Desiderata: Process, analyze and learn from network data [Kolaczyk’09]
Network Science analytics

- **Desiderata:** Process, analyze and learn from network data [Kolaczyk’09]
  - Network as graph $G = (\mathcal{V}, \mathcal{E}, W)$: encode pairwise relationships
  - Interest here not in $G$ itself, but in data associated with nodes in $\mathcal{V}$
  - Object of study is a graph signal $x$
  - Networks are common and interesting. So are graph signals?
Network of economic sectors of the United States

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

Oil extraction (OG), Petroleum and coal products (PC), Construction (CO)
Administrative services (AS), Professional services (MP)
Credit intermediation (FR), Securities (SC), Real state (RA), Insurance (IC)

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
- $\mathcal{E} = \text{Output of sector } i \text{ is an input to sector } j$ (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 $x$ = output per sector = disaggregated GDP
  - Network structure used to, e.g., reduce GDP estimation noise

- Signal is as interesting as the network itself. Arguably more
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 $\mathbf{x}$?
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 \( \Rightarrow \) \( \tilde{x} = F^H x \)
  \[ F_{kn} = \frac{e^{j2\pi kn/N}}{\sqrt{N}} \]

- Fourier transform \( \Rightarrow \) Projection on eigenvector space of cycle
Covariances and principal components

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

- 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) $\Rightarrow \tilde{x} = \mathbf{V}^H x$

- PCA transform $\Rightarrow$ Projection on eigenvector space of (inverse) covariance
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) $\Rightarrow \tilde{x} = V^H x$

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

GSP: Extend principles to general graphs and corresponding (graph) signals
Part I: Fundamentals

Motivation and preliminaries

Part I: Fundamentals

  Graphs 101
  Graph signals and the shift operator
  Graph Fourier Transform (GFT)
  Graph filters and network processes

Part II: Applications

  Sampling graph signals
  Stationarity of graph processes
  Network topology inference

Concluding remarks
Formally, a graph $G$ (or a network) is a triplet $(\mathcal{V}, \mathcal{E}, \mathcal{W})$

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

$\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of edges defined as ordered pairs $(n, m)$
- Write $\mathcal{N}(n) = \{m \in \mathcal{V} : (m, n) \in \mathcal{E}\}$ as the in-neighbors of $n$

$\mathcal{W} : \mathcal{E} \to \mathbb{R}$ 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, $\mathcal{W} : \mathcal{E} \to \mathbb{R}_{++}$

Unweighted graphs $\Rightarrow w_{nm} \in \{0, 1\}$, for all $(n, m) \in \mathcal{E}$

Undirected graphs $\Rightarrow (n, m) \in \mathcal{E}$ if and only if $(m, n) \in \mathcal{E}$ and $w_{nm} = w_{mn}$, for all $(n, m) \in \mathcal{E}$
Time, Images, and Covariances

- **Time:** Unweighted and directed graphs
  - $V = \{0, 1, \ldots, 23\}$
  - $E = \{(0, 1), (1, 2), \ldots, (22, 23), (23, 0)\}$
  - $W : (n, m) \mapsto 1$, for all $(n, m) \in E$

- **Images:** Unweighted and undirected graphs
  - $V = \{1, 2, 3, \ldots, 9\}$
  - $E = \{(1, 2), (2, 3), \ldots, (8, 9), (1, 4), \ldots, (6, 9)\}$
  - $W : (n, m) \mapsto 1$, for all $(n, m) \in E$

- **Covariances:** Weighted and undirected graphs
  - $V = \{1, 2, 3, 4\}$
  - $E = \{(1, 1), (1, 2), \ldots, (4, 4)\} = V \times V$
  - $W : (n, m) \mapsto \sigma_{nm} = \sigma_{mn}$, 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 $G = (V, E, W)$, the adjacency matrix $A \in \mathbb{R}^{N \times N}$ is

\[
A_{nm} = \begin{cases} 
  w_{nm}, & \text{if } (n, m) \in E \\
  0, & \text{otherwise}
\end{cases}
\]

- 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 $|\mathcal{N}(i)|$
  - In directed graphs, have out-degree and an in-degree

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

- The $k$th power of $A$ describes $k$–hop neighborhoods of each node.
  - $[A^k]_{ij}$ non-zero only if there exists a path of length $k$ from $i$ to $j$
  - Support of $A^k$: pairs that can be reached in $k$ hops
Laplacian of a graph

- 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

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

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

\[
\begin{pmatrix}
3 & -1 & 0 & -2 \\
-1 & 6 & -3 & -2 \\
0 & -3 & 4 & -1 \\
-2 & -2 & -1 & 5
\end{pmatrix}
\]
Spectral properties of the Laplacian

- Denote by $\lambda_i$ and $v_i$ the eigenvalues and eigenvectors of $L$

- $L$ is positive semi-definite
  - $x^T L x = \frac{1}{2} \sum_{(i,j) \in E} w_{ij} (x_i - x_j)^2 \geq 0$, for all $x$
  - All eigenvalues are nonnegative, i.e. $\lambda_i \geq 0$ for all $i$

- A constant vector $1$ is an eigenvector of $L$ with eigenvalue $0$
  \[
  [L1]_i = \sum_{j \in N(i)} w_{ij}(1 - 1) = 0
  \]
  - Thus, $\lambda_1 = 0$ and $v_1 = (1/\sqrt{N}) 1$

- In connected graphs, it holds that $\lambda_i > 0$ for $i = 2, \ldots, N$
  - Multiplicity{\(\lambda = 0\)} = number of connected components
Motivation and preliminaries

Part I: Fundamentals
- Graphs 101
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Sampling graph signals
- Stationarity of graph processes
- Network topology inference

Concluding remarks
Graph signals

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

$\Rightarrow$ Defined on the vertices of the graph (data tied to nodes)

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

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

$\Rightarrow x_n$ denotes the signal value at the $n$-th vertex in $\mathcal{V}$

$\Rightarrow$ Implicit ordering of vertices (same as in $A$ or $L$)

$\Rightarrow$ Data associated with links of $G$ $\Rightarrow$ Use line graph of $G$

$$
\begin{pmatrix}
  x_0 \\
  x_1 \\
  x_2 \\
  \vdots \\
  x_9
\end{pmatrix} =
\begin{pmatrix}
  0.6 \\
  0.7 \\
  0.3 \\
  \vdots \\
  0.7
\end{pmatrix}
$$
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

\[ \mathbf{x}_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \]

Sample patient 1 with subtype 1

\[ \mathbf{x}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \]

Sample patient 2 with subtype 1

- To understand a graph signal, the structure of \( G \) must be considered
To understand and analyze $\mathbf{x}$, useful to account for $G$'s structure

Associated with $G$ is the graph-shift operator $\mathbf{S} \in \mathbb{R}^{N \times N}$

$S_{ij} = 0$ for $i \neq j$ 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

Set $S = A_{dc}$

What is $Sx$?

$S$ 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**

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

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

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

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

Part I: Fundamentals
   Graphs 101
   Graph signals and the shift operator
   Graph Fourier Transform (GFT)
   Graph filters and network processes

Part II: Applications
   Sampling graph signals
   Stationarity of graph processes
   Network topology inference

Concluding remarks
Discrete Fourier Transform (DFT)

Let \( x \) be a temporal signal, its DFT is \( \tilde{x} = F^H x \), with 
\[
F_{kn} = \frac{1}{\sqrt{N}} e^{j \frac{2\pi}{N} kn}.
\]

- Equivalent description, provides insights
- Oftentimes, more parsimonious (bandlimited)
- 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 $S = VΛV^{-1}$ 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 $\tilde{x}_k = v_k^Hx = <v_k, x>$
  ⇒ 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
  $\Rightarrow$ Notion of bandlimitedness: $x = \sum_{k=1}^{K} \tilde{x}_k v_k$
  $\Rightarrow$ Sampling, compression, filtering, pattern recognition
Meaning of the eigenvalues

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

- Q: What about the eigenvalues $\lambda_k = \Lambda_{kk}$?
  - $\Rightarrow$ When $S = A_{dc}$, we get $\lambda_k = e^{-j\frac{2\pi}{N}(k-1)}$
  - $\Rightarrow$ $\lambda_k$ can be viewed as frequencies!!

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

- Q: Does this carry over for graph signals? $\Rightarrow$ No, only if $S = L$
  - $\Rightarrow \{\lambda_k\}_{k=1}^N$ will be very important when analyzing graph filters
Consider a graph $G$, let $x$ be a signal on $G$, and set $S = L$.

Define the quadratic form

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

$\Rightarrow$ $x^T L x$ 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 $v_k$
  - Let us plot some of the eigenvectors of $L$ (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 the 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 the same metabolism.
- Alternatively, consider a genetic network.
  - Genetic profiles become graph signals on a 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 $\mathbf{x}_i \in \{0, 1\}^{2458}$

- 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

  $k = 3 \Rightarrow 13.3\%$, $k = 5 \Rightarrow 12.9\%$, $k = 7 \Rightarrow 14.6\%$

- 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 ($S = L$)

Example of signal $x_i$

Frequency representation $\tilde{x}_i$
Define the **discriminative power** of frequency $v_k$ as

$$DP(v_k) = \frac{\left| \sum_{i:y_i=1} \tilde{x}_i(k) - \sum_{i:y_i=2} \tilde{x}_i(k) \right|}{\sum_{i} \left| \tilde{x}_i(k) \right|}$$

- Normalized difference between the mean GFT coefficient for $v_k$

  ⇒ Among patients with serous and endometrioid subtypes

- Discriminative power is not equal across frequencies

The discriminative power defined is one of many proper heuristics
Retaining most discriminative frequencies

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

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

- Perform iGFT to get the graph signal $\hat{x}_i$ ⇒ Classify with k-NN

![Graph showing error percentage for different frequencies and k values](image)
Motivation and preliminaries

Part I: Fundamentals
- Graphs 101
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Sampling graph signals
- Stationarity of graph processes
- Network topology inference

Concluding remarks
Linear (shift-invariant) graph filter

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

Focus on linear filters
$\Rightarrow$ 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 $[\Psi]_{k,l} := \lambda_k^{l-1}$, we have $\tilde{h} = \Psi h \Rightarrow$ Defs can be rendered equivalent
$\Rightarrow$ More on this later, now focus on Def1
Graph filters as linear network operators

- **Def1** says \( H = \sum_{l=0}^{L} h_l S^l \)

- Suppose \( H \) acts on a graph signal \( x \) to generate \( y = Hx \)

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

  \( y \) 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

  \( y \) is a linear combination of successive shifted versions of \( x \)

- **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, Sandryhaila14]
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 \( \mathbf{y} \)

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

\[ \mathbf{y} = 1 \mathbf{x}^{(0)} + 1 \mathbf{x}^{(1)} + 0.5 \mathbf{x}^{(2)} = \begin{pmatrix} 1.0 \\ 2.5 \\ 1.5 \\ 1.5 \\ 1.5 \\ 0.0 \end{pmatrix} \]
Def 2 says $H = V \text{diag}(\tilde{h}) V^{-1}$

$\Rightarrow$ Since $S = V \Lambda V^{-1}$, Def 1 $H = \sum_{l=0}^{L} h_l S^l$ yields

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

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

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

$\Rightarrow$ $\tilde{H} := \sum_{l=0}^{L} h_l \Lambda^l$ modulates the freq. coefficients $\tilde{\mathbf{x}}$ to obtain $\tilde{\mathbf{y}}$

$\Rightarrow$ $V$ brings the signal $\tilde{\mathbf{y}}$ back to the graph domain $\mathbf{y}$
Def2 says $H = V \text{diag}(\tilde{h}) V^{-1}$

⇒ Since $S = V \Lambda V^{-1}$, Def1 $H = \sum_{l=0}^{L} h_{l} S^{l}$ yields

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

The application $H x$ of filter $H$ to $x$ can be split into three parts

⇒ $V^{-1}$ takes signal $x$ to the graph freq. domain $\tilde{x}$

⇒ $\tilde{H} := \sum_{l=0}^{L} h_{l} \Lambda^{l}$ modulates the freq. coefficients $\tilde{x}$ to obtain $\tilde{y}$

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

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

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

⇒ Output at frequency $k$ depends only on input at frequency $k$

$$\tilde{y}_{k} = \tilde{h}_{k} \tilde{x}_{k}$$
Relation between $\tilde{h}$ and $h$ in a more friendly manner?

⇒ Since $\tilde{h} = \text{diag}(\sum_{l=0}^{L} h_l \Lambda^l)$, we have that $\tilde{h}_k = \sum_{l=0}^{L} h_l \lambda_k^l$

⇒ Define the Vandermonde matrix $\Psi$ as

$$\Psi := \begin{pmatrix}
1 & \lambda_1 & \ldots & \lambda_1^L \\
\vdots & \vdots & & \vdots \\
1 & \lambda_N & \ldots & \lambda_N^L 
\end{pmatrix}$$

Frequency response of a graph filter

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

Relation between $\tilde{h}$ and $h$ in a more friendly manner?
⇒ Since $\tilde{h} = \text{diag}(\sum_{l=0}^{L} h_l \Lambda^l)$, we have that $\tilde{h}_k = \sum_{l=0}^{L} h_l \lambda_k^l$
⇒ Define the Vandermonde matrix $\Psi$ as

$$
\Psi := \begin{pmatrix}
1 & \lambda_1 & \ldots & \lambda_1^L \\
& \vdots & \ddots & \vdots \\
& & \lambda_N & \ldots & \lambda_N^L \\
1 & \lambda_1 & \ldots & \lambda_1^L \\
\end{pmatrix}
$$

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 $k \neq k'$
More on the frequency response

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

- Since $\tilde{h} = \psi h$ ⇒ If $\lambda_k = \lambda_k'$, then
  ⇒ 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 $\lambda_k = e^{-j \frac{2\pi}{N} (k-1)}$

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

⇒ The frequency response is the DFT of the impulse response

$$
\tilde{h} = F^H h
$$
Frequency response for graph signals and filters

- 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 $S$
  - Some of the properties are not true for graphs
  - Several options to generalize operations
Deltas and system identification

- In time, given canonical delta $\delta_1 = e_1 = [1, 0, ..., 0]^T$
  - Any other delta $e_i$ is a shifted version of $e_1$
  - Fourier transform is constant: $\tilde{e}_1 = 1$ (all freqs. are excited)
  - Output to $e_1$ characterizes the filter, in fact output to $e_1$ is $h$

- In graph SP none of the above is true!

- Shifting is not translation $\Rightarrow e_j$ is not a shifted version of $e_i$

- GFT $\tilde{e}_i = V^{-1}e_i \Rightarrow$ how strongly node $i$ expresses each freq.
  - $\tilde{e}_i$ is not an eigenvector, $\tilde{e}_i \neq 1$, and entries of $\tilde{e}_i$ can be zero

- System id? $\Rightarrow$ Let $y_i$ be filter output when $x = e_i$, then we have
  \[
  h = \Psi^{-1}\text{diag}^{-1}(V^{-1}e_i)V^{-1}y_i = \Psi^{-1}\text{diag}^{-1}(\tilde{e}_i)V^{-1}y_i
  \]

  $\Rightarrow$ More involved, works if $[\tilde{e}_i]_k$ are non-zero (see [Segarra16] for blind id)
Implementing graph filters: frequency or space

- Frequency or space?

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

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

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

- Space definition useful for modeling
  - Diffusion, percolation, opinion formation, ... (more on this soon)
Implementing graph filters: frequency or space

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

- 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]
Linear network processes via graph filters

- Consider 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 \([x_{t-1}]_j, j \in \mathcal{N}(i)\)

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

\( \Rightarrow x_t = H x_0 \), with \( H \) a polynomial of \( S \) \( \Rightarrow \) linear graph filter
Consider 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 \([x_{t-1}]_j, j \in \mathcal{N}(i)\)

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

\( \Rightarrow x_t = Hx_0, \) with \( H \) a polynomial of \( S \) \( \Rightarrow \) 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 diffusion dynamics \( x_t = \alpha S x_{t-1} + w \)

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

⇒ When \( t \to \infty \):
\[
x_\infty = (I - \alpha S)^{-1} w \Rightarrow \text{AR graph filter}
\]
Before finite-time dynamics (FIR filters)

Consider now diffusion dynamics \( x_t = \alpha S x_{t-1} + w \)

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

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

Higher orders [Isufi-Leus16]

⇒ \( M \) successive diffusion dynamics ⇒ AR of order \( M \)

⇒ Process is the sum of \( M \) parallel diffusions ⇒ 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

\[ x_t = H_t^a(S)x_{t-1} + H_t^b(S)w \Rightarrow x_t = H_t^A(S)x_0 + H_t^B(S)w \]

\[ \Rightarrow y = x_t, \text{ sequential/parallel application, linear combination} \]

- 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
Graph filters: Summary

- **Linear** graph-signal operators that account for the structure of $G$
  $\Rightarrow$ Polynomials of $S$, orthogonal operators on the frequency domain

- A few take-home messages
  $\Rightarrow$ Output is weighted sum of shifted inputs
  $\Rightarrow$ Shifting is not a translation, but a local diffusion
  $\Rightarrow$ Distributed implementation across $L$-hop neighborhood
  $\Rightarrow$ GFT given by Vandermonde matrix $\Psi$, different from $V^{-1}$
  $\Rightarrow$ System identification more involved

- Graph filter design
  $\Rightarrow$ Designing $h$, designing $\tilde{h}$, and... designing $S$?

- Useful to process signals, but also to model networked phenomena
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 $\alpha$
  \[
  m_1(x) = \sum_{t=1}^{T} \mathbf{1}\left\{ \frac{\|x(t) - x(t-1)\|_2}{\|x(t)\|_2} > \alpha \right\}
  \]

- **Meso**: Cluster brain states and count the changes in clusters
  \[
  m_2(x) = \sum_{t=1}^{T} \mathbf{1}\{c(t) \neq c(t-1)\}
  \]
  ⇒ where $c(t)$ is the cluster to which $x(t)$ belongs.

- **Macro**: Sample entropy. Measure of complexity of time series
  \[
  m_3(x) = -\log \left( \frac{\sum_t \sum_{s \neq t} \mathbf{1}\{\|\bar{x}_3(t) - \bar{x}_3(s)\|_\infty > \alpha\}}{\sum_t \sum_{s \neq t} \mathbf{1}\{\|\bar{x}_2(t) - \bar{x}_2(s)\|_\infty > \alpha\}} \right)
  \]
  ⇒ 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

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

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

- Analyzing diffusion in the frequency domain

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

⇒ Freq. response $\tilde{h}_k = 1/(1 + \beta \lambda_k)$

- 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 $x_{null}$

$$x_{null}(t) = x_{diff}(\pi_t)$$

$\Rightarrow \pi_t$ is a random permutation of the time indices

- Correlation between variability ($m_1$, $m_2$, and $m_3$) and learning?
- We consider three time series of brain activity
  - The original fMRI data $x$
  - The filtered data $x_{diff}$
  - The null signal $x_{null}$
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
  - Graphs 101
  - Graph signals and the shift operator
  - Graph Fourier Transform (GFT)
  - Graph filters and network processes

Part II: Applications
  - Sampling graph signals
  - Stationarity of graph processes
  - 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

- **Statistical GSP: understanding random graph signals**

- **Network topology inference**
  - Infer shift from collection of network diffused signals

- Many more (glad to discuss or redirect):
  - Filter banks
  - Windowing, convolution, duality...
  - Nonlinear GSP
Motivation and preliminaries

Part I: Fundamentals
- Graphs 101
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Sampling graph signals
- Stationarity of graph processes
- Network topology inference

Concluding remarks
Sampling and interpolation are cornerstone problems in classical SP
⇒ How to recover a signal using only a few observations?
⇒ Need to limit the degrees of freedom: subspace, smoothness
⇒ What are reasonable observation models for graph signals?

Subset selection model ⇒ Static graph signal
⇒ Have access to a subset of the nodes
Motivation and preliminaries

▶ **Sampling and interpolation** are cornerstone problems in classical SP
  ⇒ How to recover a signal using only a few observations?
  ⇒ Need to limit the degrees of freedom: subspace, smoothness
  ⇒ What are reasonable observation models for graph signals?

▶ **Subset selection** model ⇒ Static graph signal
  ⇒ Have access to a subset of the nodes

▶ **Node aggregation** model
  ⇒ Incorporate local graph structure into the observation model
  ⇒ Recover signal from local observations at one node
How to find $x \in \mathbb{R}^N$ using $P < N$ observations?

⇒ Our focus on **bandlimited** signals, but other models possible

⇒ $\tilde{x} = V^{-1}x$ sparse

⇒ $x = \sum_{k \in K} \tilde{x}_k v_k$, with $|K| = K < N$

⇒ $S$ involved in generation of $x$

⇒ Agnostic to the particular form of $S$
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

⇒ Correspond to the previous two observation models
⇒ Selection [Anis14, Chen15, Tsitsvero15, Puy15, Wang15]
⇒ Aggregation [Segarra15], [Marques15]
⇒ Hybrid scheme combining both ⇒ Space-shift sampling
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
  \[\Rightarrow\] Give rise to **selection** and **aggregation** sampling
Intuitive generalization to graph signals
\[ C \in \{0, 1\}^{P \times N} \text{ (matrix } P \text{ rows of } I_N) \]
\[ \text{Sampled signal is } \bar{x} = Cx \]

Goal: recover \( x \) based on \( \bar{x} \)

Assume that the support of \( \mathcal{K} \) is known (w.l.o.g. \( \mathcal{K} = \{ k \}_{k=1}^K \))

Since \( \bar{x}_k = 0 \) for \( k > K \), define \( \tilde{x}_K := [\bar{x}_1, ..., \bar{x}_K]^T = E^T_K \bar{x} \)

Approach: use \( \bar{x} \) 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 $P \geq K$

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

\[ \Rightarrow (CV_K) \text{ submatrix of } V \]
Selection sampling: Recovery

- Number of samples $P \geq K$

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

\[ \Rightarrow (CV_K) \text{ submatrix of } V \]

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)

\[ \Rightarrow \text{Columns of } V_K(CV_K)^{-1} \text{ are the interpolators} \]
Selection sampling: Recovery

- Number of samples $P \geq K$

$$\tilde{x} = Cx = CV_K \tilde{x}_K$$

$$\Rightarrow (CV_K) \text{ submatrix of } V$$

Recall of selection sampling

If $\operatorname{rank}(CV_K) \geq K$, $x$ 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)
  $$\Rightarrow \text{Columns of } V_K(CV_K)^{-1} \text{ are the interpolators}$$

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

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

- Consider shifted (aggregated) signals $y^{(l)} = S'x$
  - $y^{(l)} = Sy^{(l-1)}$ → 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
  \[ \tilde{y}_i = Cy_i \]

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

- **Goal:** recover $x$ based on $\bar{y}_i \Rightarrow$ Same approach than before
  - 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 $\psi_{kl} = \lambda_k^{l-1}$

**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!
  - Node $i$ has to be able to capture frequencies in $\mathcal{K}$
  - The frequencies have to distinguishable
Aggregation sampling: Recovery

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

- Define $\bar{\mathbf{u}}_i := \mathbf{V}_K^T \mathbf{e}_i$ and recall $\Psi_{kl} = \frac{1}{\lambda_k - 1}$

Recovery of aggregation sampling

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

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

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

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

- Bandlimited signals: Signals that can be well estimated locally
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:**
\[ y_4 = [-0.55, 1.27, -2.94]^T \]
\[ \Rightarrow [y_4]_1 = x_4 = -0.55, \quad [y_4]_2 = x_2 + x_3 + x_5 + x_6 + x_7 = 1.27 \]
\[ \Rightarrow \text{For this example, any node guarantees recovery} \]
\[ \Rightarrow \text{Selection sampling fails if, e.g.,} \{1, 3, 4\} \]
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
  \[ \Rightarrow \text{additive noise } w_i \]
  \[ \Rightarrow \text{covariance } R_w^{(i)} \]
- BLUE interpolation

\[
\hat{x}_K^{(i)} = [\Psi_i^H C^H (R_w^{(i)})^{-1} C \Psi_i]^{-1} \Psi_i^H C^H (R_w^{(i)})^{-1} \bar{z}_i
\]

- If \( P = K \), then \( \hat{x}^{(i)} = V_K (C \Psi_i)^{-1} \bar{z}_i \)
Presence of noise

- **Linear observation model**
  ⇒ additive noise \( w_i \)
  ⇒ covariance \( R^{(i)}_w \)

- **BLUE interpolation**

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

- 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.** $\Rightarrow$ submatrix of unitary $\mathbf{V}_K$
  - **Aggr.** $\Rightarrow$ Vander. $\times$ diag
    - $[\mathbf{u}_i]_k \neq 0$ and $\lambda_k \neq \lambda_{k'}$ $\Rightarrow$ full-spark

\[ \tilde{\mathbf{x}}^* = \arg \min_{\tilde{\mathbf{x}}} \| \tilde{\mathbf{x}} \|_0 \quad \text{s.t.} \quad \mathbf{C}_i \tilde{\mathbf{x}} = \mathbf{C}_\Psi \tilde{\mathbf{x}} \]

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

- Noisy case: sampling nodes critical
Unknown frequency support

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

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

- If full spark ⇒ $P = 2K$ 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
  \( p = 0.15, 0.20, 0.25, \)  
  \( K = 3, \) 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

**Selection:** 4 nodes, 1 sample  
**Space-shift:** 2 nodes, 2 samples  
**Aggregat.:** 1 node, 4 samples

- 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 closed-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
  
  $\Rightarrow$ 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>$[Sx]_i$</td>
<td>$[S^2x]_i$</td>
</tr>
<tr>
<td>$[x]_i$</td>
<td>$[x]_j$</td>
<td>$[x]_k$</td>
</tr>
<tr>
<td>$[Sx]_i$</td>
<td>$[Sx]_j$</td>
<td>$[Sx]_k$</td>
</tr>
<tr>
<td>$[S^2x]_i$</td>
<td>$[S^2x]_j$</td>
<td>$[S^2x]_k$</td>
</tr>
<tr>
<td>$[S^3x]_i$</td>
<td>$[S^3x]_j$</td>
<td>$[S^3x]_k$</td>
</tr>
<tr>
<td>$[x]_i$</td>
<td>$[Sx]_i$</td>
<td>$[x]_j$</td>
</tr>
</tbody>
</table>
More on sampling graph signals

- **Beyond bandlimitedness**
  - Smooth signals [Chen15]
  - Parsimonious in kernelized domain [Romero16]

- **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
- Graphs 101
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Sampling graph signals
- Stationarity of graph processes
- Network topology inference

Concluding remarks
Motivation and context

- We frequently encounter stochastic processes
- Statistical SP $\Rightarrow$ tools for their understanding

- Stationarity facilitates the analysis of random signals in time
  $\Rightarrow$ Statistical properties are time-invariant

- We seek to extend the concept of stationarity to graph processes
  $\Rightarrow$ Network data and irregular domains motivate this
  $\Rightarrow$ Lack of regularity leads to multiple definitions

- Classical SSP can be generalized: spectral estimation, periodograms,...
  $\Rightarrow$ Better understanding and estimation of graph processes
  $\Rightarrow$ Related works: [Girault 15], [Perraudin 16]
Prelimaries: Weak stationarity in time

(1) **Correlation** of stationary discrete time signals is invariant to shifts

\[ C_x := \mathbb{E}[xx^H] = \mathbb{E}[x^H(n-l)x(n-l)] = \mathbb{E}[S^lx(S^lx)^H] \]

(2) Signal is the output of a LTI filter \( H \) excited with white noise \( w \)

\[ x = Hw, \quad \text{with} \quad \mathbb{E}[ww^H] = I \]

(3) The covariance matrix \( C_x \) is **diagonalized** by the Fourier matrix

\[ C_x = F\text{diag}(p)F^H \]

- The process has a **power spectral density** \( \Rightarrow p := \text{diag}(F^HC_xF) \)
- Each of these definitions can be generalized to graph signals
Definition (shift invariance)

Process $\mathbf{x}$ is weakly stationary with respect to $\mathbf{S}$ if and only if $(b > c)$

$$
\mathbb{E}\left[(\mathbf{S}^a \mathbf{x}) ((\mathbf{S}^H)^b \mathbf{x})^H\right] = \mathbb{E}\left[(\mathbf{S}^{a+c} \mathbf{x}) ((\mathbf{S}^H)^{b-c} \mathbf{x})^H\right]
$$

- Use $a$ and $b$ shifts as reference. Shift by $c$ forward and backward
  - Signal is stationary if these shifts do not alter its covariance
- It reduces to $\mathbb{E}[\mathbf{x}\mathbf{x}^H] = \mathbb{E}[(\mathbf{S}'\mathbf{x})(\mathbf{S}'\mathbf{x})^H]$ when $\mathbf{S}$ is a directed cycle
- Time shift is orthogonal, $\mathbf{S}^H = \mathbf{S}^{-1}$ ($a = 0$, $b = N$ and $c = l$)
- Need reference shifts because $\mathbf{S}$ can change energy of the signal
Definition (filtering of white noise)

Process \( x \) is weakly stationary with respect to \( S \) if it can be written as the output of linear graph filter \( H \) with white input \( w \)

\[
x = Hw, \quad \text{with } \mathbb{E} \left[ ww^H \right] = I
\]

- The filter \( H \) is linear shift invariant if \( \Rightarrow H(Sx) = S(Hx) \)
- Equivalently, \( H \) polynomial on the shift operator \( \Rightarrow H = \sum_{l=0}^{L} h_l S^l \)
- Filter \( H \) determines color \( \Rightarrow C_x = \mathbb{E} \left[ (Hw)(Hw)^H \right] = HH^H \)
Definition (Simultaneous diagonalization)

Process $x$ is weakly stationary with respect to $S$ if the covariance $C_x$ and the shift $S$ are simultaneously diagonalizable.

$$S = V \Lambda V^H \implies C_x = V \text{diag}(p) V^H$$

- Equivalent to time definition because $F$ diagonalizes cycle graph
- The process has a power spectral density $\Rightarrow p := \text{diag}(V^H C_x V)$
Definition (Simultaneous diagonalization)

Process $x$ is weakly stationary with respect to $S$ if the covariance $C_x$ and the shift $S$ are simultaneously diagonalizable.

$$S = V \Lambda V^H \quad \Rightarrow \quad C_x = V \text{diag}(p) V^H$$

- Equivalent to time definition because $F$ diagonalizes cycle graph.
- The process has a power spectral density $\Rightarrow p := \text{diag}(V^H C_x V)$
Equivalence of definitions and PSD

- Have introduced three equally valid definitions of weak stationarity
  - They are different but, under mild conditions, equivalent

**Proposition**

Process $\mathbf{x}$ has shift invariant correlation matrix $\Leftrightarrow$ it is the output of a linear shift invariant filter $\Leftrightarrow$ Covariance jointly diagonalizable with shift

- Shift and Filtering $\Rightarrow$ How stationary signals look like (local invariance)
- Simultaneous Diagonalization $\Rightarrow$ A PSD exists $\Rightarrow$ $\mathbf{p} := \text{diag}(\mathbf{V}^H \mathbf{C}_x \mathbf{V})$ $\Rightarrow$ The PSD collects the eigenvalues of $\mathbf{C}_x$ and is nonnegative

**Proposition**

Let $\mathbf{x}$ be stationary in $\mathbf{S}$ and define the process $\tilde{\mathbf{x}} := \mathbf{V}^H \mathbf{x}$. Then, it holds that $\tilde{\mathbf{x}}$ is uncorrelated with covariance matrix $\mathbf{C}_{\tilde{\mathbf{x}}} = \mathbb{E} [\tilde{\mathbf{x}} \tilde{\mathbf{x}}^H] = \text{diag}(\mathbf{p})$. 
Weak stationary graph processes examples

Example (White noise)

- White noise $w$ is stationary in any graph shift $S = VΛV^H$
- Covariance $C_w = \sigma^2 I$ simultaneously diagonalizable with all $S$

Example (Covariance matrix graphs and Precision matrices)

- Every process is stationary in the graph defined by its covariance matrix
- If $S = C_x$, shift $S$ and covariance $C_x$ diagonalized by same basis
- Process is also stationary on precision matrix $S = C_x^{-1}$

Example (Heat diffusion processes and ARMA processes)

- Heat diffusion process in a graph $\Rightarrow x = \alpha_0(I - \alpha L)^{-1}w$
- Stationary in $L$ since $\alpha_0(I - \alpha L)^{-1}$ is a polynomial on $L$
- Any autoregressive moving average (ARMA) process on a graph
Power spectral density examples

Example (White noise)

- Power spectral density \( \Rightarrow p = \text{diag}(V^H(\sigma^2 I)V) = \sigma^2 I \)

Example (Covariance matrix graphs and Precision matrices)

- Power spectral density \( \Rightarrow p = \text{diag}(V^H(V\Lambda V^H)V) = \text{diag}(\Lambda) \)

Example (Heat diffusion processes and ARMA processes)

- Power spectral density \( \Rightarrow p = \text{diag}\left[\alpha_0^2 (I - \alpha \Lambda)^{-2}\right] \)
Given a process $x$, the covariance of $\tilde{x} = V^H x$ is given by

$$C_{\tilde{x}} := \mathbb{E} [\tilde{x}\tilde{x}^H] = \mathbb{E} [(V^H x)(V^H x)^H] = \text{diag}(p)$$

**Periodogram** ⇒ Given samples $\{x_r\}_{r=1}^R$, average GFTs of samples

$$\hat{p}_{pg} := \frac{1}{R} \sum_{r=1}^R |\tilde{x}_r|^2 = \frac{1}{R} \sum_{r=1}^R |V^H x_r|^2$$

**Correlogram** ⇒ Replace $C_x$ in PSD definition by sample covariance

$$\hat{p}_{cg} := \text{diag} \left( V^H \hat{C}_x V \right) := \text{diag} \left( V^H \left[ \frac{1}{R} \sum_{r=1}^R x_r x_r^H \right] V \right)$$

**Periodogram and correlogram** lead to identical estimates $\hat{p}_{pg} = \hat{p}_{cg}$
Theorem

If the process $x$ is Gaussian, periodogram estimates have bias and variance

- **Bias** $\Rightarrow b_{pg} := E[\hat{p}_{pg}] - p = 0$

- **Variance** $\Rightarrow \Sigma_{pg} := E[(\hat{p}_{pg} - p)(\hat{p}_{pg} - p)^H] = \frac{2}{R} \text{diag}^2(p)$

- The periodogram is **unbiased** but the **variance** is not too good
  $\Rightarrow$ **Quadratic in** $p$. Same as time processes

- Alternative nonparametric methods to reduce variance
  $\Rightarrow$ **Average windowed periodogram**
  $\Rightarrow$ **Filterbanks**
  $\Rightarrow$ Bias - variance tradeoff characterized [Marques16, Segarra16]
Until now non-parametric estimators: parametric estimation?

View $\mathbf{x}$ as output of a graph filter with $P \ll N$ parameters

$$
\mathbf{p} = \text{diag} \left( \mathbb{E} \left[ (\mathbf{V}^H \mathbf{x})(\mathbf{V}^H \mathbf{x})^H \right] \right) = \text{diag} \left( \mathbf{V}^H (\mathbf{C}_x) \mathbf{V} \right)
$$

Use $\mathbf{x}$ to obtain $\widehat{\mathbf{C}}_x$

$$
\widehat{\rho}_{cg} = \text{diag} \left( \mathbf{V}^H \widehat{\mathbf{C}}_x \mathbf{V} \right)
$$
Parametric PSD estimation

- Until now non-parametric estimators: parametric estimation?
  ⇒ View $x$ as output of a graph filter with $P \ll N$ parameters

$$p = \text{diag} \left( \mathbb{E} \left[ (V^H x)(V^H x)^H \right] \right) = \text{diag} \left( V^H (C_x) V \right)$$

Use $x$ to obtain $\hat{C}_x$  ⇒  $\hat{p}_{cg} = \text{diag} \left( V^H \hat{C}_x V \right)$

$$p = \text{diag} \left( (\Psi h)(\Psi h)^H \right) = \text{diag} \left( \Psi (hh^H) \Psi^H \right)$$

Use $x$ to obtain $\hat{h}$  ⇒  $\hat{p}_{par} = \text{diag} \left( \Psi (\hat{h}\hat{h}^H) \Psi^H \right)$

- Key how to use $x$ to estimate $\hat{h}$
Parametric PSD estimation: finding $h$

- How to use realization $x$ to estimate $\hat{h}$
  - Input $w$ white and $x$ stationary ⇒ Rely on correlations
  - Problem can be formulated in the vertex or in the frequency domain

- Vertex domain: $\hat{h} = \arg\min_h d(C(h), \hat{C}(x))$
  - $C(h) = H(h, S)H(h, S)^H$
  - $\hat{C}(x) = xx^H$

- Frequency domain: $\hat{h} = \arg\min_h d(p(h), \hat{p}(x))$
  - $p(h) = \text{diag}(\Psi(hh^H)\Psi^H)$
  - $\hat{p}(x) = \text{diag}(V^H(xx^H)V)$

- $C(\cdot)$ and $p(\cdot)$ quadratic on $h$
  - General estimation nonconvex ⇒ If $d(\cdot, \cdot)$ quadratic, phase retrieval
  - Particular cases ($S \succeq 0$ and $h \succeq 0$) tractable ⇒ Not in time
Parametric PSD estimation: an example

- Intuition on parametric estimation
  - Better performance because less parameters must be estimated
  - Give rise to smoother PSD estimates

- Example: Laplacian diffusion with positive coefficients
  - Diffusion will favor exponential spectral decay

\[ p = \text{diag}(\mathbf{V}^H(C_x)\mathbf{V}) \]
Parametric PSD estimation: an example

- Intuition on parametric estimation
  - Better performance because less parameters must be estimated
  - Give rise to smoother PSD estimates

- Example: Laplacian diffusion with positive coefficients
  - Diffusion will favor exponential spectral decay
Parametric PSD estimation: an example

► Intuition on parametric estimation
  ⇒ Better performance because less parameters must be estimated
  ⇒ Give rise to smoother PSD estimates

► Example: Laplacian diffusion with positive coefficients
  ⇒ Diffusion will favor exponential spectral decay

\[ p = \text{diag} \left( \mathbf{V}^H (\mathbf{C}_x) \mathbf{V} \right) \]
\[ \hat{p}_{\text{par}} = \text{diag} \left( \psi (\hat{\mathbf{h}}^H \hat{\mathbf{h}}) \psi^H \right) \]
\[ \hat{p}_{\text{cg}} = \text{diag} \left( \mathbf{V}^H \hat{\mathbf{C}}_x \mathbf{V} \right) \]
Alternative parametric formulations

- Chose vertex or frequency based on the type of filter
  ⇒ MA, AR, ARMA can be generalized

- If the number of parameters (filter degree) not known
  ⇒ Use regularizers so that $\mathbf{h}$ is sparse

- Alternative parametric models
  ⇒ $\mathbf{x}$ sum of frequency basis ⇒ $\tilde{\mathbf{x}} = \mathbf{V}^H \mathbf{x}$ is sparse
  ⇒ $\mathbf{x}$ is diffusion of a sparse initial state ⇒ $\mathbf{w}$ is sparse
Average periodogram

- MSE of periodogram as a function of the nr. of observations $R$

- Baseline ER random graph ($N = 100$ and $p = 0.05$) and $S = A$

- Observe filtered white Gaussian noise and estimate PSD

- Normalized MSE evolves as $2/R$ as expected
  ⇒ Invariant to size, topology, and PSD

- Same behavior observed in non-Gaussian processes (theory not valid)
Windowed average periodogram

- Performance of local windows and random windows
- Block stochastic graph ($N = 100$, 10 communities) and small world
- Process filters white noise with different number of taps

- The use of windows introduces bias but reduces total error (MSE)
- Local windows work better than random windows
  ⇒ Advantage of local windows is larger for local processes
Opinion source identification

- **Opinion diffusion** in Zachary’s karate club network \((N = 34)\)
- **Observed opinion** \(x\) obtained by diffusing sparse white rumor \(w\)

- Given \(\{x_r\}_{r=1}^R\) generated from unknown \(\{w_r\}_{r=1}^R\)
  \(\Rightarrow\) Diffused through filter of unknown nonnegative coefficients \(\beta\)
- **Goal** \(\Rightarrow\) **Identify the support** of each rumor \(w_r\)
- **First** \(\Rightarrow\) **Estimate** \(\beta\) from Moving Average PSD estimation
- **Second** \(\Rightarrow\) **Solve** \(R\) sparse linear regressions to recover \(\text{supp}(w_r)\)

![Graph showing source identification error versus number of observations]

- Source identification error
- Number of observations
PSD of face images

- PSD estimation for spectral signatures of faces of different people
- 100 grayscale face images \( \{x_i\}_{i=1}^{100} \in \mathbb{R}^{10304} \) (10 images \( \times \) 10 people)
- Consider \( x_i \) as realization graph process that is Stationary on \( \hat{C}_x \)
- Construct \( \hat{C}_x^{(j)} = \mathbf{V}(j) \Lambda_c^{(j)} \mathbf{V}^H(j) \) based on images of person \( j \)

- Process of person \( j \) approximately stationary in \( \hat{C}_x \) (left)
- Use windowed average periodogram to estimate PSD of new face
Stationarity: Takeaways

► Extended the notion of weak stationarity for graph processes
► **Three definitions** inspired in stationary time processes
  ⇒ Shown all of them to be **equivalent**

► Defined **power spectral density** and studied its estimation
► Generalized classical **non-parametric estimation** methods
  ⇒ Periodogram and correlogram where shown to be equivalent
  ⇒ Windowed average periodogram, filter banks
► Generalized classical ARMA **parametric estimation** methods
  ⇒ Particular cases tractable

► **Extensions**
  ⇒ Other parametric schemes
  ⇒ **Space-time** variation
Network topology inference

Motivation and preliminaries

Part I: Fundamentals
  Graphs 101
  Graph signals and the shift operator
  Graph Fourier Transform (GFT)
  Graph filters and network processes

Part II: Applications
  Sampling graph signals
  Stationarity of graph processes
  Network topology inference

Concluding remarks
Motivation and context

- **Network topology inference** from nodal observations [Kolaczyk’09]
  ⇒ Approaches use Pearson correlations to construct graphs [Brovelli04]
  ⇒ Partial correlations and conditional dependence [Friedman08, Karanikolas16]

- **Key in neuroscience** [Sporns’10]
  ⇒ Functional net inferred from activity
Motivation and context

- Network **topology inference** from nodal observations [Kolaczyk’09]
  - Approaches use **Pearson correlations** to construct graphs [Brovelli04]
  - Partial correlations and conditional dependence [Friedman08, Karanikolas16]

- Key in neuroscience [Sporns’10]
  - Functional net inferred from activity

- Most GSP works: How known graph $S$ affects signals and filters

- Here, reverse path: How to use **GSP to infer the graph topology**?
  - Gaussian graphical models [Egilmez16]
  - Smooth signals [Dong15], [Kalofolias16]
  - Stationary signals [Segarra16], [Pasdeloup16]
  - Directed graphs [Mei-Moura15], [Shen16]

†Segarra, Marques, Mateos, Ribeiro, *Network Topology Identification from Spectral Templates*, IEEE SSP16
‡Segarra, Marques, Mateos, Ribeiro, *Network Topology Inference from Spectral Templates*, JSTSP (sub.)
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** $V$
  ⇒ Graph sparsification, network deconvolution,…
Step 1: Eigenvectors from graph stationarity

- Given a set of signals \( \{x_r\}_{r=1}^R \) find \( S \)
  - We view signals as samples of random graph process \( x \)
  - \( AS. x \) is stationary in \( S \)

- Equivalent to “\( x \) is the linear diffusion of a white input”

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

- i.e. \( x = Hw \) ⇒ Examples: heat diffusion, structural equation models

\[
x = (I - \alpha L)^{-1}w \quad x = Ax + w
\]

- We say the graph shift \( S \) explains the structure of signal \( x \)

- Key point after assuming stationarity: eigenvectors of the covariance
Step 1: Eigenvectors from covariance

- The covariance matrix of the stationary signal $x = Hw$ is

$$C_x = E[xx^T] = HE[(ww^T)]H^T = HH^T$$

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

$$C_x = V|\sum_{l=0}^{L-1} h_l \Lambda^l|^2 V^T = V \text{diag}(p) V^T$$

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

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

Observations

(a) Identifying $S \rightarrow$ identifying the eigenvalues

(b) Correlation methods $\rightarrow$ eigenvalues are kept unchanged

(c) Precision methods $\rightarrow$ eigenvalues are inverted
Step 1: Other sources of spectral templates

1) Graph sparsification
   - Goal: given $S_f$ find sparser $S$ with same eigenvectors
     $$S_f = V_f \Lambda_f V_f^T$$ and set $V = V_f$
     $$\Rightarrow$$ Otentimes referred to as network deconvolution problem
Step 1: Other sources of spectral templates

1) Graph sparsification
   - Goal: given $S_f$ find sparser $S$ with same eigenvectors
     - Find $S_f = V_f \Lambda_f V_f^T$ and set $V = V_f$
     - Sometimes referred to as network deconvolution problem

2) Nodal relation assumed by a given transform
   - GSP: decompose $S = V \Lambda V^T$ and set $V^T$ as GFT
   - SP: some transforms $T$ known to work well on specific data
   - Goal: given $T$, set $V^T = T$ and identify $S$ ⇒ intuition on data relation

DCTs: i–iii
Step 1: Other sources of spectral templates

1) **Graph sparsification**
   - Goal: given $S_f$ find sparser $S$ with same eigenvectors
     - Find $S_f = V_f \Lambda_f V_f^T$ and set $V = V_f$
     - Often times referred to as network deconvolution problem

2) **Nodal relation assumed by a given transform**
   - GSP: decompose $S = V \Lambda V^T$ and set $V^T$ as GFT
   - SP: some transforms $T$ known to work well on specific data
   - Goal: given $T$, set $V^T = T$ and identify $S$ ⇒ intuition on data relation

3) **Implementation of linear network operators**
   - Goal: distributed implementation of linear operator $B$ via graph filter
     ⇒ Feasible if $B$ and $S$ share eigenvectors ⇒ Like 1) with $S_f = B$
Step 2: Obtaining the eigenvalues

- Given \( V \), there are many possible \( S = V \text{diag}(\lambda)V^T \)
  - We can use extra knowledge/assumptions to choose one graph
  - Of all graphs, select one that is optimal in some sense

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

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

\[
S := \{S \mid S_{ij} \geq 0, \ S \in \mathcal{M}^N, \ S_{ii} = 0, \ \sum_j S_{1j} = 1\}
\]
  - 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 \((f(\lambda) = -\lambda_2)\)
Size of the feasibility set

- Feasible set in (1) small ⇒ helps optimization
  ⇒ Search over $\lambda \in \mathbb{R}^N$
  ⇒ $N$ linear constraints $S_{ii} = 0$
- To be rigorous define
  ⇒ $W := \mathbf{V} \odot \mathbf{V}$ (Khatri-Rao)
  ⇒ $\mathcal{D}$ index set diagonal elements

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.

- Take-aways
  ⇒ Convex and small feasibility set ⇒ Exhaustive search affordable
  ⇒ $\text{rank}(W_D) = N - 1$ arises in practice
Non-convex objective: Sparse recovery

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

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

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

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

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

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

- Recall that $W := V \otimes V$
- 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 $K$ 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) $\text{rank}(R_K) = |K|$; and
2) There exists a constant $\delta > 0$ such that

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

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

- So far, two-step algorithm based on perfect spectral templates
  ⇒ However, perfect knowledge of $\mathbf{V}$ may not be available
  ⇒ Robust designs?

**Q1:** How to modify the optimization in step 2?
  ⇒ Distance for noise, orthogonal subspace for incomplete

**Q2:** Recovery guarantees?
Noisy spectral templates

We might have access to $\hat{V}$, a noisy version of the spectral templates.

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

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

How does the noise in $\hat{V}$ affect the recovery?
Noisy spectral templates

We might have access to \( \hat{V} \), a noisy version of the spectral templates.

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

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

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

Stability (robustness) can be established.

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 \( \mathcal{K} \)
Incomplete spectral templates

- Partial access to $\mathbf{V}$ ⇒ Only $K$ known eigenvectors $\mathbf{V}_K = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$

⇒ E.g., if (sample) covariance is rank-deficient

$$\min_{\{\mathbf{S}, \mathbf{S}_\tilde{K}, \lambda\}} \|\mathbf{S}\|_1 \quad \text{s. to} \quad \mathbf{S} = \mathbf{S}_\tilde{K} + \sum_{k=1}^{K} \lambda_k \mathbf{v}_k \mathbf{v}_k^T, \quad \mathbf{S} \in \mathcal{S}, \quad \mathbf{S}_\tilde{K} \mathbf{v}_k = 0$$
Incomplete spectral templates

**Partial access to $\mathbf{V}$**  ⇒  Only $K$ known eigenvectors $\mathbf{V}_K = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$

⇒ E.g., if (sample) covariance is rank-deficient

$$\min_{\{\mathbf{S}, \mathbf{S}_\bar{K}, \lambda\}} \|\mathbf{S}\|_1 \text{  s. to  } \mathbf{S} = \mathbf{S}_\bar{K} + \sum_{k=1}^{K} \lambda_k \mathbf{v}_k \mathbf{v}_k^T, \quad \mathbf{S} \in \mathcal{S}, \quad \mathbf{S}_\bar{K} \mathbf{v}_k = 0$$

**How does the (partial) knowledge of $\mathbf{V}$ affect the recovery?**

⇒ A bit more involved, conditions 1) and 2) depend now on $\mathbf{V}_K$

⇒ For $K = N$, guarantees boil down to the noiseless case

**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)

- 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 $\hat{\mathbf{V}} = [\mathbf{v}_1, \ldots, \mathbf{v}_K]$
  - Obtained from a low-pass diffusion process
  - Repeated eigenvalues in $\mathbf{C}_x$ introduce rotation ambiguity in $\mathbf{V}$

![Graph showing recovery error vs. number of spectral templates](image)

- 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 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

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 $V$
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
Topology ID: Takeaways

- 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
Motivation and preliminaries

Part I: Fundamentals
- Graphs 101
- Graph signals and the shift operator
- Graph Fourier Transform (GFT)
- Graph filters and network processes

Part II: Applications
- Sampling graph signals
- Stationarity of graph processes
- 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/percolations 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
  - Sampling, interpolation *(network control)*
  - Input and system ID *(rumor 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
Reflections of a reluctant convert

- 2014: GSP $\equiv$ projecting on the eigenvector space of the graph shift
  $\Rightarrow$ An interesting idea, but can’t be much useful

- 2015: GSP is also about generalizing the notion of a time shift
  $\Rightarrow$ More interesting, but still, can’t be that much useful

- 2016: This is actually much more interesting than I thought
  $\Rightarrow$ **Theory** questions are more interesting than they look at first
  $\Rightarrow$ **Analogies** are powerful. Yield good questions and solutions
  $\Rightarrow$ **Graph signals** exist and they have to be analyzed
Interesting theory questions

- Sampling of graph signals ⇒ There’s more than one generalization
  ⇒ Selection of optimal sampling sets

- Statistical signal processing on graphs ⇒ Generalizes PCA
  ⇒ Estimation of power spectral densities
  ⇒ Ergodic (law of large numbers) theorems for graph signals
  ⇒ Strict sense stationarity

- Topology identification ⇒ A graph that explains a signal
  ⇒ There’s more than one graph and more than one solution
  ⇒ Graph prior knowledge ⇒ E.g., small number of links
  ⇒ Filter prior knowledge ⇒ E.g., graphical lasso

- Nonlinear processing. Wavelets. Algebraic and Topological SP
The power of analogies in applications of GSP

- Bandlimited representations $\sim$ compression $\sim$ sampling $\sim$ filtering
  $\Rightarrow$ These analogies have not been exploited in irregular domains

- Dimensionality reduction $\Rightarrow$ Signal is low pass on covariance graph
  $\Rightarrow$ Then, sampling should work as well. It does $\Rightarrow$ Sketching

- Collaborative filtering $\Rightarrow$ Ratings low pass on user preference graph
  $\Rightarrow$ Explicit in latent factor models $\Rightarrow$ Low rank
  $\Rightarrow$ Implicit in nearest neighbors rules $\Rightarrow$ Ratings $\hat{x} = Ax$
  $\Rightarrow$ Improve ratings using (graph) filter design
Application domains

- Gene regulatory networks
  - Interactions between genes and between genes and environment

- Brain signal analysis
  - Very large and very strong literature on brain network analysis
  - Classifying neural disorders, predicting learning ability.
  - They analyze networks because they can’t analyze signals

- Economics, finance, social sciences
  - Relationships between actors used more often than apparent
  - Does bank $a$ fail if bank $b$ does?
  - GDP estimation
  - Disaggregate election outcomes per state / county / district

- An orthogonal observation: Not all data is big
Questions? Feel free to contact us.

- Santiago segarra@mit.edu,
- Antonio: antonio.garcia.marques@urjc.es,
- Alejandro: aribeiro@seas.upenn.edu.

Slides and references: