Graph Filters and Its Parameterization

Ph.D candidate in Computational Science and Engineering
Yonsei Univ.

Jin-Duk Park

This slides are made by referring to the following materials:

- Youtube video: "**Xiaowen Dong: Learning graphs from data: A signal processing perspective",** https://www.youtube.com/watch?v=2ds4A11DSOw&list=PLe0J3_6vYq7tgrMvHC6M7w6Ncqsy38flm
- Shuman, David I., et al. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." IEEE signal processing magazine 30.3 (2013): 83-98.
- Bianchi, Filippo Maria, et al. "Graph neural networks with convolutional arma filters." IEEE transactions on pattern analysis and machine intelligence 44.7 (2021): 3496-3507.
- Tremblay, Nicolas, Paulo Gonçalves, and Pierre Borgnat. "Design of graph filters and filterbanks." Cooperative and Graph Signal Processing. Academic Press, 2018. 299-324.
- Kipf, Thomas N., and Max Welling. "Semi-supervised classification with graph convolutional networks." ICLR 2017

Reading group material

Filtering in Data

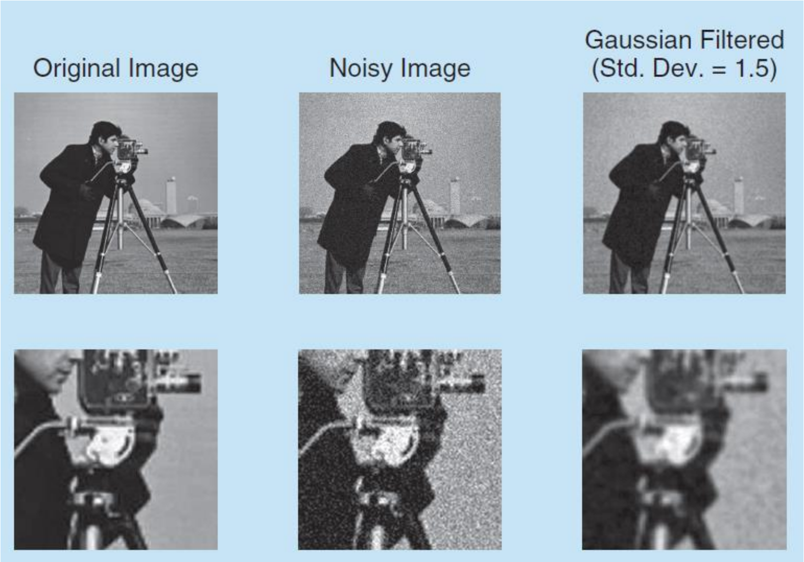
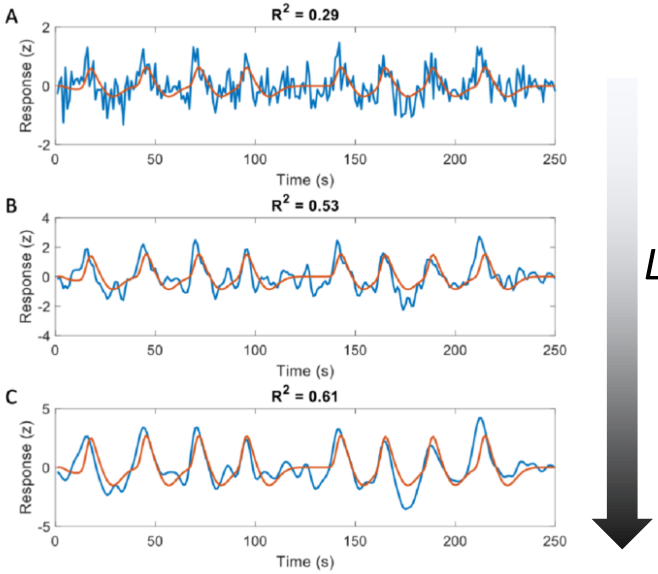


Image data

ID	Age	Gender	Preferred cola	Brand Ratings: Coca-Cola	Diet Coke	Coke Zero	Pepsi	Diet Pepsi	Pepsi Max
1	25 to 29	Female	Pepsi Max	2	5	2	3	1	4
2	45 to 49	Male	Pepsi Max	5	1	5	5	3	4
3	25 to 29	Female	Diet Coke	5	4	2	3	1	1
4	25 to 29	Female	Coca-Cola	4	2	2	2	2	2
5	55 to 64	Female	Diet Coke	3	4	3	3	4	2
6	55 to 64	Female	Diet Pepsi	3	3	3	3	4	4
7	50 to 54	Female	Coke Zero	2	3	5	2	2	2
8	35 to 39	Female	Coca-Cola	4	2	5	3	2	5
9	65 or more	Male	Diet Pepsi	5	5	3	5	5	3
10	45 to 49	Female	Coke Zero	4	4	4	5	5	3
11	45 to 49	Male	Coca-Cola	4	1	1	4	1	1
12	55 to 64	Male	Coca-Cola	5	2	2	5	2	2
13	55 to 64	Male	Coca-Cola	5	2	2	3	2	2
14	30 to 34	Male	Pepsi Max	3	2	5	3	3	5
15	65 or more	Female	Diet Pepsi	2	4	2	5	4	2

Tabular data



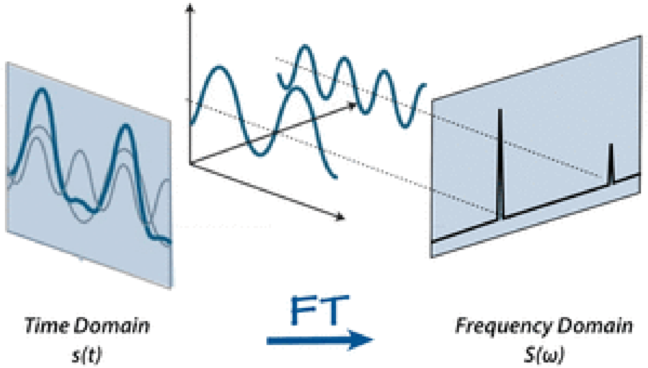
Low-pass filtered

Time-series data

<u>Data filtering</u> is a prominent technique that can be applicable for various types of data

https://www.researchgate.net/figure/Low-pass-filtering-the-time-series-improves-goodness-of-fit-quantified-by-R-2-The-same_fig1_336026361

Fourier Transform

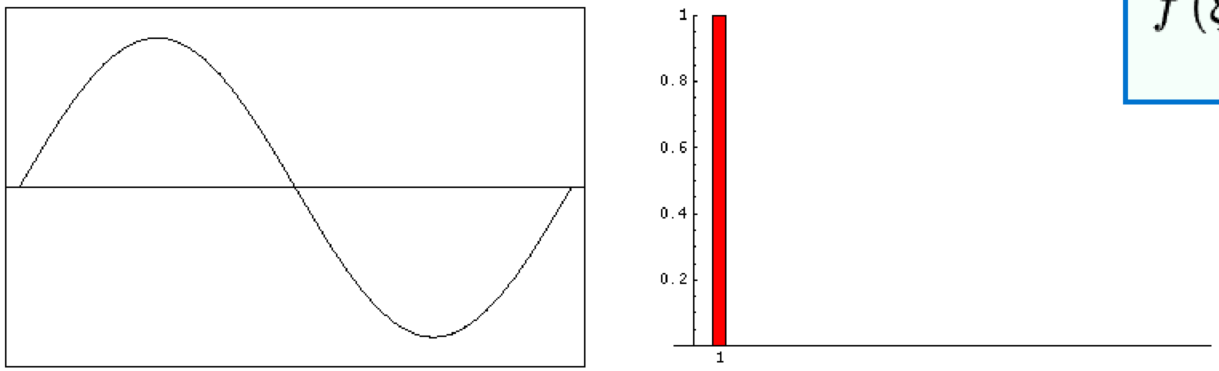


- **Fourier Transform**

A mathematical procedure a signal in the time domain to a complex number in the frequency domain

Fourier transform

$$\hat{f}(\xi) = \int_{-\infty}^{\infty} f(x)\, e^{-i2\pi\xi x}\, dx.$$



https://mriquestions.com/fourier-transform-ft.html

Graph Spectral Decomposition

- **Graph** is a flexible model for representing data in many problems
- **Fourier Transform**

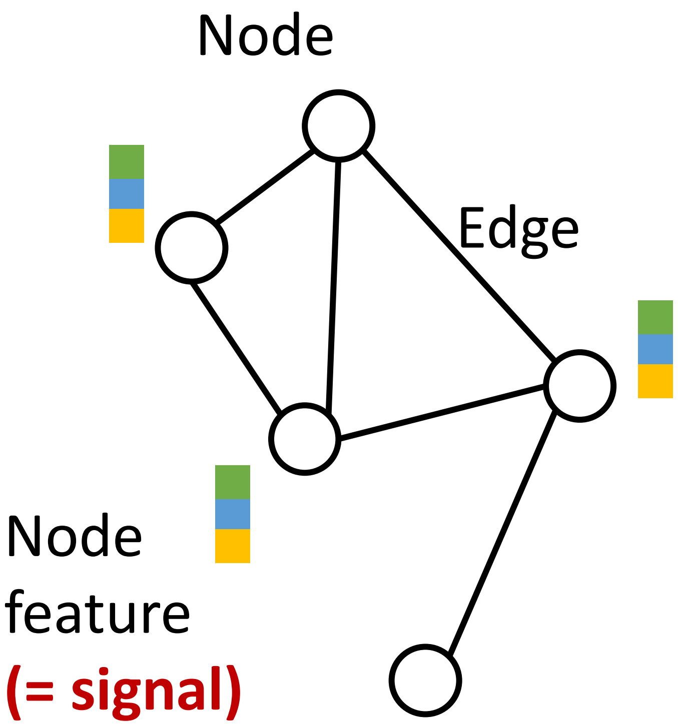
A mathematical procedure a signal in the time domain to a complex number in the frequency domain

$$f(x) \overset{\mathcal{F}}{\longleftrightarrow} \hat{f}(\xi)$$

Graph Fourier Transform (GFT)

How to analyze **graph in frequency domain**, via GFT?

Graph and Graph Signals



- Nodes (vertices), edges

$$G = (V, E)$$

- **Graph signal**

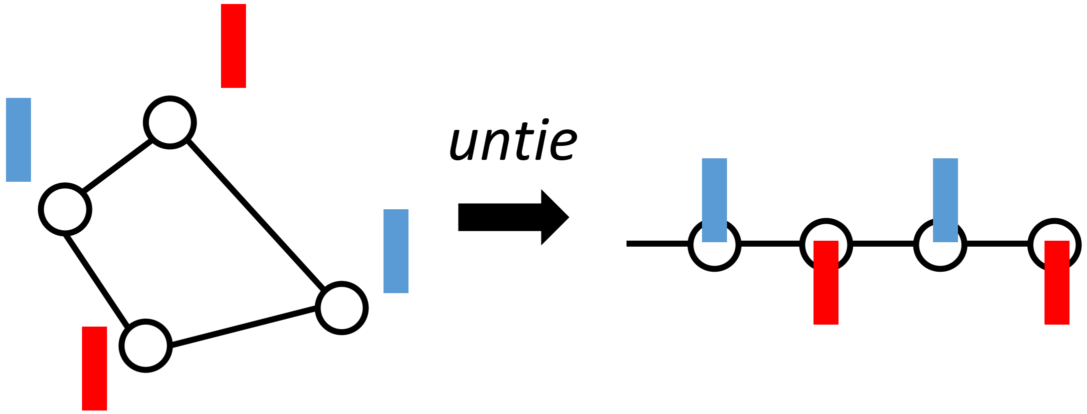
$$\mathbf{x} = x_1, x_2, \ldots$$

Graph Frequency

- **Graph signal**

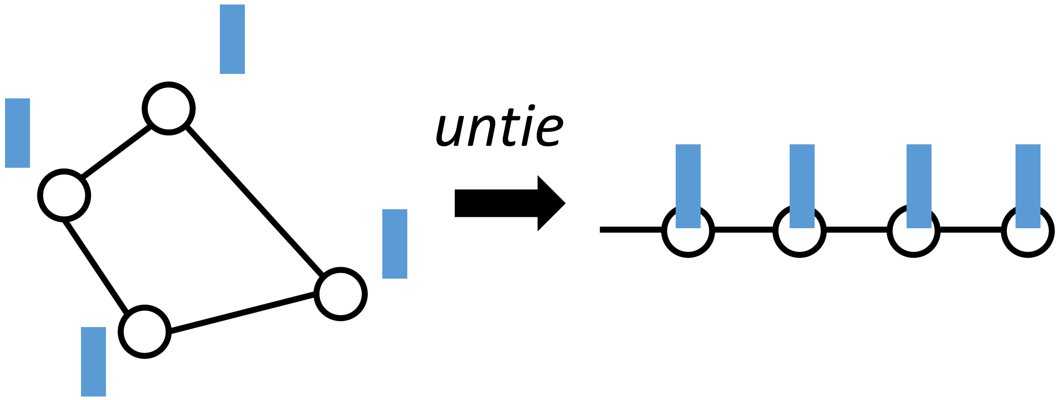
$$\mathbf{x} = x_1, x_2, \ldots$$

High Frequency



Connected components have different signals

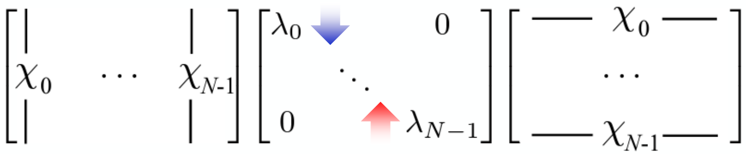
Low Frequency



Connected components have same (similar) signals

Reference Operator for GFT

** We only consider real-valued component of eigenvalues now*



- **Graph Spectral Decomposition**

  - Reference operator should be **diagonalizable**

$$\mathbf{R} \in \mathbb{R}^{N \times N}$$

Reference operator

Eigendecomposition

$$\mathbf{R} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}$$

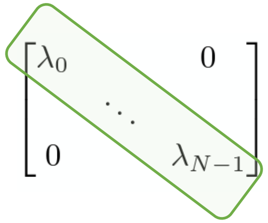
Definition 1.2 (Graph Fourier Transform). *For a given diagonalizable reference operator* $\mathbf{R} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}$ *acting on a graph* $\mathcal{G}$*, the GFT of a graph signal* $\mathbf{x} \in \mathbb{R}^N$ *is:*

$$\mathbb{F}_{\mathcal{G}}\,\mathbf{x} \doteq \hat{\mathbf{x}} \doteq \mathbf{U}^{-1}\mathbf{x}. \quad (2)$$

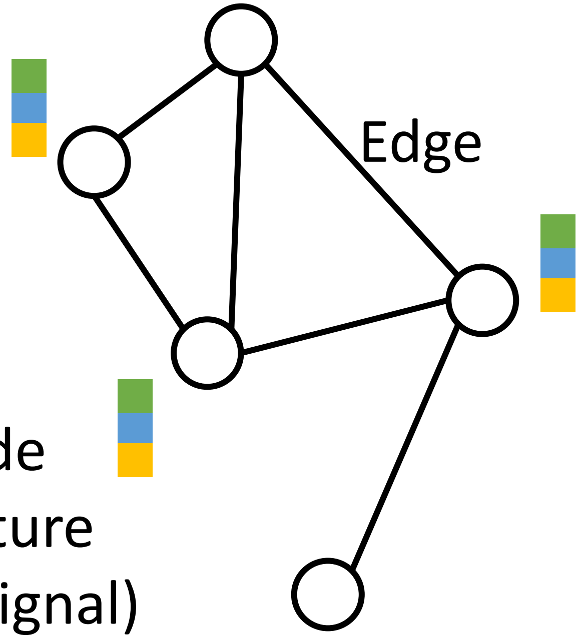
- **Graph Frequency**

Definition 1.3 (Graph frequency). *Let* $\mathbf{R}$ *be a reference operator. If its eigenvalues are real, the generalized graph frequency* ν *of a graph Fourier mode* $\mathbf{u}_k$ *is:*

$$\nu(\mathbf{u}_k) = \lambda_k \geqslant 0. \quad (10)$$



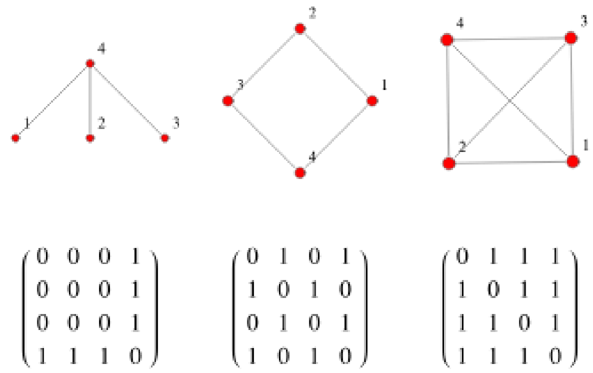
Algebraic representations of Graph



- **Adjacency matrix** $\mathbf{A}$

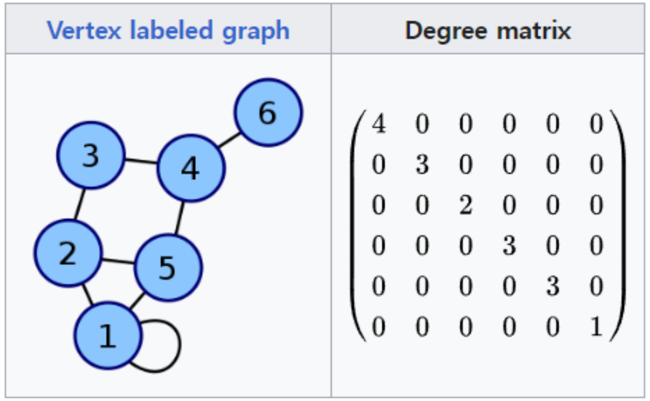
  - each entry: edge weight
  - if zero: not connected

https://mathworld.wolfram.com/AdjacencyMatrix.html



- **Degree matrix** $\mathbf{D} = diag(d_i)$

  - Sum of edge weights for each node



https://en.wikipedia.org/wiki/Degree_matrix

- **Combinatorial Laplacian matrix**

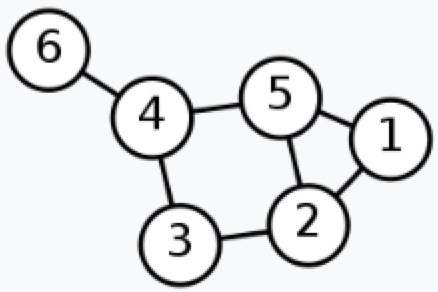
$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

  - Measure difference between own signal and its neighbors
  - Widely selected as a **reference operator**

Reference Operator in Graph

- **Combinatorial Laplacian matrix**

$$\mathbf{L} = \mathbf{D} - \mathbf{A}$$

Labelled graph	Degree matrix	Adjacency matrix	Laplacian matrix
	$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$

https://en.wikipedia.org/wiki/Laplacian_matrix

Total variation of the whole graph can be measured by

$$\frac{1}{2}\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{N}_i}(f(j)-f(i))^2 = f^T\mathbf{L}f$$

Other Possible Choices of Reference Operator

Other possible choices of $\mathbf{R}$

- *Normalized Laplacian*

$$\mathbf{L}_n = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$$

All eigenvalues lie between 0 and 2

- *Adjacency matrix*

$$\mathbf{A}$$

Eigenbasis $\boldsymbol{U}$ of $\boldsymbol{A}$ and the eigenbasis of the deformed Laplacian, $\boldsymbol{L_d} = \boldsymbol{I} - \frac{\boldsymbol{A}}{||\boldsymbol{A}||_2}$, are the same

- *Random walk Laplacian*

$$\mathbf{L}_{rw} = \mathbf{D}^{-1}\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$$

Same eigenvalues with $\boldsymbol{L_n}$, but the Fourier basis are not orthonomal

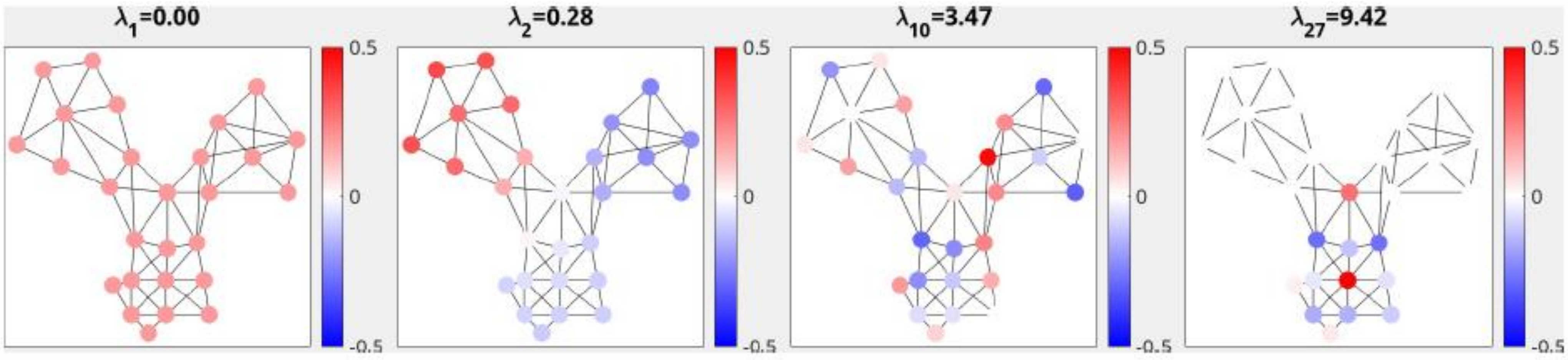
- *ETC (Consensus operator, geometric Laplacian, deformed Laplacian, …)*

Reference Operator for GFT

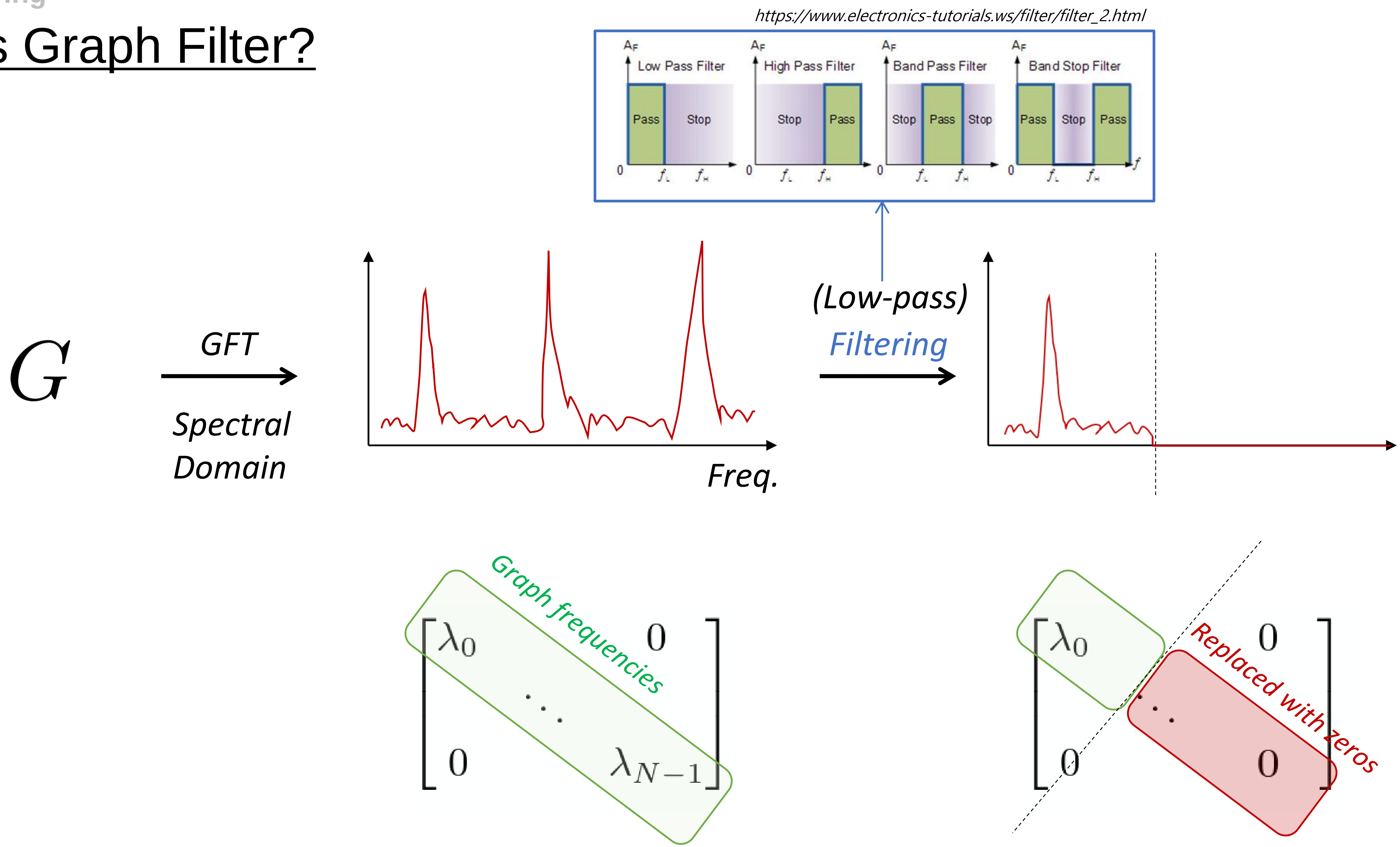
$$\mathbf{L} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^{-1}$$

$$\begin{bmatrix} | & & | \\ \chi_0 & \cdots & \chi_{N\text{-}1} \\ | & & | \end{bmatrix} \begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \begin{bmatrix} — & \chi_0 & — \\ & \cdots & \\ — & \chi_{N\text{-}1} & — \end{bmatrix}$$

[Ortega et al., 2018]

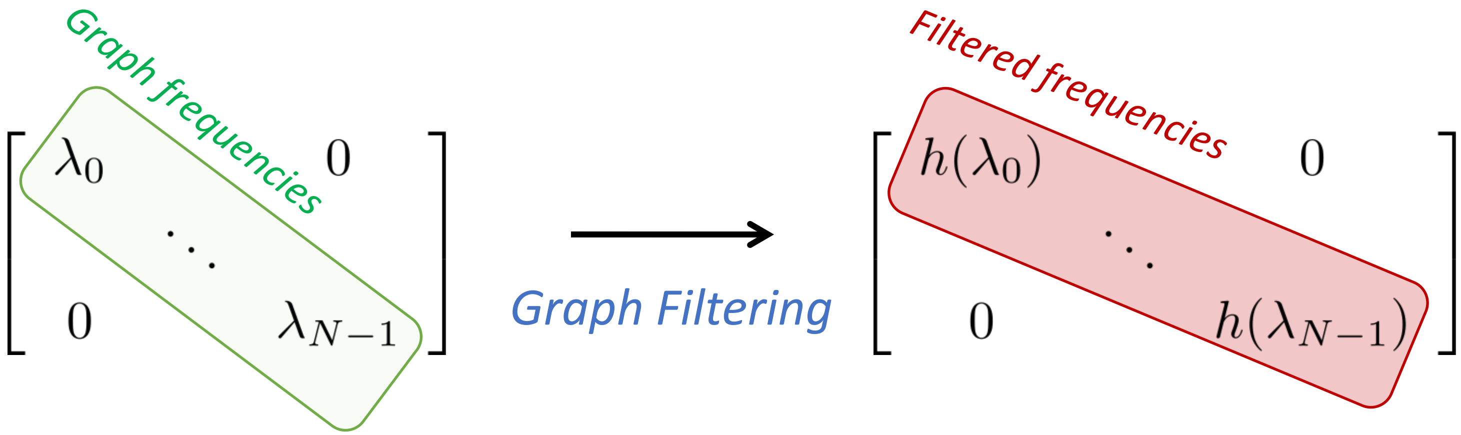


What is Graph Filter?



General Definition of Graph Filter

We can generally design any filter with *function.*



Definition (Graph Filter) $\mathbf{H}$

$$\mathbf{R} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$$

$$h : \mathbb{R}^+ \to \mathbb{R}$$

$$\nu \to h(\nu).$$

$$\mathbf{H} = \mathbf{U}h(v(\mathbf{\Lambda}))\mathbf{U}^{-1}$$

Examples of Filters

INDEX

- Constant filter
- *Ideal low-pass with cut-off frequency*
- *Kronecker delta*
- *Heat kernel*

$$h(\nu) = c$$

$$\begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \xrightarrow{\text{Graph Filtering}} \begin{bmatrix} c & & 0 \\ & \ddots & \\ 0 & & c \end{bmatrix}$$

All frequencies are allowed to pass

Usecase (response by ChatGPT)

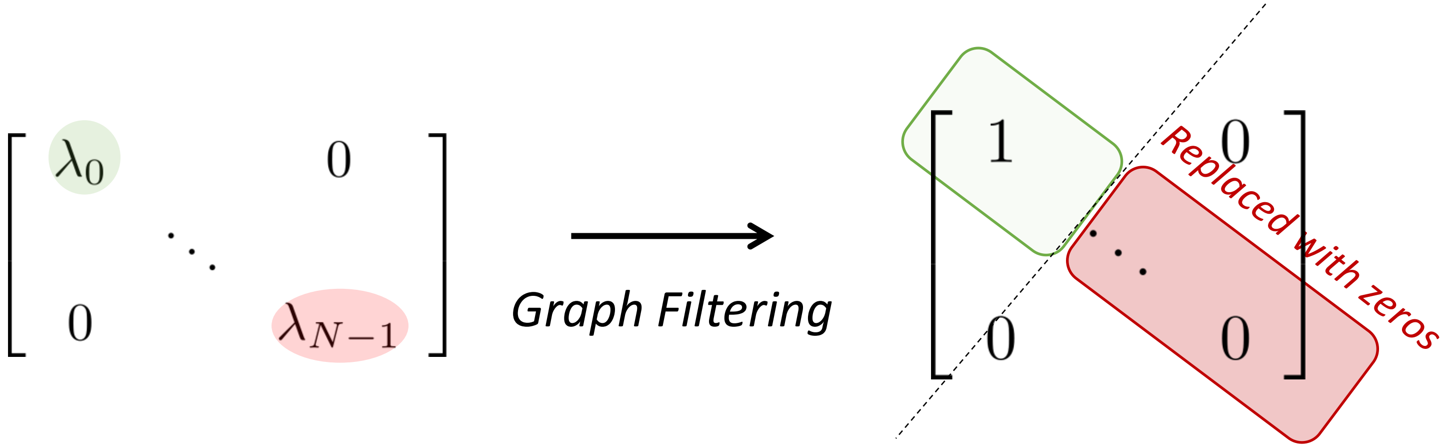
1) **Graph reconstruction**: in case graph signals are incomplete or corrupted (help infer missing signals)
2) **Graph denoising:** Suppress high-frequency and sharpen low-frequency

Examples of Filters

INDEX

- Constant filter
- *Ideal low-pass with cut-off frequency*
- *Kronecker delta*
- *Heat kernel*

$$h(\nu) = 1 \text{ if } \nu \leq \nu_c \text{ and } 0 \text{ otherwise}$$



Usecase

1) **Recommendation**: Application where low-frequency is important (similar connectivity pattern)
2) **Graph Denoising**
3) **Graph Convolutional Networks**

Examples of Filters

INDEX

$$h(\nu) = \delta_{\nu,\nu^*}$$

Frequencies with certain conditions (ν^*) are allowed pass

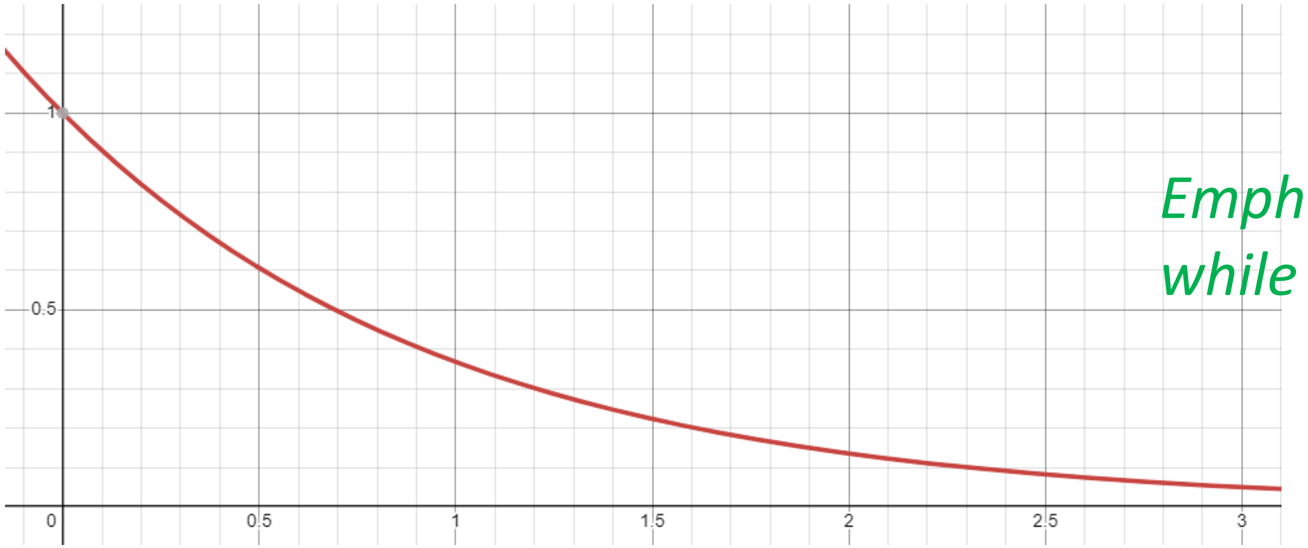
$$\begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \xrightarrow[\textit{Graph Filtering}]{} \begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & 0 \end{bmatrix}$$

Examples of Filters

INDEX

$$h(\nu) = e^{-\frac{\nu}{\nu_0}}$$

$$\begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \xrightarrow{\textit{Graph Filtering}} \begin{bmatrix} e^{-\frac{\lambda_0}{\nu_0}} & & 0 \\ & \ddots & \\ 0 & & e^{-\frac{\lambda_{N-1}}{\nu_0}} \end{bmatrix}$$



Emphasize low-frequencies, while weaken high-freqeuncies

Learning Filters by Parameterization

Filter Parameterization

- However, **finding optimal filters** for a given task is **difficult**

What if we **learn** graph filter?

- **Parameterizing** graph filter

$$\begin{bmatrix} \lambda_0 & & 0 \\ & \ddots & \\ 0 & & \lambda_{N-1} \end{bmatrix} \rightarrow \begin{bmatrix} \theta_0 & & 0 \\ & \ddots & \\ 0 & & \theta_{N-1} \end{bmatrix}$$

Learning Filters by Parameterization

Spectral Graph Convolutions

$$h * x = \mathbf{U} h(\mathbf{\Lambda}) \mathbf{U}^{-1} x$$

- Eigendecomposition is expansive for large graph
- Mutilpication with $\boldsymbol{U}$: $O(N^2)$

↓

***Approximated** by a truncated expansion of*
***Chebyshev polynomials** up to K-th order (Hammond et al., 2011)*

$$h_\theta * x = \sum_{k=0}^{K} \theta_k T_k(\hat{L}) x$$

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$$

$$T_0(x) = 1, T_1(x) = x$$

→ *Now, it only requires* $O(|E|)$!

GCN: Layer-wise Linear Model

$$h_\theta * x = \sum_{k=0}^{K} \theta_k T_k(\hat{L})x$$

K-localized *since it is a K-th order polynomial in the Laplacian*

1. *Linearize with **K=1***
2. *Instead, **stack multiple layers** (deeper model)*

$$h_\theta * x = \theta_0 x + \theta_1 (L - I_N) x$$

Integrate two free parameters, for practicallity

$$h_\theta * x = \theta(I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}})x$$

** Renormalization tick*

$$I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$$

$$\tilde{A} = A + I_N$$

1. *Eigenvalues in [0,2] -> stacking layer -> unstable*
2. *Renormalization trick*

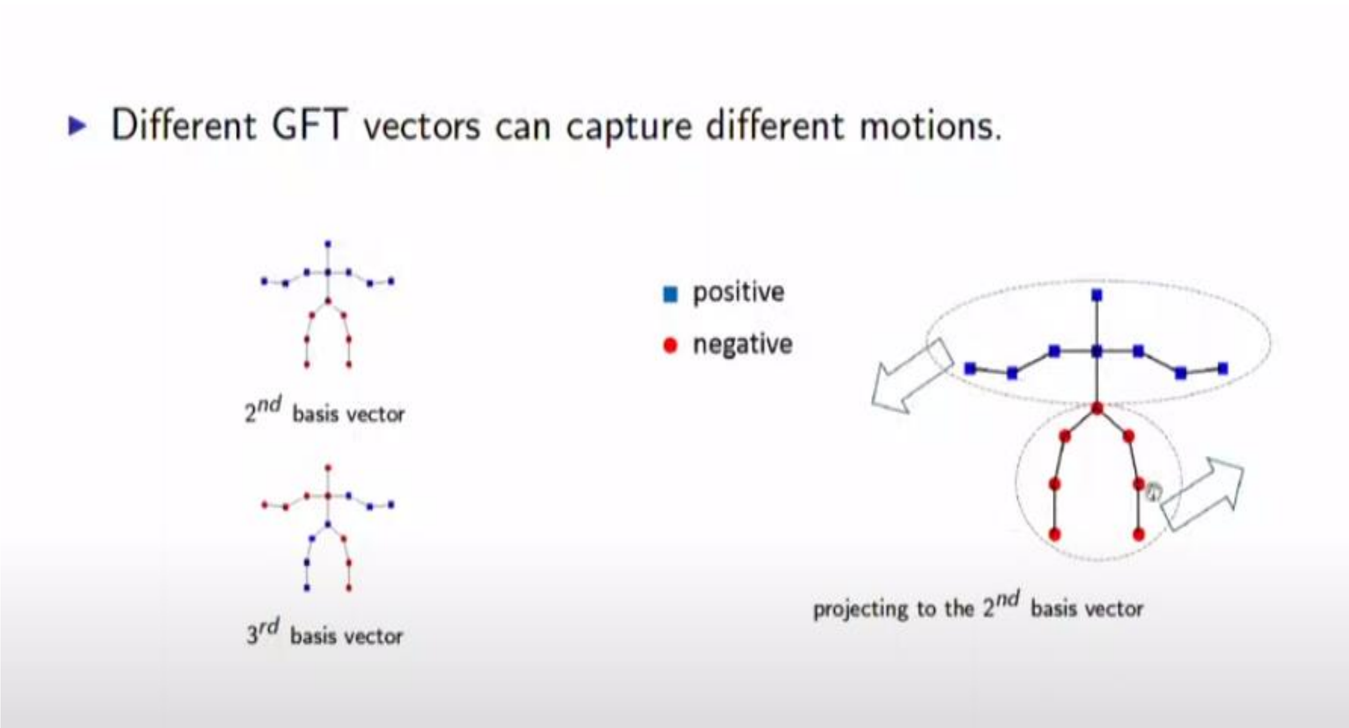
$$Z = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} X \Theta$$

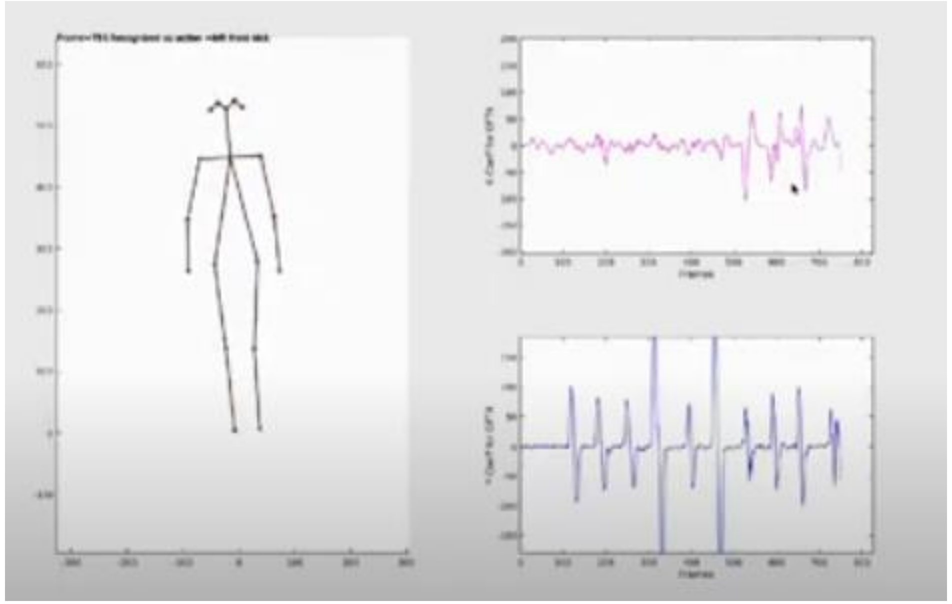
Become efficient operation with $O(|E|)$

Graph Convolutional Networks

GSP Application example: motion capturing

GSP can be used for motion capturing

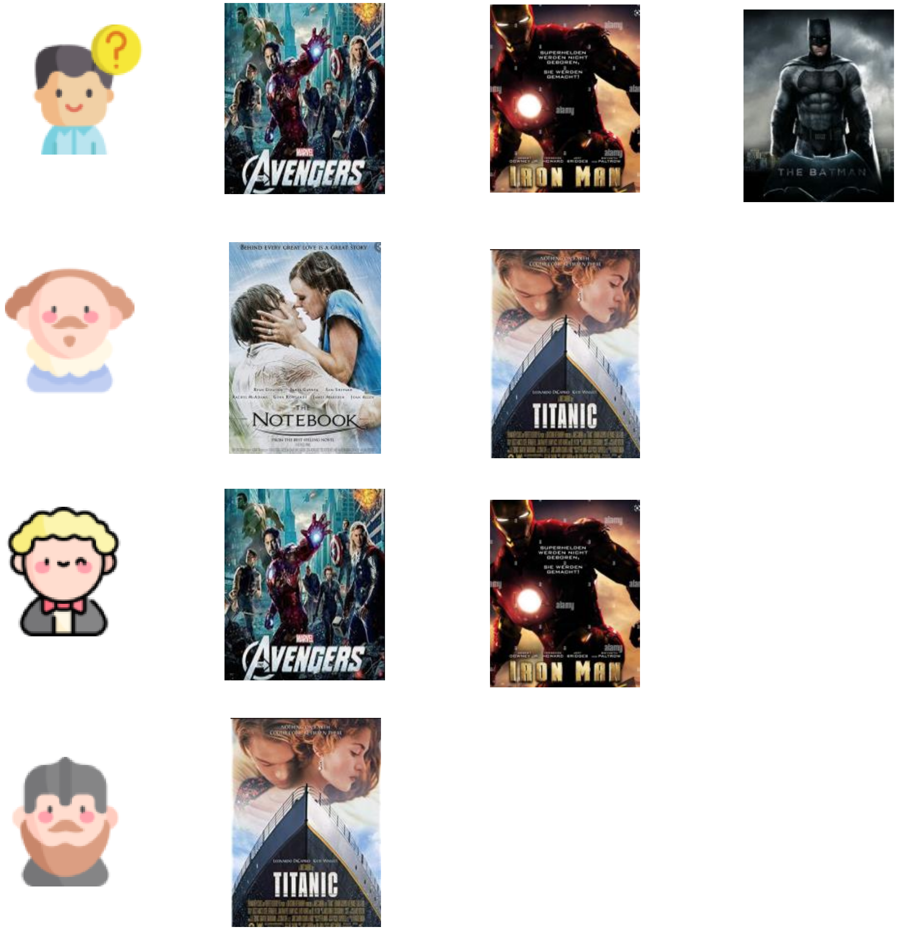




https://www.youtube.com/watch?v=296S-zh3WnU&list=PLeMXQeoNccrjCFyCUUpvAkCcsREa88WFQ:
2017-2018 ECE Distinguished Lecture Series - Antonio Ortega, University of Southern California

GSP Application example: recommendation

GSP can be used for recommendation (CF)



CF: Recommendation based on **similarity of historical interactions.**

GF-CF [Shen et al., 2021]

$$s_u = r_u \left(\tilde{R}^T \tilde{R} + \alpha D_I^{-\frac{1}{2}} \bar{U} \bar{U}^T D_I^{\frac{1}{2}} \right)$$

➡ Low-pass filtered (Top-k eigenvectors)

Takeaways

- Graph signal processing is a prominent area for processing irregular graph domain
- GNNs can be viewed as learning graph filter, by its parameterization
- GSP can be applied in various domains includes recommender system (next topic)

"Success is not final, failure is not fatal:
it is the courage to continue that counts."
- Winston Churchill

Thank you!

jindeok6@yonsei.ac.kr