Inferring Networks and Network Properties from Graph Dynamic Processes

Santiago Segarra

Electrical and Computer Engineering
Rice University
segarra@rice.edu
http://segarra.rice.edu

One World SP Seminar Series, July 16-17, 2020
Desiderata: Process, analyze and learn from network data [Kolaczyk09]
Desiderata: Process, analyze and learn from network data [Kolaczyk09]

Network as graph $G$: encodes pairwise relationships between agents

Interest not only in $G$, also in network data associated with the nodes

Combine Network Science and Signal Processing and Machine Learning to leverage the structure of networks for the better understanding of data defined on them
Understanding Networks

- Understanding Network Data
- Applications
- Graphons
- Metric projections
- Hierarchical clustering
- Overlapping clustering
- Node centralities
- Authorship attribution
- Early modern English plays
- Shakespearean Literature
- Political Science
- Neuroscience
- Dual graphs
- Network inference
- Stationarity
- Deconvolution
- Signal distance
- Reconstruction
- Graph neural nets
- Median graph filters
- Reconstruct
- Filter design
- Sampling
- Median graph filters
- Deconvolution
Motivating examples – Processing signals

Interpolate a brain signal from local observations

Compress a signal in an irregular domain

Localize the source of a rumor

Smooth an observed network profile

Predict the evolution of a network process

Infer the topology where the signals reside
Consider graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{W})$. **Graph signals** are mappings $x : \mathcal{V} \rightarrow \mathbb{R}$

- Defined on the nodes of the graph

May be represented as a vector $x \in \mathbb{R}^N$

- $x_i$ denotes the signal value at the $i$-th vertex in $\mathcal{V}$

$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{10} \end{bmatrix} = \begin{bmatrix} 0.6 \\ 0.7 \\ 0.3 \\ \vdots \\ 0.7 \end{bmatrix}$
Associated with $G$ is the graph-shift operator $S \in \mathbb{R}^{N \times N}$

- $S$ can take nonzero values in the edges of $G$ or in its diagonal
  \[ \Rightarrow S \text{ transformation that can be computed locally at the nodes} \]

**Ex:** Adjacency $A$ and Laplacian $L = D - A$ matrices
Associated with $G$ is the **graph-shift operator** $S \in \mathbb{R}^{N \times N}$

- $S$ can take nonzero values in the edges of $G$ or in its diagonal
  \[ \Rightarrow S \text{ transformation that can be computed locally at the nodes} \]

**Ex:** Adjacency $A$ and Laplacian $L = D - A$ matrices
Associated with $G$ is the **graph-shift operator** $S \in \mathbb{R}^{N \times N}$

- $S$ can take nonzero values in the edges of $G$ or in its diagonal
  - $S$ transformation that can be computed locally at the nodes

**Ex:** Adjacency $A$ and Laplacian $L = D - A$ matrices
Graph Fourier Transform (GFT)

Let $S = V \Lambda V^{-1}$ be the shift associated with $G$

The Graph Fourier Transform (GFT) of $x$ is defined as

$$\tilde{x} = V^{-1}x$$

While the inverse GFT (iGFT) of $\tilde{x}$ is defined as $x = V\tilde{x}$
Let $S = V \Lambda V^{-1}$ be the shift associated with $G$.

The Graph Fourier Transform (GFT) of $x$ is defined as

$$\tilde{x} = V^{-1}x$$

While the inverse GFT (iGFT) of $\tilde{x}$ is defined as $x = V\tilde{x}$.
Let $S = V \Lambda V^{-1}$ be the shift associated with $G$.

The Graph Fourier Transform (GFT) of $x$ is defined as

$$\tilde{x} = V^{-1}x$$

While the inverse GFT (iGFT) of $\tilde{x}$ is defined as $x = V\tilde{x}$

$$v_k^T L v_k = \sum_{(i,j) \in \mathcal{E}} A_{ij} ([v_k]_i - [v_k]_j)^2 = TV(v_k)$$

$\lambda_1 = 0 \leq \lambda_2 \leq \lambda_3 \leq \lambda_{20}$
The Graph Fourier Transform (GFT) of $x$ is defined as

$$\tilde{x} = V^{-1}x$$

While the inverse GFT (iGFT) of $\tilde{x}$ is defined as $x = V\tilde{x}$

$$\lambda_k = \lambda_k v_k^T v_k = v_k^T L v_k = \sum_{(i,j) \in \mathcal{E}} A_{ij}([v_k]_i - [v_k]_j)^2 = TV(v_k)$$
Let \( S = V \Lambda V^{-1} \) be the shift associated with \( G \).

The Graph Fourier Transform (GFT) of \( x \) is defined as

\[
\tilde{x} = V^{-1}x
\]

While the inverse GFT (iGFT) of \( \tilde{x} \) is defined as \( x = V\tilde{x} \).

\[
\lambda_k = \lambda_k v_k^T v_k = v_k^T L v_k = \sum_{(i,j) \in \mathcal{E}} A_{ij} ([v_k]_i - [v_k]_j)^2 = TV(v_k)
\]

\( V_1 \) \hspace{1cm} \( V_2 \) \hspace{1cm} \( V_3 \) \hspace{1cm} \( V_{20} \)

\( \lambda_1 = 0 \leq \lambda_2 \leq \lambda_3 \leq \lambda_{20} \)
A graph filter $H : \mathbb{R}^N \rightarrow \mathbb{R}^N$ is a map between graph signals of the form Polynomial in $S$ of degree $L$, with coeff. $h = [h_0, \ldots, h_L]^T$ [Sandryhaila-Moura13]

$$H := h_0 S^0 + h_1 S^1 + \ldots + h_L S^L = \sum_{l=0}^{L} h_l S^l$$

If $y := Hx$, Def 1 says $y = \sum_{l=0}^{L} h_l x^{(l)}$ (shifted versions of $x$)

$\Rightarrow x^{(l)} := S^l x = S x^{(l-1)}$ can be found locally and sequentially
Network topology inference

Understanding Networks

- Graphons
- Metric projections
- Hierarchical clustering
- Overlapping clustering
- Node centralities
- Authorship attribution
- Early modern English plays
- Shakespearean Literature
- Neuroscience
- Political Science

Understanding Network Data

- Dual graphs
- Stationarity
- Sampling
- Signal distance
- Deconvolution
- Reconstruction
- Filter design
- Median graph filters

Applications

- Graph neural nets
- Stationarity
- Signal distance
- Filter design
- Sampling
- Deconvolution
- Reconstruction

Network inference
Network topology inference from nodal observations [Kolaczyk09]

- Approaches use Pearson correlations to construct graphs [Brovelli04]
- Partial correlations and conditional dependence [Friedman08]
- [Banerjee08], [Lake10], [Slawski15], [Meinshausen06], [Karanikolas16]
Network topology inference from nodal observations [Kolaczyk09]

⇒ Approaches use Pearson correlations to construct graphs [Brovelli04]
⇒ Partial correlations and conditional dependence [Friedman08]
⇒ [Banerjee08], [Lake10], [Slawski15], [Meinshausen06], [Karanikolas16]

Key in neuroscience [Sporns10]

⇒ Functional net inferred from activity
Network topology inference and GSP

Network **topology inference** from nodal observations [Kolaczyk09]
  ⇒ Approaches use **Pearson correlations** to construct graphs [Brovelli04]
  ⇒ Partial correlations and conditional dependence [Friedman08]
  ⇒ [Banerjee08], [Lake10], [Slawski15], [Meinshausen06], [Karanikolas16]

Key in neuroscience [Sporns10]
  ⇒ Functional net inferred from activity

Most GSP works assume that $S$ (hence the graph) is known
  ⇒ Analyze how the characteristics of $S$ affect the signals and filters

We take the reverse path
  ⇒ How to use **GSP to infer the graph topology**? [TSIPN17] [SPMag19]
  ⇒ [Dong16], [Kalofolias16], [Mei17], [Shen17], [Pasdeloup17], [Egilmez17]
Connecting the dots

▶ Recent tutorials on learning graphs from data
▶ IEEE Signal Processing Magazine and Proceedings of the IEEE

▶ IEEE Trans. on Signal and Information Processing over Networks
▶ Issue on Network Topology Inference earlier this year
Setup

- Undirected network $G$ with unknown graph shift $S$
- Observe signals $\{y_i\}_{i=1}^P$ defined on the unknown graph
**Problem formulation**

**Setup**
- Undirected network $G$ with unknown graph shift $S$
- Observe signals $\{y_i\}_{i=1}^P$ defined on the unknown graph

**Problem statement**

Given observations $\{y_i\}_{i=1}^P$, determine the network $S$ knowing that $\{y_i\}_{i=1}^P$ are outputs of a diffusion process on $S$. 

---

$Y_1$ $Y_2$ $Y_3$
Generating structure of a diffusion process

- Signal $y_i$ is the response of a linear diffusion process to input $x_i$

$$y_i = \alpha_0 \prod_{l=1}^{\infty} (1 - \alpha_l S) x_i = \sum_{l=0}^{\infty} \beta_l S^l x_i, \quad i = 1, \ldots, P$$

$\Rightarrow$ Common generative model, e.g., heat diffusion, consensus
Generating structure of a diffusion process

- Signal $y_i$ is the response of a linear diffusion process to input $x_i$

\[ y_i = \alpha_0 \prod_{l=1}^{\infty} (1 - \alpha_l S) x_i = \sum_{l=0}^{\infty} \beta_l S^l x_i, \quad i = 1, \ldots, P \]

⇒ Common generative model, e.g., heat diffusion, consensus

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

\[ y_i = \left( \sum_{l=0}^{L-1} h_l S^l \right) x_i := H x_i, \quad i = 1, \ldots, P \]

⇒ Graph filter $H$ is shift invariant [Sandryhaila-Moura’13]
⇒ $H$ diagonalized by the eigenvectors $V$ of the shift operator
Signal $y_i$ is the response of a linear diffusion process to input $x_i$

$$y_i = \alpha_0 \prod_{l=1}^{\infty} (1 - \alpha_l S) x_i = \sum_{l=0}^{\infty} \beta_l S^l x_i, \quad i = 1, \ldots, P$$

Common generative model, e.g., heat diffusion, consensus

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

$$y_i = \left( \sum_{l=0}^{L-1} h_l S^l \right) x_i := H x_i, \quad i = 1, \ldots, P$$

Graph filter $H$ is shift invariant [Sandryhaila-Moura’13]

$H$ diagonalized by the eigenvectors $V$ of the shift operator

Goal: estimate undirected network $S$ from signal realizations $\{y_i\}_{i=1}^P$

Unknowns: filter order $L$, coefficients $\{h_l\}_{l=1}^{L-1}$, inputs $\{x_i\}_{i=1}^P$
Blueprint of our solution

STEP 1: Estimate the eigenvectors of $S$

STEP 2: Find eigenvalues via optimization

A priori info and desirable features

$\{y_i\}_{i=1}^P$
Blueprint of our solution

STEP 1: Estimate the eigenvectors of $\mathbf{S}$

A priori info and desirable features

STEP 2: Find eigenvalues via optimization

$\hat{\mathbf{V}}$: noisy

Sparsity and shift operator feasibility

$\{y_i\}_{i=1}^P$
Step 1: Obtaining the eigenvectors of $S$

- $y$ is the output of a **local diffusion** of a white input

$$y = \alpha_0 \prod_{l=1}^{\infty} (I - \alpha_l S)x = \left( \sum_{l=0}^{N-1} h_l S^l \right)x := Hx$$
Step 1: Obtaining the eigenvectors of $S$

- $y$ is the output of a local diffusion of a white input

$$ y = \alpha_0 \prod_{l=1}^{\infty} (I - \alpha_l S) x = \left( \sum_{l=0}^{N-1} h_l S^l \right) x := Hx $$

- The covariance $C_y$ of $y$ shares $V$ with $S$

$$ C_y = H^2 = h_0^2 I + 2h_0 h_1 S + h_1^2 S^2 + \ldots $$
Step 1: Obtaining the eigenvectors of $S$

- $y$ is the output of a local diffusion of a white input

$$y = \alpha_0 \prod_{l=1}^{\infty} (I - \alpha_l S)x = \left( \sum_{l=0}^{N-1} h_l S^l \right)x := Hx$$

- The covariance $C_y$ of $y$ shares $V$ with $S$

$$C_y = H^2 = h_0^2 I + 2h_0 h_1 S + h_1^2 S^2 + ...$$

- Mapping $S \rightarrow C_y$ is polynomial

  - Correlation methods $\Rightarrow C_y = S$
  - Precision methods (graphical Lasso) $\Rightarrow C_y = S^{-1}$
  - Structural EM methods $\Rightarrow C_y = (I - S)^{-2}$
Step 2: Convex recovery of the eigenvalues

- Use extra knowledge/assumptions to find the eigenvalues
  - Of all graphs, select one that is optimal in some sense

\[
S_0^* := \arg\min_{S, \lambda} \|S\|_0 \quad \text{s. to} \quad S = \sum_{k=1}^{N} \lambda_k v_k v_k^T, \quad S \in S
\]

- Set \( S \) contains all admissible scaled adjacency matrices

\[
S := \{S \mid S_{ij} \geq 0, \quad S \in \mathcal{M}^N, \quad S_{ii} = 0, \quad \sum_j S_{1j} = 1\}
\]
Step 2: Convex recovery of the eigenvalues

- Use extra knowledge/assumptions to find the eigenvalues
- Of all graphs, select one that is optimal in some sense

\[ S_0^* := \operatorname{argmin}_{S, \lambda} \|S\|_0 \quad \text{s. to} \quad S = \sum_{k=1}^{N} \lambda_k v_k v_k^T, \quad S \in S \]

- Set $S$ contains all admissible scaled adjacency matrices

\[ S := \{S \mid S_{ij} \geq 0, \ S \in \mathcal{M}^N, S_{ii} = 0, \ \sum_j S_{1j} = 1\} \]

- Non-convex problem, relax to $\ell_1$-norm minimization, e.g., [Tropp06]

\[ S_1^* := \operatorname{argmin}_{S, \lambda} \|S\|_1 \quad \text{s. to} \quad S = \sum_{k=1}^{N} \lambda_k v_k v_k^T, \quad S \in S \]

- Does the solution $S_1^*$ coincide with the $\ell_0$ solution $S_0^*$?
Recovery guarantee for $\ell_1$ relaxation

- Define $W := V \odot V$
- Build $M := (I - WW^\dagger)_{D^c}$ the orthogonal projector onto $\text{range}(W)$
  - $\Rightarrow$ Construct $R := [M, \ e_1 \otimes 1_{N-1}]$
  - $\Rightarrow$ Denote by $\mathcal{K}$ the indices of the support of $s_0^* = \text{vec}(S_0^*)$

$S_1^*$ and $S_0^*$ coincide if the two following conditions are satisfied:
1) $\text{rank}(R_{\mathcal{K}}) = |\mathcal{K}|$; and
2) There exists a constant $\delta > 0$ such that

\[
\psi_R := \|I_{\mathcal{K}^c}(\delta^{-2}RR^T + I_{\mathcal{K}^c}^T I_{\mathcal{K}^c})^{-1}I_{\mathcal{K}^c}^T\|_\infty < 1.
\]

- Cond. 1) ensures uniqueness of solution $S_1^*$
- Cond. 2) guarantees existence of a dual certificate for $\ell_0$ optimality
Recovery guarantee for $\ell_1$ relaxation

- Define $W := V \odot V$
- Build $M := (I - WW^\dagger)_{D^c}$ the orthogonal projector onto $\text{range}(W)$
  - Construct $R := [M, \ e_1 \otimes 1_{N-1}]$
  - Denote by $\mathcal{K}$ the indices of the support of $s_0^* = \text{vec}(S_0^*)$

$S_1^*$ and $S_0^*$ coincide if the two following conditions are satisfied:
1) $\text{rank}(R_{\mathcal{K}}) = |\mathcal{K}|$; and
2) There exists a constant $\delta > 0$ such that

$$\psi_R := \|I_{\mathcal{K}^c}(\delta^{-2}RR^T + I_{\mathcal{K}^c}^T I_{\mathcal{K}^c}^{-1} I_{\mathcal{K}^c})^{-1} I_{\mathcal{K}^c}^T\|_{\infty} < 1.$$ 

- Cond. 1) ensures uniqueness of solution $S_1^*$
- Cond. 2) guarantees existence of a dual certificate for $\ell_0$ optimality
Robust shift identification

STEP 1: Estimate the eigenvectors of $\tilde{S}$

A priori info and desirable features

STEP 2: Find eigenvalues via optimization

$\{y_i\}_{i=1}^P$ → $\hat{S}$

$\hat{V}$: noisy

Sparsity and shift operator feasibility

Q1: How to modify the optimization problem to make it robust?

Q2: Recovery guarantees in this robust setting?
Robust shift identification

STEP 1: Estimate the eigenvectors of $S$

STEP 2: Find eigenvalues via optimization

A priori info and desirable features

Sparsity and shift operator feasibility

Q1: How to modify the optimization problem to make it robust?

Q2: Recovery guarantees in this robust setting?
We might have access to \( \hat{\mathbf{V}} \), a noisy version of the eigenvectors with 

\[
\min_{\{\mathbf{S}, \lambda, \hat{\mathbf{S}}\}} \|\mathbf{S}\|_1 \quad \text{s. to} \quad \hat{\mathbf{S}} = \sum_{k=1}^{N} \lambda_k \hat{\mathbf{v}}_k \hat{\mathbf{v}}_k^T, \quad \mathbf{S} \in S, \quad d(\mathbf{S}, \hat{\mathbf{S}}) \leq \epsilon
\]

How does the noise in \( \hat{\mathbf{V}} \) affect the recovery?
We might have access to \( \hat{V} \), a noisy version of the eigenvectors.

With \( d(\cdot,\cdot) \) denoting a (convex) distance between matrices,

\[
\min_{\{S,\lambda,\hat{S}\}} \|S\|_1 \quad \text{s. to} \quad \hat{S} = \sum_{k=1}^{N} \lambda_k \hat{v}_k \hat{v}_k^T, \quad S \in S, \quad d(S,\hat{S}) \leq \epsilon
\]

How does the noise in \( \hat{V} \) affect the recovery?

Conditions 1) and 2) but based on \( \hat{R} \), guaranteed

\[
d(S^*,S^*_0) \leq C\epsilon
\]

\( \Rightarrow \) \( \epsilon \) large enough to guarantee feasibility of \( S^*_0 \)

\( \Rightarrow \) Constant \( C \) depends on \( \hat{V} \) and the support \( \mathcal{K} \)
Performance comparisons

- Comparison with **graphical lasso** and **sparse correlation** methods
  - Evaluated on 100 realizations of ER graphs with $N = 20$ and $p = 0.2$

![Graphical representation with data points and lines indicating performance comparisons between different methods.]

- Graphical lasso **implicitly assumes a filter** $H_1 = (\rho I + S)^{-1/2}$
  - For this filter our method works, but not as well

- For **general diffusion filters** $H_2$ our method still works fine
Inferring the structure of a protein

- Our method can be used to **sparsify a given network**
  - Keep direct and important edges or relations
  - Discard indirect relations that can be explained by direct ones

- Use eigenvectors $\hat{V}$ of given network as noisy eigenvectors of $S$

**Ex:** Infer **contact between amino-acid residues** in BPT1 BOVIN
  - Use mutual information of amino-acid covariation as input

- Network deconvolution assumes a specific filter model [Feizi13]
  - We achieve better performance by being agnostic to this
Sensitivity of recovered edges

- **Sensitivity** of the top edge predictions
  - Fraction of the real contact edges recovered

- For $\epsilon = 0$ we force $S$ to be mutual information matrix $S'$

- For larger values of $\epsilon$, we get a better recovery
A rich framework for network inference

\[
\{x_i\}_{i=1}^P \xrightarrow{H(S)} \{y_i\}_{i=1}^P \xrightarrow{\text{Network Inference}} \hat{S}
\]
A rich framework for network inference

$H \in \mathcal{H}$

$\{x_i\}_{i=1}^P \xrightarrow{H(S)} \{y_i\}_{i=1}^P \xrightarrow{\text{Network Inference}} \hat{S}$

- Prior knowledge on the filter class [Segarra et al’17] [Zhu et al’20]
A rich framework for network inference

\[ \sim \mathcal{N}(0, C_x) \]

\[ \{x_i\}_{i=1}^P \xrightarrow{\text{H}(S)} \{y_i\}_{i=1}^P \xrightarrow{\text{Network Inference}} \hat{S} \]

- Prior knowledge on the filter class [Segarra et al’17] [Zhu et al’20]
- Colored inputs to the diffusion process [Shafipour et al’17, ’19]
A rich framework for network inference

\[ S \neq S^T \]

Prior knowledge on the filter class [Segarra et al’17] [Zhu et al’20]

Colored inputs to the diffusion process [Shafipour et al’17, ’19]

Inference for directed graphs [Shafipour et al’18]
A rich framework for network inference

\[
\{x_i^{(1)}\}_{i=1}^P \xrightarrow{H(S_1)} \{y_i^{(1)}\}_{i=1}^P \\
\{x_i^{(2)}\}_{i=1}^P \xrightarrow{H(S_2)} \{y_i^{(2)}\}_{i=1}^P
\]

- Prior knowledge on the filter class [Segarra et al’17] [Zhu et al’20]
- Colored inputs to the diffusion process [Shafipour et al’17, ’19]
- Inference for directed graphs [Shafipour et al’18]
- Joint inference of multiple networks [Segarra et al’17]
A rich framework for network inference

Prior knowledge on the filter class [Segarra et al’17] [Zhu et al’20]
Colored inputs to the diffusion process [Shafipour et al’17, ’19]
Inference for directed graphs [Shafipour et al’18]
Joint inference of multiple networks [Segarra et al’17]
Recovering the community structure [Wai’18,’19] [Roddenberry’20]
A rich framework for network inference

\[ S \sim \mathcal{D}(\Omega) \]

- Prior knowledge on the filter class [Segarra et al'17] [Zhu et al'20]
- Colored inputs to the diffusion process [Shafipour et al'17, '19]
- Inference for directed graphs [Shafipour et al'18]
- Joint inference of multiple networks [Segarra et al'17]
- Recovering the community structure [Wai’18,’19] [Roddenberry’20]
- Estimating the graph model coefficients [Schaub et al’20]
A rich framework for network inference

Prior knowledge on the filter class [Segarra et al’17] [Zhu et al’20]
- Colored inputs to the diffusion process [Shafipour et al’17, ’19]
- Inference for directed graphs [Shafipour et al’18]
- Joint inference of multiple networks [Segarra et al’17]
- Recovering the community structure [Wai’18,’19] [Roddenberry’20]
- Estimating the graph model coefficients [Schaub et al’20]
- Recovering the node centralities [Roddenberry et al’20] [He at al’20]
What if we have low-rank data?

- **Low-rankness** is a prevalent feature in graph signals / network data

**Social network data / opinion:**

- **Polarization** is common in social networks’ opinions

**Gene network data:**

- Number of experiments available is limited due to time and labor cost.

- **Bad news**: oftentimes we do not have **full-rank** data

[Chua et al., PNAS, 2006]
Community detection

Remedy: relax the goal of learning the whole graph

- Understanding the community structure gives a macroscopic (or reduced resolution) view of the graph
  ⇒ useful for network analysis and influence maximization
- Typical approach requires perfect knowledge of the graph
A **Blind Community Detection** method to detect communities from graph signals without learning/storing the graph itself.
Given observations \( \{y_i\}_{i=1}^P \), determine the communities in \( G \) when:

(AS1) \( \{y_i\}_{i=1}^P \) are the outputs of a low-rank diffusion on \( G \)
Problem statement

Given observations $\{y_i\}_{i=1}^P$, determine the communities in $G$ when:

(AS1) $\{y_i\}_{i=1}^P$ are the outputs of a low-rank diffusion on $G$

$$y = \sum_{\ell=0}^L h_\ell S^\ell w = H(S)w$$
Given observations \( \{y_i\}_{i=1}^P \), determine the communities in \( G \) when:

(AS1) \( \{y_i\}_{i=1}^P \) are the outputs of a low-rank diffusion on \( G \)

\[
y = \sum_{\ell=0}^{L} h_\ell S^\ell w = H(S)w
\]

\[
w = Bz, \quad B \in \mathbb{R}^{N \times R}, \quad R \ll N.
\]
Given observations $\{y_i\}_{i=1}^P$, determine the communities in $G$ when:

(AS1) $\{y_i\}_{i=1}^P$ are the outputs of a low-rank diffusion on $G$

$$y = \sum_{\ell=0}^L h_\ell S^\ell w = H(S)w$$

$$w = Bz, \quad B \in \mathbb{R}^{N \times R}, \quad R \ll N.$$  

- Additional unknowns:
  - The filter $H$ $\Rightarrow$ Unknown $L$ and $\{h_\ell\}_{\ell=0}^L$
  - Tall matrix $B$
  - Input $z$ $\Rightarrow$ Assume statistical knowledge $\mathbb{E}[zz^\top] = I$
Quick review of spectral clustering

- Undirected graph $G = (V, E, A)$
- Let $V = C_1 \cup \cdots \cup C_K$ with $C_k \cap C_{k'} = \emptyset$, $k \neq k'$.
- “Best” clustering is achieved by minimizing:

$$\text{RatioCut}(C_1, \ldots, C_K) := \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i \in C_k} \sum_{j \in \overline{C_k}} A_{ij}$$

- **Spectral clustering** tackles it via low rank approximation —

Let $S := \text{Diag}(A1) - A$ be the graph Laplacian, and $V_K \in \mathbb{R}^{N \times K}$ as the collection of its smallest $K$ singular vectors. Perform $K$-means to:

$$\min_{C_1, \ldots, C_K} F(C_1, \ldots, C_K) := \sum_{k=1}^{K} \sum_{i \in C_k} \left\| \mathbf{v}_{i}^{\text{row}} - \frac{1}{|C_k|} \sum_{j \in C_k} \mathbf{v}_{j}^{\text{row}} \right\|_2^2,$$

where $\mathbf{v}_{i}^{\text{row}}$ is the $i$th row vector of $V_K$. 
Blind Community Detection (BlindCD):

1. Sample Covariance:
\[ \hat{C}_y = \frac{1}{T} \sum_{t=1}^{T} y^{(t)}(y^{(t)})^\top \]

2. Stack the top-$K$ singular vectors as \( \hat{P}_K \in \mathbb{R}^{N \times K} \).

3. Apply $K$-means on the $N$ row vectors of $\hat{P}_K$.

- A **blind** method as the graph and graph filter are unknown.
- If $\hat{P}_K$ spans the **same subspace** as $V_K$, then BlindCD is accurate.
High-level view of the problem

\[ z \rightarrow B \rightarrow w \rightarrow H(S) \rightarrow y \]
High-level view of the problem

\[
z \rightarrow B \rightarrow w \rightarrow H(S) \rightarrow y
\]
High-level view of the problem

\[ z \rightarrow B \rightarrow w \rightarrow H(S) \rightarrow y \rightarrow \hat{C}_y \rightarrow \{\hat{C}_1, \ldots, \hat{C}_K\} \]

**GENERATIVE MODEL**

**OBSERVATIONS**

**PROPOSED ALGORITHM**

Spectral Clustering
High-level view of the problem

\[
\begin{align*}
z & \rightarrow B & \rightarrow w & \rightarrow H(S) & \rightarrow y & \rightarrow \hat{C}_y \rightarrow \{\hat{C}_1, \ldots, \hat{C}_K\} \\
S & \rightarrow \{C_1, \ldots, C_K\} & \text{ORACLE} & \text{PROPOSED ALGORITHM} & \text{Spectral Clustering} & \text{OBSERVATIONS} & \text{GENERATIVE MODEL}
\end{align*}
\]
High-level view of the problem

\[ \begin{align*}
Z & \rightarrow B \rightarrow w \rightarrow H(S) \rightarrow y \\
\hat{C}_y & \rightarrow \{\hat{C}_1, \ldots, \hat{C}_K\} \\
S & \rightarrow \{C_1, \ldots, C_K\} \\
\end{align*} \]

How similar are these recovered communities?
Recall: Spectral clustering requires the smallest $K$ eigenvectors of $S$.

Specifically, we want $V_K$ from $V := (V_K \ V_{N-K})$ [recall $S = V \Lambda V^\top$].

Under the low-rank model, $w = Bz$, we have

$$C_y = HBB^\top H^\top = V(Diag(\tilde{h}))(V^\top B)(B^\top V)(Diag(\tilde{h}))V^\top$$

Intuitively, the singular vectors $V_K$ are in $C_y$ if

$$\operatorname{rank}(V_K^\top B) = K, \ V_{N-K}^\top B \approx 0$$

$$\tilde{h}_i \begin{cases} \neq 0, & i = 1, \ldots, K, \\ \approx 0, & i \geq K + 1. \end{cases}$$
**Definition:** a graph filter is said to be \((K, \eta)\)-low pass if

\[
\frac{\tilde{h}_{K+1}}{\tilde{h}_{K}} \leq \eta < 1, \quad \tilde{h}_1 \geq \tilde{h}_2 \geq \cdots \geq \tilde{h}_N \geq 0,
\]

where \( \tilde{h}_i := \left[ \sum_{\ell=0}^{L-1} h_{\ell} \Lambda^\ell \right]_{ii}, \ i = 1, \ldots, N. \)

- **An ideal low-pass graph filter** have \(\eta = 0.\)
- **Example 1** — **consensus dynamics**

\[
H_1(S) = (I - \alpha S)^{L-1} \implies \eta = \left( \frac{1 - \alpha \lambda_{K+1}}{1 - \alpha \lambda_K} \right)^{L-1}.
\]

- **Example 2** — **steady-state of DeGroot dynamics**

\[
H_2(S) = (I + c^{-1}S)^{-1} \implies \eta = \frac{1 + c^{-1} \lambda_K}{1 + c^{-1} \lambda_{K+1}}.
\]
Theoretical guarantee of BlindCD

Recall $F(C_1, \ldots, C_K) := \sum_{k=1}^{K} \sum_{i \in C_k} \| v_i^{\text{row}} - \frac{1}{|C_k|} \sum_{j \in C_k} v_j^{\text{row}} \|_2$. 

Let $F^* := \min_{C_1, \ldots, C_K} F(C_1, \ldots, C_K)$ and $\hat{C}_1, \ldots, \hat{C}_K$ be the communities found by BlindCD, we have:

Under some conditions (e.g., $\|C_y - \hat{C}_y\|$ is small, $R \geq K$), we have

$$\sqrt{F(\hat{C}_1, \ldots, \hat{C}_K)} - \sqrt{F^*} \leq \sqrt{8K} \left( \sqrt{\frac{\gamma^2}{1 + \gamma^2}} + \frac{\|C_y - \hat{C}_y\|_2}{\delta} \right),$$

where $\gamma \leq \eta \cdot \|V_{N-K}^\top BQ_K\|_2 \cdot \| (V_K^\top BQ_K)^{-1} \|_2$,

and $Q_K$ is a set of $K$ orthogonal vectors.

The error of BlindCD is decomposed into two parts —

- Non-ideal low-pass graph filter and mismatch between $V_K, B$.
- Due to insufficient samples in covariance estimation.
Infer partition of the network of \( n = 50 \) states of USA
\( i \)-th rollcall data is mapped into a graph signal \( y_i \in \mathbb{R}^{50} \)
United States Senate data

- **Infer partition** of the network of $n = 50$ states of USA
- *i*-th rollcall data is mapped into a graph signal $y_i \in \mathbb{R}^{50}$
Conclusions

- GSP approach to network inference in the graph spectral domain
  - Two step approach: i) Obtain $V$; ii) Estimate $S$ given $V$

- How to obtain the spectral templates $V$
  - Based on covariance of diffused signals
  - Other sources: network operators, network deconvolution
Conclusions

- GSP approach to network inference in the graph spectral domain
  - Two step approach: i) Obtain $V$; ii) Estimate $S$ given $V$

- How to obtain the spectral templates $V$
  - Based on covariance of diffused signals
  - Other sources: network operators, network deconvolution

- Infer $S$ via convex optimization
  - Objectives promote desirable physical properties
  - Constraints encode a priori information on structure
  - Robust formulations for noisy (and incomplete) templates
Whenever recovering the whole graph is not feasible

⇒ What can data tell us about the graph?
⇒ Try to recover coarser features ⇒ Communities
Conclusions

- Whenever recovering the whole graph is not feasible
  - What can data tell us about the graph?
  - Try to recover coarser features ⇒ Communities

How similar are these recovered communities?
Conclusions

Whenever recovering the whole graph is not feasible
⇒ What can data tell us about the graph?
⇒ Try to recover coarser features ⇒ Communities

How similar are these recovered communities?

\[
gap \leq f(B, H, P)
\]
Thank you

Antonio Marques (K. Juan Carlos)
Gonzalo Mateos (U. Rochester)
Hoi-To Wai (CUHK)
Asuman Ozdaglar (MIT)
Michael Schaub (RWTH Aachen)
Yuhao Wang (MIT)
Yu Zhu (Rice)

Alejandro Ribeiro (UPenn)
Rasoul Shafipour (U. Rochester)
Anna Scaglione (ASU)
Ali Jadbabaie (MIT)
John Tsitsiklis (MIT)
Caroline Uhler (MIT)
T. Mitchell Roddenberry (Rice)


References: Conference papers (1)


