Graph Signal Processing: Fundamentals and Applications to Diffusion Processes

A. G. Marques†, G. Mateos*, A. Ribeiro, and S. Segarra

†King Juan Carlos University – antonio.garcia.marques@urjc.es
*University of Rochester – gmateosb@ece.rochester.edu

Thanks: Weiyu Huang and Geert Leus

IEEE SAM Tutorial
Rio de Janeiro, Brazil
July 10, 2016
Desiderata: Process, analyze and learn from network data [Kolaczyk’09]

Network as graph $G = (V, E)$: encode pairwise relationships

Interest here not in $G$ itself, but in data associated with nodes in $V$

$\Rightarrow$ Object of study is a graph signal $x$

Q: Graph signals common and interesting as networks are?
Network of economic sectors of the United States

- Bureau of Economic Analysis of the U.S. Department of Commerce
- $\mathcal{E} = \text{Output of sector } i \text{ is an input to sector } j \text{ (62 sectors in } \mathcal{V})$

**Oil and Gas**

- Oil extraction (OG), Petroleum and coal products (PC), Construction (CO)

**Services**

- Administrative services (AS), Professional services (MP)
- Credit intermediation (FR), Securities (SC), Real state (RA), Insurance (IC)

**Finance**

- Only interactions stronger than a threshold are shown
Network of economic sectors of the United States

- Bureau of Economic Analysis of the U.S. Department of Commerce
- $\mathcal{E} = \text{Output of sector } i \text{ is an input to sector } j \text{ (62 sectors in } \mathcal{V})$

- A few sectors have widespread strong influence (services, finance, energy)
- Some sectors have strong indirect influences (oil)
- The heavy last row is final consumption

- This is an interesting network $\Rightarrow$ Signals on this graph are as well
Disaggregated GDP of the United States

- Signal $x = \text{output per sector} = \text{disaggregated GDP}$
  - Network structure used to, e.g., reduce GDP estimation noise

- Signal is as interesting as the network itself. Arguably more
  - Same is true on brain connectivity and fMRI brain signals, ...
  - Gene regulatory networks and gene expression levels, ...
  - Online social networks and information cascades, ...
  - Alignment of customer preferences and product ratings, ...
Graph signal processing

- **Graph SP**: broaden classical SP to graph signals [Shuman et al.’13]
  
  ⇒ **Our view**: GSP well suited to study network (diffusion) processes

- **As.**: Signal properties related to topology of $G$ (locality, smoothness)
  
  ⇒ Algorithms that fruitfully leverage this relational structure

- **Q**: Why do we expect the graph structure to be useful in processing $x$?
Importance of signal structure in time

- Signal and Information Processing is about exploiting signal structure

- Discrete time described by cyclic graph
  - Time $n$ follows time $n-1$
  - Signal value $x_n$ similar to $x_{n-1}$

- Formalized with the notion of frequency

- Cyclic structure $\Rightarrow$ Fourier transform $\Rightarrow \tilde{x} = F^H x$

- Fourier transform $\Rightarrow$ Projection on eigenvector space of cycle

\[
F_{kn} = \frac{e^{j2\pi kn/N}}{\sqrt{N}}
\]
Covariances and principal components

- Random signal with mean $\mathbb{E}[x] = 0$ and covariance $C_x = \mathbb{E}[xx^H]$
  $\Rightarrow$ Eigenvector decomposition $C_x = V\Lambda V^H$

- Covariance matrix $C_x$ is a graph
  $\Rightarrow$ Not a very good graph, but still

- Precision matrix $C_x^{-1}$ a common graph too
  $\Rightarrow$ Conditional dependencies of Gaussian $x$

- Covariance matrix structure $\Rightarrow$ Principal components (PCA) $\Rightarrow \tilde{x} = V^Hx$

- PCA transform $\Rightarrow$ Projection on eigenvector space of (inverse) covariance

- Q: Can we extend these principles to general graphs and signals?
Formally, a graph $G$ (or a network) is a triplet $(V, E, W)$

$V = \{1, 2, \ldots, N\}$ is a finite set of $N$ nodes or vertices

$E \subseteq V \times V$ is a set of edges defined as ordered pairs $(n, m)$

Write $\mathcal{N}(n) = \{m \in V : (m, n) \in E\}$ as the in-neighbors of $n$

$W : E \rightarrow \mathbb{R}$ is a map from the set of edges to scalar values $w_{nm}$

- Represents the level of relationship from $n$ to $m$
- Often weights are strictly positive, $W : E \rightarrow \mathbb{R}_{++}$

- **Unweighted** graphs $\Rightarrow w_{nm} \in \{0, 1\}$, for all $(n, m) \in E$
- **Undirected** graphs $\Rightarrow (n, m) \in E$ if and only if $(m, n) \in E$ and $w_{nm} = w_{mn}$, for all $(n, m) \in E$
Unweighted and directed graphs (e.g., time)

- $\mathcal{V} = \{0, 1, \ldots, 23\}$
- $\mathcal{E} = \{(0, 1), (1, 2), \ldots, (22, 23), (23, 0)\}$
- $W : (n, m) \mapsto 1$, for all $(n, m) \in \mathcal{E}$

Unweighted and undirected graphs (e.g., image)

- $\mathcal{V} = \{1, 2, 3, \ldots, 9\}$
- $\mathcal{E} = \{(1, 2), (2, 3), \ldots, (8, 9), (1, 4), \ldots, (6, 9)\}$
- $W : (n, m) \mapsto 1$, for all $(n, m) \in \mathcal{E}$

Weighted and undirected graphs (e.g., covariance)

- $\mathcal{V} = \{1, 2, 3, 4\}$
- $\mathcal{E} = \{(1, 1), (1, 2), \ldots, (4, 4)\} = \mathcal{V} \times \mathcal{V}$
- $W : (n, m) \mapsto \sigma_{nm} = \sigma_{mn}$, for all $(n, m)$
Adjacency matrix

- **Algebraic graph theory**: matrices associated with a graph $G$
  - Adjacency $A$ and Laplacian $L$ matrices
  - **Spectral graph theory**: properties of $G$ using spectrum of $A$ or $L$

- Given $G = (\mathcal{V}, \mathcal{E}, W)$, the adjacency matrix $A \in \mathbb{R}^{N \times N}$ is
  \[
  A_{nm} = \begin{cases} 
  w_{nm}, & \text{if } (n, m) \in \mathcal{E} \\
  0, & \text{otherwise}
  \end{cases}
  \]

- Matrix representation incorporating all information about $G$
  - For **unweighted** graphs, positive entries represent connected pairs
  - For **weighted** graphs, also denote proximities between pairs
Degree and $k$-hop neighbors

- If $G$ is unweighted and undirected, the degree of node $i$ is $|\mathcal{N}(i)|$
  \[\Rightarrow\text{ In directed graphs, have out-degree and in-degree}\]

- Using the adjacency matrix in the undirected case
  \[\Rightarrow\text{ For node } i: \text{deg}(i) = \sum_{j \in \mathcal{N}(i)} A_{ij} = \sum_{j} A_{ij}\]
  \[\Rightarrow\text{ For all } N \text{ nodes: } d = A1 \rightarrow \text{Degree matrix: } D := \text{diag}(d)\]

- Q: Can this be extended to $k$-hop neighbors? \(\rightarrow\) Powers of $A$
  \[\Rightarrow [A^k]_{ij} \text{ non-zero only if there exists a path of length } k \text{ from } i \text{ to } j\]
  \[\Rightarrow \text{Support of } A^k: \text{pairs that can be reached in } k \text{ hops}\]
Given undirected $G$ with $A$ and $D$, the Laplacian matrix $L \in \mathbb{R}^{N \times N}$ is

$$L = D - A$$

Equivalently, $L$ can be defined element-wise as

$$L_{ij} = \begin{cases} 
\deg(i), & \text{if } i = j \\
-w_{ij}, & \text{if } (i, j) \in E \\
0, & \text{otherwise}
\end{cases}$$

Normalized Laplacian: $\mathcal{L} = D^{-1/2}LD^{-1/2}$ (we will focus on $L$)
Spectral properties of the Laplacian

- Denote by $\lambda_i$ and $v_i$ the eigenvalues and eigenvectors of $L$
- $L$ is positive semi-definite
  \[ x^T L x = \frac{1}{2} \sum_{(i,j) \in E} w_{ij} (x_i - x_j)^2 \geq 0, \text{ for all } x \]
  \[ \Rightarrow \text{ All eigenvalues are nonnegative, i.e. } \lambda_i \geq 0 \text{ for all } i \]
- A constant vector $1$ is an eigenvector of $L$ with eigenvalue $0$
  \[ [L1]_i = \sum_{j \in N(i)} w_{ij} (1 - 1) = 0 \]
  \[ \Rightarrow \text{ Thus, } \lambda_1 = 0 \text{ and } v_1 = (1/\sqrt{N}) \mathbf{1} \]
- In connected graphs, it holds that $\lambda_i > 0$ for $i = 2, \ldots, N$
  \[ \Rightarrow \text{ Multiplicity}\{\lambda = 0\} = \text{ number of connected components} \]
Part I: Fundamentals

Motivation and preliminaries

Part I: Fundamentals
  - Graph signals and the shift operator
  - Graph Fourier Transform (GFT)
  - Graph filters and network processes

Part II: Applications
  - Filter design for network operators
  - Sampling graph signals
  - Blind identification of graph filters
  - Network topology inference

Concluding remarks
Consider graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{W})$. **Graph signals** are mappings $x : \mathcal{V} \rightarrow \mathbb{R}$

- Defined on the vertices of the graph (data tied to nodes)

**Ex:** Opinion profile, buffer congestion levels, neural activity, epidemic

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

  - $x_n$ denotes the signal value at the $n$-th vertex in $\mathcal{V}$
  - Implicit ordering of vertices (same as in $A$ or $L$)

\[
x = \begin{bmatrix}
x_0 \\
x_1 \\
x_2 \\
\vdots \\
x_9
\end{bmatrix} = \begin{bmatrix}
0.6 \\
0.7 \\
0.3 \\
\vdots \\
0.7
\end{bmatrix}
\]

- Data associated with links of $G$ ⇒ Use line graph of $G$
Graph signals – Genetic profiles

- Graphs representing gene-gene interactions
  - Each node denotes a single gene (loosely speaking)
  - Connected if their coded proteins participate in same metabolism

- Genetic profiles for each patient can be considered as a graph signal
  - Signal on each node is 1 if mutated and 0 otherwise

\[
x_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}
\]

Sample patient 1 with subtype 1

\[
x_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}
\]

Sample patient 2 with subtype 1

- To understand a graph signal, the structure of $G$ must be considered
Graph-shift operator

- To understand and analyze $\mathbf{x}$, useful to account for $G$’s structure

- Associated with $G$ is the graph-shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$
  \[ \Rightarrow S_{ij} = 0 \text{ for } i \neq j \text{ and } (i, j) \notin \mathcal{E} \] (captures local structure in $G$)

- $\mathbf{S}$ can take nonzero values in the edges of $G$ or in its diagonal

- Ex: Adjacency $\mathbf{A}$, degree $\mathbf{D}$, and Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ matrices
Relevance of the graph-shift operator

**Q:** Why is $S$ called shift? **A:** Resemblance to time shifts

$$S = A_{dc}$$

What is $Sx$?

$$\begin{pmatrix} 0 \\ 0 \\ x_2 \\ x_3 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ x_2 \\ x_3 \\ 0 \\ 0 \end{pmatrix}$$

**$S$ will be building block for GSP algorithms** (More soon)

⇒ Same is true in the time domain (filters and delay)
Local structure of graph-shift operator

**S** represents a linear transformation that can be computed locally at the nodes of the graph. More rigorously, if \( y \) is defined as \( y = Sx \), then node \( i \) can compute \( y_i \) if it has access to \( x_j \) at \( j \in \mathcal{N}(i) \).

- **Straightforward because** \([S]_{ij} \neq 0\) only if \( i = j \) or \((j, i) \in \mathcal{E}\)

- **What if** \( y = S^2x \)?
  - \( \Rightarrow \) Like powers of \( A \): neighborhoods
  - \( \Rightarrow y_i \) found using values within 2-hops

\[
\begin{pmatrix}
  y_1 \\
  y_2 \\
  y_3 \\
  y_4 \\
  y_5 \\
  y_6
\end{pmatrix}
\begin{pmatrix}
  S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\
  S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\
  0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\
  0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\
  S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\
  0 & 0 & 0 & S_{64} & 0 & S_{66}
\end{pmatrix}
\begin{pmatrix}
  x_1 \\
  x_2 \\
  x_3 \\
  x_4 \\
  x_5 \\
  x_6
\end{pmatrix}
\]

\[[S^2]_{3,5} = S_{3,2}S_{2,5} + S_{3,4}S_{4,5}\]

\[
S^2 = \begin{pmatrix}
  S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\
  S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\
  0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\
  0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\
  S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\
  0 & 0 & 0 & S_{64} & 0 & S_{66}
\end{pmatrix}
\begin{pmatrix}
  S_{11} & S_{12} & 0 & 0 & S_{15} & 0 \\
  S_{21} & S_{22} & S_{23} & 0 & S_{25} & 0 \\
  0 & S_{32} & S_{33} & S_{34} & 0 & 0 \\
  0 & 0 & S_{43} & S_{44} & S_{45} & S_{46} \\
  S_{51} & S_{52} & 0 & S_{54} & S_{55} & 0 \\
  0 & 0 & 0 & S_{64} & 0 & S_{66}
\end{pmatrix}
\]

Marques, Mateos, Ribeiro, Segarra
Graph SP: Fundamentals and Applications
Motivation and preliminaries

Part I: Fundamentals
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Filter design for network operators
- Sampling graph signals
- Blind identification of graph filters
- Network topology inference

Concluding remarks
Let $\mathbf{x}$ be a temporal signal, its DFT is $\tilde{\mathbf{x}} = \mathbf{F}^H \mathbf{x}$, with $F_{kn} = \frac{1}{\sqrt{N}} e^{j \frac{2\pi}{N} kn}$

-q Equivalent description, provides insights
-q Oftentimes, more parsimonious (bandlimited)
-q Facilitates the design of SP algorithms: e.g., filters

Many other transformations (orthogonal dictionaries) exist

Q: What transformation is suitable for graph signals?
Graph Fourier Transform (GFT)

- Useful transformation? ⇒ $S$ involved in generation/description of $x$  
  ⇒ 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
  $$x = V\tilde{x}$$

  ⇒ Eigenvectors $V = [v_1, ..., v_N]$ are the frequency basis (atoms)

- Additional structure
  ⇒ If $S$ is normal, then $V^{-1} = V^H$ and $\tilde{x}_k = v_k^H x = <v_k, x>$
  ⇒ Parseval holds, $\|x\|^2 = \|\tilde{x}\|^2$

- GFT ⇒ Projection on eigenvector space of shift operator $S$
Is this a reasonable transform?

- Particularized to cyclic graphs $\Rightarrow$ GFT $\equiv$ Fourier transform
- Particularized to covariance matrices $\Rightarrow$ GFT $\equiv$ PCA transform
- But really, this is an empirical question. GFT of disaggregated GDP

- GFT transform characterized by a few coefficients
  - Notion of bandlimitedness: $x = \sum_{k=1}^{K} \tilde{x}_k v_k$
  - Sampling, compression, filtering, pattern recognition
Eigenvalues as frequencies

- Columns of $\mathbf{V}$ are the frequency atoms: $\mathbf{x} = \sum_k \tilde{x}_k \mathbf{v}_k$

- **Q:** What about the eigenvalues $\lambda_k = \Lambda_{kk}$
  - When $S = A_{dc}$, we get $\lambda_k = e^{-j \frac{2\pi}{N} k}$
  - $\lambda_k$ can be viewed as frequencies!!

- In time, well-defined relation between frequency and variation
  - Higher $k$ $\Rightarrow$ higher oscillations
  - Bounds on total-variation: $TV(\mathbf{x}) = \sum_n (x_n - x_{n-1})^2$

- **Q:** Does this carry over for graph signals?
  - No in general, but if $S = \mathbf{L}$ there are interpretations for $\lambda_k$
  - $\{\lambda_k\}_{k=1}^N$ will be very important when analyzing graph filters
Interpretation of the Laplacian

Consider a graph $G$, let $x$ be a signal on $G$, and set $S = L$

$\Rightarrow y = Sx$ is now $y = Lx \Rightarrow y_i = \sum_{j \in N(i)} w_{ij} (x_i - x_j)$

$\Rightarrow j$-th term is large if $x_j$ is very different from neighboring $x_i$

$\Rightarrow y_i$ measures difference of $x_i$ relative to its neighborhood

We can also define the quadratic form $x^T S x$

$x^T L x = \frac{1}{2} \sum_{(i,j) \in E} w_{ij} (x_i - x_j)^2$

$\Rightarrow x^T L x$ quantifies the (aggregated) local variation of signal $x$

$\Rightarrow$ Natural measure of signal smoothness w.r.t. $G$

Q: Interpretation of frequencies $\{\lambda_k\}_{k=1}^N$ when $S = L$?

$\Rightarrow$ If $x = v_k$, we get $x^T L x = \lambda_k \Rightarrow$ local variation of $v_k$

$\Rightarrow$ Frequencies account for local variation, they can be ordered

$\Rightarrow$ Eigenvector associated with eigenvalue 0 is constant
Frequencies of the Laplacian

- Laplacian eigenvalue $\lambda_k$ accounts for the local variation of $\mathbf{v}_k$

  $\Rightarrow$ Let us plot some of the eigenvectors of $\mathbf{L}$ (also graph signals)

- Ex: gene network, $N = 10$, $k = 1$, $k = 2$, $k = 9$

- Ex: smooth natural images, $N = 2^{16}$, $k = 2$, ..., $6$
Application: Cancer subtype classification

- Patients diagnosed with same disease exhibit different behaviors
- Each patient has a genetic profile describing gene mutations
- Would be beneficial to infer phenotypes from genotypes
  ⇒ Targeted treatments, more suitable suggestions, etc.
- Traditional approaches consider different genes to be independent
  ⇒ Not ideal, as different genes may affect same metabolism
- Alternatively, consider genetic network
  ⇒ Genetic profiles become graph signals on genetic network
  ⇒ We will see how this consideration improves subtype classification
Genetic network

- **Undirected and unweighted** gene-to-gene interaction graph
  - 2458 nodes are genes in human DNA related to breast cancer
  - An edge between two genes represents interaction
    ⇒ Coded proteins participate in the same metabolic process

- **Adjacency matrix** of the gene-interaction network
Genetic profiles

- **Genetic profile of 240 women with breast cancer**
  - 44 with **serous** subtype and 196 with **endometrioid** subtype
  - Patient $i$ has an associated profile $x_i \in \{0, 1\}^{2458}$

- **Mutations** are very varied across patients
  - Some **patients** present a lot of mutations
  - Some **genes** are consistently mutated across patients

- **Q:** Can we use **genetic profiles** to **classify** patients across **subtypes**?
Improving $k$-nearest neighbor classification

- Distance between genetic profiles $\Rightarrow d(i, j) = \| x_i - x_j \|_2$
  $\Rightarrow N$-fold cross-validation error from $k$-NN classification

  \[
  k = 3 \Rightarrow 13.3\%, \quad k = 5 \Rightarrow 12.9\%, \quad k = 7 \Rightarrow 14.6\%
  \]

- **Q:** Can we do any better using graph signal processing?

- Each genetic profile $x_i$ is a graph signal on the genetic network
  $\Rightarrow$ Look at the frequency components $\tilde{x}_i$ using the GFT
  $\Rightarrow$ Use as shift operator $S$ the Laplacian of the genetic network

Example of signal $x_i$  

Frequency representation $\tilde{x}_i$
Define the **distinguishing power** of frequency $\mathbf{v}_k$ as

$$DP(\mathbf{v}_k) = \frac{\sum_{i:y_i=1} \tilde{x}_i(k)}{\sum_i \mathbf{1}\{y_i = 1\}} - \frac{\sum_{i:y_i=2} \tilde{x}_i(k)}{\sum_i \mathbf{1}\{y_i = 2\}} \bigg/ \sum_i |\tilde{x}_i(k)|,$$

- Normalized difference between the mean GFT coefficient for $\mathbf{v}_k$
  - Among patients with *serous* and *endometrioid* subtypes
- Distinguishing power is not equal across frequencies

The distinguishing power defined is one of many proper heuristics
Increasing accuracy by selecting the best frequencies

- Keep information in frequencies with higher distinguishing power
  ⇒ Filter, i.e., multiply $\tilde{x}_i$ by $\text{diag}(\tilde{h}^p)$ where

$$[\tilde{h}^p]_k = \begin{cases} 1, & \text{if } DP(v_k) \geq p\text{-th percentile of } DP \\ 0, & \text{otherwise} \end{cases}$$

- Then perform inverse GFT to get the graph signal $\hat{x}_i$
Motivation and preliminaries

Part I: Fundamentals
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Filter design for network operators
- Sampling graph signals
- Blind identification of graph filters
- Network topology inference

Concluding remarks
A graph filter $H : \mathbb{R}^N \rightarrow \mathbb{R}^N$ is a map between graph signals.

Focus on linear filters

⇒ map represented by an $N \times N$ matrix

**DEF1:** Polynomial in $S$ of degree $L$, with coeff. $h = [h_0, \ldots, h_L]^T$

$$H := h_0 S^0 + h_1 S^1 + \ldots + h_L S^L = \sum_{l=0}^{L} h_l S^l \quad \text{[Sandryhaila13]}$$

**DEF2:** Orthogonal operator in the frequency domain

$$H := V \text{diag}(\tilde{h}) V^{-1}, \quad \tilde{h}_k = g(\lambda_k)$$

With $[\Psi]_{k,l} := \lambda_k^{-1}$, we have $\tilde{h} = \Psi h$ ⇒ Defs can be rendered equivalent

⇒ More on this later, now focus on DEF1
DEF1 says $H = \sum_{l=0}^{L} h_l S^l$

Suppose $H$ acts on a graph signal $x$ to generate $y = Hx$

⇒ If we define $x^{(l)} := S^l x = Sx^{(l-1)}$

$$y = \sum_{l=0}^{L} h_l x^{(l)}$$

$y$ is a linear combination of successive shifted versions of $x$

After introducing $S$, we stressed that $y = Sx$ can be computed locally

⇒ $x^{(l)}$ can be found locally if $x^{(l-1)}$ is known

⇒ The output of the filter can be found in $L$ local steps

A graph filter represents a linear transformation that

⇒ Accounts for local structure of the graph

⇒ Can be implemented distributedly in $L$ steps

⇒ Only requires info in $L$-neighborhood [Shuman13, Sandyhaila14]
An example of a graph filter

\[ \mathbf{x} = [-1, 2, 0, 0, 0, 0]^T, \quad \mathbf{h} = [1, 1, 0.5]^T, \quad \mathbf{y} = (\sum_{l=0}^{L} h_l \mathbf{S}) \mathbf{x} = \sum_{l=0}^{L} h_l \mathbf{x}^{(l)} \]

\[ \mathbf{S} = \mathbf{A} = \begin{pmatrix}
0 & 1 & 0 & 0 & 1 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 \\
1 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0
\end{pmatrix} \]

\[ \mathbf{y} = \sum_{l=0}^{L} h_l \mathbf{S}^l \mathbf{x} = \sum_{l=0}^{L} h_l \mathbf{x}^{(l)} \]

\[ \mathbf{y} = h_0 \mathbf{x}^{(0)} + h_1 \mathbf{x}^{(1)} + h_2 \mathbf{x}^{(2)} \]

Given \( \mathbf{x} = [-1, 2, 0, 0, 0, 0]^T \) and \( \mathbf{h} = [1, 1, 0.5]^T \) \( \Rightarrow \) Find \( \{ \mathbf{x}^{(0)}, \mathbf{x}^{(1)}, \mathbf{x}^{(2)} \} \) \( \Rightarrow \) Find \( \mathbf{y} \)

\[ \begin{pmatrix}
0 & 0 & 0 \\
0 & 2 & -1 \\
2 & -1 & 3 \\
0 & 0 & 3 \\
-1 & 2 & 0
\end{pmatrix} \]

\[ \mathbf{x}^{(0)} = \mathbf{x} = \begin{pmatrix}
-1 \\
2 \\
0 \\
0 \\
0 \\
1
\end{pmatrix} \]

\[ \mathbf{x}^{(1)} = \mathbf{S} \mathbf{x}^{(0)} = \begin{pmatrix}
2 \\
-1 \\
2 \\
0 \\
1 \\
0
\end{pmatrix} \]

\[ \mathbf{x}^{(2)} = \mathbf{S} \mathbf{x}^{(1)} = \begin{pmatrix}
0 \\
3 \\
-1 \\
3 \\
1 \\
0
\end{pmatrix} \]

\[ \mathbf{y} = 1 \mathbf{x}^{(0)} + 1 \mathbf{x}^{(1)} + 0.5 \mathbf{x}^{(2)} = \begin{pmatrix}
1.0 \\
2.5 \\
1.5 \\
1.5 \\
1.5 \\
0.0
\end{pmatrix} \]
Recalling that $S = \mathbf{V}\Lambda\mathbf{V}^{-1}$, we may write

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

The application $H\mathbf{x}$ of filter $H$ to $\mathbf{x}$ can be split into three parts:

$\Rightarrow$ $\mathbf{V}^{-1}$ takes signal $\mathbf{x}$ to the graph frequency domain $\tilde{\mathbf{x}}$

$\Rightarrow \tilde{H} := \sum_{l=0}^{L} h_l\Lambda^l$ modifies the frequency coefficients to obtain $\tilde{\mathbf{y}}$

$\Rightarrow \mathbf{V}$ brings the signal $\tilde{\mathbf{y}}$ back to the graph domain $\mathbf{y}$

Since $\tilde{H}$ is diagonal, define $\tilde{H} =: \text{diag}(\tilde{h})$

$\Rightarrow \tilde{h}$ is the frequency response of the filter $H$

$\Rightarrow$ Output at frequency $k$ depends only on input at frequency $k$

$$\tilde{y}_k = \tilde{h}_k\tilde{x}_k$$
Frequency response and filter coefficients

- Relation between $\tilde{h}$ and $h$ in a more friendly manner?
  - Since $\tilde{h} = \text{diag}(\sum_{l=0}^{L} h_{l} \lambda_{l}^{l})$, we have that $\tilde{h}_{k} = \sum_{l=0}^{L} h_{l} \lambda_{k}^{l}$
  - Define the Vandermonde matrix $\Psi$ as
    
    $$
    \Psi := \begin{pmatrix}
    1 & \lambda_{1} & \ldots & \lambda_{1}^{L} \\
    \vdots & \vdots & \ddots & \vdots \\
    1 & \lambda_{N} & \ldots & \lambda_{N}^{L}
    \end{pmatrix}
    $$

Frequency response of a graph filter

If $h$ are the coefficients of a graph filter, its frequency response is $\tilde{h} = \Psi h$

- Given a desired $\tilde{h}$, we can find the coefficients $h$ as
  
  $$
  h = \Psi^{-1} \tilde{h}
  $$

  - Since $\Psi$ is Vandermonde, invertible as long as $\lambda_{k} \neq \lambda_{k}'$, for $k \neq k'$

Marques, Mateos, Ribeiro, Segarra
Graph SP: Fundamentals and Applications
More on the frequency response

- Since $h = \Psi^{-1} \tilde{h}$ \(\Rightarrow\) If all $\{\lambda_k\}_{k=1}^N$ distinct, then
  \(\Rightarrow\) Any $\tilde{h}$ can be implemented with at most $L+1 = N$ coefficients

- Since $h = \Psi \tilde{h}$ \(\Rightarrow\) If $\lambda_k = \lambda_k'$, then
  \(\Rightarrow\) The corresponding frequency response will be the same $\tilde{h}_k = \tilde{h}_k'$

- For the particular case when $S = A_{dc}$, we have that $\lambda_k = e^{-j \frac{2\pi}{N} (k-1)}$

\[
\Psi = \begin{pmatrix}
1 & 1 & \cdots & 1 \\
1 & e^{-j \frac{2\pi (1)(1)}{N}} & \cdots & e^{-j \frac{2\pi (1)(N-1)}{N}} \\
\vdots & \vdots & \ddots & \vdots \\
1 & e^{-j \frac{2\pi (N-1)(1)}{N}} & \cdots & e^{-j \frac{2\pi (N-1)(N-1)}{N}}
\end{pmatrix} = F^H
\]

\(\Rightarrow\) The frequency response is the DFT of the impulse response

\[
\tilde{h} = F^H h
\]
Suppose that we have a signal $x$ and filter coefficients $h$.

For time signals, it holds that the output $y$ is

$$\tilde{y} = \text{diag}(F^H h) F^H x$$

For graph signals, the output $y$ in the frequency domain is

$$\tilde{y} = \text{diag}(\Psi h) V^{-1} x$$

The GFT for filters is different from the GFT for signals:

- Symmetry is lost, but both depend on spectrum of $S$.
- Many of the properties are not true for graphs.
- Several options to generalize operations.
Suppose that our goal is to find $h$ given $x$ and $y$

⇒ Using the previous expressions

$$h = \Psi^{-1}\text{diag}^{-1}(V^{-1}x)V^{-1}y$$

In time, if we set $x = [1, 0, ..., 0]^T = e_1$ (i.e., $\tilde{x} = 1$), we have

⇒ $h = F\text{diag}^{-1}(1)F^Hy = y$ → $h$ is the impulse response

In the graph domain

If we set $x = e_i$, then $h = \Psi^{-1}\text{diag}^{-1}(\tilde{e}_i)V^{-1}y$, where

⇒ $\tilde{e}_i := V^{-1}e_i \equiv$ how strongly node $i$ expresses each of the freqs.

⇒ Problem if $\tilde{e}_i$ has zero entries

Alternatively we can get $\tilde{x} = 1$ by setting $x = V1$ and then

⇒ $h = \Psi^{-1}\text{diag}^{-1}(\tilde{x})V^{-1}y = \Psi^{-1}V^{-1}y$
Implementing graph filters: frequency or space

- Frequency or space?

\[ y = V \text{diag}(\tilde{h}) V^{-1} x \text{ vs. } y = \sum_{l=0}^{L} h_l S^l x \]

- In space: leverage the fact that \( Sx \) can be computed locally
  \( \Rightarrow \) Signal \( x \) is percolated \( L \) times to find \( \{x^{(l)}\}_{l=0}^{L} \)
  \( \Rightarrow \) Every node finds its own \( y_i \) by computing \( \sum_{l=0}^{L} h_l [x^{(l)}]_i \)

- Frequency implementation useful for processing if, e.g.,
  \( \Rightarrow \) Filter bandlimited and eigenvectors easy to find
  \( \Rightarrow \) Low complexity [Anis16, Tremblay16]

- Space definition useful for modeling
  \( \Rightarrow \) Diffusion, percolation, opinion formation, ... (more on this soon)

- More on filter design
  \( \Rightarrow \) Chebyshev polyn. [Shuman12]; AR-MA [Isufi-Leus15]; Node-var. [Segarra15]; Time-var. [Isufi-Leus16]; Median filters [Segarra16]
Consider a linear dynamics of the form

\[ x_t - x_{t-1} = \alpha J x_{t-1} \Rightarrow x_t = (I - \alpha J)x_{t-1} \]

If \( x \) is network process \( \Rightarrow [x_t]_i \) depends only on \( [x_{t-1}]_j, j \in \mathcal{N}(i) \)

\[ [S]_{ij} = [J]_{ij} \Rightarrow x_t = (I - \alpha S)x_{t-1} \Rightarrow x_t = (I - \alpha S)^t x_0 \]

\[ x_t = H x_0, \text{ with } H \text{ a polynomial of } S \Rightarrow \text{linear graph filter} \]

If the system has memory \( \Rightarrow \) output weighted sum of previous exchanges (opinion dynamics) \( \Rightarrow \) still a polynomial of \( S \)

\[ y = \sum_{t=0}^{T} \beta^t x_t \Rightarrow y = \sum_{t=0}^{T} (\beta I - \beta \alpha S)^t x_0 \]

Everything holds true if \( \alpha_t \) or \( \beta_t \) are time varying.
Diffusion dynamics and AR (IIR) filters

- Before finite-time dynamics (FIR filters)
- Consider now a diffusion dynamics $x_t = \alpha S x_{t-1} + w$

$$x_t = \alpha^t S^t x_0 + \sum_{t' = 0}^{t} \alpha^{t'} S^{t'} w$$

$\Rightarrow$ When $t \to \infty$: $x_\infty = (I - \alpha S)^{-1} w \Rightarrow$ AR graph filter

- Higher orders [Isufi-Leus16]
  $\Rightarrow$ $M$ successive diffusion dynamics $\Rightarrow$ AR of order $M$
  $\Rightarrow$ Process is the sum of $M$ parallel diffusions $\Rightarrow$ ARMA order $M$

$$x_\infty = \prod_{m=1}^{M}(I - \alpha_m S)^{-1} w \quad x_\infty = \sum_{m=1}^{M}(I - \alpha_m S)^{-1} w$$
General linear network processes

- Combinations of all the previous are possible

\[ x_t = H^a_t(S)x_{t-1} + H^b_t(S)w \Rightarrow x_t = H^A_t(S)x_0 + H^B_t(S)w \]

\( \Rightarrow y = x_t \), sequential/parallel application, linear combination

- Expands range of processes that can be modeled via GSP
- Coefficients can change according to some control inputs

- A number of linear processes can be modeled using graph filters
  - Theoretical GSP results can be applied to distributed networking
  - Deconvolution, filtering, system id, ...
  - Beyond linearity possible too (more at the end of the talk)

- Links with control theory (of networks and complex systems)
  - Controllability, observability
Why do some people learn faster than others?  
⇒ Can we answer this by looking at their brain activity?

Brain activity during learning of a motor skill in 112 cortical regions  
⇒ fMRI while learning a piano pattern for 20 individuals

Pattern is repeated, reducing the time needed for execution  
⇒ Learning rate = rate of decrease in execution time

Define a functional brain graph  
⇒ Based on correlated activity

fMRI outputs a series of graph signals  
⇒ $x(t) \in \mathbb{R}^{112}$ describing brain states

Does brain state variability correlate with learning?
We propose three different measures capturing different time scales
⇒ Changes in micro, meso, and macro scales

- **Micro**: instantaneous changes higher than a threshold \( \alpha \)

\[
m_1(x) = \sum_{t=1}^{T} 1 \left\{ \frac{\|x(t) - x(t-1)\|_2}{\|x(t)\|_2} > \alpha \right\}
\]

- **Meso**: Cluster brain states and count the changes in clusters

\[
m_2(x) = \sum_{t=1}^{T} 1 \{ c(t) \neq c(t-1) \}
\]
⇒ where \( c(t) \) is the cluster to which \( x(t) \) belongs.

- **Macro**: Sample entropy. Measure of complexity of time series

\[
m_3(x) = -\log \left( \frac{\sum_{t} \sum_{s \neq t} 1 \{ \| \bar{x}_3(t) - \bar{x}_3(s) \|_\infty > \alpha \}}{\sum_{t} \sum_{s \neq t} 1 \{ \| \bar{x}_2(t) - \bar{x}_2(s) \|_\infty > \alpha \}} \right)
\]
⇒ Where \( \bar{x}_r(t) = [x(t), x(t+1), \ldots, x(t+r-1)] \)
Diffusion as low-pass filtering

- We diffuse each time signal $x(t)$ across the brain graph

$$x_{\text{diff}}(t) = (I + \beta L)^{-1}x(t)$$

$\Rightarrow$ where Laplacian $L = \mathbf{V} \Lambda \mathbf{V}^{-1}$ and $\beta$ represents the diffusion rate

- Analyzing diffusion in the frequency domain

$$\tilde{x}_{\text{diff}}(t) = (I + \beta \Lambda)^{-1}V^{-1}x(t) = \text{diag}(\tilde{h})\tilde{x}(t)$$

$\Rightarrow$ where $\tilde{h}_i = 1/(1 + \beta \lambda_i)$

- Diffusion acts as low-pass filtering
- High freq. components are attenuated
- $\beta$ controls the level of attenuation
Computing correlation for three signals

- **Variability** measures consider the order of brain signal activity
- As a **control**, we include in our analysis a **null signal** time series \( x_{\text{null}} \)

\[
x_{\text{null}}(t) = x_{\text{diff}}(\pi_t)
\]

⇒ where \( \pi_t \) is a random permutation of the time indices

- Correlation between **variability** \((m_1, m_2, \text{ and } m_3)\) and **learning**?
- We consider **three** time series of brain activity
  ⇒ The original fMRI data \( x \)
  ⇒ The filtered data \( x_{\text{diff}} \)
  ⇒ The **null** signal \( x_{\text{null}} \)
Low-pass filtering reveals correlation

- Correlation coeff. between learning rate and brain state variability

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>Filtered</th>
<th>Null</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m_1$</td>
<td>0.211</td>
<td>0.568</td>
<td>0.182</td>
</tr>
<tr>
<td>$m_2$</td>
<td>0.226</td>
<td>0.611</td>
<td>0.174</td>
</tr>
<tr>
<td>$m_3$</td>
<td>0.114</td>
<td>0.382</td>
<td>0.113</td>
</tr>
</tbody>
</table>

- Correlation is clear when the signal is filtered
  - Result for original signal similar to null signal
- Scatter plots for original, filtered, and null signals ($m_2$ variability)
Part II: Applications

Motivation and preliminaries

Part I: Fundamentals
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Filter design for network operators
- Sampling graph signals
- Blind identification of graph filters
- Network topology inference

Concluding remarks
Application domains

- Design graph filters to approximate desired network operators
- Sampling bandlimited graph signals
- Blind graph filter identification
  ⇒ Infer diffusion coefficients from observed output
- Network topology inference
  ⇒ Infer shift from collection of network diffused signals
- Many more (not covered, glad to discuss or redirect):
  ⇒ Statistical GSP, stationarity and spectral estimation
  ⇒ Filter banks
  ⇒ Windowing, convolution, duality...
  ⇒ Nonlinear GSP
Distributed network operators

- Design **graph filters** to implement a given **linear transformation**
  - Implementation is **distributed** by construction
  - Conditions for **perfect** and **approximate** implementation
  - [Shuman11], [Sandryhaila14], [Safavi15], [Chen15]

- Given a **linear transformation** $\mathbf{B}$, find the **filter coefficients** $\mathbf{h}$ s. t.
  \[
  \mathbf{B} = \sum_{l=0}^{L-1} h_l \mathbf{S}^l
  \]
  \[
  \mathbf{B} = \begin{bmatrix}
  \text{[Color Diagram]}
  \end{bmatrix}
  \]

- **Graph-shift operator** $\mathbf{S}$ is given
  - Well-suited for cases where $\mathbf{S}$ is a **network process**
  - E.g., diffusion in a social network
  - Agents exchange information and weigh info observed
  - Choosing $\mathbf{h}$ $\Rightarrow$ **fixing the weights**
Perfect implementation of linear graph operators [Segarra15]

The linear transformation $B$ can be implemented using a graph filter $H$ if the following conditions hold true:

i) Matrices $B$ and $S$ are simultaneously diagonalizable.

ii) If $\lambda_{k1} = \lambda_{k2}$, then $\gamma_{k1} = \gamma_{k2}$; and $L \geq \# \{\lambda_k\}_{k=1}^N$ distinct.

- i) $\implies$ frequency basis of $B$ and $S$ the same $\implies$ necessary
- ii) $\implies$ two equal freqs. in $S$ must be equal in $B$ $\implies$ necessary

- **Restrictive conditions but not impossible** to satisfy $\implies$ Consensus $B_{\text{con}} = \mathbf{1}\mathbf{1}^T$
  - favors i) and ii) because it is rank-one

- If satisfied: $h^* = \Psi^{-1}\gamma$, where $\gamma = [\gamma_1, ..., \gamma_N]^T$ are eigvals. of $B$
Approximate design

- When perfect reconstruction is infeasible $\Rightarrow$ minimize error metric
  $\Rightarrow$ Design $Hx$ to resemble $Bx$ (or $H$ to resemble $B$)
  $\Rightarrow$ Minimizing $\|(H - B)R_x(H - B)^T\|_z$ (with $R_x = I$ if unknown)

- MSE coefficients: $h^* = \Theta_{R_x}^\dagger b_{R_x} = (\Theta_{R_x}^T \Theta_{R_x})^{-1} \Theta_{R_x}^T b_{R_x}$
  $\Rightarrow$ with $\Theta_{R_x} := [\text{vec}(IR_x^{1/2}), \ldots, \text{vec}(S^{L-1}R_x^{1/2})]$, $b_{R_x} := \text{vec}(BR_x^{1/2})$

- Worst-case error coefficients:
  $$\{h^*, s^*\} = \arg\min_{\{h, s\}} s$$
  s. to $$s \begin{bmatrix} sI & V\text{diag}(\psi h)V^{-1} - B \\ (V\text{diag}(\psi h)V^{-1} - B)^T & sR_x^{-1} \end{bmatrix} \succeq 0.$$  

- Additional assumptions can be incorporated
Consensus and rank-1 transformations

Consensus

- Local implementation of the consensus operator $B_{\text{con}} = \mathbf{1} \mathbf{1}^T / N$

Proposition [Segarra16]
If $G$ is connected and the desired operator $B_{\text{rk1}}$ is rank one, then there exists an $S$ such that $B_{\text{rk1}}$ can be written as a graph filter $\sum_{l=0}^{N-1} h_l S^l$.

- Constructive proof, for consensus $S = L$
- Consensus is achieved in finite time [Sandryhaila-Kar-Moura14]
- Key: $B$ low-rank (repeated eigenvalues) $\Rightarrow$ well-suited for approx.

- We compare the performance of: 1) Asymptotic fastest distributed linear averaging (FDLA), 2) Graph filter approx.
Finite-time consensus

- Define the graph-shift operator $S = W$
  $\Rightarrow \text{Where } \lim_{k \to \infty} W^k = B_{\text{con}} \text{ with fastest convergence}$
- Plot average errors across the 100 graphs with 10 nodes
- Compare worst-case and mean error design (50 nodes)

![Graphs showing error comparison]

- Smaller error than FDLA for intermediate $K$
  $\Rightarrow \text{When } K = N - 1 = 9, \text{ perfect recovery}$
- The price to pay is that $\{\lambda_k\}_{k=1}^N$ need to be known
- Consistent performance of mean error and worst case designs
Node-variant graph filters: definition

- A generalization of graph filters [Segarra16]:

\[
H_{nv} := \sum_{l=0}^{L-1} \text{diag}(h^{(l)}) S^l
\]

⇒ When \( h^{(l)} = h_1 \) ⇒ regular (node-invariant) filter

In general, when \( H_{nv} \) is applied to a signal \( x \)

⇒ Each node applies different weights to the shifted signals \( S^l x \)

⇒ More flexible and still distributed, not shift-invariant
Collect the coefficients of node $i$ in $h_i$, such that $[h_i]_l = [h^{(l)}]_i$.

Focus on the filter output at node $i$, $e_i^T H_{nv} x$

$$\eta_i^T = e_i^T H_{nv} = \sum_{l=0}^{L-1} [h_i]_l e_i^T \Lambda^{l'} V^{-1}$$

Defining $u_i := V^T e_i$

$$\eta_i^T = u_i^T \left( \sum_{l=0}^{L-1} [h_i]_l \Lambda^{l'} \right) V^{-1} = u_i^T \text{diag}(\Psi h_i) V^{-1}$$

The output of the filter at node $i$, $\eta_i^T x$ is the inner product of

$\Rightarrow V^{-1} x \Rightarrow$ the frequency representation of the input, and
$\Rightarrow u_i \Rightarrow$ how strongly the frequencies are expressed by node $i$
$\Rightarrow$ Modulated by $\Psi h_i \Rightarrow$ Frequency response associated to $i$
Perfect reconstruction with node-variant filters

- **Node-variant filters** can implement a large class of transformations
  \[ B = \sum_{l=0}^{L-1} \text{diag}(h^{(l)})S^l \]

- **TH:** Always possible if \( V \) non-zero and \( \{\lambda_k\} \) distinct

- **Application in distributed processing:** analog network coding
  \( B \) is a binary matrix (input-output pairs)

- **Example:** \( G \) undirected, with \( N = 10 \), \( S = A \), sources 3 and 6
  \[ \text{Node 3 tx to 1, 4, 6, 7, and 10; node 6 to the remaining ones} \]
  \[ \text{Node invariant unable to implement } B \]
Motivation and preliminaries

Part I: Fundamentals
  Graph signals and the shift operator
  Graph Fourier Transform (GFT)
  Graph filters and network processes

Part II: Applications
  Filter design for network operators
  Sampling graph signals
  Blind identification of graph filters
  Network topology inference

Concluding remarks
Motivation and preliminaries

- **Sampling** and interpolation are cornerstone problems in classical SP
  - How recover a signal using only a few observations?
  - Need to limit the degrees of freedom: subspace, smoothness

- **Graph signals**: sampling thoroughly investigated
  - Most works assume only a few values are observed
  - \([\text{Anis14, Chen15, Tsitsvero15, Puy15, Wang15}]\)

- **Alternative approach** [Marques16, Segarra16]
  - GSP is well-suited for distributed networking
  - Incorporate local graph structure into the observation model
  - Recover signal using distributed local graph operators
Sampling bandlimited graph signals: Overview

- **Sampling** is likely to be most important inverse problem
  - How to find \( \mathbf{x} \in \mathbb{R}^N \) using \( P < N \) observations?

- Our focus on **bandlimited** signals, but other models possible
  - \( \tilde{\mathbf{x}} = \mathbf{V}^{-1} \mathbf{x} \) sparse
  - \( \mathbf{x} = \sum_{k \in \mathcal{K}} \tilde{x}_k \mathbf{v}_k \), with \( |\mathcal{K}| = K < N \)
  - \( \mathbf{S} \) involved in generation of \( \mathbf{x} \)
  - Agnostic to the particular form of \( \mathbf{S} \)

- Two sampling schemes were introduced in the literature
  - **Selection** [Anis14, Chen15, Tsitsvero15, Puy15, Wang15]
  - **Aggregation** [Segarra15], [Marques15]
  - **Hybrid** scheme combining both \( \Rightarrow \) **Space-shift** sampling

- More involved, theoretical benefits, practical benefits in distr. setups
Revisiting sampling in time

- There are two ways of interpreting sampling of time signals.
- We can either freeze the signal and sample values at different times.
- We can fix a point (present) and sample the evolution of the signal.
- Both strategies coincide for time signals but not for general graphs.
  ⇒ Give rise to selection and aggregation sampling.
Selection sampling: Definition

- Intuitive generalization to graph signals
  \[ C \in \{0, 1\}^{P \times N} \text{ (matrix } P \text{ rows of } I_N) \]
  \[ \Rightarrow \text{ Sampled signal is } \tilde{x} = CX \]

- Goal: recover \( x \) based on \( \tilde{x} \)
  \[ \Rightarrow \text{ Assume that the support of } K \text{ is known (w.l.o.g. } K = \{k\}_{k=1}^{K} \) \]
  \[ \Rightarrow \text{ Since } \tilde{x}_k = 0 \text{ for } k > K, \text{ define } \tilde{x}_K := [\tilde{x}_1, \ldots, \tilde{x}_K]^T = E_K^T \tilde{x} \]

- Approach: use \( \tilde{x} \) to find \( \tilde{x}_K \), and then recover \( x \) as
  \[ x = V(E_K \tilde{x}_K) = (VE_K) \tilde{x}_K = V_K \tilde{x}_K \]
Selection sampling: Recovery

- Number of samples $P \geq K$

\[
\tilde{x} = Cx = CV_K \tilde{x}_K
\]

$\Rightarrow (CV_K)$ submatrix of $V$

Recovery of selection sampling

If $\text{rank}(CV_K) \geq K$, $x$ can be recovered from the $P$ values in $\tilde{x}$ as

\[
x = V_K \tilde{x}_K = V_K (CV_K)^{\dagger} \tilde{x}
\]

- With $P = K$, hard to check invertibility (by inspection)
  $\Rightarrow$ Columns of $V_K(CV_K)^{-1}$ are the interpolators

- In time ($S = A_{dc}$), if the samples in $C$ are equally spaced
  $\Rightarrow (CV_K)$ is Vandermonde (DFT) and $V_K(CV_K)^{-1}$ are sincs
Aggregation sampling: Definition

- Idea: incorporating $S$ to the sampling procedure
  ⇒ Reduces to classical sampling for time signals

- Consider shifted (aggregated) signals $y^{(l)} = S'y$
  ⇒ $y^{(l)} = Sy^{(l-1)}$ ⇒ found sequentially with only local exchanges

- Form $y_i = [y_i^{(0)}, y_i^{(1)}, ..., y_i^{(N-1)}]^T$ (obtained locally by node $i$)

- The sampled signal is $\bar{y}_i = C y_i$

- Goal: recover $x$ based on $\bar{y}_i$
Aggregation sampling: Recovery

- Goal: recover $x$ based on $\bar{y}_i$ $\Rightarrow$ Same approach than before
  $\Rightarrow$ Use $\bar{y}_i$ to find $\tilde{x}_K$, and then recover $x$ as $x = V_K \tilde{x}_K$

- Define $\bar{u}_i := V_K^T e_i$ and recall $\Psi_{kl} = \lambda_k^{-1}$

**Recovery of aggregation sampling**

Signal $x$ can be recovered from the first $K$ samples in $\bar{y}_i$ as

$$x = V_K \tilde{x}_K = V_K \text{diag}^{-1}(\bar{u}_i)(C \Psi^T E_K)^{-1} \bar{y}_i$$

provided that $[\bar{u}_i]_k \neq 0$ and all $\{\lambda_k\}_{k=1}^K$ are distinct.

- If $C = E_K^T$, node $i$ can recover $x$ with info from $K - 1$ hops!
  $\Rightarrow$ Node $i$ has to be able to capture frequencies in $\mathcal{K}$
  $\Rightarrow$ The frequencies have to be distinguishable

- **Bandlimited signals**: Signals that can be well estimated locally
Aggregation and selection sampling: Example

- In time ($S = A_{dc}$), selection and aggregation are equivalent
  \[ \Rightarrow \text{Differences for a more general graph?} \]

- Erdős-Rényi
  
  $p = 0.2$, $S = A$, $K = 3$,
  
  non-smooth

- First 3 observations at node 4: $y_4 = [0.55, 1.27, 2.94]^T$
  
  $\Rightarrow [y_4]_1 = x_4 = -0.55$, $[y_4]_2 = x_2 + x_3 + x_5 + x_6 + x_7 = 1.27$
  
  $\Rightarrow$ For this example, any node guarantees recovery
  
  $\Rightarrow$ Selection sampling fails if, e.g., $\{1, 3, 4\}$
Sampling: Discussion and extensions

- Discussion on aggregation sampling
  - Observation matrix: diagonal times Vandermonde
  - Very appropriate in distributed scenarios
  - Different nodes will lead to different performance (soon)
  - Types of signals that are actually bandlimited (role of S)

- Three extensions:
  - Sampling in the presence of noise
  - Unknown frequency support
  - Space-shift sampling (hybrid)
Presence of noise

- Linear observation model: \( \tilde{z}_i = C\psi_i\tilde{x}_K + Cw_i \) and \( x = V_K\tilde{x}_K \)

- BLUE interpolation (\( \psi_i \) either selection or aggregation)

\[
\hat{x}_K^{(i)} = [\psi_i^H C^H (\tilde{R}_w(i))^{-1} C\psi_i]^{-1} \psi_i^H C^H (\tilde{R}_w(i))^{-1} \tilde{z}_i
\]

⇒ If \( P = K \), then \( \hat{x}^{(i)} = V_K (C\psi_i)^{-1} \tilde{z}_i \)

- Error covariances \( (R_e^{(i)}, \tilde{R}_e^{(i)}) \) in closed form ⇒ Noise covariances?
  ⇒ Colored, different models: white noise in \( z_i \), in \( x \), or in \( \tilde{x}_K \)

- Metric to optimize?

  ⇒ \( \text{trace}(R_e^{(i)}), \lambda_{\text{max}}(R_e^{(i)}), \log \det(\tilde{R}_e^{(i)}), \left[ \text{trace} \left( \tilde{R}_e^{(i)^{-1}} \right) \right]^{-1} \)

- Select \( i \) and \( C \) to min. error ⇒ Depends on metric and noise [Marques16]
Unknown frequency support

- Falls into the class of sparse reconstruction: observation matrix?
  - Selec. ⇒ submatrix of unitary $V_K$
  - Aggr. ⇒ Vander. $\times$ diag
  \[
  [u_i]^T_k \neq 0 \text{ and } \lambda_k \neq \lambda_{k'} \Rightarrow \text{full-spark}
  \]

- Joint recovery and support identification (noiseless)
  \[
  \tilde{x}^* := \arg \min_{\tilde{x}} \|\tilde{x}\|_0 \quad \text{s.t.} \quad Cy_i = C\Psi_i \tilde{x}.
  \]

- If full spark $\Rightarrow P = 2K$ samples suffice
  - Different relaxations are possible
  - Conditioning will depend on $\Psi_i$ (e.g., how different $\{\lambda_k\}$ are)

- Noisy case: sampling nodes critical
Recovery with unknown support: Example

- Erdős-Rényi
  \[ p = 0.15, 0.20, 0.25, \]
  \[ K = 3, \text{ non-smooth} \]

- Three different shifts: \( A, (I - A) \) and \( \frac{1}{2}A^2 \)
Space-shift sampling

- **Space-shift** sampling (hybrid) ⇒ Multiple nodes and multiple shifts

<table>
<thead>
<tr>
<th>Selection: 4 nodes, 1 sample</th>
<th>Space-shift: 2 nodes, 2 samples</th>
<th>Aggregat.: 1 node, 4 samples</th>
</tr>
</thead>
</table>

- Section and aggregation sampling as particular cases
- With \( \tilde{U} := \text{diag}(\tilde{u}_1), ..., \text{diag}(\tilde{u}_N) \)^T, the sampled signal is

  \[
  \tilde{z} = C \left( I \otimes (\Psi^T E_K) \right) \tilde{U} \tilde{x}_K + Cw
  \]

- As before, BLUE and error covariance in close-form
- Optimizing sample selection more challenging
- More structured schemes easier: e.g., message passing

  ⇒ Node \( i \) knows \( y_i^{(l)} \) ⇒ node \( i \) knows \( y_j^{(l')} \) for all \( j \in \mathcal{N}_i \) and \( l' < l \)
Sampling the US economy

- 62 economic sectors in USA + 2 artificial sectors
  - Graph: average flows in 2007-2010, $S = A$
  - Signal $x$: production in 2011
  - $x$ is approximately bandlimited with $K = 4$
Sampling the US economy: Results

- Setup 1: we add different types of noise
  \[ \Rightarrow \text{Error depends on sampling node: better if more connected} \]

- Setup 2: we try different shift-space strategies

<table>
<thead>
<tr>
<th>Sampling strategy</th>
<th>Min. error</th>
<th>Median error</th>
</tr>
</thead>
<tbody>
<tr>
<td>( [x]_i )</td>
<td>.0035</td>
<td>.019</td>
</tr>
<tr>
<td>( [Sx]_i )</td>
<td>.0039</td>
<td>4.2</td>
</tr>
<tr>
<td>( [S^2x]_i )</td>
<td>.0035</td>
<td>.030</td>
</tr>
<tr>
<td>( [S^3x]_i )</td>
<td>.0035</td>
<td>.0055</td>
</tr>
<tr>
<td>( [x]_j )</td>
<td>.0035</td>
<td>.0040</td>
</tr>
<tr>
<td>( [Sx]_j )</td>
<td>.0035</td>
<td>.039</td>
</tr>
</tbody>
</table>
More on sampling graph signals

- **Beyond bandlimitedness**
  - Smooth signals [Chen15]
  - Parsimonious in kernelized domain [Romero-Giannakis16]

- **Strategies to select the sampling nodes**
  - Random (sketching) [Varma15]
  - Optimal reconstruction [Marques16, Chepuri-Leus16]
  - Designed based on posterior task [Gama16]

- **And more...**
  - Low-complexity implementations [Tremblay16, Anis16]
  - Local implementations [Wang14, Segarra15]
  - Unknown spectral decomposition [Anis16]
Motivation and preliminaries

Part I: Fundamentals
   Graph signals and the shift operator
   Graph Fourier Transform (GFT)
   Graph filters and network processes

Part II: Applications
   Filter design for network operators
   Sampling graph signals
   Blind identification of graph filters
   Network topology inference

Concluding remarks
Diffusion processes as graph filter outputs

- **Q:** Upon observing a graph signal \( y \), how was this signal generated?

- **Postulate the following generative model**
  - An originally **sparse** signal \( x = x^{(0)} \)
  - **Diffused via linear graph dynamics** \( S \)  \( \Rightarrow \) \( x^{(l)} = S x^{(l-1)} \)
  - Observed \( y \) is a linear combination of the diffused signals \( x^{(l)} \)

\[
    y = \sum_{l=0}^{L} h_l x^{(l)} = \sum_{l=0}^{L} h_l S^l x = H x
\]

- **Model:** Observed network process as output of a graph filter
  - View few elements in \( \text{supp}(x) =: \{i : x_i \neq 0\} \) as **seeds**
Motivation and problem statement

▶ **Ex:** Global opinion/belief profile formed by spreading a rumor
⇒ What was the rumor? Who started it?
⇒ How do people weigh in peers’ opinions to form their own?

▶ **Problem:** Blind identification of graph filters with sparse inputs
▶ **Q:** Given $S$, can we find $x$ and the combination weights $h$ from $y = Hx$?
⇒ Extends classical blind deconvolution to graphs
Blind graph filter identification

- Leverage frequency response of graph filters \((U := V^{-1})\)

\[ y = Hx \quad \Rightarrow \quad y = V \text{diag}(\Psi h)Ux \]

\(\Rightarrow y\) is a bilinear function of the unknowns \(h\) and \(x\)

- Problem is ill-posed \(\Rightarrow (L + 1) + N\) unknowns and \(N\) observations

\(\Rightarrow \text{As.}: \ x\) is \(S\)-sparse i.e., \(\|x\|_0 := |\text{supp}(x)| \leq S\)

- Blind graph filter identification \(\Rightarrow\) Non-convex feasibility problem

\[
\text{find } \{h, x\}, \quad \text{s. to } \quad y = V \text{diag}(\Psi h)Ux, \quad \|x\|_0 \leq S
\]
“Lifting” the bilinear inverse problem

Key observation: Use the Khatri-Rao product $\odot$ to write $\mathbf{y}$ as

$$\mathbf{y} = \mathbf{V} (\psi^T \odot \mathbf{U}^T)^T \text{vec}(\mathbf{x}_h^T)$$

Reveals $\mathbf{y}$ is a linear combination of the entries of $\mathbf{Z} := \mathbf{x}_h^T$

$\mathbf{Z}$ is of rank-1 and row-sparse $\Rightarrow$ Linear matrix inverse problem

$$\min \text{rank}(\mathbf{Z}), \quad \text{s. to } \mathbf{y} = \mathbf{V} (\psi^T \odot \mathbf{U}^T)^T \text{vec}(\mathbf{Z}), \quad \|\mathbf{Z}\|_{2,0} \leq S$$

$\Rightarrow$ Pseudo-norm $\|\mathbf{Z}\|_{2,0}$ counts the nonzero rows of $\mathbf{Z}$

$\Rightarrow$ Matrix “lifting” for blind deconvolution [Ahmed etal’14]

Rank minimization s. to row-cardinality constraint is NP-hard. Relax!
Algorithmic approach via convex relaxation

- **Key property:** \( \ell_1 \)-norm minimization promotes sparsity \([\text{Tibshirani}'94]\)
  - Nuclear norm \( \|Z\|_* := \sum_i \sigma_i(Z) \) a convex proxy of rank \([\text{Fazel}'01]\)
  - \( \ell_2,1 \) norm \( \|Z\|_{2,1} := \sum_i \|z_i^T\|_2 \) surrogate of \( \|Z\|_{2,0} \) \([\text{Yuan-Lin}'06]\)

- **Convex relaxation**

\[
\min_Z \|Z\|_* + \alpha \|Z\|_{2,1}, \quad \text{s. to } y = V(\Psi^T \odot U^T)^T \text{vec}(Z)
\]

⇒ Scalable algorithm using method of multipliers

- Refine estimates \( \{h, x\} \) via iteratively-reweighted optimization
  ⇒ Weights \( \alpha_i(k) = (\|z_i(k)^T\|_2 + \delta)^{-1} \) per row \( i \), per iteration \( k \)

- **Noisy and partial observations** ⇒ Adjust constraints
  - Noise in \( y \): \( \|y - V(\Psi^T \odot U^T)^T \text{vec}(Z)\| \leq \varepsilon \)
  - Sampling via selection matrix \( C \): \( y_C = CV(\Psi^T \odot U^T)^T \text{vec}(Z) \)
Exact recovery guarantees

- **Exact recovery** ⇒ Success of the convex relaxation
  - Random model on the graph structure [Ling-Stromher’15]
  - Probabilistic guarantees depend on graph spectrum

\[ P_{\text{rec}} \geq 1 - N^{-O(\rho_u^{-1}(S))}, \quad \rho_u(S) := \max_{l \in \{1,\ldots,N\}} \max_{\Omega \in \Omega_S^N} \|u_l, \Omega\|_2^2 \]

- Blind deconvolution (in time) most favorable graph setting

Details in arXiv:1604.07234v1 [cs.IT]
Numerical tests: Recovery rates

- **Recovery rates** over an \((L, S)\) grid and 20 trials
  - Successful recovery when \(\|x^*(h^*)^T - xh^T\|_F < 10^{-3}\)

- **ER (left), ER reweighted \(\ell_{2,1}\) (center), brain reweighted \(\ell_{2,1}\) (right)**

- Exact recovery over non-trivial \((L, S)\) region
  - Reweighted optimization markedly improves performance
  - Encouraging results even for real-world graphs
Numerical tests: Brain graph

- Human brain graph with $N = 66$ regions, $L = 3$ and $S = 3$

- Proposed method also outperforms alternating-minimization solver
  - Unknown $\text{supp}(x) \approx$ Need twice as many observations
  - Stable to Gaussian noise in $y$ ($\sigma^2 = 0.01$)
Multiple output signals

- Suppose we have access to $P$ output signals $\{y_p\}^{P}_{p=1}$

Goal: Identify common filter $H$ fed by multiple unobserved inputs $x_p$
Formulation

- **As.:** \( \{x_p\}_{p=1}^P \) are \( S \)-sparse with common support

- Concatenate outputs \( \bar{y} := [y_1^T, \ldots, y_P^T]^T \) and inputs \( \bar{x} := [x_1^T, \ldots, x_P^T]^T \)

- Unknown rank-one matrices \( Z_p := x_p h^T \). Stack them
  - Vertically in rank one \( \bar{Z}_v := [Z_1^T, \ldots, Z_P^T]^T = \bar{x} h^T \in \mathbb{R}^{NP \times L} \)
  - Horizontally in row sparse \( \bar{Z}_h := [Z_1, \ldots, Z_P] \in \mathbb{R}^{N \times PL} \)

- **Convex formulation**

\[
\min_{\{Z_p\}_{p=1}^P} \|\bar{Z}_v\|_* + \tau \|\bar{Z}_h\|_{2,1}, \quad \text{s. to } \bar{y} = \left( I_P \otimes \left( V \left( \Psi^T \odot U^T \right)^T \right) \right) \text{vec}(\bar{Z}_h)
\]

\( \Rightarrow \) Relax (As.): \( \|\bar{Z}_h\|_{2,1} \leftrightarrow \|\bar{Z}_v\|_{2,1} = \sum_{p=1}^P \|Z_p\|_{2,1} \)
Numerical tests: Multiple signals, recovery rates

- **Recovery rates** over an \((L, S)\) grid and 20 trials
  - Successful recovery when \(\|\hat{x}h^T - \bar{x}h^T\|_F < 10^{-3}\)

- **ER (left), ER reweighted \(\ell_{2,1}\) (center), brain reweighted \(\ell_{2,1}\) (right)**

- Leveraging multiple output signals aids the blind identification task
Blind ID: Takeaways

▷ Extended blind deconvolution of space/time signals to graphs
  ⇒ Key: model diffusion process as output of graph filter
▷ Rank and sparsity minimization subject to model constraints
  ⇒ “Lifting” and convex relaxation yield efficient algorithms
▷ Exact recovery conditions ⇒ Success of the convex relaxation
  ⇒ Probabilistic guarantees that depend on the graph spectrum
▷ Consideration of multiple sparse inputs aids recovery

▷ Envisioned application domains
  (a) Opinion formation in social networks
  (b) Identify sources of epileptic seizure
  (c) Trace “patient zero” for an epidemic outbreak

▷ Unknown shift $S$ ⇒ Network topology inference
Network topology inference

Motivation and preliminaries

Part I: Fundamentals
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Filter design for network operators
- Sampling graph signals
- Blind identification of graph filters
- Network topology inference

Concluding remarks
Motivation and context

Network topology inference from nodal observations
⇒ Approaches use Pearson correlations to construct graphs
⇒ Partial correlations and conditional dependence

Paramount importance in neuroscience
⇒ Functional net inferred from activity

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

We take the reverse path
⇒ How to use GSP to infer the graph topology?
⇒ [Dong15, Mei15, Pavez16, Pasdeloup16]
Generating structure of a diffusion process

- Signal $x$ is the response of a linear diffusion process to a white input

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

$\Rightarrow$ Common generative model. Heat diffusion if $\alpha_k$ constant

- We say the graph shift $S$ explains the structure of signal $x$

- It follows from Cayley Hamilton that we can write diffusion as

$$x = \left( \sum_{l=0}^{N-1} h_l S^l \right) w := Hw$$

$\Rightarrow$ $H$ diagonalized by the eigenvectors of the shift operator
Our approach for topology inference

- We propose a **two-step approach** for graph topology identification

  **STEP 1:** Identify the eigenvectors of the shift

  **STEP 2:** Identify eigenvalues to obtain a suitable shift

- Beyond diffusion ⇒ alternative sources for *spectral templates* ¥
STEP 1: Obtaining the eigenvectors

- The covariance matrix of the signal $x$ is

$$C_x = E \left[ (Hw(Hw)^H) \right] = H E \left[ (ww^H) \right] H^H = HH^H$$

- Since $H$ is diagonalized by $V$, so is the covariance $C_x$

$$C_x = V \left| \sum_{l=0}^{L-1} h_l \Lambda^l \right|^2 V^H = V \text{diag}(|\tilde{h}|^2) V^H$$

- Any shift with eigenvectors $V$ can explain $x$

  $$\Rightarrow G \text{ and its specific eigenvalues have been obscured by diffusion}$$

Observations

(a) There are many shifts that can explain a signal $x$

(b) Identifying the shift $S$ is just a matter of identifying the eigenvalues

(c) In correlation methods the eigenvalues are kept unchanged

(d) In precision methods the eigenvalues are inverted
Other sources of spectral templates

1) Implementation of linear network operators
   - Goal: distributed implementation of linear operator \( B \) via graph filter
     \( \Rightarrow \) \( B \) and \( S \) sharing \( V \) is beneficial for implementation
   - Given a pre-specified \( B \)
     \( \Rightarrow \) Use its eigenvectors as spectral templates to generate a shift \( S \)
     \( \Rightarrow \) The goal here not to identify a shift, but to design one
     Ex.: consensus \( \Rightarrow \) Laplacian of the smallest connected graph

2) Relationship between nodes of a signal
   - Particular transforms \( T \) are known to work well on specific data
     \( \Rightarrow \) Such transform assumes an implicit relation among data \( \Rightarrow S \)
     \( \Rightarrow \) Identification of that relation can provide insights \( V^H = T \)

DCTs: i–iii
STEP 2: Obtaining the eigenvalues

- We can use extra knowledge/assumptions to choose one graph
  - Of all graphs, select one that is optimal in some sense

\[
S^* := \underset{S, \lambda}{\text{argmin}} \quad f(S, \lambda) \quad \text{s. to} \quad S = \sum_{k=1}^{N} \lambda_k v_k v_k^H, \quad S \in S
\]  

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

\[
S := \{S \mid S_{ij} \geq 0, \quad S \in \mathcal{M}^N, \quad S_{ii} = 0, \quad \sum_j S_{1j} = 1\}
\]
  - Can accommodate Laplacian matrices as well

- Problem is convex if we select a convex objective \( f(S, \lambda) \)
  - Minimum energy \( (f(S) = \|S\|_F) \), Fast mixing \( (f(\lambda) = -\lambda_2) \)
The feasibility set in (1) is generally small

⇒ Define $W := V \odot V$ where $\odot$ is the Khatri-Rao product

⇒ Denote by $D$ the index set such that $\text{vec}(S)_D = \text{diag}(S)$

Assume that (1) is feasible, then it holds that $\text{rank}(W_D) \leq N - 1$. If $\text{rank}(W_D) = N - 1$, then the feasible set of (1) is a singleton.

Convex feasibility set ⇒ Search for the optimal solution may be easy

Simulations will show that $\text{rank}(W_D) = N - 1$ arises in practice
Sparse recovery

- Whenever the feasibility set of (1) is non-trivial
  \[ f(S, \lambda) \] determines the features of the recovered graph

  **Ex:** Identify the sparsest shift \( S_0^* \) that explains observed signal structure
  \[ \Rightarrow \text{Set the cost } f(S, \lambda) = \|S\|_0 \]

- Problem is not convex, but can relax to \( \ell_1 \) norm minimization

  \[
  S_1^* := \arg\min_{S, \lambda} \|S\|_1 \quad \text{s. to} \quad S = \sum_{k=1}^{N} \lambda_k v_k v_k^H, \quad S \in S
  \]

- Does the solution \( S_1^* \) coincide with the \( \ell_0 \) solution \( S_0^* \)?
Recovery guarantee

- Denoting by \( \mathbf{m}_i^T \) the \( i \)-th row of \( \mathbf{M} := (\mathbf{I} - \mathbf{WW}^\dagger)_{\mathcal{D}^c} \)
  - Construct \( \mathbf{R} := [\mathbf{m}_2 - \mathbf{m}_1, \ldots, \mathbf{m}_{N-1} - \mathbf{m}_1, \mathbf{m}_N, \ldots, \mathbf{m}_{|\mathcal{D}^c|}]^T \)
  - Denote by \( \mathcal{K} \) the indices of the support of \( \mathbf{s}^* = \text{vec}(\mathbf{S}^*_0) \)

\( \mathbf{S}^*_1 \) and \( \mathbf{S}^*_0 \) coincide if the two following conditions are satisfied:
1) \( \text{rank}(\mathbf{R}_{\mathcal{K}}) = |\mathcal{K}| \); and
2) There exists a constant \( \delta > 0 \) such that
\[
\psi_{\mathbf{R}} := \| \mathbf{I}_{\mathcal{K}^c} (\delta^{-2} \mathbf{R} \mathbf{R}^T + \mathbf{I}_{\mathcal{K}^c} \mathbf{I}_{\mathcal{K}^c})^{-1} \mathbf{I}_{\mathcal{K}}^T \|_{\infty} < 1.
\]

- Cond. 1) ensures uniqueness of solution \( \mathbf{S}^*_1 \)
- Cond. 2) guarantees existence of a dual certificate for \( \ell_0 \) optimality
We might have access to \( \hat{V} \), a noisy version of the spectral templates. With \( d(\cdot, \cdot) \) denoting a (convex) distance between matrices,

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

Recovery result similar to the noiseless case can be derived. Conditions under which we are guaranteed \( d(S^*, S_0^*) \leq C\epsilon \)

Partial access to \( V \) \Rightarrow Only \( K \) known eigenvectors \([v_1, \ldots, v_K]\)

\[
\min_{\{S, S_{\bar{K}}, \lambda\}} \|S\|_1 \quad \text{s. to} \quad S = S_{\bar{K}} + \sum_{k=1}^{K} \lambda_k v_k v_k^T, \quad S \in S, \quad S_{\bar{K}} v_k = 0
\]

Incomplete and noisy scenarios can be combined.
Topology inference in random graphs

- **Erdős-Rényi graphs** of varying size $N \in \{10, 20, \ldots, 50\}$
  - $\Rightarrow$ Edge probabilities $p \in \{0.1, 0.2, \ldots, 0.9\}$
- **Recovery rates** for adjacency (left) and normalized Laplacian (mid)

  ![Graphs showing recovery rates for adjacency and normalized Laplacian](image)

- Recovery is easier for **intermediate values of $p$**
- Rate of recovery related to the $\text{rank}(W_D)$ (histogram $N=10, p=0.2$)
  - $\Rightarrow$ When rank is $N-1$, recovery is **guaranteed**
  - $\Rightarrow$ As rank decreases, there is a detrimental effect on recovery
Sparse recovery guarantee

- Generate 1000 ER random graphs ($N = 20, p = 0.1$) such that
  - Feasible set is not a singleton
  - Cond. 1) in sparse recovery theorem is satisfied

- Noiseless case: $\ell_1$ norm guarantees recovery as long as $\psi_R < 1$

- Condition is sufficient but not necessary
  - Tightest possible bound on this matrix norm
Inferring brain graphs from noisy templates

- Identification of structural brain graphs $N = 66$
- Test recovery for noisy spectral templates $\hat{V}$
  ⇒ Obtained from sample covariances of diffused signals

- Recovery error decreases with increasing number of observed signals
  ⇒ More reliable estimate of the covariance  ⇒ Less noisy $\hat{V}$
- Brain of patient 1 is consistently the hardest to identify
  ⇒ Robustness for identification in noisy scenarios
- Traditional methods like graphical lasso fail to recover $S$
Inferring social graphs from incomplete templates

- Identification of multiple social networks $N = 32$
  - Defined on the same node set of students from Ljubljana
- Test recovery for incomplete spectral templates $\hat{V} = [v_1, \ldots, v_K]$
  - Obtained from a low-pass diffusion process
  - Repeated eigenvalues in $C_x$ introduce rotation ambiguity in $V$

Recovery error decreases with increasing nr. of spectral templates
  - Performance improvement is sharp and precipitous
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 lasso comparison](image)

- Graphical lasso implicitly assumes a filter $H_1 = (\rho I + S)^{-1/2}$
  - For this filter spectral templates work, but not as well (MLE)

- For general diffusion filters $H_2$ spectral templates still work fine
Inferring direct relations

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

Ground truth  Mutual info.  Network deconv.  Our approach

- Network deconvolution assumes a specific filter model [Feizi13]
  ⇒ We achieve better performance by being agnostic to this
Network topology inference cornerstone problem in Network Science
  ▶ Most GSP works analyze how $S$ affect signals and filters
  ▶ Here, reverse path: How to use GSP to infer the graph topology?

Our GSP approach to network topology inference
  ⇒ 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 too: net operators, data transforms

Infer $S$ via convex optimization
  ⇒ Objectives promotes desirable properties
  ⇒ Constraints encode structure a priori info and structure
  ⇒ Formulations for perfect and imperfect templates
  ⇒ Sparse recovery results for both adjacency and Laplacian
Wrapping up

Motivation and preliminaries

Part I: Fundamentals
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Filter design for network operators
- Sampling graph signals
- Blind identification of graph filters
- Network topology inference

Concluding remarks
Concluding remarks

- **Network science** and big data pose new challenges
  - GSP can contribute to **solve** some of those challenges
  - Well suited for **network (diffusion) processes**

- **Central elements in GSP**: graph-shift operator and Fourier transform

- **Graph filters**: operate graph signals
  - Polynomials of the shift operator that can be implemented **locally**

- **Network diffusion/percolation processes via graph filters**
  - Successive/parallel combination of **local linear dynamics**
  - Possibly time-varying diffusion coefficients
  - Accurate to model certain setups
  - GSP yields insights on how those processes behave
Concluding remarks

- **GSP results** can be applied to solve practical problems
  - Filter design (design of distributed operators)
  - Sampling, interpolation (network control)
  - Blind deconvolution (source ID), shift design (network topology ID)

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
Looking ahead

- First step to challenging problems: social nets, brain signals

- Motivates further research:
  - Statistical modeling
  - Space-time variation
  - Changing topologies
  - Nonlinear approaches
  - Local, reduced-complexity algorithms

- Thanks!
  - Contact: antonio.garcia.marques@urjc.es  gmateosb@ece.rochester.edu
    ssegarra@seas.upenn.edu  aribeiro@seas.upenn.edu
  - Slides on stationarity available at:
We include a list of our published work in graph signal processing (GSP) categorized by topic. We also include relevant works by other authors. This latter list is not intended to be exhaustive but rather its purpose is to guide the interested reader to pertinent publications in different areas of graph signal processing.
Sampling bandlimited graph signals


Interpolating graph signals


Graph filter design and network operators


References: our work

**Blind graph deconvolution**


**GSP-Based Network Topology Inference**


References: our work

**Stationary Graph Processes**

**Median Graph Filters**
General references


Filtering


Sampling


Interpolation and Reconstruction


**Topology Inference**


**Stationarity**


Topics of interest

- Graph-signal transforms and filters
- Non-linear graph SP
- Statistical graph SP
- Prediction and learning in graphs
- Network topology inference
- Network tomography
- Control of network processes
- Signals in high-order graphs
- Graph algorithms for network analytics
- Graph-based distributed SP algorithms
- Graph-based image and video processing
- Communications, sensor and power networks
- Neuroscience and other medical fields
- Web, economic and social networks

Symposium dates: December 8-9, 2016

Organizers:
Michael Rabbat (McGill Univ.)
Gonzalo Mateos (Univ. of Rochester)
Antonio Marques (King Juan Carlos Univ.)