Learning to Identify Sources of Network Diffusion

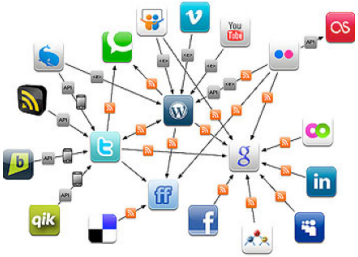
Gonzalo Mateos

Dept. of ECE and Goergen Institute for Data Science
University of Rochester
gmateosb@ece.rochester.edu
http://www.hajim.rochester.edu/ece/sites/gmateos

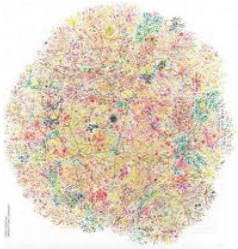
Co-author: Chang Ye
Acknowledgment: NSF Awards CCF-1750428, CCF-1934962 and ECCS-1809356

EUSIPCO 2022, Belgrade, Serbia
August 30, 2022

Online social media



Internet



Clean energy and grid analytics

- Network as undirected graph $G = (\mathcal{V}, \mathcal{E})$: encode pairwise relationships
- Desiderata: Process, analyze and learn from network data [Kolaczyk'09]

  $\Rightarrow$ Study graph signals, data associated with N nodes in $\mathcal{V}$
- Ex: Opinion profile, buffer congestion levels, neural activity, epidemic

- Graph signals mappings $x : \mathcal{V} \to \mathbb{R}$, represented as vectors $\mathbf{x} \in \mathbb{R}^N$
  - $\Rightarrow$ **As.:** Signal properties related to topology of G
- To process graph signals $\Rightarrow$ Graph-shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$
  - $\Rightarrow$ Local $S_{ij} = 0$ for $i \neq j$ and $(i, j) \notin \mathcal{E}$ $\Rightarrow$ Ex: $\mathbf{A}$ or $\mathbf{L} = \mathbf{D} - \mathbf{A}$
  - $\Rightarrow$ Spectrum of symmetric $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\top}$

- Graph signals mappings $x : \mathcal{V} \to \mathbb{R}$, represented as vectors $\mathbf{x} \in \mathbb{R}^N$
  - $\Rightarrow$ **As.:** Signal properties related to topology of G
- To process graph signals $\Rightarrow$ Graph-shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$
  - $\Rightarrow$ Local $S_{ij} = 0$ for $i \neq j$ and $(i,j) \notin \mathcal{E}$ $\Rightarrow$ Ex: $\mathbf{A}$ or $\mathbf{L} = \mathbf{D} - \mathbf{A}$
  - $\Rightarrow$ Spectrum of symmetric $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$
- Graph Fourier Transform (GFT) for signals: $\tilde{\mathbf{x}} = \mathbf{V}^\top \mathbf{x}$
- Graph filters $\mathbf{H} : \mathbb{R}^N \to \mathbb{R}^N$ are maps between graph signals
  - $\Rightarrow$ Polynomial in $\mathbf{S}$ with coefficients $\mathbf{h} \in \mathbb{R}^L$ $\Rightarrow$ $\mathbf{H} := \sum_{l=0}^{L-1} h_l \mathbf{S}^l$
  - $\Rightarrow$ Orthogonal frequency operator: $\mathbf{H} = \mathbf{V}\text{diag}(\tilde{\mathbf{h}})\mathbf{V}^\top$
  - $\Rightarrow$ Freq. response (GFT for filters): $\tilde{\mathbf{h}} = \mathbf{\Psi}\mathbf{h}$ and $[\mathbf{\Psi}]_{k,l} = {\lambda_k}^{l-1}$

- **Q:** Upon observing a graph signal $\mathbf{y}$, how was this signal generated?
- Postulate $\mathbf{y}$ is the response of linear diffusion to a sparse input $\mathbf{x}$

$$\mathbf{y} \;=\; \alpha_0 \prod_{l=1}^{\infty} (\mathbf{I} - \alpha_l \mathbf{S})\mathbf{x} \;=\; \sum_{l=0}^{\infty} \beta_l \mathbf{S}^l \mathbf{x}$$

$\Rightarrow$ Common generative model, e.g., heat diffusion, consensus

- **Q:** Upon observing a graph signal $\mathbf{y}$, how was this signal generated?
- Postulate $\mathbf{y}$ is the response of linear diffusion to a sparse input $\mathbf{x}$

$$\mathbf{y} \;=\; \alpha_0 \prod_{l=1}^{\infty} (\mathbf{I} - \alpha_l \mathbf{S})\mathbf{x} \;=\; \sum_{l=0}^{\infty} \beta_l \mathbf{S}^l \mathbf{x}$$

  $\Rightarrow$ Common generative model, e.g., heat diffusion, consensus

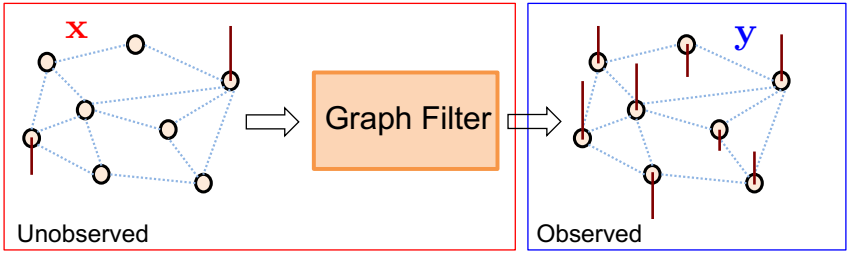
- Cayley-Hamilton asserts we can write diffusion as ($L \leq N$)

$$\mathbf{y} = \left(\sum_{l=0}^{L-1} h_l \mathbf{S}^l \right) \mathbf{x} := \mathbf{H}\mathbf{x}$$

- **Model:** Observed network process as output of a graph filter

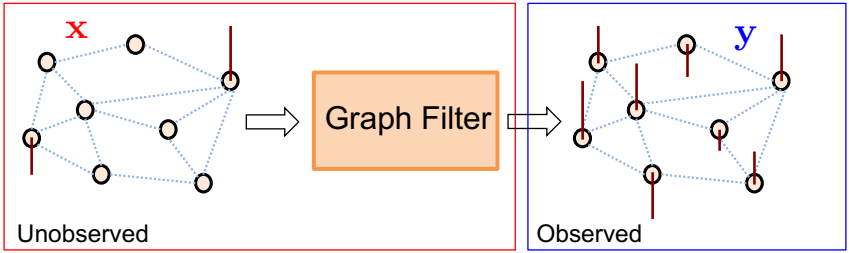
  $\Rightarrow$ View few elements in $\text{supp}(\mathbf{x}) =: \{i : x_i \neq 0\}$ as sources

- Ex: Global opinion/belief profile formed by spreading a rumor
  $\Rightarrow$ What was the rumor? Who started it?



- **Problem:** Blind identification of graph filters with multiple sparse inputs
  $\Rightarrow$ Suppose we observe P output signals $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_P] \in \mathbb{R}^{N \times P}$

- Ex: Global opinion/belief profile formed by spreading a rumor
  - $\Rightarrow$ What was the rumor? Who started it?



- **Problem:** Blind identification of graph filters with multiple sparse inputs
  - $\Rightarrow$ Suppose we observe P output signals $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_P] \in \mathbb{R}^{N \times P}$
- **Q:** Given $\mathbf{S}$, can we find sparse $\mathbf{X}$ and the filter coeffs. $\mathbf{h}$ from $\mathbf{Y} = \mathbf{H}\mathbf{X}$?
  - $\Rightarrow$ Extends classical blind deconvolution to graphs
  - $\Rightarrow$ Localization of sources that diffuse on the network

- Leverage frequency response of graph filters

$$\mathbf{Y} = \mathbf{H}\mathbf{X} \;\Rightarrow\; \mathbf{Y} = \mathbf{V}\text{diag}(\boldsymbol{\Psi}\mathbf{h})\mathbf{V}^{\top}\mathbf{X}$$

  $\Rightarrow$ $\mathbf{Y}$ is a bilinear function of the unknowns $\mathbf{h}$ and $\mathbf{X}$

- Ill-posed problem $\Rightarrow$ $L + NP$ unknowns and NP observations

  $\Rightarrow$ **As.:** $\mathbf{X}$ has S-sparse columns i.e., $\|\mathbf{X}\|_0 := |\text{supp}(\mathbf{X})| \leq PS$

- Leverage frequency response of graph filters

$$\mathbf{Y} = \mathbf{H}\mathbf{X} \;\Rightarrow\; \mathbf{Y} = \mathbf{V}\text{diag}(\boldsymbol{\Psi}\mathbf{h})\mathbf{V}^{\top}\mathbf{X}$$

  $\Rightarrow$ $\mathbf{Y}$ is a bilinear function of the unknowns $\mathbf{h}$ and $\mathbf{X}$

- Ill-posed problem $\Rightarrow$ $L + NP$ unknowns and NP observations

  $\Rightarrow$ **As.:** $\mathbf{X}$ has S-sparse columns i.e., $\|\mathbf{X}\|_0 := |\text{supp}(\mathbf{X})| \leq PS$

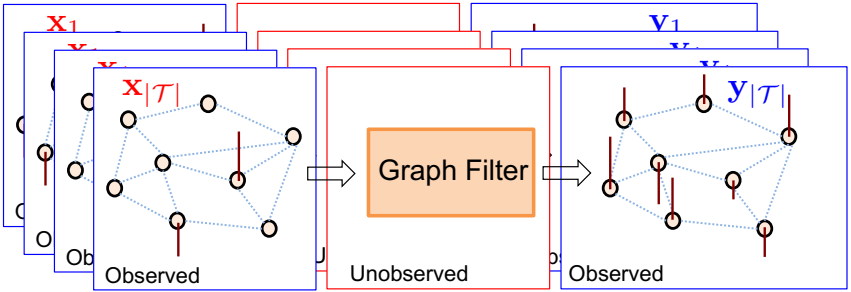
- Blind graph filter identification $\Rightarrow$ Non-convex feasibility problem

$$\text{find } \{\mathbf{h}, \mathbf{X}\}, \quad \text{s. to } \mathbf{Y} = \mathbf{V}\text{diag}(\boldsymbol{\Psi}\mathbf{h})\mathbf{V}^{\top}\mathbf{X}, \;\; \|\mathbf{X}\|_0 \leq PS$$

  $\Rightarrow$ Identifiability for Bernoulli-Gaussian model on $\mathbf{X}$ [Li et al'17]

S. Segarra, G. Mateos, A. G. Marques and A. Ribeiro, "Blind identification of graph filters," *IEEE TSP*, 2017

- Given independent training samples $\mathcal{T} := \{\mathbf{X}_i, \mathbf{Y}_i\}_{i=1}^{|\mathcal{T}|}$
  - $\Rightarrow$ Drawn from joint distribution of filters and sparse sources



- **Goal:** learn the parametric mapping $\hat{\mathbf{X}} = \Phi(\mathbf{Y}; \mathbf{\Theta})$ by minimizing a loss

$$L(\mathbf{\Theta}) := \frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} \ell(\mathbf{X}_i, \Phi(\mathbf{Y}_i; \mathbf{\Theta}))$$

- Source localization on graphs
  - Maximum-likelihood estimator optimal for trees [Pinto et al'12]
  - Scalable under restrictive dependency assumptions [Feizi el al'16]
  - Non-convex estimators of sparse sources [Pena et al'16], [Hu et al'16]
- Blind identification of graph filters [Segarra et al'17]
  - Matrix lifting can hinder applicability to large graphs
- Blind identification of invertible graph filters [Ye et al'18]
  - Convex formulation amenable to e.g., ADMM solvers
  - Multi-signal case with arbitrary supports

- Source localization on graphs
  - Maximum-likelihood estimator optimal for trees [Pinto et al'12]
  - Scalable under restrictive dependency assumptions [Feizi el al'16]
  - Non-convex estimators of sparse sources [Pena et al'16], [Hu et al'16]
- Blind identification of graph filters [Segarra et al'17]
  - Matrix lifting can hinder applicability to large graphs
- Blind identification of invertible graph filters [Ye et al'18]
  - Convex formulation amenable to e.g., ADMM solvers
  - Multi-signal case with arbitrary supports
- **Our contribution:** data-driven deep learning solution rooted in GSP

  $\Rightarrow$ Unroll and truncate the model-based ADMM iterations

  $\Rightarrow$ Trainable parametric architecture is interpretable

  $\Rightarrow$ Parameter efficient and offers controllable complexity

- Inverse filter $\mathbf{G} = \mathbf{H}^{-1}$ is also a graph filter on G [Sandryhaila-Moura'13]
  - $\Rightarrow$ Requires $\tilde{h}_i = \sum_{l=0}^{L-1} h_l \lambda_i^l \neq 0$, for all $i = 1, ..., N$
  - $\Rightarrow$ Inverse-filter coefficients $\mathbf{g} \in \mathbb{R}^N$, frequency response $\tilde{\mathbf{g}} = \mathbf{\Psi g}$
- Recast as linear inverse problem [Wang-Chi'16], [Ye et al'18]

$$\min_{\{\tilde{\mathbf{g}}, \mathbf{X}\}} \|\mathbf{X}\|_0, \quad \text{s. to} \quad \mathbf{X} = \mathbf{V}\text{diag}(\tilde{\mathbf{g}})\mathbf{V}^\top \mathbf{Y}, \quad \mathbf{X} \neq \mathbf{0}$$

- Still NP hard.

- Inverse filter $\mathbf{G} = \mathbf{H}^{-1}$ is also a graph filter on G [Sandryhaila-Moura'13]
  - $\Rightarrow$ Requires $\tilde{h}_i = \sum_{l=0}^{L-1} h_l \lambda_i^l \neq 0$, for all $i = 1, ..., N$
  - $\Rightarrow$ Inverse-filter coefficients $\mathbf{g} \in \mathbb{R}^N$, frequency response $\tilde{\mathbf{g}} = \mathbf{\Psi g}$
- Recast as linear inverse problem [Wang-Chi'16], [Ye et al'18]

$$\min_{\{\tilde{\mathbf{g}}, \mathbf{X}\}} \|\mathbf{X}\|_0, \quad \text{s. to} \quad \mathbf{X} = \mathbf{V}\text{diag}(\tilde{\mathbf{g}})\mathbf{V}^\top \mathbf{Y}, \quad \mathbf{X} \neq \mathbf{0}$$

- Still NP hard. Relax! and minimize convex $\|\mathbf{X}\|_1$

$$\hat{\tilde{\mathbf{g}}} = \underset{\tilde{\mathbf{g}}}{\text{argmin}} \|(\mathbf{Y}^\top \mathbf{V} \odot \mathbf{V})\tilde{\mathbf{g}}\|_1, \quad \text{s. to} \quad \mathbf{1}^\top \tilde{\mathbf{g}} = c$$

  - $\Rightarrow$ Constraint fixes the scale and avoids all-zero solution
  - $\Rightarrow$ ℓ_1-synthesis problem, can be solved via ADMM

- Let $\mathbf{Z} = \mathbf{Y}^\top \mathbf{V} \odot \mathbf{V}$ and $\mathbf{x} = \mathrm{vec}(\mathbf{X})$, rewrite problem as

$$\min_{\{\tilde{\mathbf{g}}, \mathbf{x}\}} \|\mathbf{x}\|_1, \quad \text{s. to} \quad \mathbf{1}^\top \tilde{\mathbf{g}} = c,\ \mathbf{Z}\tilde{\mathbf{g}} - \mathbf{x} = \mathbf{0}$$

- ADMM solver ($\mathbf{\Gamma} := \rho_\lambda \mathbf{Z}^\top \mathbf{Z} + \rho_\mu \mathbf{1}_N \mathbf{1}_N^\top$, dual variables $\{\boldsymbol{\lambda}, \mu\}$)

$$\begin{aligned}
\tilde{\mathbf{g}}[k+1] &= \mathbf{\Gamma}^{-1}\left[\mathbf{Z}^\top(\rho_\lambda \mathbf{x}[k] - \boldsymbol{\lambda}[k]) + (\rho_\mu c - \mu[k])\mathbf{1}_N\right], \\
\mathbf{x}[k+1] &= \mathcal{S}_{\rho_\lambda^{-1}}(\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \boldsymbol{\lambda}[k]/\rho_\lambda), \\
\boldsymbol{\lambda}[k+1] &= \boldsymbol{\lambda}[k] + \rho_\lambda(\mathbf{Z}\tilde{\mathbf{g}}[k+1] - \mathbf{x}[k+1]), \\
\mu[k+1] &= \mu[k] + \rho_\mu(\mathbf{1}_N^\top \tilde{\mathbf{g}}[k+1] - c)
\end{aligned}$$

- Limitations
  - Step-sizes ρ_λ, ρ_μ need to be tuned
  - Hundreds or thousands of iterations until convergence
  - Matrix inversion may hinder scalability to large graphs
- **Idea:** unroll the iterations and learn the parameters from dataset $\mathcal{T}$

- Map ADMM updates as sub-layers within a layer of the deep net
  - $\Rightarrow$ Stack K of the resulting layers to form $\Phi(\mathbf{Y};\boldsymbol{\Theta})$
  - $\Rightarrow$ End-to-end learning of $\boldsymbol{\Theta}$ using mini-batch SGD

Filter sub-layer $\mathcal{G}_k$

$$\tilde{\mathbf{g}}[k+1] = (\boldsymbol{\Gamma}^{(k)})^{-1}[\mathbf{Z}^\top(\mathbf{x}[k] - \rho_1^{(k)}\boldsymbol{\lambda}[k]) + (\rho_2^{(k)}c - \rho_1^{(k)}\mu[k])\mathbf{1}_N]$$

$$\boldsymbol{\Gamma}^{(k)} = \mathbf{Z}^\top\mathbf{Z} + \rho_2^{(k)}\mathbf{1}_N\mathbf{1}_N^\top$$

$\Rightarrow$ Learn $\{\rho_1^{(k)}, \rho_2^{(k)}\}_{k=1}^K$, where $\rho_1^{(k)}, \rho_2^{(k)} \geq 0$, for $k = 1, \ldots, K$

- Map ADMM updates as sub-layers within a layer of the deep net
  - $\Rightarrow$ Stack K of the resulting layers to form $\Phi(\mathbf{Y};\boldsymbol{\Theta})$
  - $\Rightarrow$ End-to-end learning of $\boldsymbol{\Theta}$ using mini-batch SGD

Sources sub-layer $\mathcal{X}_k$

$$\mathbf{x}[k+1] = \mathcal{S}_{\tau^{(k)}}(\alpha_1^{(k)}\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \alpha_2^{(k)}\boldsymbol{\lambda}[k])$$

$\Rightarrow$ Learn $\{\alpha_1^{(k)}, \alpha_2^{(k)}, \tau^{(k)}\}$, where $\tau^{(k)} \geq 0$, for $k = 1, \ldots, K$

- Map ADMM updates as sub-layers within a layer of the deep net
  - $\Rightarrow$ Stack K of the resulting layers to form $\Phi(\mathbf{Y};\mathbf{\Theta})$
  - $\Rightarrow$ End-to-end learning of $\mathbf{\Theta}$ using mini-batch SGD

Multiplier sub-layer $\mathcal{M}_k$

$$\boldsymbol{\lambda}[k+1] = \beta_1^{(k)}\boldsymbol{\lambda}[k] + \beta_2^{(k)}\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \beta_3^{(k)}\mathbf{x}[k+1]$$

$$\mu[k+1] = \gamma_1^{(k)}\mu[k] + \gamma_2^{(k)}\mathbf{1}_N^\top\tilde{\mathbf{g}}[k+1] + \gamma_3^{(k)}c$$

$\Rightarrow$ Learn $\{\beta_i^{(k)}\}_{i=1}^3, \{\gamma_i^{(k)}\}_{i=1}^3$, for $k = 1, \ldots, K$

- Design considerations
  - Additional parameters to broaden the model's expressive power
  - Forgo the parameter sharing constraint imposed by the unrolling
- Source location predictions: $\Phi(\mathbf{Y}_{\text{test}}, \hat{\mathbf{\Theta}}) = \text{unvec}[(\mathbf{Y}_{\text{test}}^\top\mathbf{V} \odot \mathbf{V})\tilde{\mathbf{g}}[K]]$

Algorithm 1 ADMM for Source Localization on Graphs

Require : $\rho_\lambda, \rho_\mu, c$

Initialize : $\{\mathbf{x}[0], \boldsymbol{\lambda}[0], \mu[0]\}$ at random

for $k = 0, 1, 2, \ldots$ **do**

$\tilde{\mathbf{g}}[k+1] \leftarrow \boldsymbol{\Gamma}^{-1}\left[\mathbf{Z}^\top(\rho_\lambda \mathbf{x}[k] - \boldsymbol{\lambda}[k]) + (\rho_\mu c - \mu[k])\mathbf{1}_N\right]$

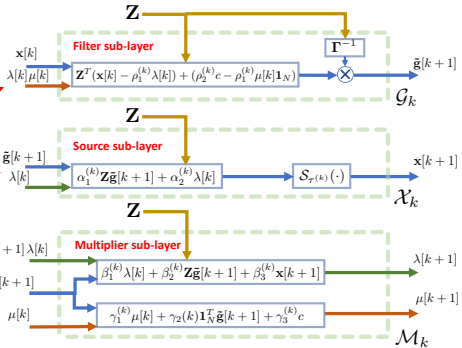
$\mathbf{x}[k+1] \leftarrow \mathcal{S}_{\rho_\lambda^{-1}}(\mathbf{Z}\tilde{\mathbf{g}}[k+1] + \boldsymbol{\lambda}[k]/\rho_\lambda)$

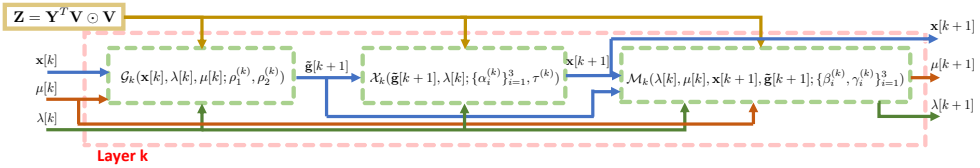
$\boldsymbol{\lambda}[k+1] \leftarrow \boldsymbol{\lambda}[k] + \rho_\lambda(\mathbf{Z}\tilde{\mathbf{g}}[k+1] - \mathbf{x}[k+1])$

$\mu[k+1] \leftarrow \mu[k] + \rho_\mu(\mathbf{1}_N^\top \tilde{\mathbf{g}}[k+1] - c)$

end for

return $\{\tilde{\mathbf{g}}[k+1], \mathbf{x}[k+1]\}$





k-th layer of the SLoG-Net architecture

- Consider undirected random graphs with $\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$
  - $\Rightarrow$ Erdős-Rényi with $N = 20$ and edge prob. $p = 0.3$
- Sources $\mathbf{X}_{\text{train}} \in \mathbb{R}^{N \times |\mathcal{T}|}$ from a Bernoulli-Gaussian model
  - $\Rightarrow$ $|\mathcal{T}| = 64000$ with Bernoulli parameter $\theta = 0.2$
- Filter $\mathbf{h}_{\text{train}} = (\mathbf{e}_1 + \alpha\mathbf{b})/\|\mathbf{e}_1 + \alpha\mathbf{b}\|_1$ as in [Wang-Chi'16]
  - $\Rightarrow$ $\mathbf{e}_1 = [1, 0, \ldots, 0]^\top \in \mathbb{R}^L$ and $\mathbf{b} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - $\Rightarrow$ Recovery performance increases while $\alpha \geq 0$ decreases

- Consider undirected random graphs with $\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$
  - $\Rightarrow$ Erdős-Rényi with $N = 20$ and edge prob. $p = 0.3$
- Sources $\mathbf{X}_{\text{train}} \in \mathbb{R}^{N \times |\mathcal{T}|}$ from a Bernoulli-Gaussian model
  - $\Rightarrow$ $|\mathcal{T}| = 64000$ with Bernoulli parameter $\theta = 0.2$
- Filter $\mathbf{h}_{\text{train}} = (\mathbf{e}_1 + \alpha\mathbf{b})/\|\mathbf{e}_1 + \alpha\mathbf{b}\|_1$ as in [Wang-Chi'16]
  - $\Rightarrow$ $\mathbf{e}_1 = [1, 0, \ldots, 0]^\top \in \mathbb{R}^L$ and $\mathbf{b} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  - $\Rightarrow$ Recovery performance increases while $\alpha \geq 0$ decreases
- **Figure of merit:** Relative recovery error of $\mathbf{X}$ and $\tilde{\mathbf{g}}$
  - $\Rightarrow$ $\text{RMSE}_{\mathbf{X}} = \|\hat{\mathbf{X}} - \mathbf{X}_{\text{test}}\|_F / \|\mathbf{X}_{\text{test}}\|_F$, $\text{RMSE}_{\tilde{\mathbf{g}}} = \|\hat{\tilde{\mathbf{g}}} - \tilde{\mathbf{g}}_{\text{test}}\|_2 / \|\tilde{\mathbf{g}}_{\text{test}}\|_2$
- **Figure of merit:** Source localization accuracy
  - $\Rightarrow$ $\text{ACC} = |\hat{\mathcal{I}} \cap \mathcal{I}_{\text{test}}| / |\mathcal{I}_{\text{test}}|$
  - $\Rightarrow$ Support estimate $\hat{\mathcal{I}} = \text{supp}_\kappa(\hat{\mathbf{X}})$: if $|\hat{\mathbf{X}}_{ip}| \geq \kappa$, then $(i, p) \in \hat{\mathcal{I}}$

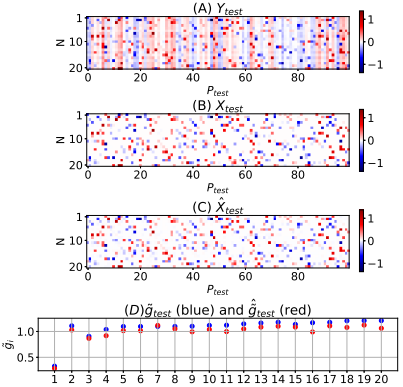
- Each epoch: randomly split $\mathbf{X}_{\text{train}}$ into $Q = 1600$ mini-batches $\{\mathbf{X}_q\}_{q=1}^{Q}$
- Then sample Q graph filter coefficients $\{\mathbf{h}_q\}_{q=1}^{Q}$, $L = 3, \alpha = 1$

  $\Rightarrow$ Generate the observations $\mathbf{Y}_q = \mathbf{V}\text{diag}(\mathbf{\Psi}_L \mathbf{h}_q)\mathbf{V}^\top \mathbf{X}_q$
- Loss function accounting for sign ambiguity

$$L(\mathbf{\Theta}) = \sum_{q=1}^{Q} \min\left(\frac{\|\mathbf{\Phi}(\mathbf{Y}_q; \mathbf{\Theta}) - \mathbf{X}_q\|_F}{\|\mathbf{X}_q\|_F}, \frac{\|\mathbf{\Phi}(\mathbf{Y}_q; \mathbf{\Theta}) + \mathbf{X}_q\|_F}{\|\mathbf{X}_q\|_F} \right)$$

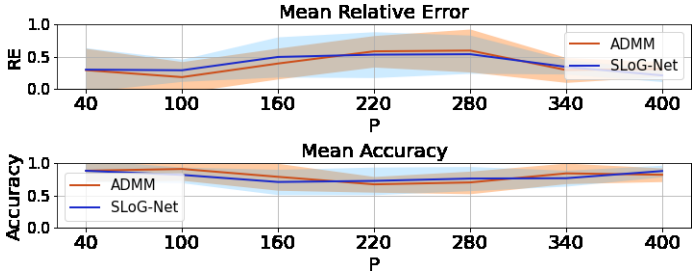
- Parameter initialization
  - $\{\rho_1^{(k)}, \rho_2^{(k)}, \tau^{(k)}\}_{k=1}^{K}$ are i.i.d. uniformly distributed in $[0, 1]$
  - All other parameters are drawn from a standard Gaussian

- Visualization of recovery performance for SLoG-Net with $K = 5$ layers



- Recovery errors are $\mathrm{RMSE}_{\mathbf{X}} = 0.090$ and $\mathrm{RMSE}_{\tilde{\mathbf{g}}} = 0.086$

- Comparisons with ADMM ($N = 20$, $P_{\text{test}} = 100$, over 50 trials)
  - Top: $\text{RMSE}_{\mathbf{X}}$ for ADMM and SLoG-Net
    $\Rightarrow$ The shaded region indicates the standard deviation
  - Bottom: ACC of support recovery, $\kappa = 0.1$
    $\Rightarrow$ Support estimate $\hat{\mathcal{I}} := \text{supp}_{\kappa}(\Phi(\mathbf{Y}_{\text{test}}; \hat{\mathbf{\Theta}}))$,
    $\Rightarrow$ Ground truth, $\mathcal{I}_{\text{test}} := \text{supp}_{\kappa}(\mathbf{X}_{\text{test}})$.



- Mean elapsed time: 0.005s for SLoG-Net, 0.067s for ADMM

Concluding summary

- Learning to identify sources of network diffusion
  - $\Rightarrow$ Data-driven trainable neural network architecture
  - $\Rightarrow$ Key: leverage inductive biases of the GSP model-based solution
- Unrolling the ADMM iterations
  - $\Rightarrow$ The SLoG-Net architecture is interpretable
  - $\Rightarrow$ Parameter and time efficient compared with iterative ADMM
- Ongoing work
  - $\Rightarrow$ Recovery of noise-corrupted signals
  - $\Rightarrow$ Effective recovery of more general graph filters
  - $\Rightarrow$ Evaluate performance on real data sets