Learning by Transference in Large Graphs

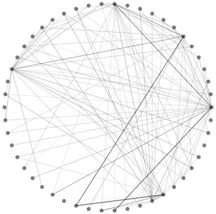
Alejandro Ribeiro

Department of Electrical and Systems Engineering
University of Pennsylvania

aribeiro@seas.upenn.edu
https://alelab.seas.upenn.edu

- **The why**: need to process information on very large graphs in a wide range of applications

  $\Rightarrow$ E.g., product recommendation systems, control of teams of autonomous agents



product similarity graph

robot swarm (Tolstaya, E. et al., 2019)

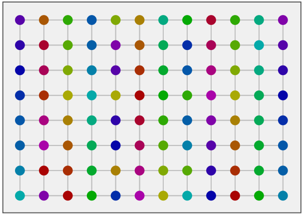
- Machine learning is solution of choice $\Rightarrow$ has been shown to outperform other existing solutions

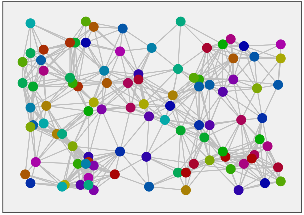
E. Tolstaya et al, *Learning Decentralized Controllers for Robot Swarms with Graph Neural Networks*, CORL 2019, https://arxiv.org/abs/1903.10527

- The how: empirical and theoretical evidence to support using neural networks

  $\Rightarrow$ Standard neural networks are not scalable $\Rightarrow$ use convolutional neural networks (CNNs)

- But convolutional neural networks only operate on regular, grid-like data...





- ... and we would like to process information on irregular structures better modeled as graphs

  $\Rightarrow$ Graph convolutions and graph neural networks (GNNs)

F. Gama et al, *Graphs, Convolutions, and Neural Networks: From Graph Filters to Graph Neural Networks*, SPMAG 2020, https://arxiv.org/pdf/2003.03777.pdf

Q1: We have empirically observed that GNNs scale. Why do they scale?

Q2: Can success of GNNs on moderate-size graphs be used to create success at large-scale?

- To answer these questions, turn to CNNs $\Rightarrow$ known to scale well for images and time sequences

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143 × 95 $\rightarrow$

205 × 136 $\rightarrow$

294 × 195 $\rightarrow$

600 × 399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$

205×136 $\rightarrow$

294×195 $\rightarrow$

600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image

- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$

205×136 $\rightarrow$

294×195 $\rightarrow$

600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$

205×136 $\rightarrow$

294×195 $\rightarrow$

600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Convolutional Neural Networks Have Limits

- Discrete time/image signals converge to continuous time/image signals $\Rightarrow \downarrow$ intrinsic dimension

143×95 $\rightarrow$

205×136 $\rightarrow$

294×195 $\rightarrow$

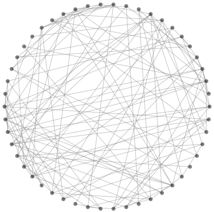
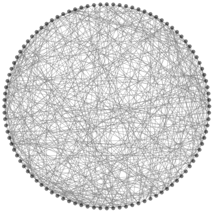
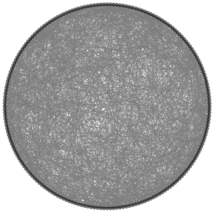
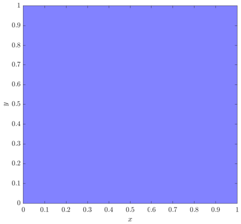
600×399

$\Rightarrow$ From SP theory, CNNs have well-defined limits on the limits of images and time signals

- A1: Intrinsic dimensionality of the problem is less than the size of the image
- A2: Training with small images is sufficient $\Rightarrow$ CIFAR 10 images are 32×32

Graphons

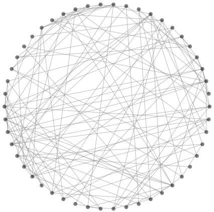
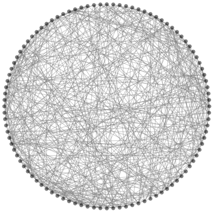
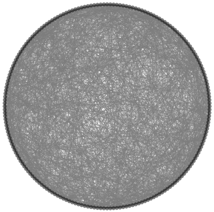
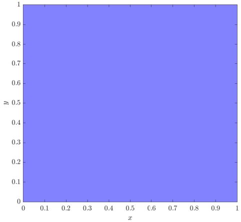
- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon

$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- A graphon can be thought of as a graph with an uncountable number of nodes

- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon

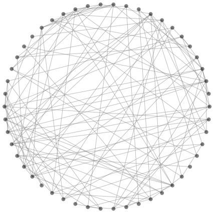





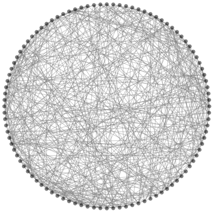
$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

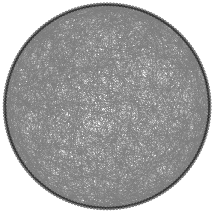
- A graphon can be thought of as a graph with an uncountable number of nodes

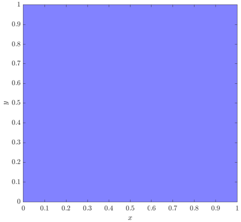
Graphons

- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon







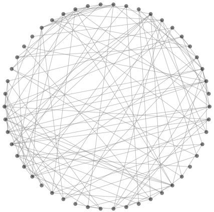
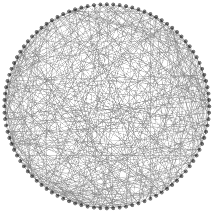
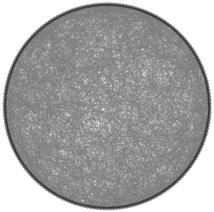
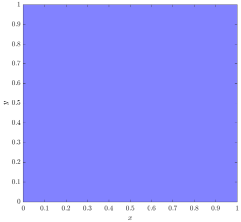


$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- A graphon can be thought of as a graph with an uncountable number of nodes

Graphons

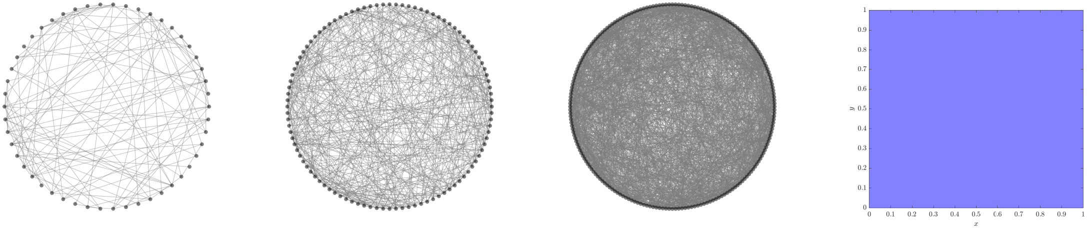
- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon

$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- A graphon can be thought of as a graph with an uncountable number of nodes

- Graphs also have limit objects that effectively limit their dimensionality $\Rightarrow$ one is the graphon

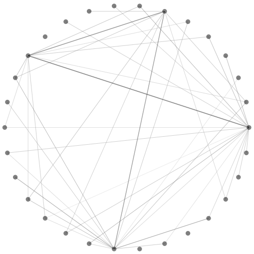


$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

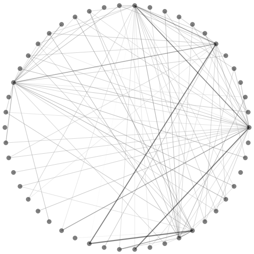
- A graphon can be thought of as a graph with an uncountable number of nodes

Large-Scale Graphs

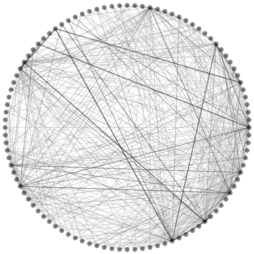
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



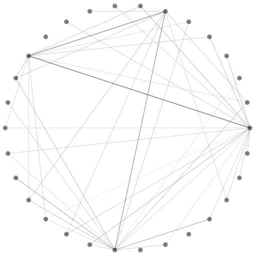
$n = 50$ products



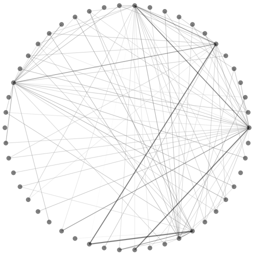
$n = 100$ products

- So do graph convolutions and graph neural networks converge to limits on the graphon?

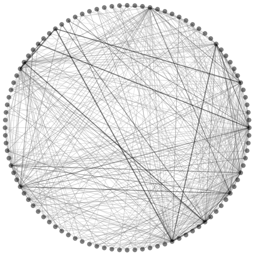
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



$n = 50$ products

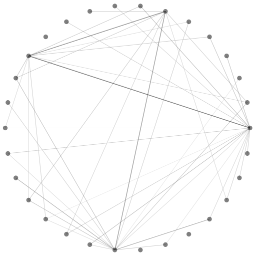


$n = 100$ products

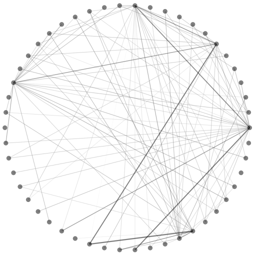
- So do graph convolutions and graph neural networks converge to limits on the graphon?

Large-Scale Graphs

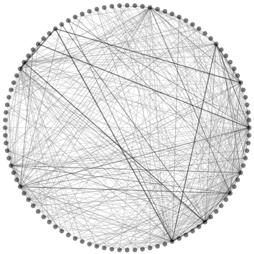
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



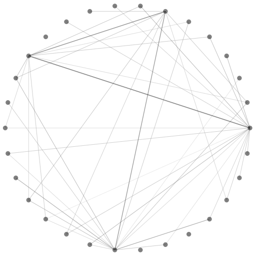
$n = 50$ products



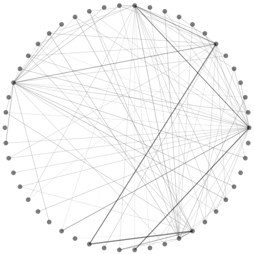
$n = 100$ products

- So do graph convolutions and graph neural networks converge to limits on the graphon?

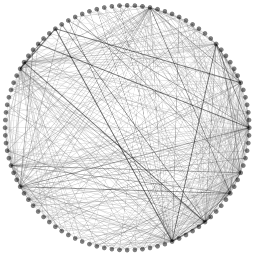
- Graphs however do not have the Euclidean structure time and image signals have in the limit



$n = 30$ products



$n = 50$ products



$n = 100$ products

- So do graph convolutions and graph neural networks converge to limits on the graphon?

Q1: We have empirically observed that GNNs scale. Why do they scale?

- A1: Because graph convolutions and GNNs have well-defined limits on graphons

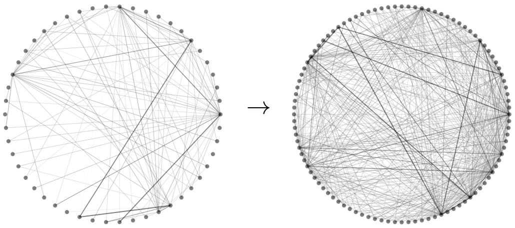
L. Ruiz et al, *Graphon Signal Processing*, TSP 2021, https://arxiv.org/abs/2003.05030

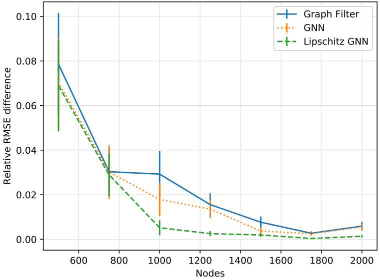
Q2: Can success of GNNs on moderate-size graphs be used to create success at large-scale?

- A2: Yes, as GNNs are transferable $\Rightarrow$ can be trained on moderate-size and executed on large-scale

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

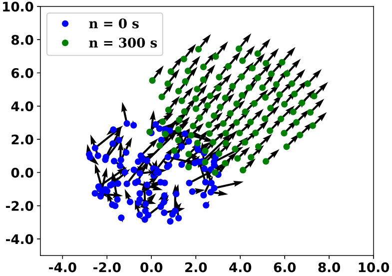
- Transferability of graph neural networks useful in practice $\Rightarrow$ recommendation system

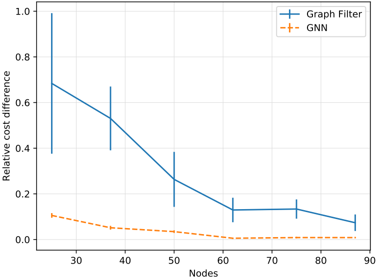




- Performance difference on training and target graphs decreases as size of training graph grows
- GNNs appear to be more transferable than graph convolutional filters $\Rightarrow$ better ML model

- Transferability of graph neural networks useful in practice $\Rightarrow$ decentralized robot control





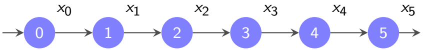
- Performance difference on training and target graphs decreases as size of training graph grows

- GNNs appear to be more transferable than graph convolutional filters $\Rightarrow$ better ML model

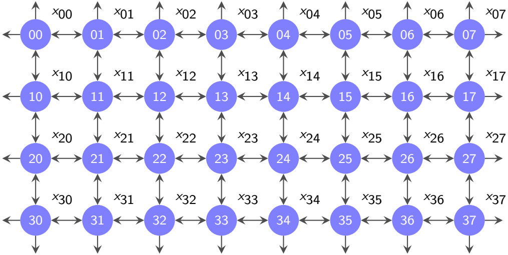
Graph Convolutions

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph



Description of images (space) with a grid graph

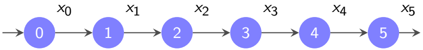


- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

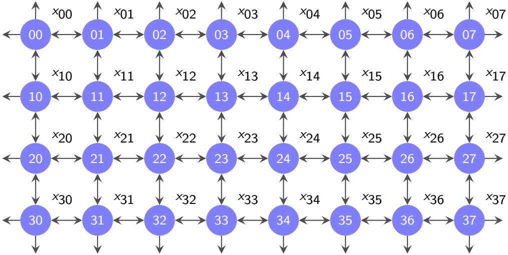
Convolutions in Time and Space

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph



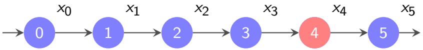
Description of images (space) with a grid graph



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

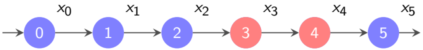


Description of images (space) with a grid graph

- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

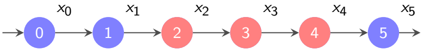


Description of images (space) with a grid graph

- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

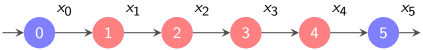


Description of images (space) with a grid graph

- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph



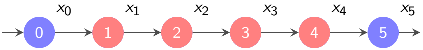
Description of images (space) with a grid graph

- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

Convolutions in Time and Space

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph



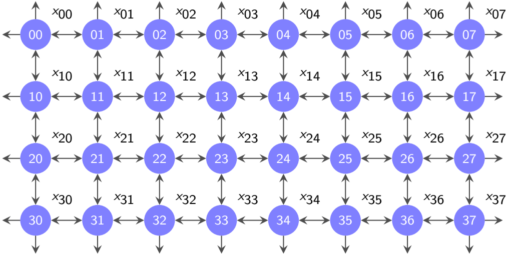
Description of images (space) with a grid graph

- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

Description of images (space) with a grid graph

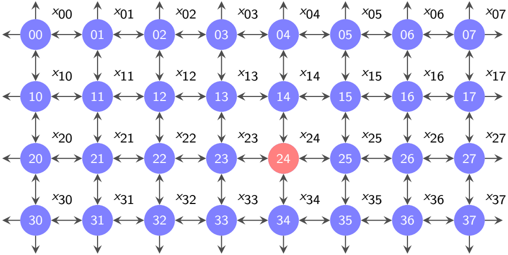


- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

Description of images (space) with a grid graph

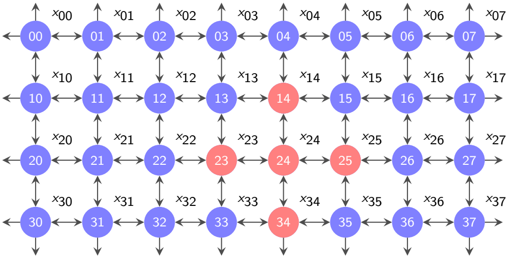


- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

Description of images (space) with a grid graph



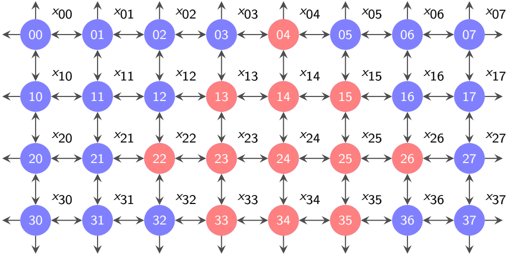
- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

Convolutions in Time and Space

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

Description of images (space) with a grid graph



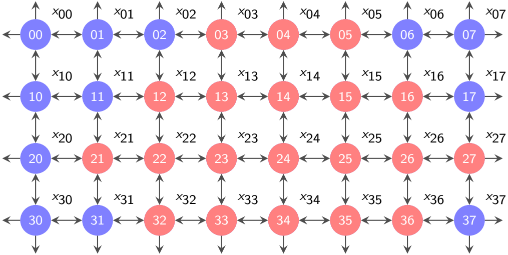
- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

Convolutions in Time and Space

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

Description of images (space) with a grid graph

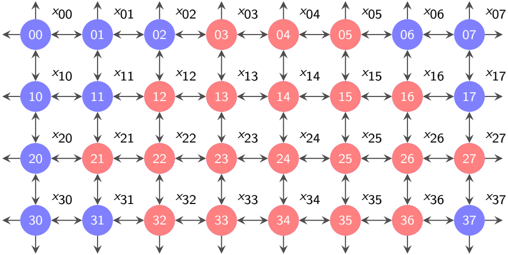


- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph

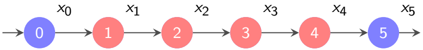
Description of images (space) with a grid graph



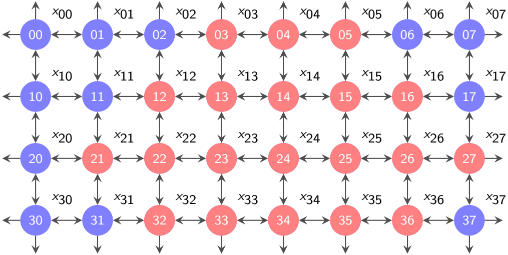
- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- Use line and grid graphs to write convolutions as polynomials on respective adjacency matrices $\mathbf{S}$

Description of time with a directed line graph



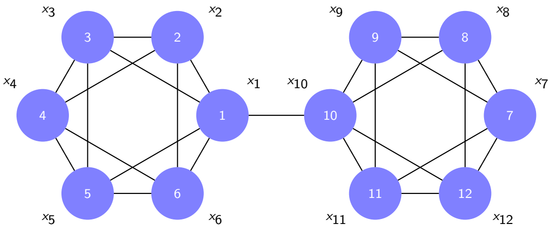
Description of images (space) with a grid graph



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

Convolutions on Graphs

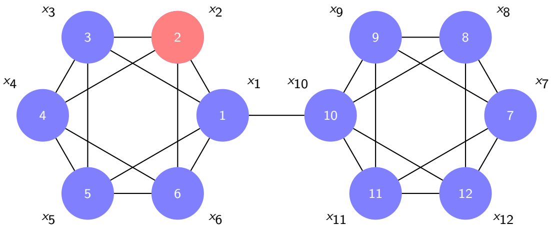
- For graph signals we define graph convolutions as polynomials on matrix representations of graphs



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} =$ $h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- To analyze their convergence to a limit object on the graphon $\Rightarrow$ need to define graphons

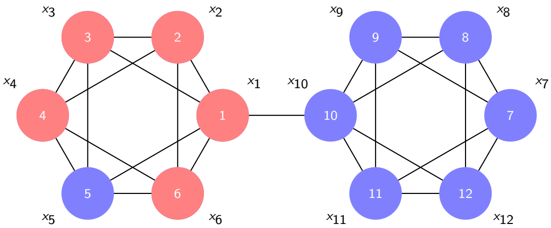
- For graph signals we define graph convolutions as polynomials on matrix representations of graphs



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0 \mathbf{S}^0\mathbf{x} + h_1 \mathbf{S}^1\mathbf{x} + h_2 \mathbf{S}^2\mathbf{x} + h_3 \mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k \mathbf{S}^k\mathbf{x}$

- To analyze their convergence to a limit object on the graphon $\Rightarrow$ need to define graphons

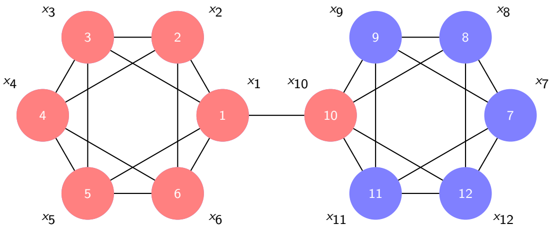
- For graph signals we define graph convolutions as polynomials on matrix representations of graphs



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- To analyze their convergence to a limit object on the graphon $\Rightarrow$ need to define graphons

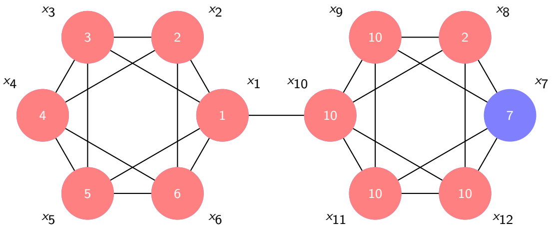
- For graph signals we define graph convolutions as polynomials on matrix representations of graphs



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$

- To analyze their convergence to a limit object on the graphon $\Rightarrow$ need to define graphons

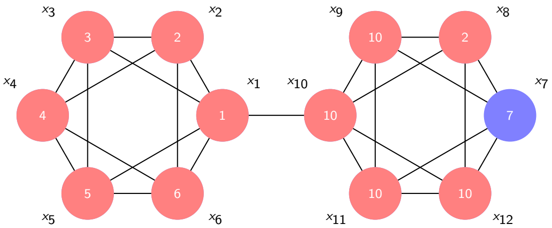
- For graph signals we define graph convolutions as polynomials on matrix representations of graphs



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = \; h_0\, \mathbf{S}^0 \mathbf{x} \; + h_1\, \mathbf{S}^1 \mathbf{x} \; + h_2\, \mathbf{S}^2 \mathbf{x} \; + h_3\, \mathbf{S}^3 \mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\, \mathbf{S}^k \mathbf{x}$

- To analyze their convergence to a limit object on the graphon $\Rightarrow$ need to define graphons

- For graph signals we define graph convolutions as polynomials on matrix representations of graphs



- Filter with coefficients h_k $\Rightarrow$ Output $\mathbf{z} = h_0 \mathbf{S}^0 \mathbf{x} + h_1 \mathbf{S}^1 \mathbf{x} + h_2 \mathbf{S}^2 \mathbf{x} + h_3 \mathbf{S}^3 \mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k \mathbf{S}^k \mathbf{x}$
- To analyze their convergence to a limit object on the graphon $\Rightarrow$ need to define graphons

Graphons

Definition (Graphon)

A graphon $\mathbf{W}$ is a bounded symmetric measurable function $\Rightarrow \mathbf{W} : [0,1]^2 \to [0,1]$

- Can think of a graphon as a weighted symmetric graph with an uncountable number of nodes

  $\Rightarrow$ Labels are graphon arguments $u \in [0,1]$, weights are graphon values $W(u,v) = W(v,u)$

- Interpreted as the limit of a sequence of graphs in the sense that densities of motifs converge

- Interpreted as a generative model of graph families by sampling edges $(u_i, u_j) \sim \mathcal{B}(\mathbf{W}(u_i, u_j))$

Definition (Graphon)

A graphon $\mathbf{W}$ is a bounded symmetric measurable function $\Rightarrow \mathbf{W} : [0,1]^2 \to [0,1]$

- Can think of a graphon as a weighted symmetric graph with an uncountable number of nodes

  $\Rightarrow$ Labels are graphon arguments $u \in [0,1]$, weights are graphon values $W(u,v) = W(v,u)$

- Interpreted as the limit of a sequence of graphs in the sense that densities of motifs converge

- Interpreted as a generative model of graph families by sampling edges $(u_i, u_j) \sim \mathcal{B}(\mathbf{W}(u_i, u_j))$

Definition (Graphon)

A graphon $\mathbf{W}$ is a bounded symmetric measurable function $\Rightarrow \mathbf{W} : [0,1]^2 \to [0,1]$

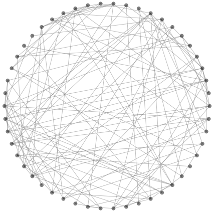
- Can think of a graphon as a weighted symmetric graph with an uncountable number of nodes

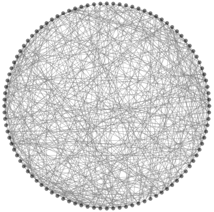
  $\Rightarrow$ Labels are graphon arguments $u \in [0,1]$, weights are graphon values $W(u,v) = W(v,u)$

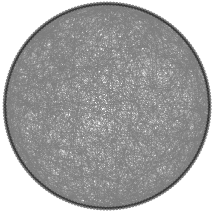
- Interpreted as the limit of a sequence of graphs in the sense that densities of motifs converge

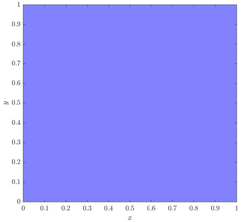
- Interpreted as a generative model of graph families by sampling edges $(u_i, u_j) \sim \mathcal{B}(\mathbf{W}(u_i, u_j))$

- A sequence of Erdos-Renyi or uniform graphs converges to Erdos-Renyi or uniform graphons





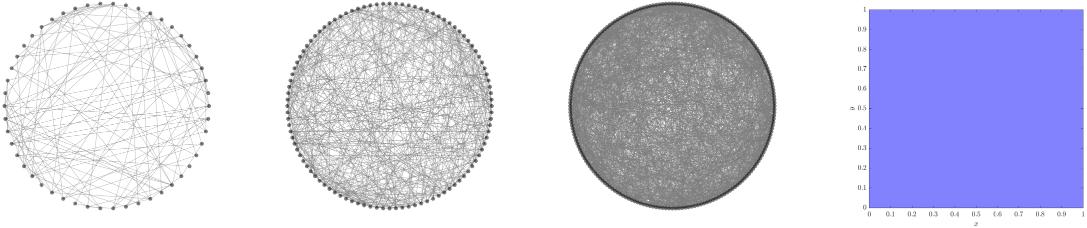




$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- The uniform graphon can be used to sample uniform graphs with 200, 100, and 50 nodes

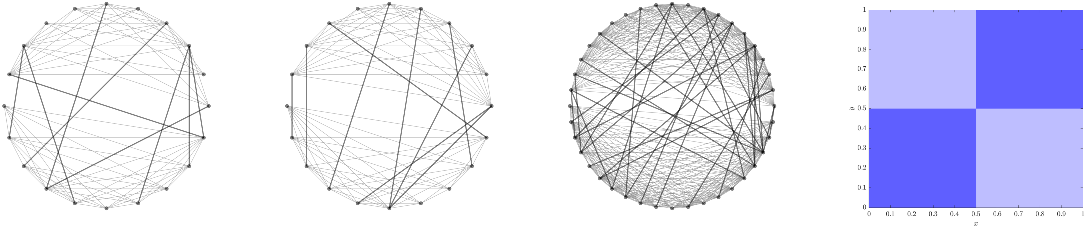
- A sequence of Erdos-Renyi or uniform graphs converges to Erdos-Renyi or uniform graphons



$n = 50$ nodes $\rightarrow$ $n = 100$ nodes $\rightarrow$ $n = 200$ nodes $\rightarrow$ Graphon $W(u, v) = p$

- The uniform graphon can be used to sample uniform graphs with 200, 100, and 50 nodes

- A sequence of stochastic block model graphs converges to stochastic block model graphons

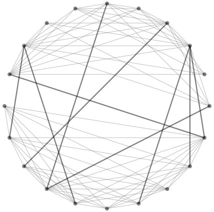
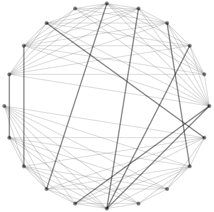
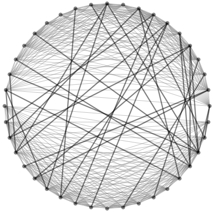
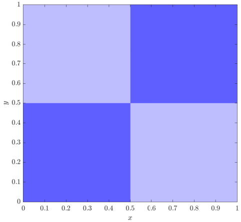


$n = 20$ nodes $\rightarrow$ $n = 30$ nodes $\rightarrow$ $n = 40$ nodes $\rightarrow$ Graphon $W(u, v)$

- The stochastic block model graphon can be used to sample SBM graphs with 40, 30, and 20 nodes

SBM as a Limit Object

- A sequence of stochastic block model graphs converges to stochastic block model graphons

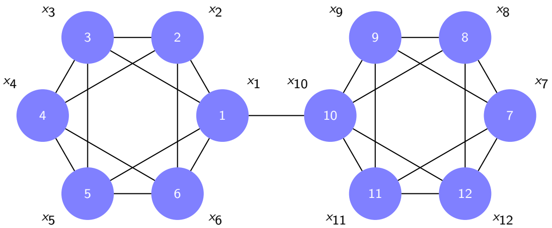





$n = 20$ nodes $\rightarrow$ $n = 30$ nodes $\rightarrow$ $n = 40$ nodes $\rightarrow$ Graphon $W(u, v)$

- The stochastic block model graphon can be used to sample SBM graphs with 40, 30, and 20 nodes

Graphon Convolutions

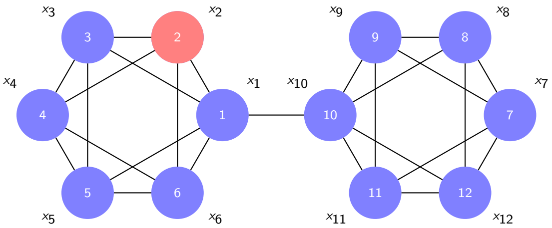
- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

$\Rightarrow$ Graphon convolutions are analogously parametrized by the graphon shift operator

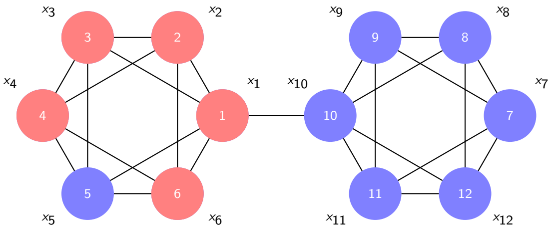
- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0\, \mathbf{S}^0\mathbf{x} \; + h_1\, \mathbf{S}^1\mathbf{x} \; + h_2\, \mathbf{S}^2\mathbf{x} \; + h_3\, \mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\, \mathbf{S}^k\mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

$\Rightarrow$ Graphon convolutions are analogously parametrized by the graphon shift operator

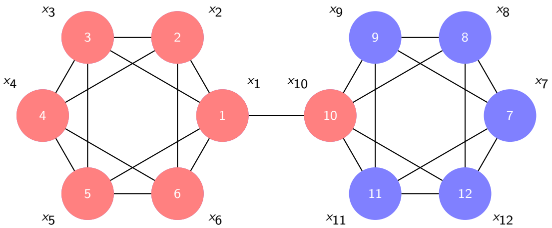
- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0\, \mathbf{S}^0 \mathbf{x} \; + h_1\, \mathbf{S}^1 \mathbf{x} \; + h_2\, \mathbf{S}^2 \mathbf{x} \; + h_3\, \mathbf{S}^3 \mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\, \mathbf{S}^k \mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

$\Rightarrow$ Graphon convolutions are analogously parametrized by the graphon shift operator

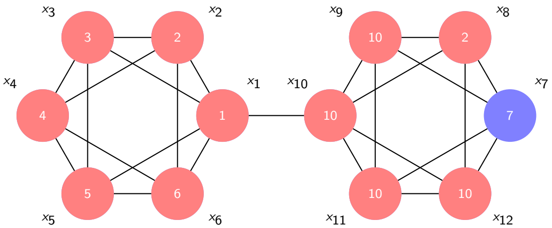
- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0 \, \mathbf{S}^0 \mathbf{x} \; + h_1 \, \mathbf{S}^1 \mathbf{x} \; + h_2 \, \mathbf{S}^2 \mathbf{x} \; + h_3 \, \mathbf{S}^3 \mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k \, \mathbf{S}^k \mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

$\Rightarrow$ Graphon convolutions are analogously parametrized by the graphon shift operator

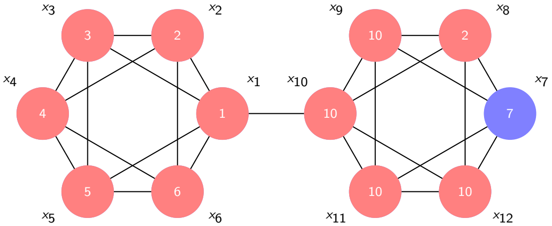
- Graph convolution $\Rightarrow$ Output $\mathbf{z} = \; h_0\,\mathbf{S}^0\mathbf{x} \; + h_1\,\mathbf{S}^1\mathbf{x} \; + h_2\,\mathbf{S}^2\mathbf{x} \; + h_3\,\mathbf{S}^3\mathbf{x} \; + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

$\Rightarrow$ Graphon convolutions are analogously parametrized by the graphon shift operator

- Graph convolution $\Rightarrow$ Output $\mathbf{z} = h_0\,\mathbf{S}^0\mathbf{x} + h_1\,\mathbf{S}^1\mathbf{x} + h_2\,\mathbf{S}^2\mathbf{x} + h_3\,\mathbf{S}^3\mathbf{x} + \ldots = \sum_{k=0}^{K-1} h_k\,\mathbf{S}^k\mathbf{x}$



- Note that the graph convolution is parametrized by the operator $\mathbf{z}_k = \mathbf{S}\mathbf{z}_{k-1}$ $\Rightarrow$ graph shift operator

$\Rightarrow$ Graphon convolutions are analogously parametrized by the graphon shift operator

- The graphon shift operator is an integral linear operator with kernel given by the graphon $\mathbf{W}$

Definition (Graphon Shift Operator)

The graphon shift operator of a graphon $\mathbf{W}$ is defined as

$$Y(v) = (T_{\mathbf{W}}X)(v) = \int_0^1 \mathbf{W}(u,v)X(u)du.$$

Definition (Graphon Convolution) (Ruiz, L., Chamon, L. F. O., Ribeiro, A., TSP'21)

A graphon convolution with coefficients $h_0, \ldots h_{K-1}$ is defined as

$$Y(v) = (T_{\mathbf{H}} X)(v) = \sum_{k=0}^{K-1} h_k \left(T_{\mathbf{W}}^{(k)} X \right)(v)$$

$$\left(T_{\mathbf{W}}^{(k)} X \right)(v) = \int_0^1 \mathbf{W}(u, v) \left(T_{\mathbf{W}}^{(k-1)} X \right)(u) du \ , \ k \geq 1 \quad \left(T_{\mathbf{W}}^{(0)} = \mathbb{I} \right)$$

- Filter with coefficients h_k $\Rightarrow$ Output $Z = \ h_0 \, \mathbf{T}_{\mathbf{W}}^{(0)} X \ + h_1 \, \mathbf{T}_{\mathbf{W}}^{(1)} X \ + h_2 \, \mathbf{T}_{\mathbf{W}}^{(2)} X \ + \ldots = \sum_{k=0}^{K-1} h_k \, \mathbf{T}_{\mathbf{W}}^{(k)} X$

M. Morency and G. Leus, *Graphon Filters: Graph Signal Processing in the Limit,* TSP 2020 https://arxiv.org/abs/2003.02099

- The graph (which is symmetric) admits the eigenvector decomposition $\mathbf{S}_n = \mathbf{V}_n \mathbf{\Lambda}_n \mathbf{V}_n^H$

Theorem (Graph frequency representation of graph filters)

Consider graph filter with coefficients h_k, graph signal $\mathbf{x}_n$ and the filtered signal $\mathbf{y}_n = \sum_{k=0}^{K-1} h_k \mathbf{S}_n^k \mathbf{x}_n$.

The Graph Fourier Transforms (GFTs) $\tilde{\mathbf{x}}_n = \mathbf{V}_n^H \mathbf{x}_n$ and $\tilde{\mathbf{y}}_n = \mathbf{V}_n^H \mathbf{y}_n$ are related by

$$\tilde{\mathbf{y}}_n = \sum_{k=0}^{K-1} h_k \mathbf{\Lambda}_n^k \tilde{\mathbf{x}}_n \qquad \Rightarrow \qquad \tilde{y}_{nj} = \sum_{k=0}^{K-1} h_k \lambda_{nj}^k \tilde{x}_{nj}$$

- This is a simple eigenvalue decomposition of the graph filter polynomial $\Rightarrow$ Nonetheless interesting

  $\Rightarrow$ Graph filters are pointwise operators in the spectral domain

- It is not only that the operator is pointwise, it also (sort of) decouples the filter from the graph

Definition (Frequency Response of a Graph Filter)

Given a graph filter with coefficients h_k the graph frequency response is the polynomial

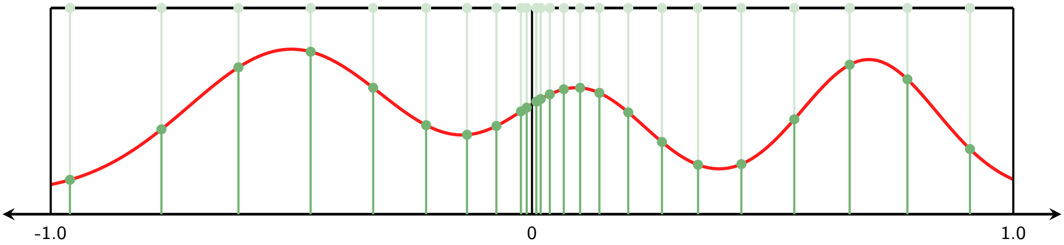
$$\tilde{h}(\lambda) = \sum_{k=0}^{K-1} h_k \lambda^k$$

- The frequency response is independent of the graph. It is a polynomial on a scalar variable λ
- Graph determines eigenvalues at which response is instantiated $\Rightarrow \tilde{y}_{nj} = \sum_{k=0}^{K-1} h_k \lambda_{nj}^k \tilde{x}_{nj} = h(\lambda_{nj})\tilde{x}_{nj}$

Graph Frequency Response

- The frequency response is independent of the graph. It is a polynomial on a scalar variable λ

- Graph determines eigenvalues at which response is instantiated $\Rightarrow \tilde{y}_{nj} = \sum_{k=0}^{K-1} h_k \lambda_{nj}^k \tilde{x}_{nj} = h(\lambda_{nj})\tilde{x}_{nj}$



- Since graphon shifts are Hilbert-Schmidt operators, the same can be done for graphon filters
- The eigenfunction representation of the graphon shift is $W(u,v) = \sum_{j \in \mathbb{Z} \setminus \{0\}} \lambda_j \phi_j(u) \varphi_j(v)$

Theorem (Graphon frequency representation of graphon filters)

Consider graphon filter with coefficients h_k, graphon signal X and the filtered signal Y The Graphon Fourier Transforms (GFTs) $\tilde{X}_j = \int_0^1 \varphi_j(u) X_(u) du$ and $\tilde{Y}_j = \int_0^1 \varphi_j(u) Y_(u) du$ are related by

$$\tilde{Y}_j = \sum_{k=0}^{K-1} h_k \lambda_j^k \tilde{X}_j$$

- Graphon filters, same as graph filters, are pointwise operators in the spectral domain

- It is not only that the operator is pointwise, it also (sort of) decouples the filter from the graphon

Definition (Frequency Response of a Graphon Filter)

Given a graph filter with coefficients h_k the graphon frequency response is the polynomial

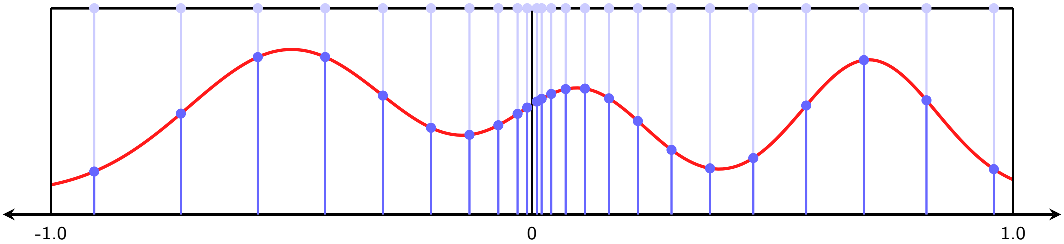
$$\tilde{h}(\lambda) = \sum_{k=0}^{K-1} h_k \lambda^k$$

- Graphon-independent. More importantly the same as the graph response for the same coefficients h_k

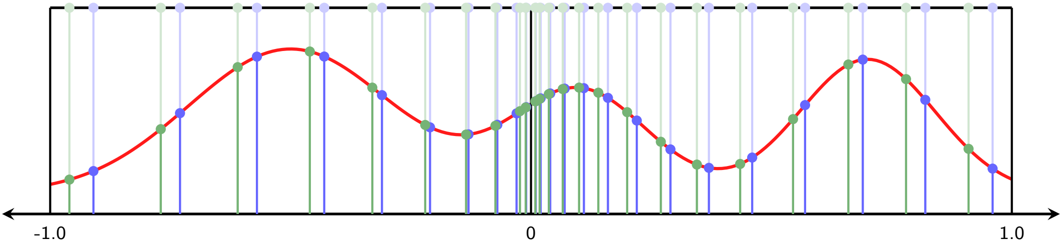
- Graphon determines eigenvalues at which response is instantiated $\Rightarrow \tilde{Y}_j = \sum_{k=0}^{K-1} h_k \lambda_j^k \tilde{X}_j = h(\lambda_j)\tilde{X}_j$

- Graphon-independent. More importantly the same as the graph response for the same coefficients h_k

- Graphon determines eigenvalues at which response is instantiated $\Rightarrow \tilde{Y}_j = \sum_{k=0}^{K-1} h_k \lambda_j^k \tilde{X}_j = h(\lambda_j)\tilde{X}_j$



- Spectral response of graph and graphon convolution is given by the same function $h(\lambda)$



- Spectral response of the graph convolution determined by evaluating $h(\lambda)$ at graph eigenvalues
- Spectral response of the graphon convolution determined by evaluating $h(\lambda)$ at graphon eigenvalues

- Graph convolutions converge to graphon convolutions $\Rightarrow$ provided that $h(\lambda)$ is Lipschitz

Theorem (Convergence of Graph Convolutions)

Given convergent graph signal sequence $(G_n, \mathbf{x}_n) \to (W, X)$ and convolutions $\mathbf{H}(\mathbf{S}_n)$ and T_H generated by the same coefficients h_k, if the spectral response $h(\lambda)$ is Lipschitz,

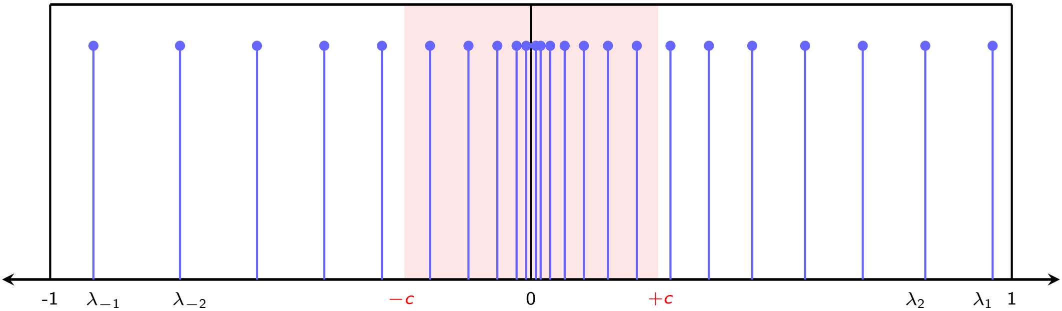
$$(\mathbf{G}_n, \mathbf{y}_n) \to (\mathbf{W}, Y)$$

i.e., the sequence of output graph signals converges to the output graphon signal.

- Lipschitz continuity restriction better understood in the graph and graphon spectral domain

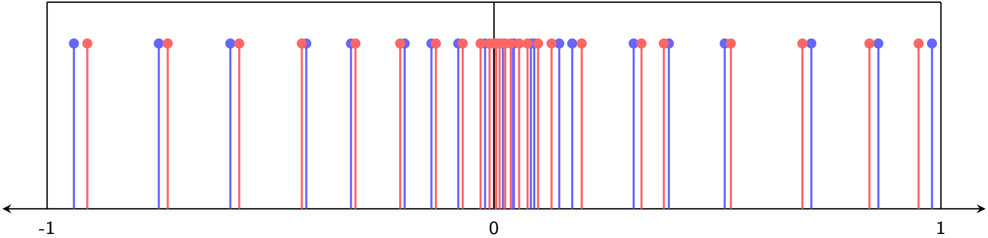
L. Ruiz et al, *Graphon Signal Processing*, TSP 2021, https://arxiv.org/abs/2003.05030

- Due to $T_{\mathbf{W}}$ being compact, graphon eigenvalues accumulate at $\lambda = 0$ $\Rightarrow \lim_{i\to\infty} \lambda_i = \lim_{i\to\infty} \lambda_{-i} = 0$



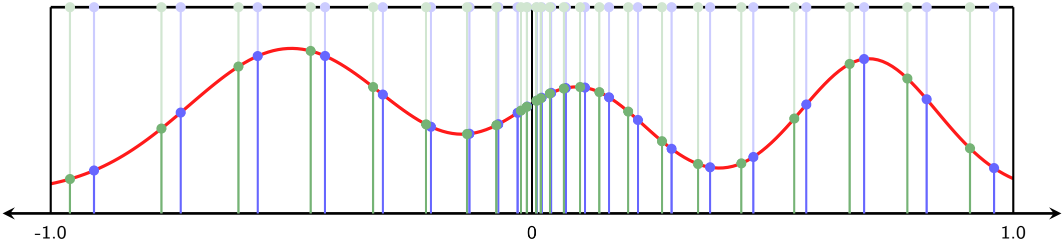
If a graph sequence $\{\mathbf{G}_n\}$ converges to a graphon $\mathbf{W}$, then

$$\lim_{n\to\infty} \frac{\lambda_j(\mathbf{S}_n)}{n} = \lambda_j(T_{\mathbf{W}}) \text{ for all } j$$ (Borgs, C. et al., 2012)



- But for $\neq j$, $\neq n_0$ are needed to show that $\exists\, n_0$ s.t. for all $n > n_0$, $\left|\frac{\lambda_j(\mathbf{S}_n)}{n} - \lambda_j(T_{\mathbf{W}})\right| < \epsilon$

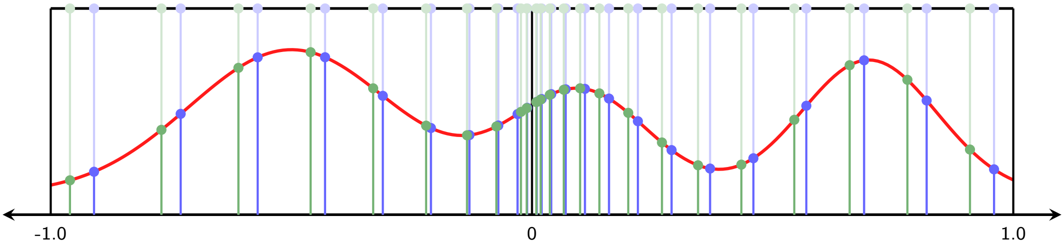
- Because eigenvalues converge, we can expect graph convolutions to converge



- But convergence near $\lambda = 0$ is complicated by eigenvalue convergence not being uniform

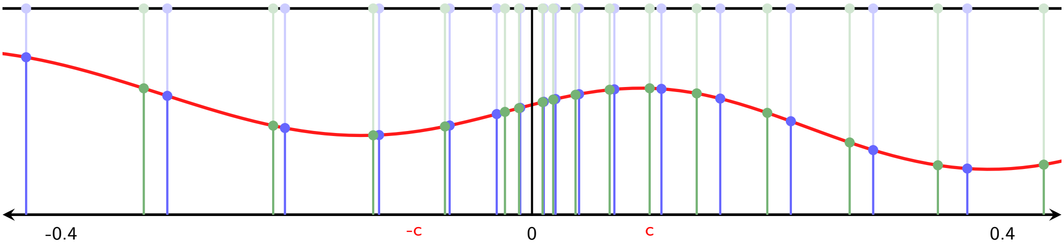
- Filters attempting to discriminate spectral components near $\lambda = 0$ do not converge

- This problem can be solved if we amplify these spectral components similarly for $|\lambda| \leq c$



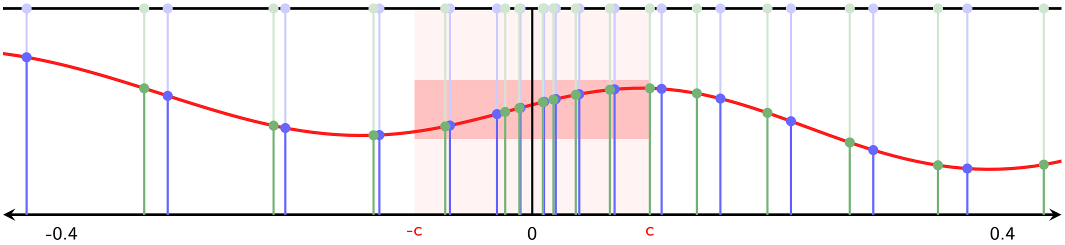
- Lipschitz filters ensure no mismatch between eigenspaces of $|\lambda_j(\mathbf{S}_n)| \leq c$ and $|\lambda_j(\mathbf{W})| \leq c$
- Lipschitz condition means that convergence comes at the cost of spectral discriminability

- This problem can be solved if we amplify these spectral components similarly for $|\lambda| \leq c$



- Lipschitz filters ensure no mismatch between eigenspaces of $|\lambda_j(\mathbf{S}_n)| \leq c$ and $|\lambda_j(\mathbf{W})| \leq c$
- Lipschitz condition means that convergence comes at the cost of spectral discriminability

- This problem can be solved if we amplify these spectral components similarly for $|\lambda| \leq c$



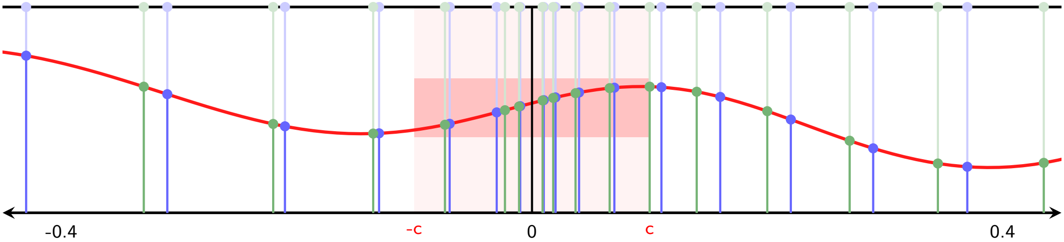
- Lipschitz filters ensure no mismatch between eigenspaces of $|\lambda_j(\mathbf{S}_n)| \leq c$ and $|\lambda_j(\mathbf{W})| \leq c$

- Lipschitz condition means that convergence comes at the cost of spectral discriminability

Transferability

- Have established an asymptotic result $\Rightarrow$ graph convolutions converge, but with a condition

- Depending on the value of the Lipschitz constant of $h(\lambda)$, convergence may be faster or slower



- In order to exploit this result in practice, need a non-asymptotic analysis for finite n

Theorem (Graphon Filter Approximation)

Consider a graph signal $(\mathbf{S}_n, \mathbf{x}_n)$ sampled from the graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $Y = T_{\mathbf{H}}X$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}\right)\|X\| + A_x(A_h c + 2)\left(\frac{1}{n}\right) + 2A_h c\|X\|$$

- Bound decreases with n $\Rightarrow$ graph filters better approximate graphon filter for large n as expected
- As $n \to \infty$ we can afford smaller bandwith c $\Rightarrow$ convergence of filters closer to $\lambda = 0$

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graphon Filter Approximation)

Consider a graph signal $(\mathbf{S}_n, \mathbf{x}_n)$ sampled from the graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $Y = T_{\mathbf{H}}X$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}\right)\|X\| + A_x(A_hc + 2)\left(\frac{1}{n}\right) + 2A_hc\|X\|$$

- Bound decreases with n $\Rightarrow$ graph filters better approximate graphon filter for large n as expected
- As $n \to \infty$ we can afford smaller bandwith c $\Rightarrow$ convergence of filters closer to $\lambda = 0$

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graphon Filter Approximation)

Consider a graph signal $(\mathbf{S}_n, \mathbf{x}_n)$ sampled from the graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $Y = T_{\mathbf{H}}X$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}\right)\|X\| + A_x(A_hc+2)\left(\frac{1}{n}\right) + 2A_hc\|X\|$$

- Bound decreases with n $\Rightarrow$ graph filters better approximate graphon filter for large n as expected
- As $n \to \infty$ we can afford smaller bandwith c $\Rightarrow$ convergence of filters closer to $\lambda = 0$

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graphon Filter Approximation)

Consider a graph signal $(\mathbf{S}_n, \mathbf{x}_n)$ sampled from the graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $Y = T_{\mathbf{H}}X$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}\right)\|X\| + A_x(A_hc + 2)\left(\frac{1}{n}\right) + 2A_hc\|X\|$$

- Discriminating around $\lambda = 0$ needs large Lipschitz constant A_h $\Rightarrow$ large approximation error
- Filters that are more discriminative (large A_h) converge more slowly with n

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graphon Filter Approximation)

Consider a graph signal $(\mathbf{S}_n, \mathbf{x}_n)$ sampled from the graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $Y = T_{\mathbf{H}}X$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}\right)\|X\| + A_x(A_hc+2)\left(\frac{1}{n}\right) + 2A_hc\|X\|$$

- Discriminating around $\lambda = 0$ needs large Lipschitz constant A_h $\Rightarrow$ large approximation error
- Filters that are more discriminative (large A_h) converge more slowly with n

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graphon Filter Approximation)

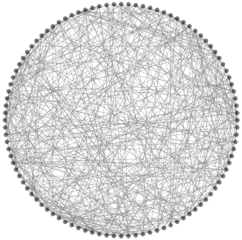
Consider a graph signal $(\mathbf{S}_n, \mathbf{x}_n)$ sampled from the graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $Y = T_{\mathbf{H}}X$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y\| \;\leq\; 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}\right)\|X\| + A_x(A_h c + 2)\left(\frac{1}{n}\right) + 2A_h c\|X\|$$

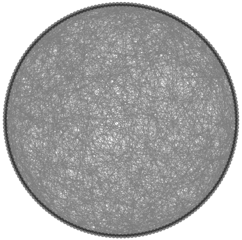
- Discriminating around $\lambda = 0$ needs large Lipschitz constant A_h $\Rightarrow$ large approximation error
- Filters that are more discriminative (large A_h) converge more slowly with n

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

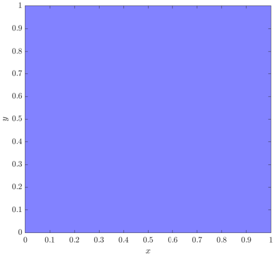
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



m nodes

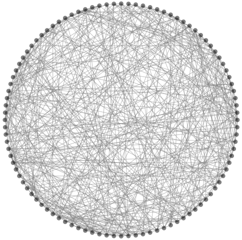


Graphon $W(u, v) = p$

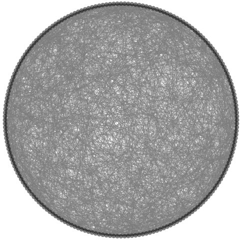
- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

Transferability Paradigm

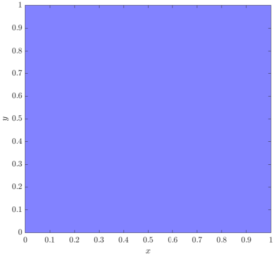
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



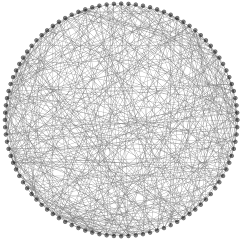
m nodes



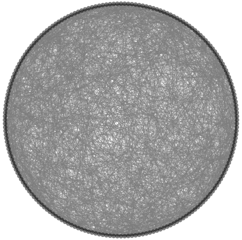
Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

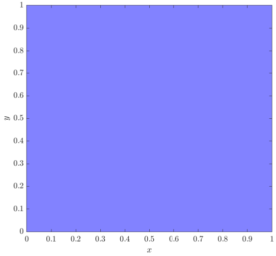
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



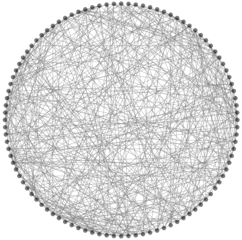
m nodes



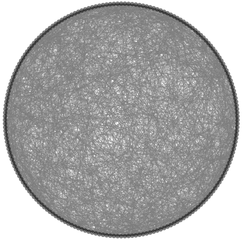
Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

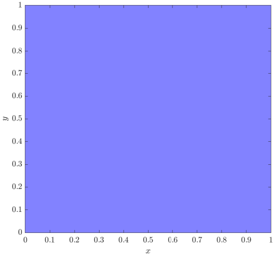
- Consider graphs $\mathbf{G}_n$ and $\mathbf{G}_m$ with $n \neq m$ nodes which are both sampled from the graphon $\mathbf{W}$
- Can upper bound the approximation error between $\mathbf{H}(\mathbf{S}_n)$ and $T_{\mathbf{H}}$. And between $\mathbf{H}(\mathbf{S}_m)$ and $T_{\mathbf{H}}$



n nodes



m nodes



Graphon $W(u, v) = p$

- By the triangle inequality, can upper bound the transferability error between $\mathbf{H}(\mathbf{S}_n)$ and $\mathbf{H}(\mathbf{S}_m)$

Theorem (Graph Filter Transferability)

Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $\mathbf{y}_m = \mathbf{H}(\mathbf{S}_m)\mathbf{x}_m$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y_m\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n} + \frac{1}{m}\right)\|X\| + A_x(A_hc + 2)\left(\frac{1}{n} + \frac{1}{m}\right) + 4A_hc\|X\|$$

- Filters that are more discriminative are more difficult to transfer
- If we fix n and m we observe emergence of a transferability vs. discriminability tradeoff

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graph Filter Transferability)

Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $\mathbf{y}_m = \mathbf{H}(\mathbf{S}_m)\mathbf{x}_m$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y_m\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n}+\frac{1}{m}\right)\|X\| + A_x(A_h c + 2)\left(\frac{1}{n}+\frac{1}{m}\right) + 4A_h c\|X\|$$

- Filters that are more discriminative are more difficult to transfer
- If we fix n and m we observe emergence of a transferability vs. discriminability tradeoff

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graph Filter Transferability)

Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $\mathbf{y}_m = \mathbf{H}(\mathbf{S}_m)\mathbf{x}_m$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y_m\| \leq 2A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n} + \frac{1}{m}\right)\|X\| + A_x(A_h c + 2)\left(\frac{1}{n} + \frac{1}{m}\right) + 4A_h c\|X\|$$

- Filters that are more discriminative are more difficult to transfer
- If we fix n and m we observe emergence of a transferability vs. discriminability tradeoff

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

Theorem (Graph Filter Transferability)

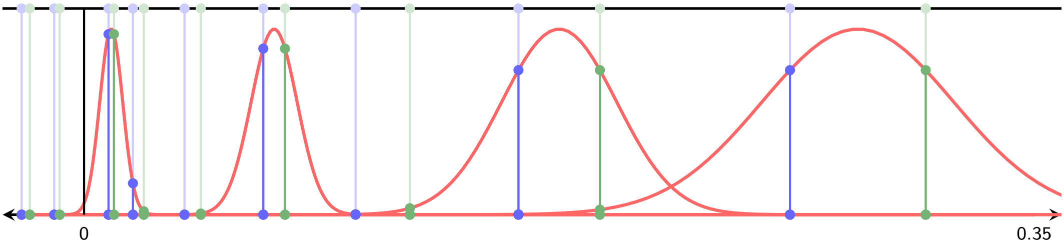
Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with convolution outputs $\mathbf{y}_n = \mathbf{H}(\mathbf{S}_n)\mathbf{x}_n$ and $\mathbf{y}_m = \mathbf{H}(\mathbf{S}_m)\mathbf{x}_m$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y_m\| \leq 2A_w \left(A_h + \pi \frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})} \right) \left(\frac{1}{n} + \frac{1}{m} \right) \|X\| + A_x(A_h c + 2) \left(\frac{1}{n} + \frac{1}{m} \right) + 4A_h c \|X\|$$

- Filters that are more discriminative are more difficult to transfer
- If we fix n and m we observe emergence of a transferability vs. discriminability tradeoff

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

- If filter is sharp near $\lambda = 0$, spectral components of $\lambda_j(\mathbf{S}_n)$ and $\lambda_j(\mathbf{W})$ are amplified differently



- Transferability and discriminability are not compatible for graph convolutional filters

Graph Neural Networks

- So far we have talked at length about graph convolutions and graphon convolutions

$\Rightarrow$ Graph Convolution

$$\mathbf{z}_n = \sum_{k=0}^{K-1} h_k \mathbf{S}_n^k \mathbf{x}_n$$

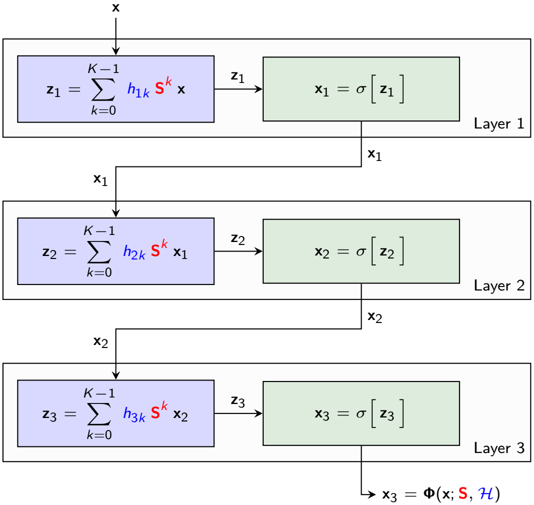
$\Rightarrow$ Graphon Convolution

$$Z = \sum_{k=0}^{K-1} h_k T_{\mathbf{W}}^{(k)} X$$

- But we have not talked much about graph neural networks and graphon neural networks

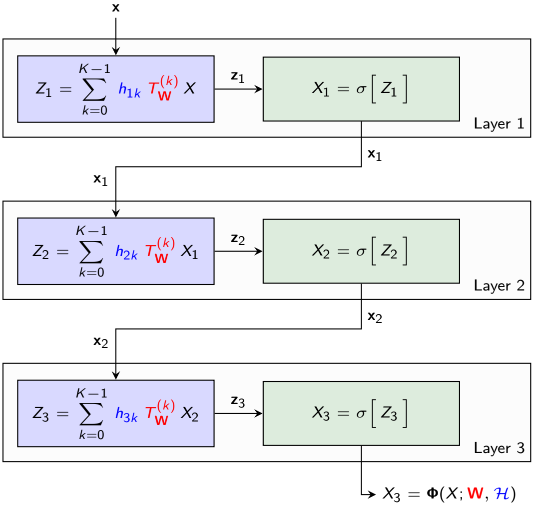
$\Rightarrow$ Graph and graphon NNs are a minor variation of graph convolutions and graphon convolutions

- A graph NN composes a cascade of layers
- Each of which are themselves compositions
  - $\Rightarrow$ Of graph convolutions $\mathbf{H}(\mathbf{S})$
  - $\Rightarrow$ With pointwise nonlinearities σ
- Define the learnable parameter set $\mathcal{H} = \{h_{kl}\}$
- GNN can be represented as $\mathbf{y} = \mathbf{\Phi}(\mathcal{H}; \mathbf{S}; \mathbf{x})$



Graphon Neural Networks

- A graphon NN (WNN) composes layers
- Each of which are themselves compositions
  - $\Rightarrow$ Of graphon convolutions $T_{\mathbf{H}}$
  - $\Rightarrow$ With pointwise nonlinearities σ
- Define the learnable parameter set $\mathcal{H} = \{h_{kl}\}$
- WNN can be represented as $Y = \mathbf{\Phi}(\mathcal{H}; \mathbf{W}; X)$



- The transferability properties of graph filters are inherited by graph neural networks

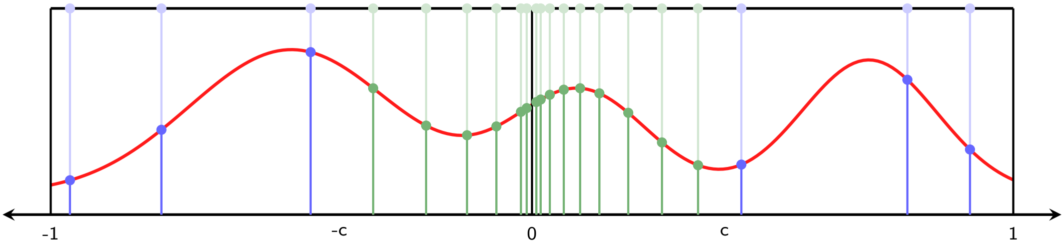
Theorem (GNN Transferability)

Consider graph signals $(\mathbf{S}_n, \mathbf{x}_n)$ and $(\mathbf{S}_m, \mathbf{x}_m)$ sampled from graphon signal (W, X) along with GNN outputs $\mathbf{y}_n = \Phi(\mathcal{H}; S_n, x_n)$ and $\mathbf{y}_m = \Phi(\mathcal{H}; S_m, x_m)$. The difference norm of the respective graphon induced signals is bounded by

$$\|Y_n - Y_m\| \leq$$
$$2LF^{L-1}A_w\left(A_h + \pi\frac{\max(B_{nc}, B_{mc})}{\min(\delta_{nc}, \delta_{mc})}\right)\left(\frac{1}{n} + \frac{1}{m}\right)\|X\| + A_x(A_hc + 2)\left(\frac{1}{n} + \frac{1}{m}\right) + 4LF^{L-1}A_hc\|X\|$$

L. Ruiz et al, *Transferability Properties of Graph Neural Networks*, https://arxiv.org/abs/2112.04629

- The difference in GNNs is that the nonlinearities scatter spectral components all over the spectrum

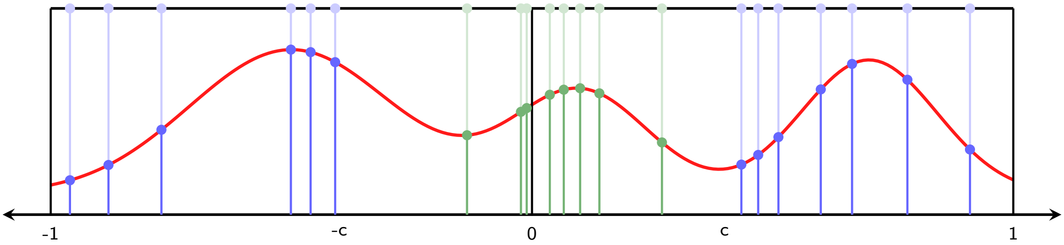


- Which allows increasing discriminability without hurting transferability. Hence:

  $\Rightarrow$ For the same level of transferability $\Rightarrow$ GNNs are more discriminative than graph filters

  $\Rightarrow$ For the same level of discriminability $\Rightarrow$ GNNs are more transferable than graph filters

- The difference in GNNs is that the nonlinearities scatter spectral components all over the spectrum

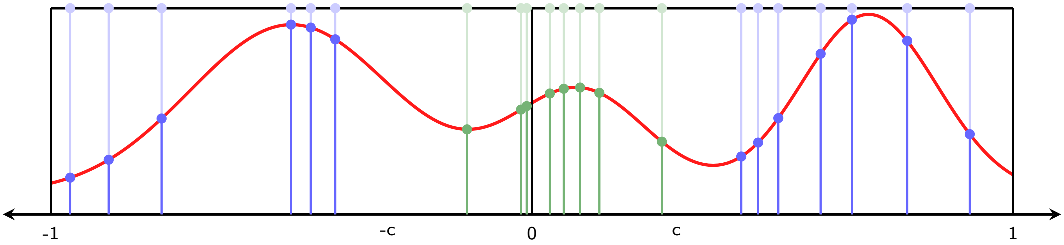


- Which allows increasing discriminability without hurting transferability. Hence:

$\Rightarrow$ For the same level of transferability $\Rightarrow$ GNNs are more discriminative than graph filters

$\Rightarrow$ For the same level of discriminability $\Rightarrow$ GNNs are more transferable than graph filters

- The difference in GNNs is that the nonlinearities scatter spectral components all over the spectrum



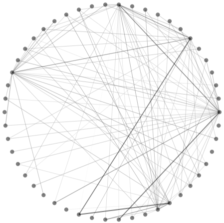
- Which allows increasing discriminability without hurting transferability. Hence:

$\Rightarrow$ For the same level of transferability $\Rightarrow$ GNNs are more discriminative than graph filters

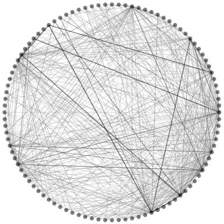
$\Rightarrow$ For the same level of discriminability $\Rightarrow$ GNNs are more transferable than graph filters

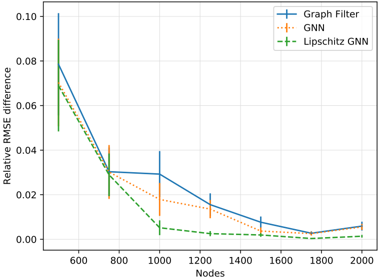
Transferability of Graph Neural Networks

- Transferability of graph neural networks observed empirically $\Rightarrow$ recommendation system



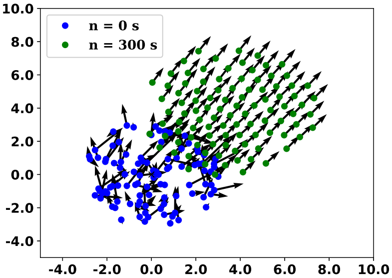
$\rightarrow$

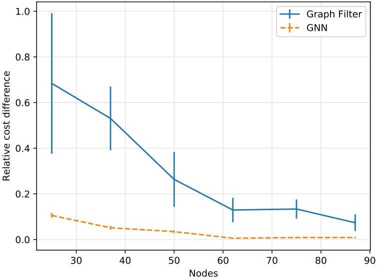




- Performance difference on training and target graphs decreases as size of training graph grows
- GNNs are more transferable than graph convolutional filters. Especially if their filters are Lipschitz

- Transferability of graph neural networks observed empirically $\Rightarrow$ decentralized robot control





- Performance difference on training and target graphs decreases as size of training graph grows
- GNNs are more transferable than graph convolutional filters. Especially if their filters are Lipschitz

Takeaways

Empirically: GNNs are more transferable than graph convolutional filters

Theoretically: GNNs are more transferable because of their mixing properties

- Empirical and theoretical evidence support using GNNs for large-scale graph machine learning