Graph Signal Processing

Alejandro Ribeiro

Dept. of Electrical and Systems Engineering

University of Pennsylvania

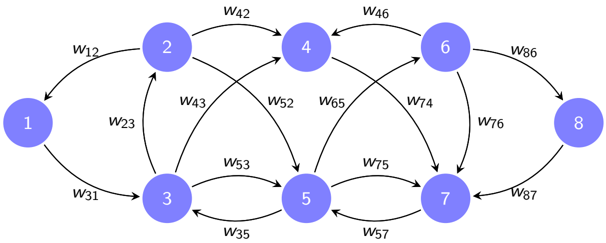
Email: aribeiro@seas.upenn.edu

Web: alelab.seas.upenn.edu

May 17, 2021

Graphs and Graph Signals

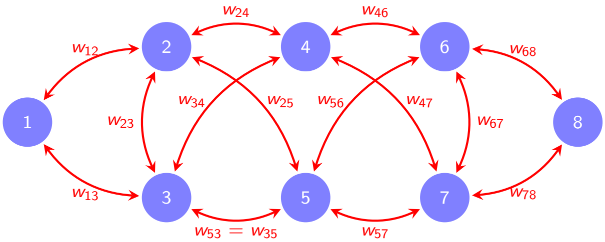
- A graph is a triplet $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, which includes vertices $\mathcal{V}$, edges $\mathcal{E}$, and weights $\mathcal{W}$
  - $\Rightarrow$ Vertices or nodes are a set of n labels. Typical labels are $\mathcal{V} = \{1, \ldots, n\}$
  - $\Rightarrow$ Edges are ordered pairs of labels (i, j). We interpret $(i, j) \in \mathcal{E}$ as "i can be influenced by j."
  - $\Rightarrow$ Weights $w_{ij} \in \mathbb{R}$ are numbers associated to edges (i, j). "Strength of the influence of j on i."



- A graph is symmetric or undirected if both, the edge set and the weight are symmetric

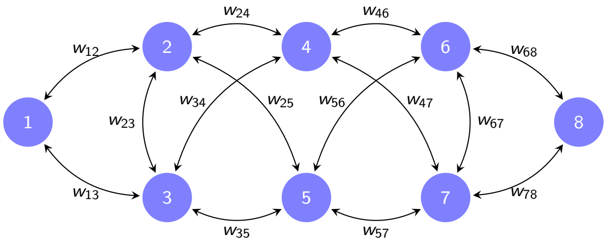
$\Rightarrow$ Edges come in pairs $\Rightarrow$ We have $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$

$\Rightarrow$ Weights are symmetric $\Rightarrow$ We must have $w_{ij} = w_{ji}$ for all $(i, j) \in \mathcal{E}$

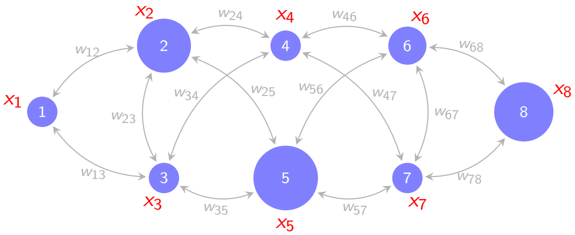


- Graphs can be directed or symmetric. Separately, they can be weighted or unweighted.

- Most of the graphs we encounter in practical situations are symmetric and weighted



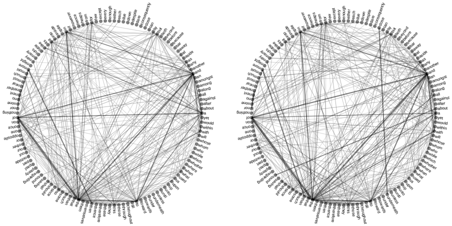
- Consider a given graph $\mathcal{G}$ with n nodes, edge set $\mathcal{E}$ and weights $\mathcal{W}$
- A graph signal is a vector $\mathrm{x} \in \mathbb{R}^n$ in which component x_i is associated with node i
- To emphasize that the graph is intrinsic to the signal we may write the signal as a pair $\Rightarrow (\mathcal{G}, \mathrm{x})$



- The graph is an expectation of proximity or similarity between components of the signal x

- Graphs are generic models of signal structure that can help to learn in several practical problems

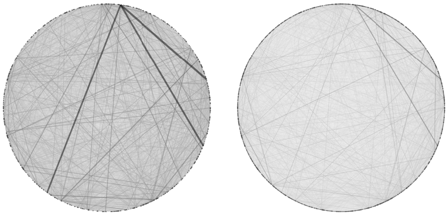
Authorship Attribution



Identify the author of a text of unknown provenance

Segarra et al '16, arxiv.org/abs/1805.00165

Recommendation Systems



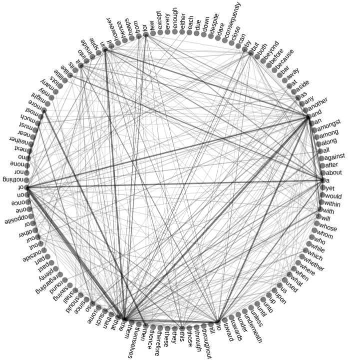
Predict the rating a customer would give to a product

Ruiz et al '18, arxiv.org/abs/1903.12575

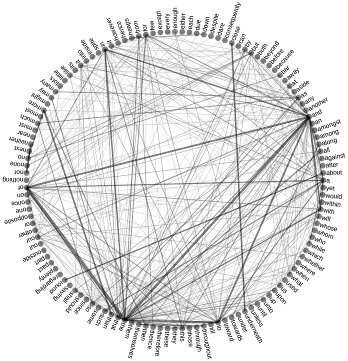
- In both cases there exists a graph that contains meaningful information about the problem to solve

- **Nodes** represent different **function words** and **edges how often words appear close** to each other
  - $\Rightarrow$ A proxy for the different ways in which different authors use the English language grammar

William Shakespeare



Christopher Marlowe

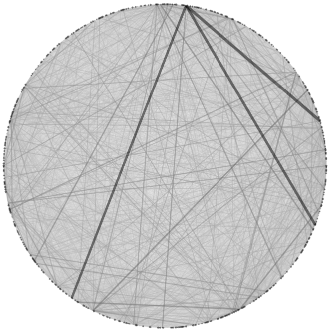


- WAN differences **differentiate the writing styles of Marlowe and Shakespeare** in, e.g., Henry VI

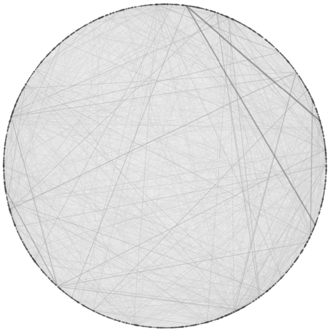
Segarra-Eisen-Egan-Ribeiro, *Attributing the Authorship of the Henry VI Plays by Word Adjacency*, Shakespeare Quarterly 2016, doi.org/10.1353/shq.2016.0024

- Nodes represent different customers and edges their average similarity in product ratings
  - ⇒ The graph informs the completion of ratings when some are unknown and are to be predicted

Variation Diagram for Original (sampled) ratings



Variation Diagram for Reconstructed (predicted) ratings



- Variation energy of reconstructed signal is (much) smaller than variation energy of sampled signal

Ruiz-Gama-Marques-Ribeiro, *Invariance-Preserving Localized Activation Functions for Graph Neural Networks*, arxiv.org/abs/1903.12575

Graphs in Multiagent Physical Systems

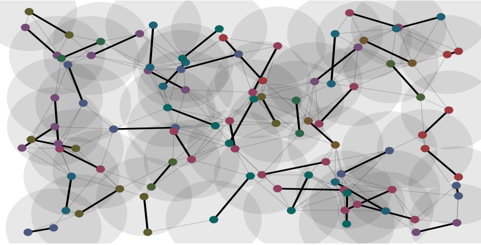
- Graphs are more than data structures $\Rightarrow$ They are models of physical systems with multiple agents

Decentralized Control of Autonomous Systems

Coordinate a team of agents without central coordination

Tolstaya et al '19, arxiv.org/abs/1903.10527

Wireless Communications Networks



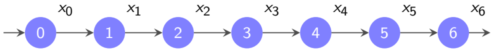
Manage interference when allocating bandwidth and power

Eisen-Ribeiro '19, arxiv.org/abs/1909.01865

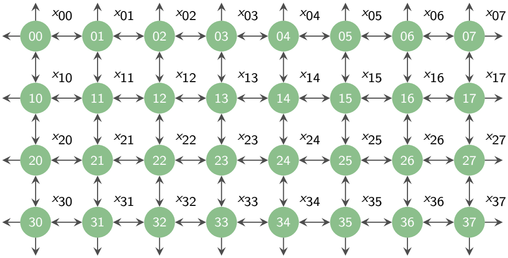
- The graph is the source of the problem $\Rightarrow$ Challenge is that goals are global but information is local

- We can describe discrete time and space using graphs that support time or space signals

Description of time with a directed line graph



Description of images (space) with a grid graph

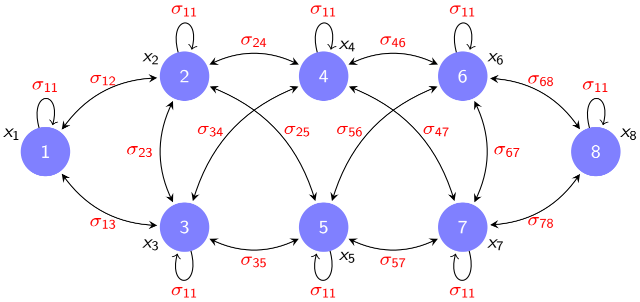


- Line graph represents adjacency of points in time. Grid graph represents adjacency of points in space

- A covariance matrix Σ with entries $((\Sigma))_{ij} = \sigma_{ij}$ is also representable with a graph

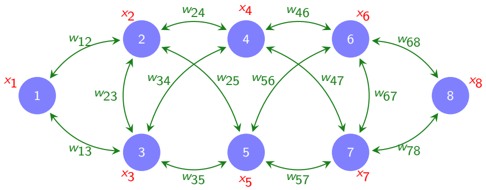
  $\Rightarrow$ One that has self loops to represent the variances σ_{ii}

- A realization x of a random signal X is a signal supported on the covariance matrix graph

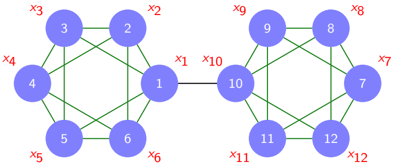


- Time and Space are pervasive and important, but still a (very) limited class of signals
- Use graphs as generic descriptors of signal structure with signal values associated to nodes and edges expressing expected similarity between signal components

A signal supported on a graph



Another signal supported on another graph

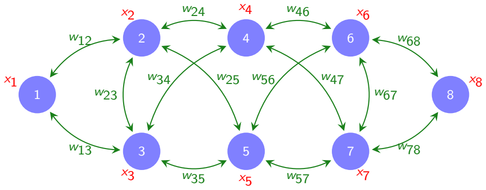


- Nodes are customers. Signal values are product ratings. Edges are cosine similarities of past scores

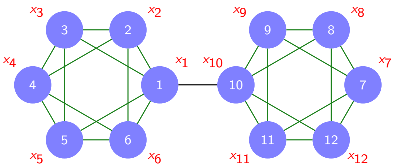
- Time and Space are pervasive and important, but still a (very) limited class of signals

- Use graphs as generic descriptors of signal structure with signal values associated to nodes and edges expressing expected similarity between signal components

A signal supported on a graph



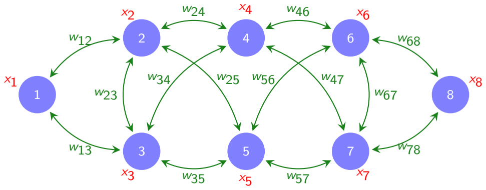
Another signal supported on another graph



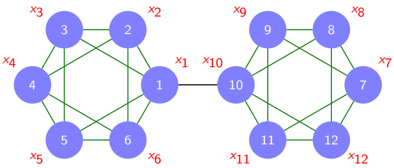
- Nodes are drones. Signal values are velocities. Edges are sensing and communication ranges

- Time and Space are pervasive and important, but still a (very) limited class of signals
- Use graphs as generic descriptors of signal structure with signal values associated to nodes and edges expressing expected similarity between signal components

A signal supported on a graph



Another signal supported on another graph

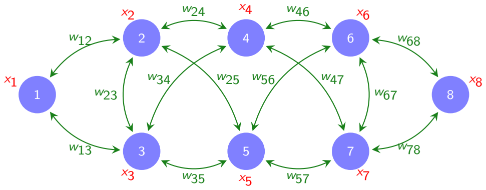


- Nodes are transceivers. Signal values are QoS requirements. Edges are wireless channels strength

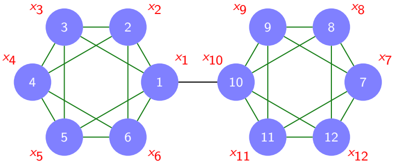
- Time and Space are pervasive and important, but still a (very) limited class of signals

- Use graphs as generic descriptors of signal structure with signal values associated to nodes and edges expressing expected similarity between signal components

A signal supported on a graph



Another signal supported on another graph

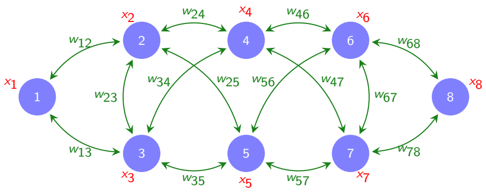


- Nodes are points in time. Signal values. Edges denote time causality

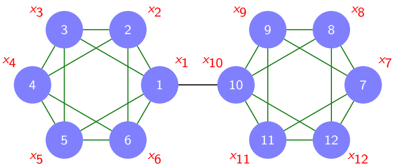
- Time and Space are pervasive and important, but still a (very) limited class of signals

- Use graphs as generic descriptors of signal structure with signal values associated to nodes and edges expressing expected similarity between signal components

A signal supported on a graph



Another signal supported on another graph

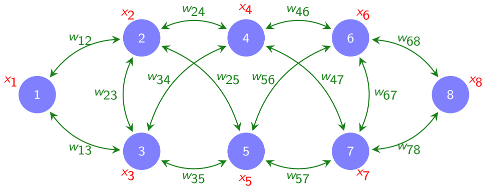


- Nodes are pixels. Signal values are luminances. Edges denote spatial proximities

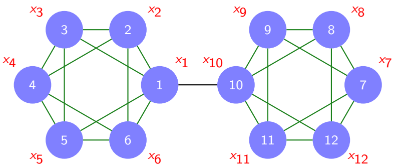
- Time and Space are pervasive and important, but still a (very) limited class of signals

- Use graphs as generic descriptors of signal structure with signal values associated to nodes and edges expressing expected similarity between signal components

A signal supported on a graph



Another signal supported on another graph



- Nodes are entries. Signal values. Edges denote crosscovariances

- Techniques to process signals on graphs that...

  $\Rightarrow$ Generalize techniques developed for time, space, and random signals

  $\Rightarrow$ Recover techniques developed for time, space, and random signals as particular cases

- Graph Fourier transform $\Rightarrow$ Recovers DFT, 2D-DFT and PCA as particular cases

- Graph Convolutional Filters $\Rightarrow$ Recovers time and spatial convolutions as particular cases

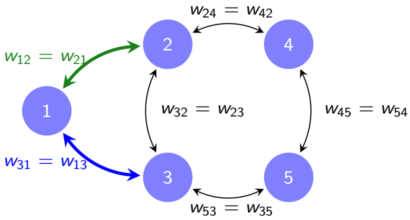
Graph Shift Operators

- Graphs have matrix representations. Which in this course, we call graph shift operators (GSOs)

- The adjacency matrix of graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ is the sparse matrix A with nonzero entries

$$A_{ij} = w_{ij}, \text{ for all } (i,j) \in \mathcal{E}$$

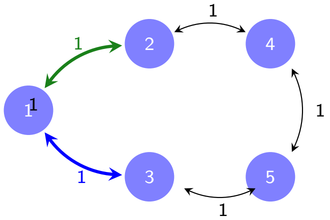
- If the graph is symmetric, the adjacency matrix is symmetric $\Rightarrow$ A = A^T . As in the example



$$A = \begin{bmatrix} 0 & w_{12} & w_{13} & 0 & 0 \\ w_{21} & 0 & w_{23} & w_{24} & 0 \\ w_{31} & w_{32} & 0 & 0 & w_{35} \\ 0 & w_{42} & 0 & 0 & w_{45} \\ 0 & 0 & w_{53} & w_{54} & 0 \end{bmatrix}.$$

- For the particular case in which the graph is unweighted. Weights interpreted as units

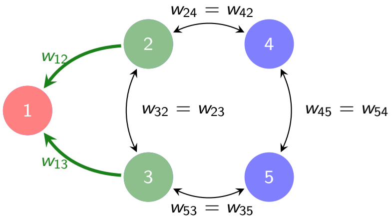
$$A_{ij} = 1, \qquad \text{for all} \quad (i,j) \in \mathcal{E}$$



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

- The neighborhood of node i is the set of nodes that influence i $\Rightarrow n(i) := \{j : (i,j) \in \mathcal{E}\}$

- Degree d_i of node i is the sum of the weights of its incident edges $\Rightarrow d_i = \sum_{j \in n(i)} w_{ij} = \sum_{j:(i,j) \in \mathcal{E}\}} w_{ij}$

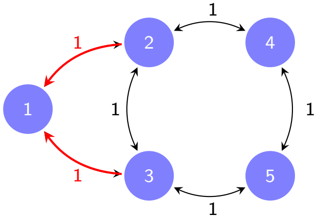


- Node 1 neighborhood $\Rightarrow n(1) = \{2, 3\}$

- Node 1 degree $\Rightarrow n(1) = w_{12} + w_{13}$

- The degree matrix is a diagonal matrix D with degrees as diagonal entries $\Rightarrow D_{ii} = d_i$

- Write in terms of adjacency matrix as $\text{D} = \text{diag}(\text{A1})$. Because $(\text{A1})_i = \sum_j w_{ij} = d_i$



$$\text{D} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

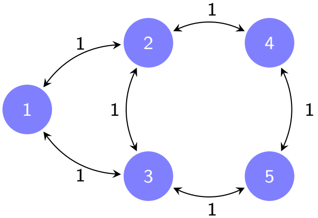
- The Laplacian matrix of a graph with adjacency matrix A is $\Rightarrow \mathsf{L} = \mathsf{D} - \mathsf{A} = \text{diag}(\mathsf{A}\mathbf{1}) - \mathsf{A}$

- Can also be written explicitly in terms of graph weights $A_{ij} = w_{ij}$

$\Rightarrow$ Off diagonal entries $\Rightarrow L_{ij} = -A_{ij} = -w_{ij}$

$\Rightarrow$ Diagonal entries $\Rightarrow L_{ii} = d_i = \sum_{j \in n(i)} w_{ij}$

$$\mathsf{L} = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 \\ -1 & -1 & 3 & 0 & -1 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & 0 & -1 & -1 & 2 \end{bmatrix}$$



- Normalized adjacency and Laplacian matrices express weights relative to the nodes' degrees

- Normalized adjacency matrix $\Rightarrow \bar{\mathrm{A}} := \mathrm{D}^{-1/2}\mathrm{A}\mathrm{D}^{-1/2}$ $\Rightarrow$ Results in entries $(\bar{\mathrm{A}})_{ij} = \dfrac{w_{ij}}{\sqrt{d_i d_j}}$

- The normalized adjacency is symmetric if the graph is symmetric $\Rightarrow \bar{\mathrm{A}}^T = \bar{\mathrm{A}}$.

- Normalized Laplacian matrix $\Rightarrow \bar{L} := D^{-1/2} L D^{-1/2}$. Same normalization of adjacency matrix

- Given definitions normalized representations $\Rightarrow \bar{L} \;=\; D^{-1/2}\Big(D - A\Big)D^{-1/2} \;=\; I - \bar{A}$

  $\Rightarrow$ The normalized Laplacian and adjacency are essentially the same linear transformation.

- Normalized operators are more homogeneous. The entries in the vector A1 tend to be similar.

- The Graph Shift Operator S is a stand in for any of the matrix representations of the graph

Adjacency Matrix	Laplacian Matrix	Normalized Adjacency	Normalized Laplacian
$\mathsf{S} = \mathsf{A}$	$\mathsf{S} = \mathsf{L}$	$\mathsf{S} = \bar{\mathsf{A}}$	$\mathsf{S} = \bar{\mathsf{L}}$

- If the graph is symmetric, the shift operator S is symmetric $\Rightarrow \mathsf{S} = \mathsf{S}^T$

- The specific choice matters in practice but most of results and analysis hold for any choice of S

Laplacians and Graph Signal Variability

- The variability of a graph signal has to be measured with respect to the structure of the graph

- The quadratic form of the graph's Laplacian provides this measure

- We are given a graph signal x and a symmetric graph with edge set $\mathcal{E}$ and edge weights w_{ij}

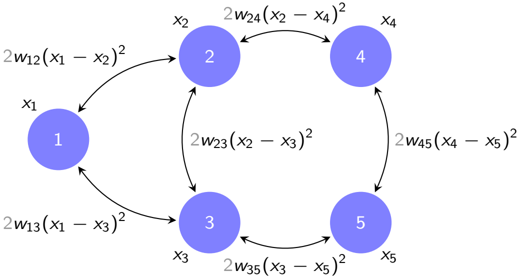
Definition (Total Variation Energy)

The total variation energy of the signal x with respect to the graph $\mathcal{G}$ is defined as

$$\text{TV}(\mathrm{x}) := \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} w_{ij}(x_i - x_j)^2$$

- $(x_i - x_j)^2 \Rightarrow$ Energy of difference between the signal values x_i and x_j observed at node i and node j
- Weighted by the edge weight w_{ij} and summed across all edges

- In the total variation energy $\text{TV}(\mathbf{x}) := \dfrac{1}{2}\sum_{(i,j)\in\mathcal{E}} w_{ij}(x_i - x_j)^2$ there is a term associated to each edge



- The factor 2 appear because the graph is symmetric. Each arrow counts for two edges

Laplacian Quadratic Form

- We are given a graph signal x and a symmetric graph with Laplacian L
- The Laplacian quadratic form is the function $\Rightarrow \mathbf{x}^T \mathbf{L} \mathbf{x}$ (row $\times$ matrix $\times$ column $=$ scalar)

Theorem (Laplacian Quadratic Form)

The Laplacian quadratic form of graph signal x is equal to its total variation energy

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

- The Laplacian quadratic form measures the variability of different graph signals

Proof of quadratic form variability

Proof:

- This is an annoying algebraic calculation
- Isolate an edge $e = (ij) \in \mathcal{E}$ and define a symmetric graph with edge e. It's Laplacian satisfies

$$((\mathsf{L}_e))_{ij} = ((\mathsf{L}_e))_{ji} = -w_{ij} \qquad ((\mathsf{L}_e))_{ii} = ((\mathsf{L}_e))_{jj} = w_{ij}$$

- Since the matrix L_e has only four nonzero entries, the quadratic form $\mathsf{x}^T\mathsf{L}_e\mathsf{x}$ satisfies

$$\mathsf{x}^T\mathsf{L}_e\mathsf{x} \;=\; x_i w_{ij} x_i \;+\; x_j w_{ij} x_j \;-\; x_i w_{ij} x_j \;-\; x_j w_{ij} x_i \;=\; w_{ij}\left(x_i - x_j\right)^2$$

- To conclude notice that we have $\mathsf{L} = \dfrac{1}{2}\displaystyle\sum_{(i,j)\in\mathcal{E}} \mathsf{L}_e$ and therefore $\Rightarrow \mathsf{x}^T\mathsf{L}\mathsf{x} = \dfrac{1}{2}\displaystyle\sum_{(i,j)\in\mathcal{E}} \mathsf{x}^T\mathsf{L}_e\mathsf{x}$ ■

- We say v_k is an eigenvector of L with associated eigenvalue λ_k if we have $Lv_k = \lambda_k v_k$

Corollary (Variability of Laplacian Eigenvectors)

The total variation energy of eigenvector v_k is its associated eigenvalue $\Rightarrow \text{TV}(v_k) = \lambda_k$

Proof: As per the Laplacian quadratic form theorem $\Rightarrow \text{TV}(v_k) = v_k^T L v_k = v_k^T \lambda_k v_k = \lambda_k$ ■

- Eigenvectors of the Laplacian represent different rates of variability $\Rightarrow$ A (graph) Fourier transform

Graph Fourier Transform

- The Graph Fourier Transform (GFT) is a tool for analyzing graph information processing systems

Eigenvectors and Eigenvalues of Shift Operator

- We work with symmetric graph shift operators $\Rightarrow \mathsf{S} = \mathsf{S}^H$

- Introduce eigenvectors v_i and eigenvalues λ_i of graph shift operator S $\Rightarrow \mathsf{S}\mathsf{v}_i = \lambda_i \mathsf{v}_i$

$\Rightarrow$ For symmetric S eigenvalues are real. We have ordered them $\Rightarrow \lambda_0 \leq \lambda_1 \leq \ldots \leq \lambda_n$

- Define eigenvector matrix $\mathsf{V} = [\mathsf{v}_1, \ldots, \mathsf{v}_n]$ and eigenvalue matrix $\Lambda = \text{diag}([\lambda_1; \ldots; \lambda_n])$

Theorem (Eigenvectors Orthogonality of Symmetric Matrices)

Consider a symmetric shift operator (matrix) S, with eigenvalues v and u associated with different eigenvalues λ and μ. The eigenvectors are orthogonal

$$\mathsf{v}^H \mathsf{u} = 0.$$

- The eigenvectors of a symmetric shift operator can be used to define a unitary transform

Proof:

- Since eigenvectors v and u are respectively associated with eigenvalues λ and μ, we have that

$$\mathsf{S}\mathsf{v} = \lambda \mathsf{v}, \qquad \mathsf{S}\mathsf{u} = \mu \mathsf{u}$$

- Since the matrix S is symmetric and real we have that $\mathsf{S}^H = \mathsf{S}$. For here, it follows that

$$\left(\mathsf{u}^H \mathsf{S} \mathsf{v}\right)^H = \mathsf{v}^H \mathsf{S}^H \mathsf{u} = \mathsf{v}^H \mathsf{S} \mathsf{u}$$

- Substitute $\mathsf{S}\mathsf{v} = \lambda \mathsf{v}$ on the leftmost side. Substitute $\mathsf{S}\mathsf{u} = \mu \mathsf{u}$ on the rightmost side.

$$\left(\lambda \mathsf{u}^H \mathsf{v}\right)^H = \left(\mathsf{u}^H \mathsf{S} \mathsf{v}\right)^H = \mathsf{v}^H \mathsf{S}^H \mathsf{u} = \mathsf{v}^H \mathsf{S} \mathsf{u} = \mu \mathsf{v}^H \mathsf{u}$$

- For this to be true with $\lambda \neq \mu$ we must have that $\mathsf{v}^H \mathsf{u} = 0$ ■

- The kth column of the eigenvector matrix $\mathsf{V} = [\mathsf{v}_1, \ldots, \mathsf{v}_n]$ is the kth eigenvector v_k of the shift S

- Since the eigenvectors v_k are orthonormal, the eigenvector matrix V is unitary $\Rightarrow \mathsf{T}^H\mathsf{T} = \mathsf{I}$

$$\begin{bmatrix} \mathsf{v}_1 & \cdots & \mathsf{v}_k & \cdots & \mathsf{v}_n \end{bmatrix}$$

$$\mathsf{V}^H\mathsf{V} = \begin{bmatrix} \mathsf{v}_1^H \\ \vdots \\ \mathsf{v}_k^H \\ \vdots \\ \mathsf{v}_n^H \end{bmatrix} \begin{bmatrix} \mathsf{v}_1^H\mathsf{v}_1 & \cdots & \mathsf{v}_1^H\mathsf{v}_k & \cdots & \mathsf{v}_1^H\mathsf{v}_n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathsf{v}_k^H\mathsf{v}_1 & \cdots & \mathsf{v}_k^H\mathsf{v}_k & \cdots & \mathsf{v}_k^H\mathsf{v}_n \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathsf{v}_n^H\mathsf{v}_n & \cdots & \mathsf{v}_n^H\mathsf{v}_k & \cdots & \mathsf{v}_n^H\mathsf{v}_n \end{bmatrix} = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 \end{bmatrix}$$

- The eigenvalue matrix Λ is a diagonal matrix with diagonal entries equal to eigenvalues of S

$$\Lambda = \begin{bmatrix} \lambda_1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_k & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & \lambda_n \end{bmatrix} = \text{diag}(\lambda_1, \ldots, \lambda_n)$$

- Eigenvalue decomposition $\Rightarrow$ We can write the shift operator as $S = V\Lambda V^H$. Indeed,

$$SV = S[v_1, \ldots, v_n] = [Sv_1, \ldots, Sv_n] = [\lambda_1 v_1, \ldots, S\lambda_n] = V\Lambda$$

- Multiply from the right by V^H and use the fact that V is unitary to eliminate $VV^H = I$

Graph Fourier Transform

Given a graph shift operator $S = V \Lambda V^H$, the graph Fourier transform (GFT) of graph signal x is

$$\tilde{x} = V^H x$$

- GFT $\equiv$ projection on the eigenspace of the graph shift operator $\Rightarrow \tilde{x}_k = v_k^H x = \langle x, v_k \rangle$
- We say $\tilde{x}$ is a graph frequency representation of x. A representation in the graph frequency domain

Inverse Graph Fourier Transform

Given a graph shift operator $S = V\Lambda V^H$, the inverse graph Fourier transform (iGFT) of GFT $\tilde{x}$ is

$$\tilde{\tilde{x}} = V\tilde{x}$$

- Given that $V^H V = I$, the iGFT of the GFT of signal x recovers the signal x

$$\tilde{\tilde{x}} = V\tilde{x} = V\left(V^H x\right) = Ix = x$$

Theorem (The GFT Preserves Energy)

The energy $\|\mathbf{x}\|^2$ of a signal and the energy of its GFT $\|\tilde{\mathbf{x}}\|^2$ are the same $\Rightarrow \|\mathbf{x}\|^2 = \|\tilde{\mathbf{x}}\|^2$

- Given that $\mathbf{V}^H\mathbf{V} = \mathbf{I}$, we have the chain of equalities

$$\|\tilde{\mathbf{x}}\|^2 = \tilde{\mathbf{x}}^H\tilde{\mathbf{x}} = \mathbf{x}^H\mathbf{V}\mathbf{V}^H\mathbf{x} = \mathbf{x}^H\mathbf{I}\mathbf{x} = \mathbf{x}^H\mathbf{x} = \|\mathbf{x}\|^2$$

- Because of inverse theorem, we can write graph signals as $\Rightarrow \mathbf{x} = \mathbf{V}\tilde{\mathbf{x}} = \sum_{k=1}^{n} \tilde{x}_k \mathbf{v}_k$

- Because of Parseval, the energy $\left|\tilde{x}_k\right|^2$ of the kth coefficient is the energy $\mathbf{v}_k$ contributes to $\mathbf{x}$

- Use the Laplacian as shift operator $\Rightarrow \mathbf{S} = \mathbf{L}$

$\Rightarrow$ Total variation energy of Laplacian eigenvectors $\Rightarrow \text{TV}(\mathbf{v}_k) = \lambda_k = \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} w_{ij}(x_i - x_j)^2$

$\Rightarrow$ Eigenvectors are sorted according to their variability $\Rightarrow \text{TV}(\mathbf{v}_1) \leq \text{TV}(\mathbf{v}_2) \leq \ldots \leq \text{TV}(\mathbf{v}_n)$

- The Laplacian GFT decomposes signals $\mathbf{x}$ into components of progressively higher variability

- This variability interpretation is true for Laplacian shift operators only

- Adjacency matrix $\Rightarrow$ If $\mathsf{S} = \mathsf{A}$ this is sort of true if the node degrees are similar

- Normalized Laplacian $\Rightarrow$ If $\mathsf{S} = \bar{\mathsf{L}}$, analogous interpretation holds for normalized variation energy

$$\bar{\text{TV}}(\mathsf{v}_k) = \frac{1}{2} \sum_{(i,j)\in\mathcal{E}} w_{ij} \left(\frac{x_i}{\sqrt{d_i}} - \frac{x_j}{\sqrt{d_j}} \right)^2$$

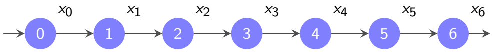
- Normalized Adjacency $\Rightarrow$ If $\mathsf{S} = \bar{\mathsf{A}}$ the same holds because eigenvectors coincide $\Rightarrow \bar{\mathsf{L}} = \mathsf{I} - \bar{\mathsf{L}}$

The GFT of Discrete Time Signals

Adjacency "Matrix" for directed line graph

- We can describe discrete time signals as signals supported on a directed line graph

Description of time with a directed line graph

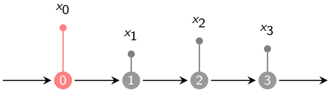


The adjacency "matrix" of a directed line graph

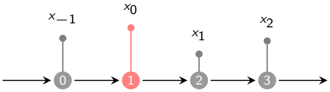
$$S = A = \begin{bmatrix} & \cdot & \cdot & \cdot & \\ \cdot & 0 & 0 & 0 & \cdot \\ \cdot & 1 & 0 & 0 & \cdot \\ \cdot & 0 & 1 & 0 & \cdot \\ \cdot & 0 & 0 & 1 & \cdot \\ & \cdot & \cdot & \cdot & \end{bmatrix}$$

- This adjacency "matrix" has a GFT associated with it. Is it related to the DTFT?

- A time shifting of a time signal means moving the signal up on the time line $\Rightarrow$ Follow the arrows



x



Sx

- Time shift is reinterpreted as multiplication by the adjacency matrix S of the line graph

$$\mathsf{S}\,\mathsf{x} \;=\; \begin{bmatrix} & \cdot & \cdot & \cdot & \\ \cdot & 0 & 0 & 0 & \cdot \\ \cdot & 1 & 0 & 0 & \cdot \\ \cdot & 0 & 1 & 0 & \cdot \\ \cdot & 0 & 0 & 1 & \cdot \\ & \cdot & \cdot & \cdot & \end{bmatrix} \begin{bmatrix} \cdot \\ x_0 \\ x_1 \\ x_2 \\ x_3 \\ \cdot \end{bmatrix} = \begin{bmatrix} \cdot \\ x_{-1} \\ x_0 \\ x_1 \\ x_2 \\ \cdot \end{bmatrix}$$

- Product Sx is such that $(\mathsf{Sx})_n = x_{n-1}$ $\Rightarrow$ Signal components move up on the time line

Time Shifting of a Complex Exponential

- Particularize to the case in which the graph signal x is a complex exponential $\Rightarrow x_n = e^{j2\pi fnTs}$

- Moving components up in the time line for this particular signal yields

$$(\mathsf{Sx})_n = e^{j2\pi f(n-1)Ts} = e^{j2\pi f(-1)Ts} e^{j2\pi f nTs} = e^{-j2\pi fTs} x_n$$

- Complex exponential x is an eigenvector of the shift "matrix" S with associated eigenvalue $e^{-j2\pi fTs}$

- Let e_{fT_s} be a discrete time complex exponential with components $x_n = e^{j2\pi fnTs}$

Theorem (GFT of a Directed Line Graph)

The components of the GFT of a discrete time signal x are $\Rightarrow \tilde{x}_k = \langle \mathrm{x}, e_{fT_s} \rangle = \sum_{-\infty}^{+\infty} x_n e^{-j2\pi fnTs}$

- Which is the exact same definition of the DTFT of the signal x

DFT, 2D-DFT, and PCA transforms are Particular Cases of GFTs

- DFT $\equiv$ GFT of directed cycle graph (connect node $n-1$ to node 1)

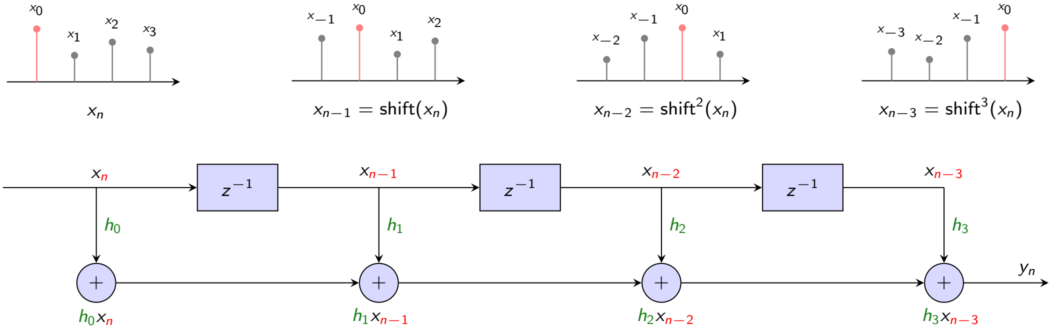
- 2D-DFT $\sim$ GFT of grid graph $\Rightarrow$ In fact, it's complicated. But true enough

- PCA *equiv* GFT of covariance marix graph $\Rightarrow$ Self evident. Same definition

Graph Convolutional Filters

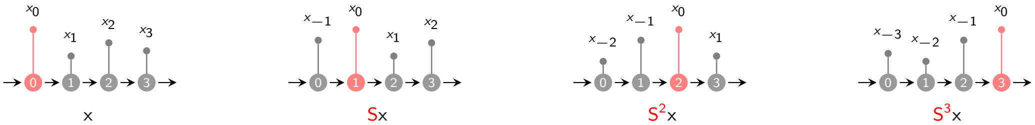
- Graph convolutional filters are the tool of choice for the linear processing of graph signals

- **Convolutional filters** process signals **in time** by leveraging the **time shift** operator



- The **time** convolution is a linear combination of **time shifted** inputs $\Rightarrow y_n = \sum_{k=0}^{K-1} h_k x_{n-k}$

- Time signals are representable as graph signals supported on a line graph S $\Rightarrow$ The pair (S, x)



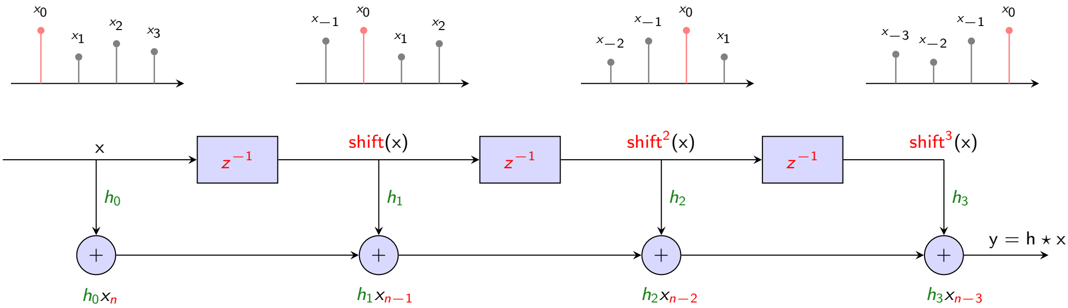
- Time shift is reinterpreted as multiplication by the adjacency matrix S of the line graph

$$\mathsf{S}^3\mathsf{x} = \mathsf{S}\left[\mathsf{S}^2\mathsf{x}\right] = \mathsf{S}\left[\mathsf{S}\left(\mathsf{S}\mathsf{x}\right)\right] = \begin{bmatrix} & \vdots & \vdots & \vdots & \\ \cdots & 0 & 0 & 0 & \cdots \\ \cdots & 1 & 0 & 0 & \cdots \\ \cdots & 0 & 1 & 0 & \cdots \\ \cdots & 0 & 0 & 1 & \cdots \\ & \vdots & \vdots & \vdots & \end{bmatrix} \begin{bmatrix} \vdots \\ x_0 \\ x_1 \\ x_2 \\ x_3 \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ x_{-3} \\ x_{-2} \\ x_{-1} \\ x_0 \\ \vdots \end{bmatrix}$$

- Components of the shift sequence are powers of the adjacency matrix applied to the original signal

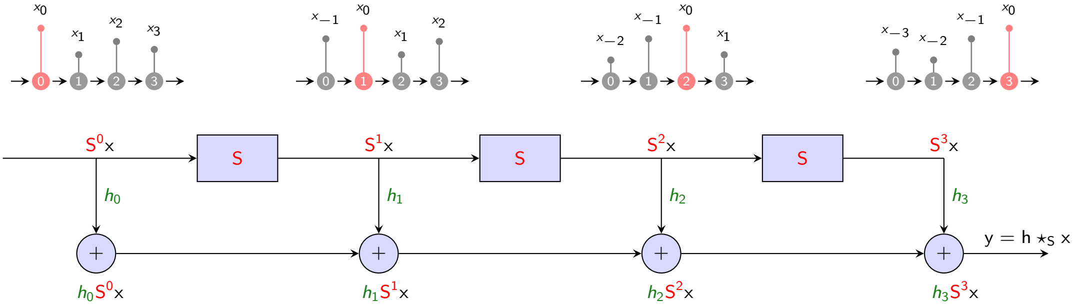
$\Rightarrow$ We can rewrite convolutional filters as polynomials on S, the adjacency of the line graph

- The convolution operation is a linear combination of shifted versions of the input signal
- But we now know that time shifts are multiplications with the adjacency matrix $\mathbf{S}$ of line graph



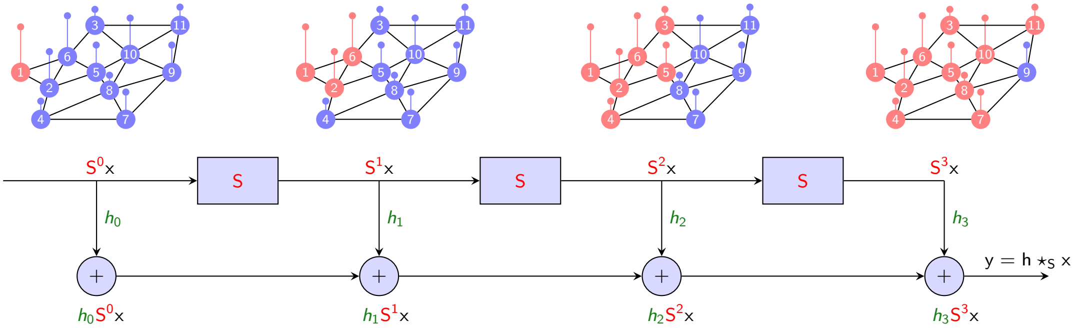
- Time convolution is a polynomial on adjacency matrix of line graph $\Rightarrow \mathbf{y} = \mathbf{h} \star \mathbf{x} = \sum_{k=0}^{K-1} h_k \mathbf{S}^k \mathbf{x}$

- The convolution operation is a linear combination of shifted versions of the input signal
- But we now know that time shifts are multiplications with the adjacency matrix S of line graph



- Time convolution is a polynomial on adjacency matrix of line graph $\Rightarrow y = h \star x = \sum_{k=0}^{K-1} h_k S^k x$

- If we let S be the shift operator of an arbitrary graph we recover the graph convolution



- Given graph shift operator S and coefficients h_k, a graph filter is a polynomial (series) on S

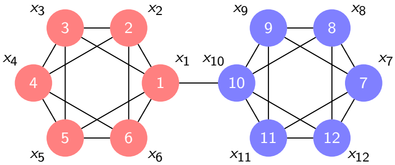
$$H(S) = \sum_{k=0}^{\infty} h_k S^k$$

- The result of applying the filter $H(S)$ to the signal x is the signal

$$y \; = \; H(S)\, x \; = \; \sum_{k=0}^{\infty} h_k S^k x$$

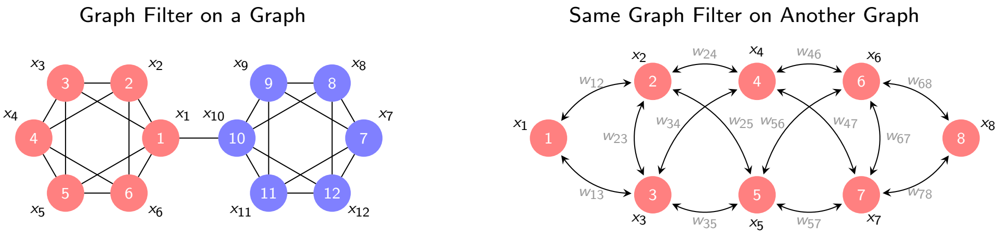
- We say that $y = h \star_S x$ is the graph convolution of the filter $h = \{h_k\}_{k=0}^{\infty}$ with the signal x

- Graph convolutions aggregate information growing from local to global neighborhoods
- Consider a signal x supported on a graph with shift operator S. Along with filter $\mathrm{h} = \{h_k\}_{k=0}^{K-1}$



- Graph convolution output $\Rightarrow \mathrm{y} = \mathrm{h} \star_{\mathrm{S}} \mathrm{x} = h_0 \mathrm{S}^0 \mathrm{x} + h_1 \mathrm{S}^1 \mathrm{x} + h_2 \mathrm{S}^2 \mathrm{x} + h_3 \mathrm{S}^3 \mathrm{x} + \ldots = \sum_{k=0}^{K-1} h_k \mathrm{S}^k \mathrm{x}$

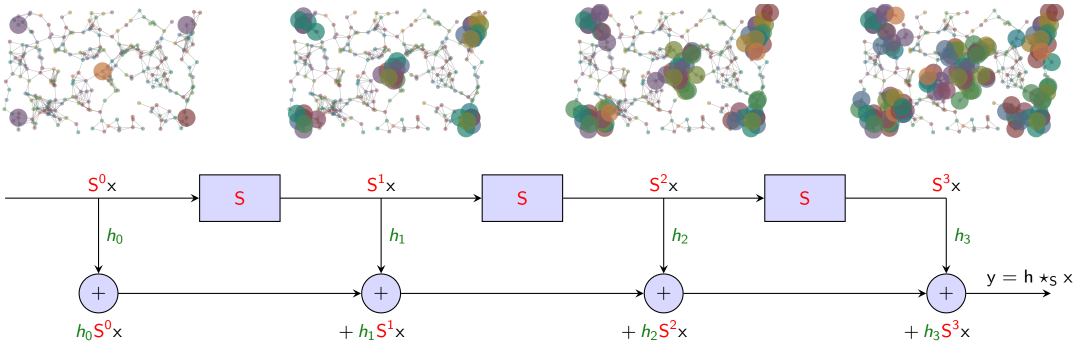
- The same filter $h = \{h_k\}_{k=0}^{\infty}$ can be executed in multiple graphs $\Rightarrow$ We can transfer the filter



- Graph convolution output $\Rightarrow y = h \star_S x = \; h_0 S^0 x + h_1 S^1 x + h_2 S^2 x + h_3 S^3 x + \ldots = \sum_{k=0}^{\infty} h_k S^k x$

- Output depends on the filter coefficients h, the graph shift operator S and the signal x

- A graph convolution is a weighted linear combination of the elements of the diffusion sequence
- Can represent graph convolutions with a shift register $\Rightarrow$ Convolution $\equiv$ Shift. Scale. Sum



Graph Frequency Response of Graph Filters

- Graph filters admit a pointwise representation when projected into the shift operator's eigenspace

Theorem (Graph frequency representation of graph filters)

Consider graph filter h with coefficients h_k, graph signal x and the filtered signal $\mathbf{y} = \sum_{k=0}^{\infty} h_k \mathbf{S}^k \mathbf{x}$.

The GFTs $\tilde{\mathbf{x}} = \mathbf{V}^H \mathbf{x}$ and $\tilde{\mathbf{y}} = \mathbf{V}^H \mathbf{y}$ are related by

$$\tilde{\mathbf{y}} = \sum_{k=0}^{\infty} h_k \Lambda^k \tilde{\mathbf{x}}$$

- The same polynomial but on different variables. One on S. The other on eigenvalue matrix Λ

Proof of Theorem

Proof: Since $\mathbf{S} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^H$, can write shift operator powers as $\mathbf{S}^k = \mathbf{V}\mathbf{\Lambda}^k\mathbf{V}^H$. Therefore filter output is

$$\mathbf{y} = \sum_{k=0}^{\infty} h_k \mathbf{S}^k \mathbf{x} = \sum_{k=0}^{\infty} h_k \mathbf{V}\mathbf{\Lambda}^k\mathbf{V}^H \mathbf{x}$$

- Multiply both sides by $\mathbf{V}^H$ on the left $\Rightarrow \mathbf{V}^H\mathbf{y} = \mathbf{V}^H \sum_{k=0}^{\infty} h_k \mathbf{V}\mathbf{\Lambda}^k\mathbf{V}^H \mathbf{x}$
- Copy and identify terms. Output GFT $\mathbf{V}^H\mathbf{y} = \tilde{\mathbf{y}}$. Input GFT $\mathbf{V}^H\mathbf{x} = \tilde{\mathbf{x}}$. Cancel out $\mathbf{V}^H\mathbf{V}$

$$\mathbf{V}^H\mathbf{y} = \mathbf{V}^H \sum_{k=0}^{\infty} h_k \mathbf{V}\mathbf{\Lambda}^k\mathbf{V}^H \mathbf{x} \qquad \Rightarrow \qquad \tilde{\mathbf{y}} = \sum_{k=0}^{\infty} h_k \mathbf{\Lambda}^k \tilde{\mathbf{x}}$$

■

- In the graph frequency domain graph filters are a diagonal matrices $\Rightarrow \tilde{\mathbf{y}} = \sum_{k=0}^{\infty} h_k \Lambda^k \tilde{\mathbf{x}}$

- Thus, graph convolutions are pointwise in the GFT domain $\Rightarrow \tilde{y}_i = \sum_{k=0}^{\infty} h_k \lambda_i^k \tilde{x}_i = \tilde{h}(\lambda_i)\tilde{x}_i$

Definition (Frequency Response of a Graph Filter)

Given a graph filter with coefficients $\mathbf{h} = \{h_k\}_{k=1}^{\infty}$, the graph frequency response is the polynomial

$$\tilde{h}(\lambda) = \sum_{k=0}^{\infty} h_k \lambda^k$$

Definition (Frequency Response of a Graph Filter)

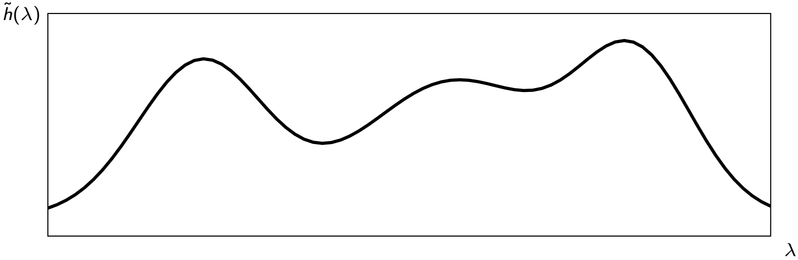
Given a graph filter with coefficients $\mathbf{h} = \{h_k\}_{k=1}^{\infty}$, the graph frequency response is the polynomial

$$\tilde{h}(\lambda) = \sum_{k=0}^{\infty} h_k \lambda^k$$

- Frequency response is the same polynomial that defines the graph filter $\Rightarrow$ but on scalar variable λ
- Frequency response is independent of the graph $\Rightarrow$ Depends only on filter coefficients
- The role of the graph is to determine the eigenvalues on which the response is instantiated

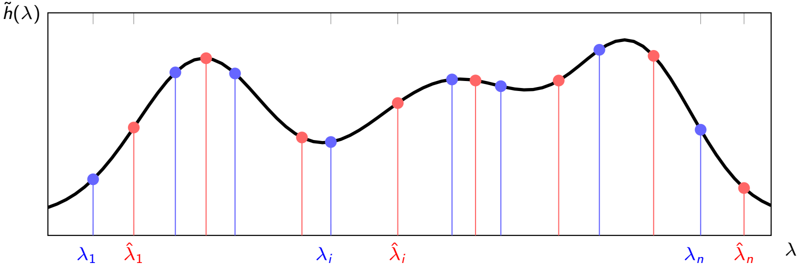
- Graph filter frequency response is a polynomial on a scalar variable λ $\Rightarrow \tilde{h}(\lambda) = \sum_{k=0}^{\infty} h_k \lambda^k$

- Completely determined by the filter coefficients $\mathrm{h} = \{h_k\}_{k=1}^{\infty}$. The Graph has nothing to do with it



- A given (another) graph instantiates the response on its given (different) specific eigenvalues λ_i

- Eigenvectors do not appear in the frequency response. They determine the meaning of frequencies.



Learning with Graph Signals

- Almost ready to introduce GNNs. We begin with a short discussion of learning with graph signals

Empirical Risk Minimization

- Machine learning (ML) on graphs (or not) $\equiv$ empirical risk minimization (ERM) on graphs (or not)
- In ERM we are given:
  - $\Rightarrow$ A training set $\mathcal{T}$ containing observation pairs $(\mathrm{x}, \mathrm{y}) \in \mathcal{T}$. Assume equal length $\mathrm{x}, \mathrm{y}, \in \mathbb{R}^n$.
  - $\Rightarrow$ A loss function $\ell(\mathrm{y}, \hat{\mathrm{y}})$ to evaluate the similarity between y and an estimate $\hat{\mathrm{y}}$
  - $\Rightarrow$ A function class $\mathcal{C}$
- Learning means finding function $\Phi^* \in \mathcal{C}$ that minimizes loss $\ell\Big(\mathrm{y}, \Phi(\mathrm{x})\Big)$ averaged over training set

$$\Phi^* = \underset{\Phi \in \mathcal{C}}{\operatorname{argmin}} \sum_{(\mathrm{x},\mathrm{y}) \in \mathcal{T}} \ell\Big(\mathrm{y}, \Phi(\mathrm{x}), \Big)$$

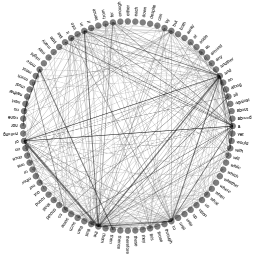
- We use $\Phi^*(\mathrm{x})$ to estimate outputs $\hat{\mathrm{y}} = \Phi^*(\mathrm{x})$ when inputs x are observed but outputs y are unknown

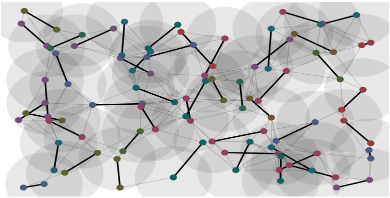
- In ERM, the function class $\mathcal{C}$ is the degree of freedom available to the system's designer

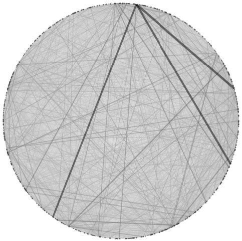
$$\Phi^* = \underset{\Phi \in \mathcal{C}}{\operatorname{argmin}} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{T}} \ell\Big(\mathbf{y}, \Phi(\mathbf{x})\Big)$$

- Designing a Machine Learning $\equiv$ finding the right function class $\mathcal{C}$

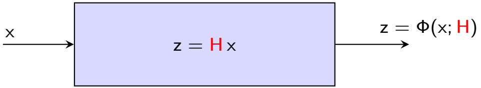
- Since we are interested in graph signals, graph convolutional filters are a good starting point







- Input / output signals $\mathbf{x}$ / $\mathbf{y}$ are graph signals supported on a common graph with shift operator $\mathbf{S}$
- Function class $\Rightarrow$ Generic Linea function mapping inputs to ooutputs $\Rightarrow$ $\Phi(\mathbf{x}) = \mathbf{H}\mathbf{x} = \Phi(\mathbf{x}; \mathbf{H})$

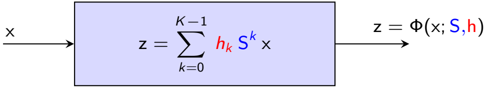


- Learn ERM solution restricted to graph filter class $\Rightarrow$ $\mathbf{h}^* = \underset{\mathbf{h}}{\operatorname{argmin}} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{T}} \ell\Big(\mathbf{y}, \Phi(\mathbf{x}; \mathbf{H})\Big)$

$\Rightarrow$ Optimization is over matrices $\mathbf{H}$. It does not take advantage of the graph

- Input / output signals x / y are graph signals supported on a common graph with shift operator S

- Function class $\Rightarrow$ graph filters of order K supported on S $\Rightarrow$ $\Phi(\mathbf{x}) = \sum_{k=0}^{K-1} h_k \mathbf{S}^k \mathbf{x} = \Phi(\mathbf{x};\mathbf{S},\mathbf{h})$



- Learn ERM solution restricted to graph filter class $\Rightarrow$ $\mathbf{h}^* = \underset{\mathbf{h}}{\operatorname{argmin}} \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{T}} \ell\Big(\mathbf{y}, \Phi(\mathbf{x};\mathbf{S},\mathbf{h})\Big)$

$\Rightarrow$ Optimization is over filter coefficients h with the graph shift operator S given

Learning Ratings in Recommendation Systems

- Formulate recommendation systems as ERM problems that predict ratings that users give to items

- In a recommendation system, we want to predict the rating a user would give to an item
- Collect ratings that some users give to some items $\Rightarrow$ These are rating histories
- Exploit product similarities to predict ratings of unseen user-item pairs
- Example 1 $\Rightarrow$ In an online store items are products and users are customers
- Example 2 $\Rightarrow$ In a movie repository items are movies and users are watchers

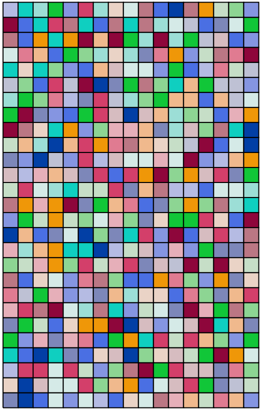
- For all items i and users u there exist ratings $\Rightarrow y_{ui}$

  $\Rightarrow$ User rating vector y_u has entries y_{ui}

- We only observe a subset of ratings $\Rightarrow x_{ui}$

  $\Rightarrow$ Observed user rating vector x_u has entries x_{ui}

  $\Rightarrow$ We assume $x_{ui} = 0$ if item i is unrated by user u



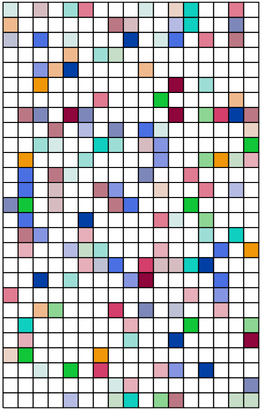
- For all items i and users u there exist ratings $\Rightarrow y_{ui}$

  $\Rightarrow$ User rating vector $\mathbf{y}_u$ has entries y_{ui}

- We only observe a subset of ratings $\Rightarrow x_{ui}$

  $\Rightarrow$ Observed user rating vector $\mathbf{x}_u$ has entries x_{ui}

  $\Rightarrow$ We assume $x_{ui} = 0$ if item i is unrated by user u



Product Ratings as Graph Signals

- Construct product similarity graph with weights w_{ij} represent likelihood of similar scores

- Interpret vector of ratings y_u of user u as a graph signal supported on the product similarity graph

- The observed ratings x_u of user u are a subsampling of this graph signal.

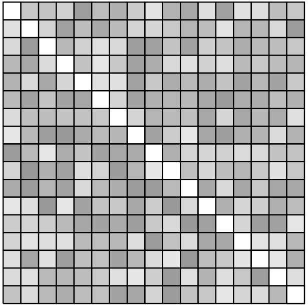
- Our goal is to learn to reconstruct the rating graph signal y_u from the observed ratings x_u

- Build similarity graph using available ratings. Use of expert knowledge is common as well

- Consider pair of products i and j. Restrict attention to set of users that rated both products $\Rightarrow \mathcal{U}_{ij}$

- Mean ratings restricted to users that rated products i and j

$$\mu_{ij} = \frac{1}{\#(\mathcal{U}_{ij})} \sum_{u \in \mathcal{U}_{ij}} x_{ui} \qquad \mu_{ji} = \frac{1}{\#(\mathcal{U}_{ij})} \sum_{u \in \mathcal{U}_{ji}} x_{uj}$$



- Similarity score = correlation restricted to users in $\mathcal{U}_{ij}$

$$\sigma_{ij} = \frac{1}{\#(\mathcal{U}_{ij})} \sum_{u \in \mathcal{U}_{ij}} \Big(x_{ui} - \mu_{ij}\Big)\Big(x_{uj} - \mu_{ji}\Big)$$

- Weights = normalized correlations $\Rightarrow w_{ij} = \sigma_{ij} \Big/ \sqrt{\sigma_{ii}\sigma_{jj}}$

Loss for Measuring Rating Prediction Quality

- Given observed ratings x_u the AI produces estimates $\Phi(\mathrm{x}_u)$. We want $\Phi(\mathrm{x}_u)$ to approximate y_u

$$\ell\Big(\mathrm{y}_u, \Phi(\mathrm{x}_u)\Big) = \frac{1}{2}\Big\|\, \mathrm{y}_u - \Phi(\mathrm{x}_u)\,\Big\|^2$$

- In reality, we want to predict the rating of specific item i

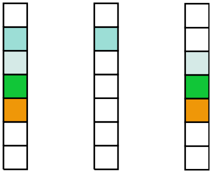
$$\ell\Big(\mathrm{y}_u, \Phi(\mathrm{x}_u)\Big) = \frac{1}{2}\Big(\mathrm{e}_i^T \mathrm{y}_u - \mathrm{e}_i^T \Phi(\mathrm{x}_u)\Big)^2$$

- Where e_i is a vector in the canonical basis $\Rightarrow (\mathrm{e}_i)_i = 1$, $(\mathrm{e}_i)_j = 0$ for $j \neq i$

- For each item i let $\mathcal{U}_i$ be the set of users that have rated i. Construct training pairs (x, y) with

$$\mathrm{y} = \left(\mathrm{e}_i^T \mathrm{x}_u\right)\mathrm{e}_i \qquad \mathrm{x} = \mathrm{x}_u - \mathrm{y} \qquad \text{for all } u \in \mathcal{U}_i, \ \text{for all } i$$

- Extract the rating x_{ui} of item i. Record into graph signal y
- Remove rating x_{ui} from x_u. Record to graph signal x
- Repeat for all users in the set $\mathcal{U}_i$ of users that rated i
- Repeat for all items $\Rightarrow$ Training set $\mathcal{T}$



- Parametrized AI $\Phi(x_u) = \Phi(x_u; \mathcal{H})$. We want to find solution of the ERM problem

$$\mathcal{H}^* = \underset{\mathcal{H}}{\operatorname{argmin}} \sum_{(x,y)\in\mathcal{T}} \Big(e_i^T y - e_i^T \Phi(x; \mathcal{H}) \Big)^2$$

- A bad idea $\Rightarrow$ Linear regression with a generic linear function.

- A good idea $\Rightarrow$ Graph filters.

Learning Ratings with Graph Filters

- We use graph filters to learn ratings in recommendation systems

- We contrast with the use of linear regression with a generic linear function

Movie Ratings Dataset

- Use MovieLens-100k as benchmark $\Rightarrow 10^6$ ratings given by $U = 943$ users to $M = 1,682$ movies

- The ratings for each movie are between 1 and 5. From one star to five starts

- Train and test several machine learning parametrizations.

Empirical Risk Minimization

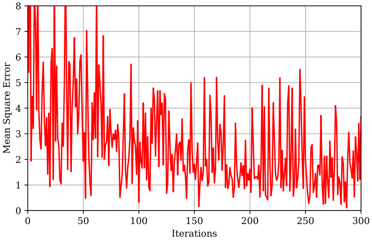
- We predict ratings using AI that results from solving the ERM problem

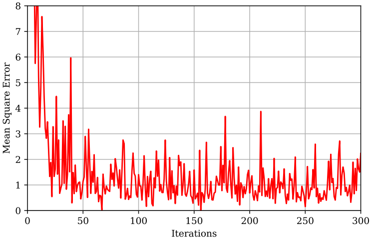
$$\mathcal{H}^* \;=\; \underset{\mathcal{H}}{\text{argmin}} \sum_{(x,y)\in\mathcal{T}} \Big(\mathbf{e}_i^T \mathbf{y} - \mathbf{e}_i^T \Phi(\mathbf{x}; \mathcal{H}) \Big)^2$$

- Parameterizations that ignore data structure $\Rightarrow$ Linear regression. Fully connected NNs

- Parameterizations that leverage data structure $\Rightarrow$ Graph filters. Graph NNs

- Linear regression reduces training MSE to about 2. Quite bad for ratings that vary from 0 to 5
- Graph filter reduces training MSE to about 1. Not too good. Humans are not that predictable

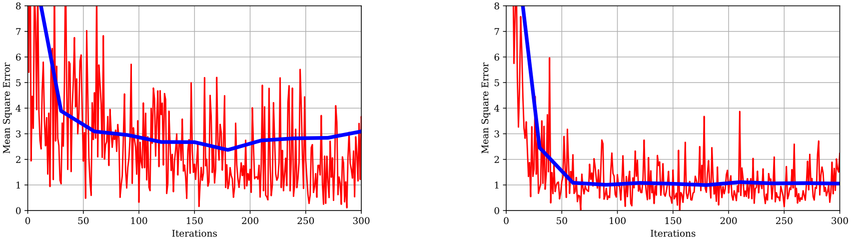




- Graph filter outperforms linear regression $\Rightarrow$ Leverages underlying permutation symmetries

- Linear regression works even worse in the test set

- The test MSE of the graph filter is about the same as the training MSE. It generalizes



- Graph filter outperforms linear regression $\Rightarrow$ Leverages underlying permutation symmetries

Permutation Equivariance of Graph Filters

- We will show that graph convolutional filters are equivariant to permutations

Permutation Matrices

Definition (Permutation matrix)

A square matrix $\mathbf{P}$ is a permutation matrix if it has binary entries so that $\mathbf{P} \in \{0, 1\}^{n \times n}$ and it further satisfies $\mathbf{P}\mathbf{1} = \mathbf{1}$ and $\mathbf{P}^T\mathbf{1} = \mathbf{1}$.

- The product $\mathbf{P}^T\mathbf{x}$ reorders the entries of the vector $\mathbf{x}$.
- The product $\mathbf{P}^T\mathbf{S}\mathbf{P}$ is a consistent reordering of the rows and columns of $\mathbf{S}$

Permutation Matrices

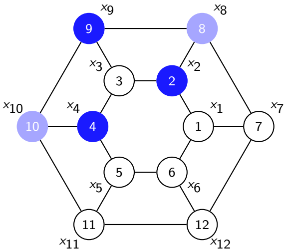
Definition (Permutation matrix)

A square matrix P is a permutation matrix if it has binary entries so that $\mathsf{P} \in \{0,1\}^{n \times n}$ and it further satisfies $\mathsf{P}\mathbf{1} = \mathbf{1}$ and $\mathsf{P}^T\mathbf{1} = \mathbf{1}$.

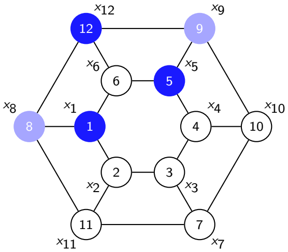
- Since $\mathsf{P}\mathbf{1} = \mathsf{P}^T\mathbf{1} = \mathbf{1}$ with binary entries $\Rightarrow$ Exactly one nonzero entry per row and column of P
- Permutation matrices are unitary $\Rightarrow \mathsf{P}^T\mathsf{P} = \mathsf{I}$. Matrix P^T undoes the reordering of matrix P

- If $(\mathbf{S}, \mathbf{x})$ is a graph signal, $(\mathbf{P}^T\mathbf{S}\mathbf{P}, \mathbf{P}^T\mathbf{x})$ is a relabeling of $(\mathbf{S}, \mathbf{x})$. Same signal. Different names

Graph signal $\mathbf{x}$ Supported on $\mathbf{S}$



Graph signal $\hat{\mathbf{x}} = \mathbf{P}^T\mathbf{x}$ supported on $\hat{\mathbf{S}} = \mathbf{P}^T\mathbf{S}\mathbf{P}$



- Processing should be label-independent $\Rightarrow$ Permutation equivariance of graph filters and GNNs

- Graph filter $H(S)$ is a polynomial on shift operator S with coefficients h_k. Outputs given by

$$H(S)x = \sum_{k=0}^{K-1} h_k S^k x$$

- We consider running the same filter on (S, x) and permuted (relabeled) $(\hat{S}, \hat{x}) = (P^T S P, P^T x)$

$$H(S)x = \sum_{k=0}^{K-1} h_k S^k x \qquad\qquad H(\hat{S})\hat{x} = \sum_{k=0}^{K-1} h_k \hat{S}^k \hat{x}$$

- Filter $H(S)x \Rightarrow$ Coefficients h_k. Input signal x. Instantiated on shift S
- Filter $H(\hat{S})\hat{x} \Rightarrow$ Same Coefficients h_k. Permuted Input signal $\hat{x}$. Instantiated on permuted shift $\hat{S}$

Theorem (Permutation equivariance of graph filters)

Consider consistent permutations of the shift operator $\hat{S} = P^T S P$ and input signal $\hat{x} = P^T x$. Then

$$H(\hat{S})\,\hat{x} \;=\; H\big(P^T S P\big)\,\big(P^T x\big) \;=\; P^T H(S)\,x$$

- Graph filters are equivariant to permutations $\Rightarrow$ Permute input and shift $\equiv$ Permute output

Proof: Write filter output in polynomial form. Use permutation definitions $\hat{\mathbf{S}} = \mathbf{P}^T\mathbf{S}\mathbf{P}$ and $\hat{\mathbf{x}} = \mathbf{P}^T\mathbf{x}$

$$\mathbf{H}(\hat{\mathbf{S}})\hat{\mathbf{x}} = \sum_{k=0}^{K-1} h_k \hat{\mathbf{S}}^k \hat{\mathbf{x}} = \sum_{k=0}^{K-1} h_k \Big(\mathbf{P}^T\mathbf{S}\mathbf{P}\Big)^k \mathbf{P}^T\mathbf{x}$$

- In the powers $\Big(\mathbf{P}^T\mathbf{S}\mathbf{P}\Big)^k$, $\mathbf{P}$ and $\mathbf{P}^T$ undo each other $(\mathbf{P}^T\mathbf{P} = \mathbf{I})$ $\Rightarrow \Big(\mathbf{P}^T\mathbf{S}\mathbf{P}\Big)^k = \mathbf{P}^T\Big(\mathbf{S}\Big)^k\mathbf{P}$

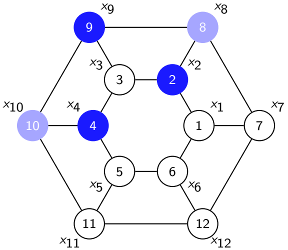
- Substitute this into filter's output expression. Cancel remaining $\mathbf{P}\mathbf{P}^T = \mathbf{I}$ product. Factor $\mathbf{P}^T$

$$\mathbf{H}(\hat{\mathbf{S}})\hat{\mathbf{x}} = \sum_{k=0}^{K-1} h_k \mathbf{P}^T\mathbf{S}^k\mathbf{P}\mathbf{P}^T\mathbf{x} = \sum_{k=0}^{K-1} h_k \mathbf{P}^T\mathbf{S}^k\mathbf{I}\mathbf{x} = \mathbf{P}^T\sum_{k=0}^{K-1} h_k \mathbf{S}^k\mathbf{x} = \mathbf{P}^T\mathbf{H}(\mathbf{S})\mathbf{x} \qquad \blacksquare$$

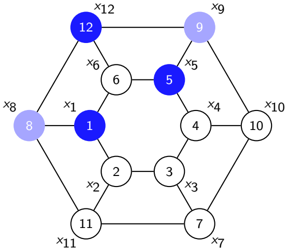
- We request signal processing independent of labeling $\Rightarrow$ Graph filters fulfill this request

$\Rightarrow$ Permute input and shift $\equiv$ Relabel input $\Rightarrow$ Permute output $\equiv$ Relabel output

Graph signal $\mathbf{x}$ Supported on $\mathbf{S}$



Graph signal $\hat{\mathbf{x}} = \mathbf{P}^T\mathbf{x}$ supported on $\hat{\mathbf{S}} = \mathbf{P}^T\mathbf{S}\mathbf{P}$

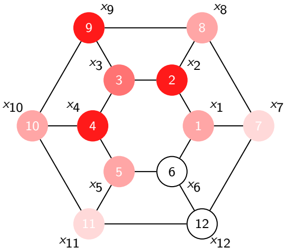


Signal Processing with Graph Filters is Independent of Labeling

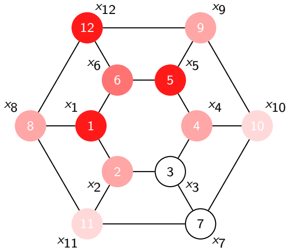
- We request signal processing independent of labeling $\Rightarrow$ Graph filters fulfill this request

$\Rightarrow$ Permute input and shift $\equiv$ Relabel input $\Rightarrow$ Permute output $\equiv$ Relabel output

Filter's output $\mathbf{H}(\mathbf{S})\mathbf{x}$ Supported on $\mathbf{S}$



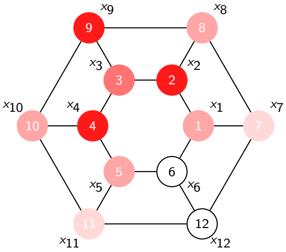
Filter's Output $\mathbf{H}(\hat{\mathbf{S}})\hat{\mathbf{x}}$ supported on $\hat{\mathbf{S}}$



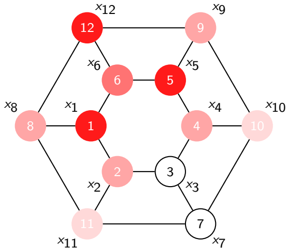
- We request signal processing independent of labeling $\Rightarrow$ Graph filters fulfill this request

$\Rightarrow$ Permute input and shift $\equiv$ Relabel input $\Rightarrow$ Permute output $\equiv$ Relabel output

Filter's output $\mathsf{H}(\mathsf{S})\mathsf{x}$ Supported on S

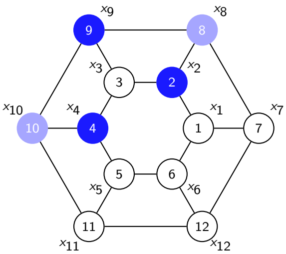


Equivariance theorem $\Rightarrow \mathsf{H}(\hat{\mathsf{S}})\hat{\mathsf{x}} = \mathsf{P}^T\mathsf{H}(\mathsf{S})\mathsf{x}$

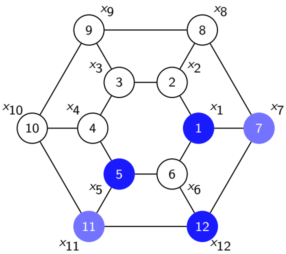


- Equivariance to permutations allows GNNs to exploit symmetries of graphs and graph signals
- By symmetry we mean that the graph can be permuted onto itself $\Rightarrow \mathsf{S} = \mathsf{P}^T\mathsf{SP}$
- Equivariance theorem implies $\Rightarrow \mathsf{H}(\mathsf{S})(\mathsf{P}^T\mathsf{x}) = \mathsf{H}(\mathsf{P}^T\mathsf{SP})(\mathsf{P}^T\mathsf{x}) = \mathsf{P}^T\mathsf{H}(\mathsf{S})(\mathsf{x})$

From observing x supported on S

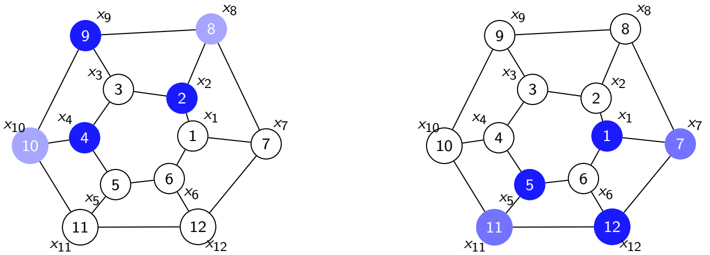


Learn to process $\mathsf{P}^T\mathsf{x}$ supported on $\mathsf{S} = \mathsf{P}^T\mathsf{SP}$



Symmetry is Rare but Quasi-Symmetry is Common

- Graph not symmetric but close to symmetric ⇒ perturbed version of a permutation of itself



- It can be shown that graph filters can lack stability to deformations ⇒ Graph Neural Networks

⇒ But this is a story for another day ⇒ Register for ESE 514. Or visit gnn.seas.upenn.edu