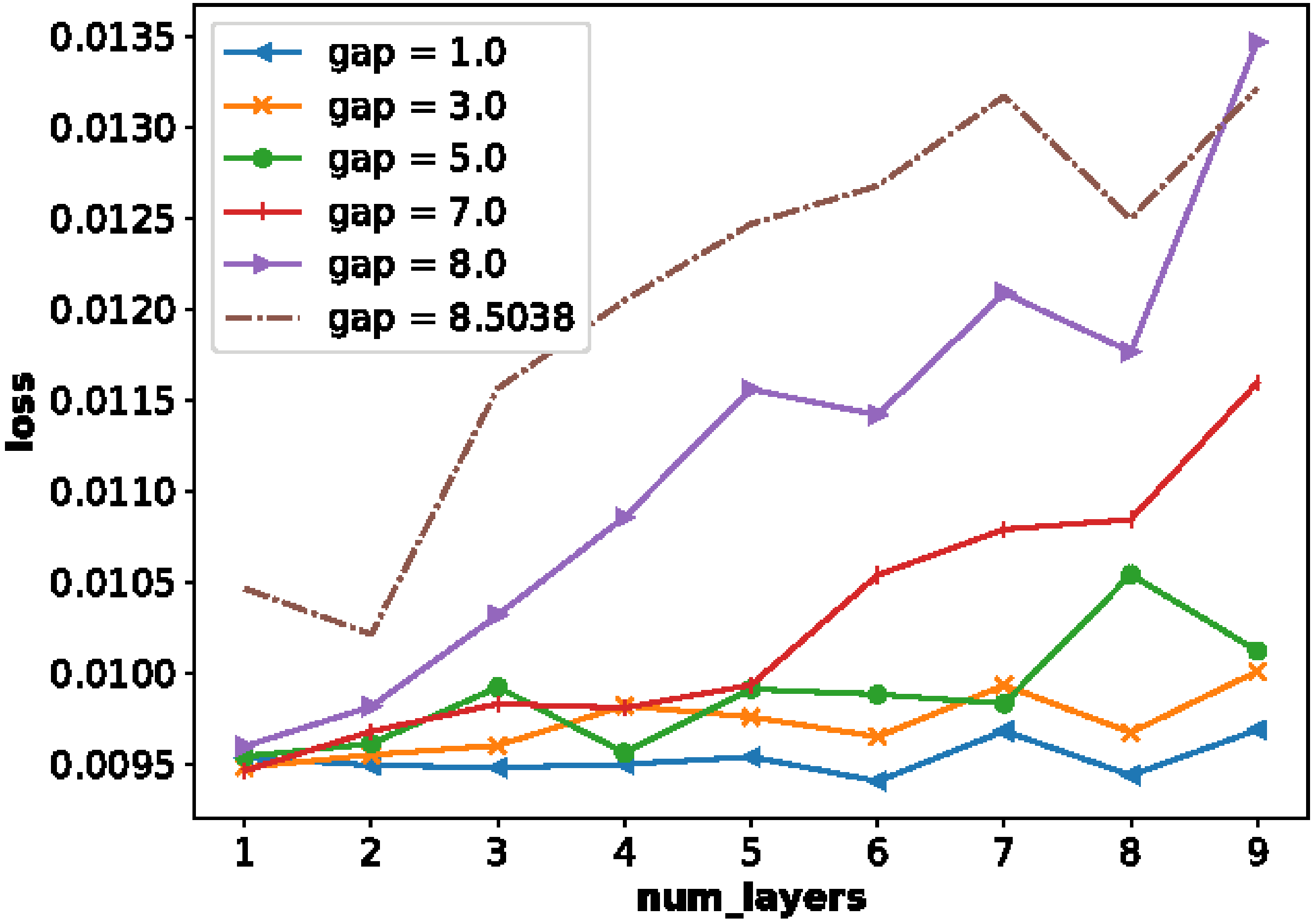
$$\mu_{N-1} = \max\left(\langle \mathbf{E}_N, \mathbf{v}_2 \mathbf{v}_2^T \rangle, \mu_N - \kappa\right)$$

eigen-gap!

[1] Jin Zeng, Saghar Bagheri, Yang Liu, Gene Cheung, Wei Hu, "**Sparse Graph Learning with Eigen-gap for Spectral Filter Training in Graph Convolutional Networks**," submitted to *EUSIPCO'22*, Belgrade, Serbia, August 2022.

GCN Training: results 1

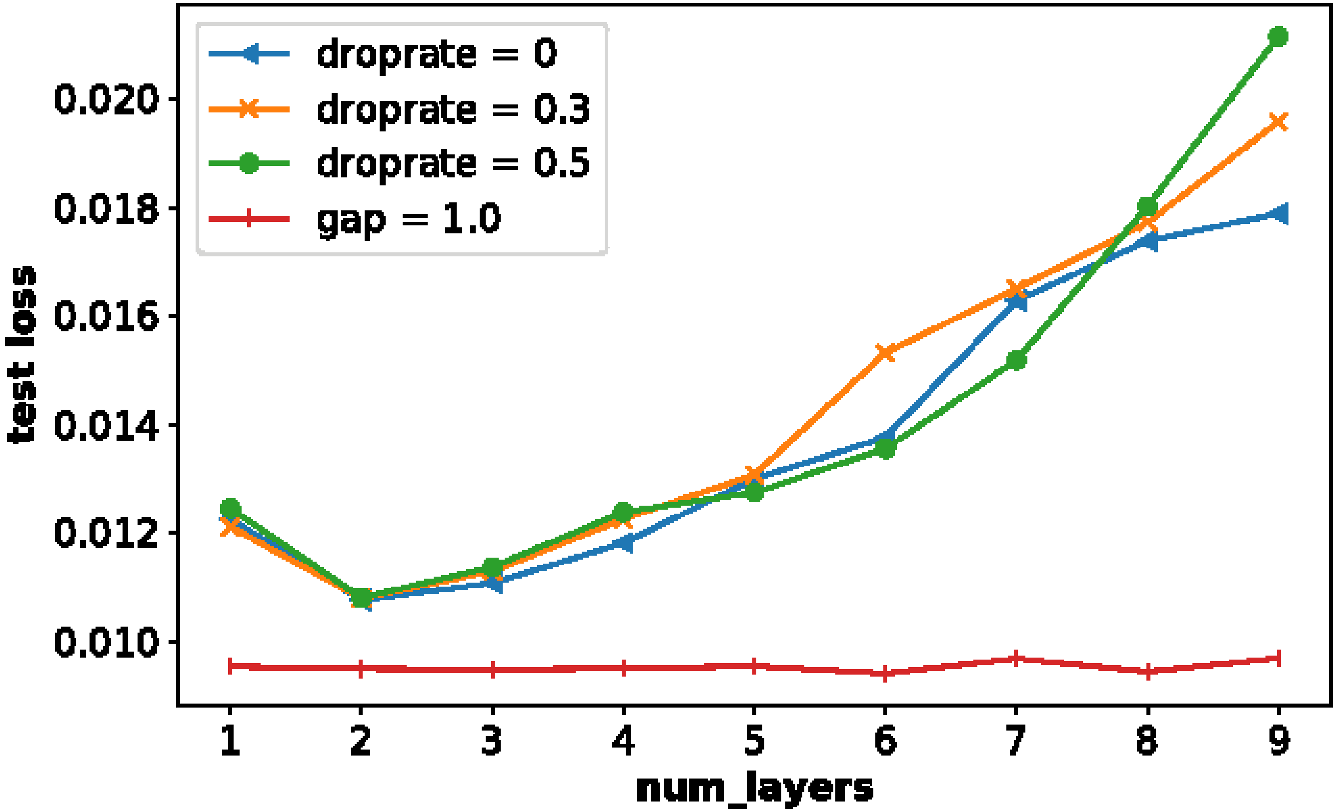
- **Data**: METR-LA contains traffic speed data in 4 months from 207 sensors in LA County.
- **Task**: predict current traffic speed using speed data from 50 to 5 minutes ago as input.



- Smaller gap, larger optimal layer.
- Smaller gap, small loss value.

GCN Training: results 2

- Compare Gap=1 to DropEdge [1] with drop rate = 0, 0.3, 0.5.



- SGL has larger optimal layer.
- SGL has small loss value.

[1] Y. Rong et al., “**Dropedge: Towards deep graph convolutional networks on node classification**,” in International Conf. on Learning Rep., 2020.

Conclusion

- **Graph Signal Processing (GSP)** studies signals on graphs.
- **Graph learning** is crucial first step for GSP.
- **Spectral graph learning** can optimize eigen-structures.
  - Image coding.
  - Filtering training in GCN.
- **Future work**:
  - Tighter integration between GSP and GCN/GNN.

Applications:
Image coding, denoising, deblurring, interpolation, contrast enhancement, light field image denoising, 3D point cloud denoising, sub-sampling, super-resolution, matrix completion, semi-supervised classifier learning, video summarization, crop yield prediction

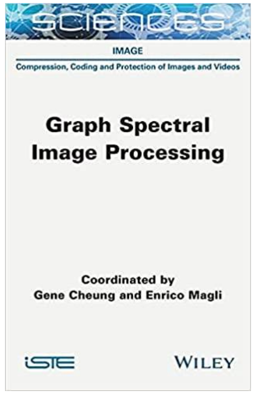
Contact Info

- **Homepage**:

https://www.eecs.yorku.ca/~genec/index.html

- **E-mail**:

genec@yorku.ca



- **New book**:

G. Cheung, E. Magli, (edited) ***Graph Spectral Image Processing***, ISTE/Wiley, August 2021.