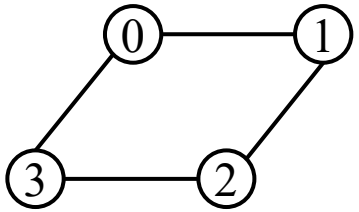
How Powerful are Spectral Graph Neural Networks

Xiyuan Wang[1], Muhan Zhang[1,2]

[1]Institute for Artificial Intelligence, Peking University

[2]Beijing Institute for General Artificial Intelligence

Graph Fourier Transform



$$A = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} \quad D = \begin{bmatrix} 2 & & & \\ & 2 & & \\ & & 2 & \\ & & & 2 \end{bmatrix}$$

Normalized Laplacian $\hat{L} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$

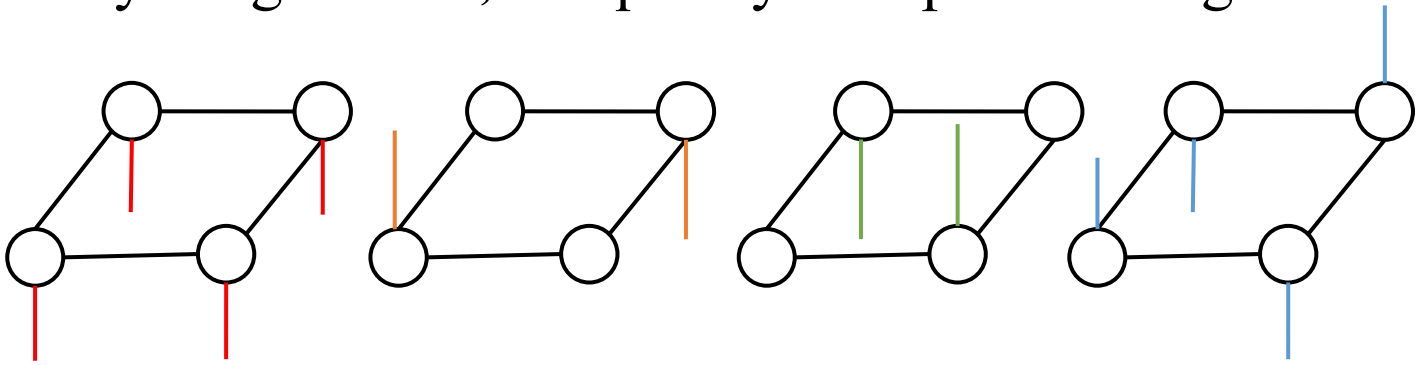
$$\hat{L} = \begin{bmatrix} 1 & -\frac{1}{2} & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & -\frac{1}{2} & 1 \end{bmatrix}$$

Eigendecomposition: $\hat{L} = U \Lambda U^T$

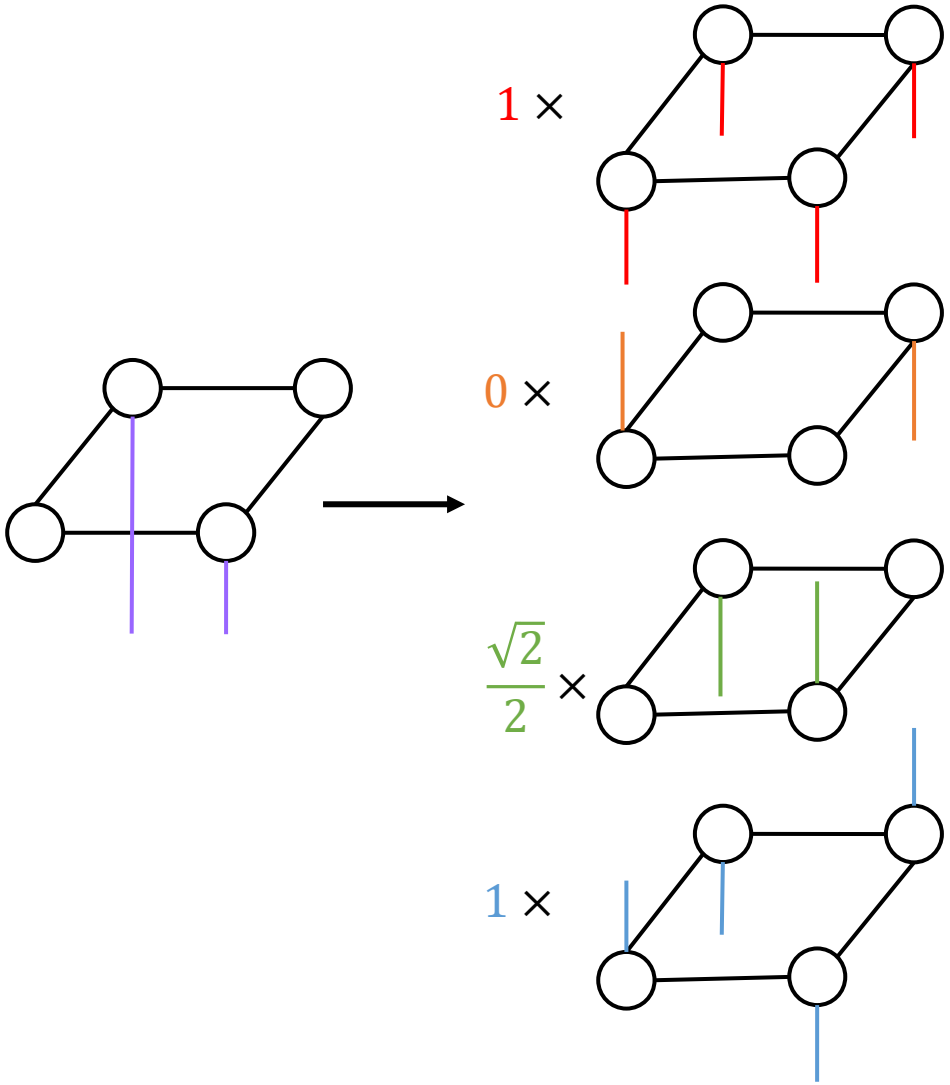
$$\Lambda = \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & 1 & \\ & & & 2 \end{bmatrix}$$

$$U = \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{\sqrt{2}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{\sqrt{2}}{2} & 0 & \frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{\sqrt{2}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{2}}{2} & 0 & \frac{1}{2} \end{bmatrix}$$

- λ_i: The i^{th} eigenvalue. U_i: The i^{th} eigenvector.
- Frequency=Eigenvalue, Frequency Component=Eigenvector



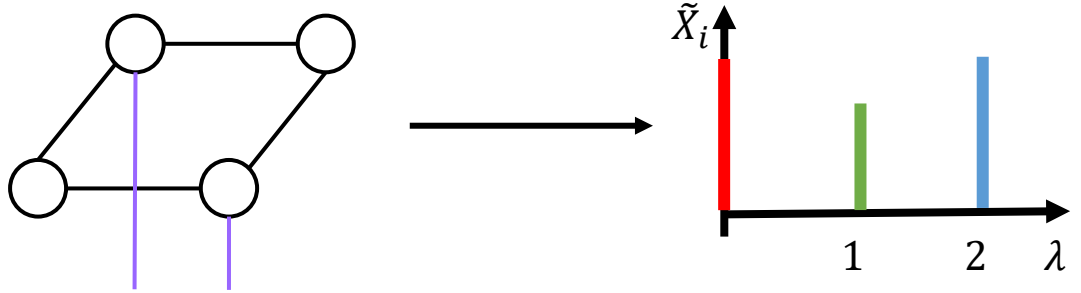
Graph Fourier Transform



Graph Domain X

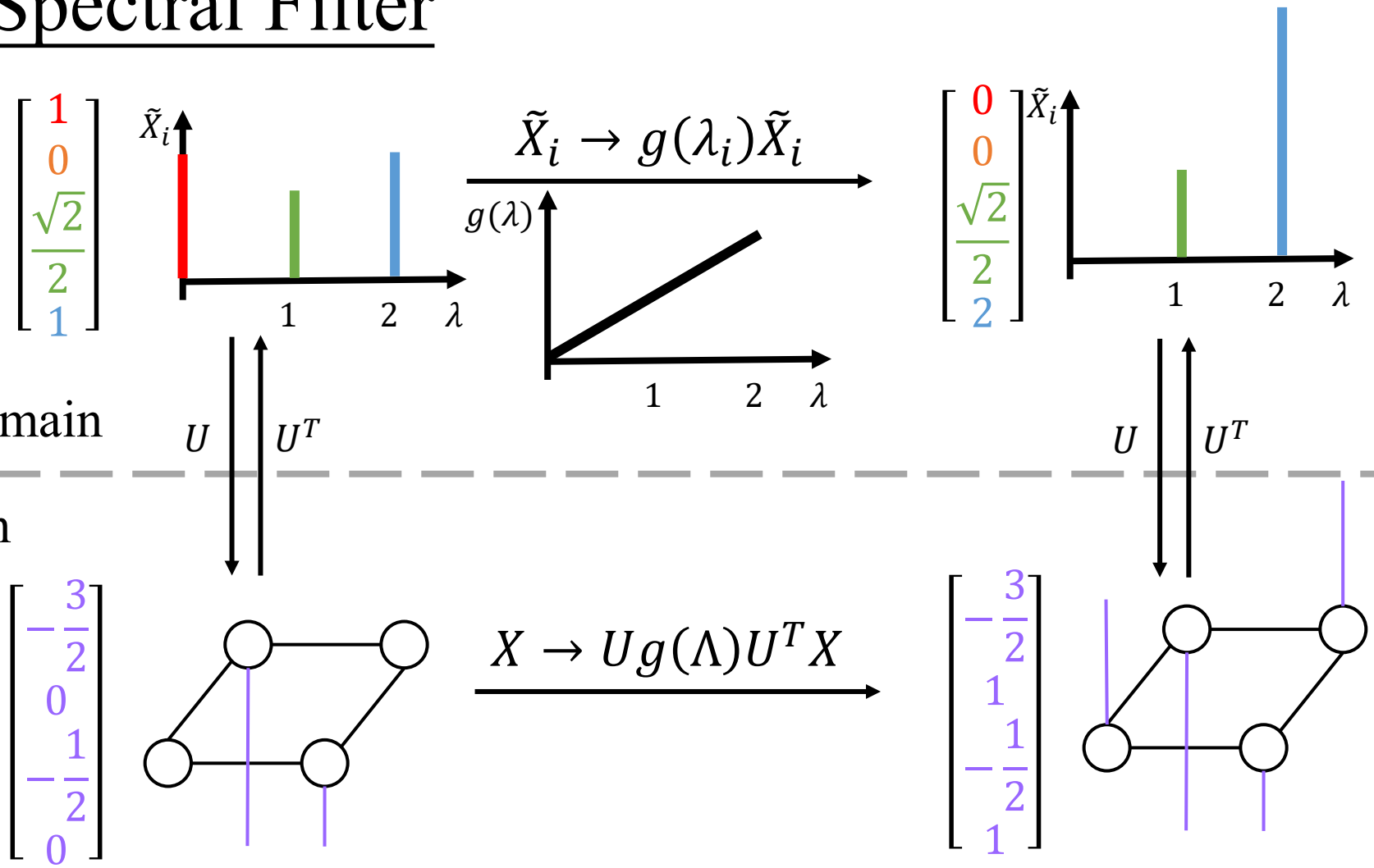
Frequency Domain $\tilde{X}$

$$\begin{bmatrix} -\frac{3}{2} \\ 0 \\ -\frac{1}{2} \\ 0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{2} & 0 & -\frac{\sqrt{2}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{\sqrt{2}}{2} & 0 & \frac{1}{2} \\ -\frac{1}{2} & 0 & \frac{\sqrt{2}}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{2}}{2} & 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \frac{\sqrt{2}}{2} \\ 1 \end{bmatrix}$$



- Frequency content: $\tilde{X}_i = U_i^T X$
  - Amplitude of frequency component U_i in X
- Graph Fourier transform $\tilde{X} = U^T X$
- Inverse graph Fourier transform $X = U\tilde{X}$

Graph Spectral Filter



- GCN: fixed filter $g(\lambda) = 2 - \lambda$
- GPRGNN: learnable filter

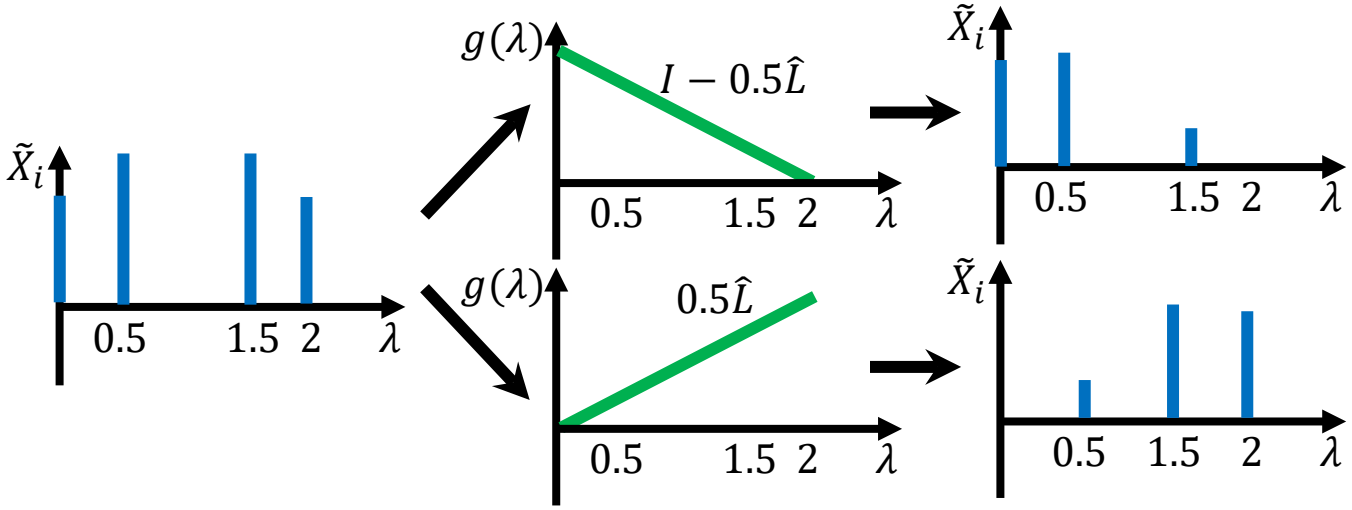
Spectral GNN

- Based on spectral graph filters.
- For node tasks on a fixed graph.
- Filter operator: polynomial $g(\hat{L}) \coloneqq \sum_{k=0}^{K} \alpha_k \hat{L}^k$
- General form: $Z = \mathrm{MLP}\left(g(\hat{L})\mathrm{MLP}(X)\right)$ → Linear GNN: $Z = g(\hat{L})XW$
  - Lower bound for the expressive power of spectral GNNs
  - Linear GNN outperforms spectral GNNs with MLPs

Universality Theorem

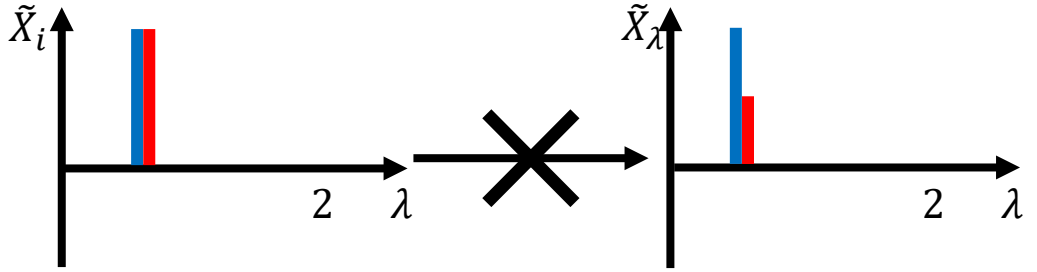
- Linear GNNs can produce any **one-dimensional** prediction if $\hat{L}$ has **no multiple eigenvalues** and the node features X **contain all frequency components**.

1. Multi-dimensional output
   - Solution: Individual filter for each output channel

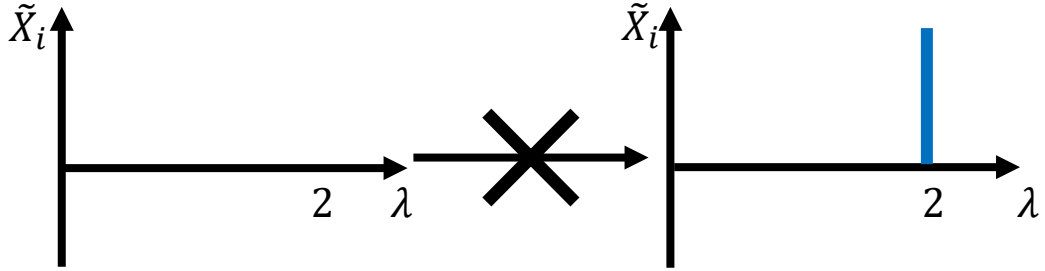


Universality Theorem

2. Multiple eigenvalue
   - Same eigenvalue→scaled by same fold



3. Missing frequency component
   - Node feature X doesn't contain some frequency component
   - $\tilde{X}_i = 0 \rightarrow \forall g,\ \tilde{X}_i \cdot g(\lambda_i) = 0$



Basis Choice

- Different polynomial bases→same expressive power
- Use a set of orthonormal bases on the signal density in frequency domain
  - Accelerate optimization
- Jacobi Basis
  - $P_0^{a,b}(z) = 1, P_1^{a,b}(z) = \frac{a-b}{2} + \frac{a+b+2}{2} z$
  - For $k \geq 2$, $P_k^{a,b}(z) = (\theta_k z + \theta'_k) P_{k-1}^{a,b}(z) - \theta''_k P_{k-2}^{a,b}(z)$
    - $\theta_k = \frac{(2k+a+b)(2k+a+b-1)}{2k(k+a+b)}$
    - $\theta'_k = \frac{(2k+a+b-1)(a^2-b^2)}{2k(k+a+b)(2k+a+b-2)}$
    - $\theta''_k = \frac{(k+a-1)(k+b-1)(2k+a+b)}{k(k+a+b)(2k+a+b-2)}$

Experiment: Real-World Datasets

DATASETS	GCN	APPNP	CHEBYNET	GPRGNN	BERNNET	JACOBICONV
CORA	$87.14_{\pm 1.01}$	$88.14_{\pm 0.73}$	$86.67_{\pm 0.82}$	$88.57_{\pm 0.69}$	$88.52_{\pm 0.95}$	$\mathbf{88.98_{\pm 0.46}}$
CITESEER	$79.86_{\pm 0.67}$	$80.47_{\pm 0.74}$	$79.11_{\pm 0.75}$	$80.12_{\pm 0.83}$	$80.09_{\pm 0.79}$	$\mathbf{80.78_{\pm 0.79}}$
PUBMED	$86.74_{\pm 0.27}$	$88.12_{\pm 0.31}$	$87.95_{\pm 0.28}$	$88.46_{\pm 0.33}$	$88.48_{\pm 0.41}$	$\mathbf{89.62_{\pm 0.41}}$
COMPUTERS	$83.32_{\pm 0.33}$	$85.32_{\pm 0.37}$	$87.54_{\pm 0.43}$	$86.85_{\pm 0.25}$	$87.64_{\pm 0.44}$	$\mathbf{90.39_{\pm 0.29}}$
PHOTO	$88.26_{\pm 0.73}$	$88.51_{\pm 0.31}$	$93.77_{\pm 0.32}$	$93.85_{\pm 0.28}$	$93.63_{\pm 0.35}$	$\mathbf{95.43_{\pm 0.23}}$
CHAMELEON	$59.61_{\pm 2.21}$	$51.84_{\pm 1.82}$	$59.28_{\pm 1.25}$	$67.28_{\pm 1.09}$	$68.29_{\pm 1.58}$	$\mathbf{74.20_{\pm 1.03}}$
ACTOR	$33.23_{\pm 1.16}$	$39.66_{\pm 0.55}$	$37.61_{\pm 0.89}$	$39.92_{\pm 0.67}$	$\mathbf{41.79_{\pm 1.01}}$	$41.17_{\pm 0.64}$
SQUIRREL	$46.78_{\pm 0.87}$	$34.71_{\pm 0.57}$	$40.55_{\pm 0.42}$	$50.15_{\pm 1.92}$	$51.35_{\pm 0.73}$	$\mathbf{57.38_{\pm 1.25}}$
TEXAS	$77.38_{\pm 3.28}$	$90.98_{\pm 1.64}$	$86.22_{\pm 2.45}$	$92.95_{\pm 1.31}$	$93.12_{\pm 0.65}$	$\mathbf{93.44_{\pm 2.13}}$
CORNELL	$65.90_{\pm 4.43}$	$91.81_{\pm 1.96}$	$83.93_{\pm 2.13}$	$91.37_{\pm 1.81}$	$92.13_{\pm 1.64}$	$\mathbf{92.95_{\pm 2.46}}$

- JacobiConv: linear GNN using Jacobi basis and individual filter for each output channel.
- JacobiConv outperforms existing spectral GNNs.
- 10% parameters & similar time overhead

Conclusion

- Linear GNNs can be universal under mild conditions.
  - No multiple eigenvalue
  - No missing frequency component
- Basis choice can affect the optimization of spectral GNNs.
- JacobiConv achieves state-of-the-art empirical performance and verifies our theory.

Thank you!

Paper ID: 2811

Code: https://github.com/GraphPKU/JacobiConv

Contact: wangxiyuan@pku.edu.cn (Xiyuan Wang, Ph.D. student at Peking University)
muhan@pku.edu.cn (Muhan Zhang, Assistant Professor at Peking Univerisity)