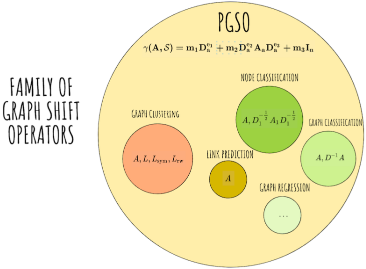
Learning Parametrised Graph Shift Operators

George Dasoulas* **Johannes Lutzeyer*** Michalis Vazirgiannis

* *Equal contribution.*

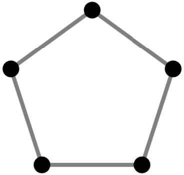


Graph Shift Operators

Definition

Graphs $G = (V, E)$ can be represented using:

- the *adjacency matrix* $A \in \{0,1\}^{n\times n}$ where $A_{ij} = 1$ iff $(i,j) \in E$.



Cycle C_5

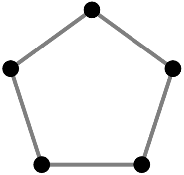
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Graph Shift Operators

Definition

Graphs $G = (V, E)$ can be represented using:

- the *adjacency matrix* $A \in \{0, 1\}^{n \times n}$ where $A_{ij} = 1$ iff $(i, j) \in E$.
- the *unnormalised graph Laplacian matrix* L is defined by $L = D - A$, where $D = \mathrm{diag}(A1_n)$.



Cycle C_5

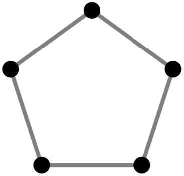
$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad L = \begin{pmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{pmatrix}$$

Graph Shift Operators

Definition

Graphs $G = (V, E)$ can be represented using:

- the *adjacency matrix* $A \in \{0,1\}^{n\times n}$ where $A_{ij} = 1$ iff $(i,j) \in E$.
- the *unnormalised graph Laplacian matrix* L is defined by $L = D - A$, where $D = \mathrm{diag}(A1_n)$.
- the *symmetric normalised graph Laplacian matrix* L_{sym} is defined by $L_{sym} = D^{-1/2}LD^{-1/2}$ and the *random-walk normalised Laplacian matrix* L_{rw} is defined as $L_{rw} = D^{-1}L$.



Cycle C_5

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix} \qquad L = \begin{pmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{pmatrix}$$

$$L_{sym} = \begin{pmatrix} 1.0 & -0.5 & 0.0 & 0.0 & -0.5 \\ -0.5 & 1.0 & -0.5 & 0.0 & 0.0 \\ 0.0 & -0.5 & 1.0 & -0.5 & 0.0 \\ 0.0 & 0.0 & -0.5 & 1.0 & -0.5 \\ -0.5 & 0.0 & 0.0 & -0.5 & 1.0 \end{pmatrix}$$

Graph Shift Operators

Definition

Graphs $G = (V, E)$ can be represented using:

- the *adjacency matrix* $A \in \{0, 1\}^{n \times n}$ where $A_{ij} = 1$ iff $(i, j) \in E$.
- the *unnormalised graph Laplacian matrix* L is defined by $L = D - A$, where $D = \mathrm{diag}(A1_n)$.
- the *symmetric normalised graph Laplacian matrix* L_{sym} is defined by $L_{sym} = D^{-1/2}LD^{-1/2}$ and the *random-walk normalised Laplacian matrix* L_{rw} is defined as $L_{rw} = D^{-1}L$.

- But there are more representations of graphs!

Definition (Sandryhaila and Moura (2013))

A matrix $S \in \mathbb{R}^{n \times n}$ is called a *Graph Shift Operator* (GSO) if it satisfies: $S_{ij} = 0$ for $i \neq j$ and $(i, j) \notin E$.

Graph Shift Operators

Definition

Graphs $G = (V, E)$ can be represented using:

- the *adjacency matrix* $A \in \{0, 1\}^{n \times n}$ where $A_{ij} = 1$ iff $(i, j) \in E$.
- the *unnormalised graph Laplacian matrix* L is defined by $L = D - A$, where $D = \mathrm{diag}(A1_n)$.
- the *symmetric normalised graph Laplacian matrix* L_{sym} is defined by $L_{sym} = D^{-1/2}LD^{-1/2}$ and the *random-walk normalised Laplacian matrix* L_{rw} is defined as $L_{rw} = D^{-1}L$.

- But there are more representations of graphs!

Definition (Sandryhaila and Moura (2013))

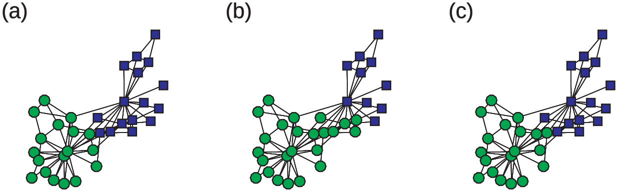
A matrix $S \in \mathbb{R}^{n \times n}$ is called a *Graph Shift Operator* (GSO) if it satisfies: $S_{ij} = 0$ for $i \neq j$ and $(i, j) \notin E$.

Note: the existence of an edge $(i, j) \in E$ does *not imply* a nonzero entry in the GSO, $S_{ij} \neq 0$. Hence, the correspondence between a GSO and a graph is not bijective in general.

GSOs in Representation Learning

GSOs in Representation Learning

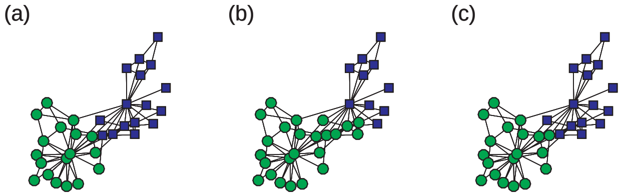
- Spectral clustering:



Spectral clustering of the karate network using A in (a), L in (b) and L_{rw} in (c) (Lutzeyer, 2020).

GSOs in Representation Learning

- Spectral clustering:



Spectral clustering of the karate network using A in (a), L in (b) and L_{rw} in (c) (Lutzeyer, 2020).

- Graph Neural Networks (GNNs), e.g., the Graph Convolutional Network (Kipf and Welling, 2017)

$$H^{(l+1)} = \sigma\left(D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}} H^{(l)} W^{(l)}\right). \qquad (1)$$

In message-passing schemes, the *sum-based aggregator* corresponds to the use of the adjacency matrix A.

Motivation

- When introducing the different standard GSO choices Butler and Chung (2017) state: *"No one matrix is best because each matrix has its own limitations in that there is some property which the matrix cannot always determine"*.

Motivation

- When introducing the different standard GSO choices Butler and Chung (2017) state: *"No one matrix is best because each matrix has its own limitations in that there is some property which the matrix cannot always determine"*.
- Graph signal processing literature: the GSO choice involves "different tradeoffs" and leads to different signal models (Deri and Moura, 2017; Ortega et al., 2018). Therefore, they recommend using *whichever GSO works best* in a particular analysis or learning task.

Motivation

- When introducing the different standard GSO choices Butler and Chung (2017) state: *"No one matrix is best because each matrix has its own limitations in that there is some property which the matrix cannot always determine"*.
- Graph signal processing literature: the GSO choice involves "different tradeoffs" and leads to different signal models (Deri and Moura, 2017; Ortega et al., 2018). Therefore, they recommend using *whichever GSO works best* in a particular analysis or learning task.

Research Questions

Question 1: *Is there a single optimal representation to encode graph structures or is the optimal representation task- and data-dependent?*

Motivation

- When introducing the different standard GSO choices Butler and Chung (2017) state: *"No one matrix is best because each matrix has its own limitations in that there is some property which the matrix cannot always determine"*.
- Graph signal processing literature: the GSO choice involves "different tradeoffs" and leads to different signal models (Deri and Moura, 2017; Ortega et al., 2018). Therefore, they recommend using *whichever GSO works best* in a particular analysis or learning task.

Research Questions

Question 1: *Is there a single optimal representation to encode graph structures or is the optimal representation task- and data-dependent?*
Question 2: *Can we learn such an optimal representation to encode graph structure in a numerically stable and computationally efficient way?*

Parametrised Graph Shift Operator

Parametrised Graph Shift Operators

Definition

We define the *Parametrised Graph Shift Operator (PGSO)*, denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{2}$$

where $A_a = A + aI_n$, $D_a = \mathrm{Diag}(A_a 1_n)$ and $S = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

Parametrised Graph Shift Operators

Definition

We define the *Parametrised Graph Shift Operator (PGSO)*, denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{2}$$

where $A_a = A + aI_n$, $D_a = \text{Diag}(A_a 1_n)$ and $S = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

Parametrised Graph Shift Operators

Definition

We define the *Parametrised Graph Shift Operator (PGSO)*, denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{2}$$

where $A_a = A + aI_n$, $D_a = \text{Diag}(A_a 1_n)$ and $S = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

Parametrised Graph Shift Operators

Definition

We define the *Parametrised Graph Shift Operator (PGSO)*, denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{2}$$

where $A_a = A + aI_n$, $D_a = \mathrm{Diag}(A_a 1_n)$ and $S = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

Parametrised Graph Shift Operators

Definition

We define the *Parametrised Graph Shift Operator (PGSO)*, denoted by $\gamma(A, \mathcal{S})$, as

$$\gamma(A, \mathcal{S}) = m_1 D_a^{e_1} + m_2 D_a^{e_2} A_a D_a^{e_3} + m_3 I_n, \tag{2}$$

where $A_a = A + aI_n,\ D_a = \mathrm{Diag}(A_a 1_n)$ and $S = (m_1, m_2, m_3, e_1, e_2, e_3, a)$.

$\mathcal{S} = (m_1,\ m_2,\ m_3,\ e_1,\ e_2,\ e_3,\ a)$	**Operator**	**Description**
$(0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0)$	A	Adjacency matrix and Summation Aggregation Operator of GNNs
$(1,\ -1,\ 0,\ 1,\ 0,\ 0,\ 0)$	$D - A$	Unnormalised Laplacian matrix L
$(1,\ 1,\ 0,\ 1,\ 0,\ 0,\ 0)$	$D + A$	Signless Laplacian matrix Q (Cvetkovic et al., 1997)
$(0,\ -1,\ 1,\ 0,\ -1,\ 0,\ 0)$	$I_n - D^{-1}A$	Random-walk Normalised Laplacian L_{rw}
$(0,\ -1,\ 1,\ 0,\ -\frac{1}{2},\ -\frac{1}{2},\ 0)$	$I_n - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$	Symmetric Normalised Laplacian L_{sym}
$(0,\ 1,\ 0,\ 0,\ -\frac{1}{2},\ -\frac{1}{2},\ 1)$	$D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}}$	Normalised Adjacency matrix of GCNs (Kipf and Welling, 2017)
$(0,\ 1,\ 0,\ 0,\ -1,\ 0,\ 0)$	$D^{-1}A$	Mean Aggregation Operator of GNNs (Xu et al., 2019)

PGSO in Graph Neural Networks

Notation:

- Let a GNN model be denoted by $\mathcal{M}(\phi(A), X)$.
- Non-parametrised function of A $\phi(A) : [0,1]^{n\times n} \rightarrow \mathbb{R}^{n\times n}$.
- Attribute matrix $X \in \mathbb{R}^{n\times d}$ (in case of an attributed graph).
- Number of aggregation layers K.

Utilization of PGSO in GNNs

1. **GNN-PGSO**: $\mathcal{M}(\phi(A), X) \rightarrow \mathcal{M}'(\gamma(A, \mathcal{S}), X)$.
2. **GNN-mPGSO (multi-PGSO)**: $\mathcal{M}(\phi(A), X) \rightarrow \mathcal{M}''(\gamma^{[K]}(A, \mathcal{S}^{[K]}), X)$, where $\gamma^{[K]}(A, \mathcal{S}^{[K]}) = [\gamma(A, \mathcal{S}^1), ..., \gamma(A, \mathcal{S}^K)]$.

PGSO in Graph Neural Networks

Notation:

- Let a GNN model be denoted by $\mathcal{M}(\phi(A), X)$.
- Non-parametrised function of A $\phi(A) : [0,1]^{n\times n} \rightarrow \mathbb{R}^{n\times n}$.
- Attribute matrix $X \in \mathbb{R}^{n\times d}$ (in case of an attributed graph).
- Number of aggregation layers K.

Utilization of PGSO in GNNs

1. **GNN-PGSO**: $\mathcal{M}(\phi(A), X) \rightarrow \mathcal{M}'(\gamma(A, \mathcal{S}), X)$.
2. **GNN-mPGSO (multi-PGSO)**: $\mathcal{M}(\phi(A), X) \rightarrow \mathcal{M}''(\gamma^{[K]}(A, \mathcal{S}^{[K]}), X)$, where $\gamma^{[K]}(A, \mathcal{S}^{[K]}) = [\gamma(A, \mathcal{S}^1), ..., \gamma(A, \mathcal{S}^K)]$.

- In simple words, we **replace** the GSO used in a GNN model by $\gamma(A, \mathcal{S})$.

PGSO in Graph Neural Networks

Notation:

- Let a GNN model be denoted by $\mathcal{M}(\phi(A), X)$.
- Non-parametrised function of A $\phi(A) : [0,1]^{n\times n} \to \mathbb{R}^{n\times n}$.
- Attribute matrix $X \in \mathbb{R}^{n\times d}$ (in case of an attributed graph).
- Number of aggregation layers K.

Utilization of PGSO in GNNs

1. **GNN-PGSO**: $\mathcal{M}(\phi(A), X) \to \mathcal{M}'(\gamma(A, \mathcal{S}), X)$.
2. **GNN-mPGSO (multi-PGSO)**: $\mathcal{M}(\phi(A), X) \to \mathcal{M}''(\gamma^{[K]}(A, \mathcal{S}^{[K]}), X)$, where $\gamma^{[K]}(A, \mathcal{S}^{[K]}) = [\gamma(A, \mathcal{S}^1), ..., \gamma(A, \mathcal{S}^K)]$.

- In simple words, we **replace** the GSO used in a GNN model by $\gamma(A, \mathcal{S})$.
- In convolutional architectures, the graph convolution is expressed as a matrix multiplication, involving the GSO $\to$ **explicit** replacement.

PGSO in Graph Neural Networks

Notation:

- Let a GNN model be denoted by $\mathcal{M}(\phi(A), X)$.
- Non-parametrised function of A $\phi(A) : [0,1]^{n \times n} \rightarrow \mathbb{R}^{n \times n}$.
- Attribute matrix $X \in \mathbb{R}^{n \times d}$ (in case of an attributed graph).
- Number of aggregation layers K.

Utilization of PGSO in GNNs

1. **GNN-PGSO**: $\mathcal{M}(\phi(A), X) \rightarrow \mathcal{M}'(\gamma(A, \mathcal{S}), X)$.
2. **GNN-mPGSO (multi-PGSO)**: $\mathcal{M}(\phi(A), X) \rightarrow \mathcal{M}''(\gamma^{[K]}(A, \mathcal{S}^{[K]}), X)$, where $\gamma^{[K]}(A, \mathcal{S}^{[K]}) = [\gamma(A, \mathcal{S}^1), ..., \gamma(A, \mathcal{S}^K)]$.

- In simple words, we **replace** the GSO used in a GNN model by $\gamma(A, \mathcal{S})$.
- In convolutional architectures, the graph convolution is expressed as a matrix multiplication, involving the GSO $\rightarrow$ **explicit** replacement.
- In message-passing architectures, the replacement is performed **implictly** as a neighborhood aggregation step.

Convolutions and message-passing

- Examples of utilisation of GNN-PGSO models ...

Convolutions and message-passing

- Examples of utilisation of GNN-PGSO models ...

1. **GCN (Kipf & Welling, 2017)**: The propagation rule is

$$H^{(l+1)} = \sigma\big(D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}} H^{(l)} W^{(l)}\big),$$

where $W^{(l)}$ is a weight matrix and σ is a non-linear activation function. The GCN-PGSO and GCN-mPGSO models are defined, respectively, as

$$H^{(l+1)} = \sigma\big(\gamma(A, \mathcal{S}) H^{(l)} W^{(l)}\big) \text{ and } H^{(l+1)} = \sigma\big(\gamma(A, \mathcal{S}^l) H^{(l)} W^{(l)}\big).$$

Convolutions and message-passing

- Examples of utilisation of GNN-PGSO models ...

1. **GCN (Kipf & Welling, 2017)**: The propagation rule is

$$H^{(l+1)} = \sigma\left(D_1^{-\frac{1}{2}} A_1 D_1^{-\frac{1}{2}} H^{(l)} W^{(l)}\right),$$

where $W^{(l)}$ is a weight matrix and σ is a non-linear activation function. The GCN-PGSO and GCN-mPGSO models are defined, respectively, as

$$H^{(l+1)} = \sigma\left(\gamma(A, \mathcal{S}) H^{(l)} W^{(l)}\right) \text{ and } H^{(l+1)} = \sigma\left(\gamma(A, \mathcal{S}^l) H^{(l)} W^{(l)}\right).$$

2. **GIN (Xu et al., 2019)**: The propagation rule is

$$h_i^{(l+1)} = \sigma\left(h_i^{(l)} W^{(l)} + \sum_{j: v_j \in \mathcal{N}(v_i)} h_j^{(l)} W^{(l)}\right).$$

The GIN-PGSO model is defined as

$$h_i^{(l+1)} = \sigma\left(\left(m_1 \left(D_a\right)_i^{e_1} + m_3\right) h_i^{(l)} W^{(l)} + \sum_{j: v_j \in \mathcal{N}(v_i)} \epsilon_{ij} h_j^{(l)} W^{(l)}\right),$$

where ϵ_{ij} are edge weights defined as $\epsilon_{ij} = m_2 \left(D_a\right)_i^{e_2} \left(D_a\right)_j^{e_3}$.

Spectral Analysis: A real spectrum

Why study the spectrum of $\gamma(A, \mathcal{S})$ *?*

Spectral Analysis: A real spectrum

Why study the spectrum of $\gamma(A, \mathcal{S})$ *?*

- In the field of Spectral Graph Theory it has been established that the study of spectral properties of the GSOs can yield a great amount of insight.

Spectral Analysis: A real spectrum

Why study the spectrum of $\gamma(A, \mathcal{S})$ *?*

- In the field of Spectral Graph Theory it has been established that the study of spectral properties of the GSOs can yield a great amount of insight.
- GNNs can be motivated as performing spectral filtering operations at each message passing operation on the input graph.

Spectral Analysis: A real spectrum

Why study the spectrum of $\gamma(A, \mathcal{S})$?

- In the field of Spectral Graph Theory it has been established that the study of spectral properties of the GSOs can yield a great amount of insight.
- GNNs can be motivated as performing spectral filtering operations at each message passing operation on the input graph.

So, let's begin:

Theorem

$\gamma(A, \mathcal{S})$ has real eigenvalues and a set of real eigenvectors independent of the parameters chosen in $\mathcal{S}$.

Spectral Analysis: Bounds on the spectral support

Theorem

Let $C_i = m_1(d_i + a)^{e_1} + m_2(d_i + a)^{e_2+e_3} a + m_3$ and $R_i = |m_2|(d_i + a)^{e_2+e_3} d_i$, where d_i denotes the degree of node v_i. Furthermore, we denote eigenvalues of $\gamma(A, \mathcal{S})$ by $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then, for all $j \in \{1, \ldots, n\}$,

$$\lambda_j \in \left[\min_{i \in \{1,\ldots,n\}} (C_i - R_i), \max_{i \in \{1,\ldots,n\}} (C_i + R_i) \right]. \tag{3}$$

Spectral Analysis: Bounds on the spectral support

Theorem

Let $C_i = m_1(d_i + a)^{e_1} + m_2(d_i + a)^{e_2+e_3} a + m_3$ and $R_i = |m_2|(d_i + a)^{e_2+e_3} d_i$, where d_i denotes the degree of node v_i. Furthermore, we denote eigenvalues of $\gamma(A, \mathcal{S})$ by $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then, for all $j \in \{1, \ldots, n\}$,

$$\lambda_j \in \left[\min_{i \in \{1,\ldots,n\}} (C_i - R_i), \max_{i \in \{1,\ldots,n\}} (C_i + R_i) \right]. \tag{3}$$

- For the parametrisation of $\gamma(A, \mathcal{S})$ corresponding to the adjacency matrix, we obtain $C_i = 0$ and $R_i = d_i$. R_i is clearly maximised by the maximum degree and therefore, from (3) the spectral support of A is equal to $[-d_{\max}, d_{\max}]$, as required.

Spectral Analysis: Bounds on the spectral support

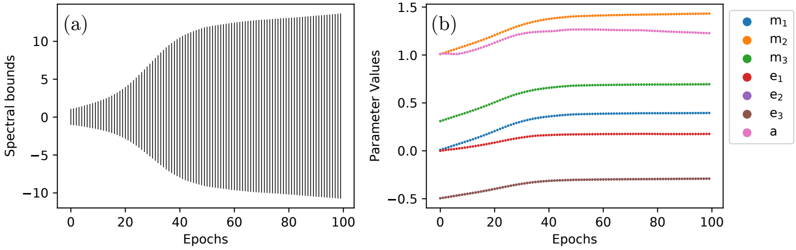
Theorem

Let $C_i = m_1(d_i + a)^{e_1} + m_2(d_i + a)^{e_2+e_3} a + m_3$ and $R_i = |m_2|(d_i + a)^{e_2+e_3} d_i$, where d_i denotes the degree of node v_i. Furthermore, we denote eigenvalues of $\gamma(A, \mathcal{S})$ by $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. Then, for all $j \in \{1, \ldots, n\}$,

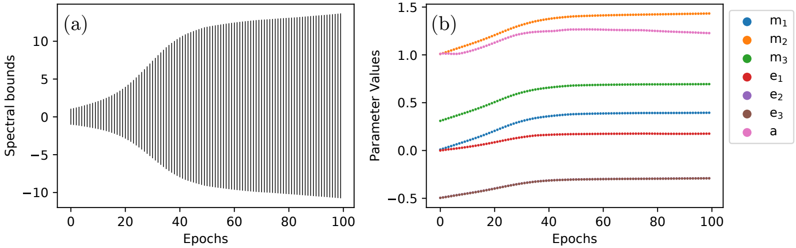
$$\lambda_j \in \left[\min_{i \in \{1,\ldots,n\}} (C_i - R_i), \max_{i \in \{1,\ldots,n\}} (C_i + R_i) \right]. \tag{3}$$

- For the parametrisation of $\gamma(A, \mathcal{S})$ corresponding to the adjacency matrix, we obtain $C_i = 0$ and $R_i = d_i$. R_i is clearly maximised by the maximum degree and therefore, from (3) the spectral support of A is equal to $[-d_{\max}, d_{\max}]$, as required.
- For the message passing operator in the GCN we obtain $C_i = 1/(d_i + 1)$ and $R_i = d_i/(d_i + 1)$. Therefore, from (3) the spectral support of the Kipf and Welling operator is restricted to lie within $[-(d_{\max} - 1)/(d_{\max} + 1), 1]$, the lower bound of this interval tends to -1 as $d_{\max} \to \infty$.

Spectral Analysis: empirical observation

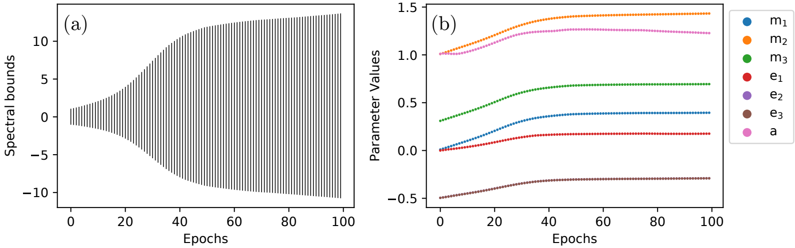


Spectral Analysis: empirical observation



- Surprisingly, the spectral support of the PGSO remains centered at 0 throughout training. It is nice to observe that this desirable property of the "renormalised" operator used by Kipf and Welling (2017) is preserved throughout training.

Spectral Analysis: empirical observation



- Surprisingly, the spectral support of the PGSO remains centered at 0 throughout training. It is nice to observe that this desirable property of the "renormalised" operator used by Kipf and Welling (2017) is preserved throughout training.
- We observe the parameters of the PGSO to be smoothly varying throughout training.

Experiments

Sparsity Interpretation of $\gamma(A, \mathcal{S})$: SBM use-case

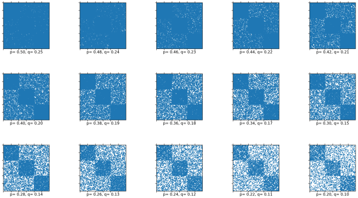
Motivation: In different tasks, we usually have either dense or sparse graphs. How does $\gamma(A, \mathcal{S})$ behave when applied to graphs with varying sparsity levels?

Sparsity Interpretation of $\gamma(A, \mathcal{S})$: SBM use-case

Motivation: In different tasks, we usually have either dense or sparse graphs. How does $\gamma(A, \mathcal{S})$ behave when applied to graphs with varying sparsity levels?

Setup:

- 15 decreasing p, q combinations.
- Fixed detectability level.
- $\forall(p, q)$ 25 sampled graphs with 3 200-node communities.
- Node classification (Dwivedi et al., 2020).
- 3-layer GCN-PGSO model.



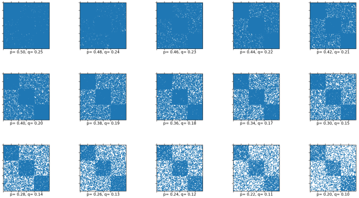
SBM adjacency matrices.

Sparsity Interpretation of $\gamma(A, \mathcal{S})$: SBM use-case

Motivation: In different tasks, we usually have either dense or sparse graphs. How does $\gamma(A, \mathcal{S})$ behave when applied to graphs with varying sparsity levels?

Setup:

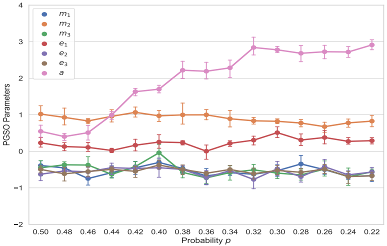
- 15 decreasing p, q combinations.
- Fixed detectability level.
- $\forall (p, q)$ 25 sampled graphs with 3 200-node communities.
- Node classification (Dwivedi et al., 2020).
- 3-layer GCN-PGSO model.



SBM adjacency matrices.

Remarks:

- The additive parameter a increases with the increasing sparsity.
- The remaining parameters remain close to constant.
- confirms the positive impact of GSO regularisation (Dall'Amico et al., 2020; Qin and Rohe, 2013)



Sensitivity analysis of $\gamma(A, \mathcal{S})$ to initialisations

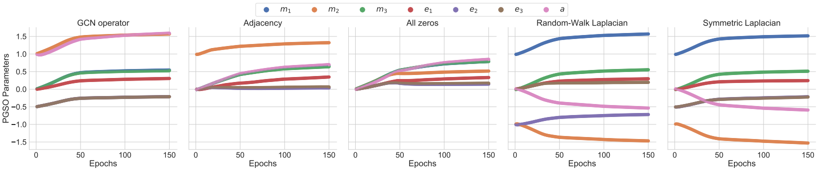
Question

- How sensitive is the model performance to the PGSO parameter initialisation?

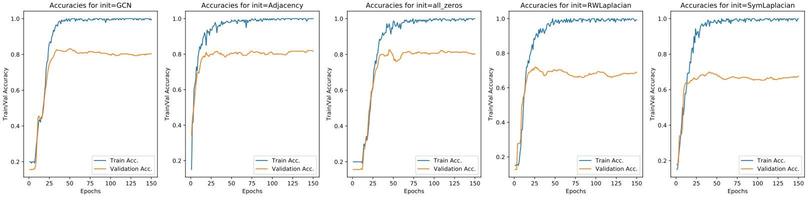
Sensitivity analysis of $\gamma(A, \mathcal{S})$ to initialisations

Question

- How sensitive is the model performance to the PGSO parameter initialisation?

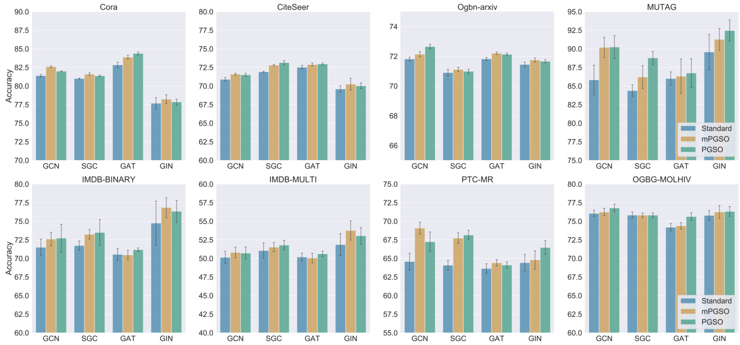


- Parameters $(m_1, m_3, e_1, e_2, e_3)$ **monotonically increase** until **convergence**.
- Parameters (m_2, a) show a "mirroring" behaviour (Laplacians vs others).
- The accuracy is **not very sensitive** to the initialisations.



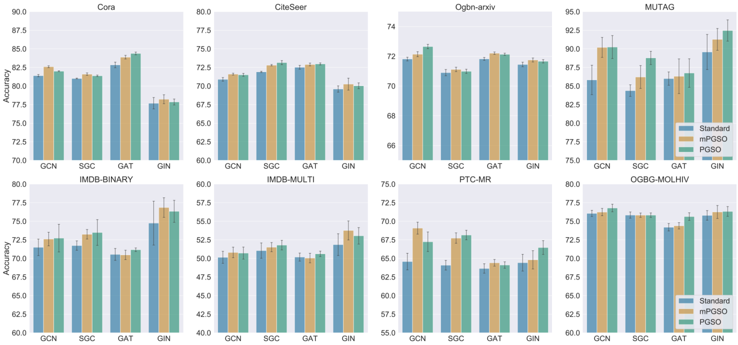
Results on Real-world datasets

- Evaluation in 8 **node classification** and **graph classification** tasks.
- Model design with 4 architectures: GCN, SGC, GAT and GIN models.
- 3 GSO variants for each model: **Standard**, **PGSO** and **mPGSO**.



Results on Real-world datasets

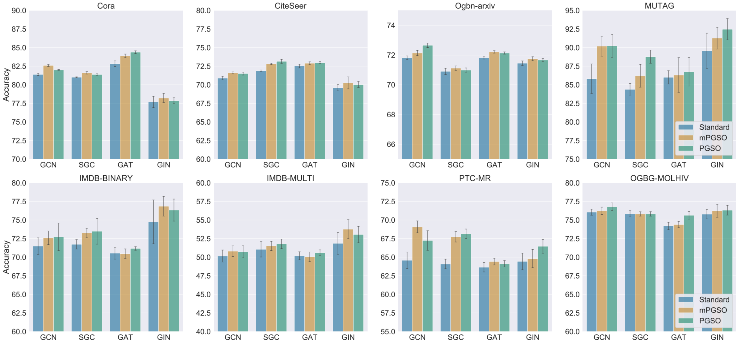
- Evaluation in 8 **node classification** and **graph classification** tasks.
- Model design with 4 architectures: GCN, SGC, GAT and GIN models.
- 3 GSO variants for each model: **Standard**, **PGSO** and **mPGSO**.



- For all datasets and architectures, the incorporation of the PGSO and/or the mPGSO **enhances** the model performance.

Results on Real-world datasets

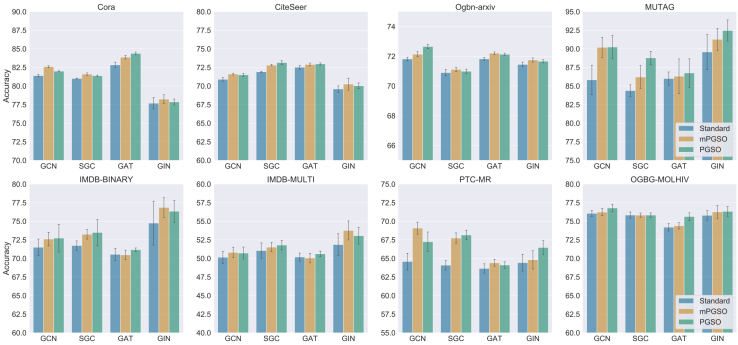
- Evaluation in 8 **node classification** and **graph classification** tasks.
- Model design with 4 architectures: GCN, SGC, GAT and GIN models.
- 3 GSO variants for each model: **Standard**, **PGSO** and **mPGSO**.



- For all datasets and architectures, the incorporation of the PGSO and/or the mPGSO **enhances** the model performance.
- The impact of PGSO is **higher** in graph classification tasks.

Results on Real-world datasets

- Evaluation in 8 **node classification** and **graph classification** tasks.
- Model design with 4 architectures: GCN, SGC, GAT and GIN models.
- 3 GSO variants for each model: **Standard**, **PGSO** and **mPGSO**.



- For all datasets and architectures, the incorporation of the PGSO and/or the mPGSO **enhances** the model performance.
- The impact of PGSO is **higher** in graph classification tasks.
- There is **no clear** winner between PGSO and mPGSO.

Conclusions

We conclude:

- We proposed a parametrised graph shift operator (PGSO) that encodes graph structures.

Conclusions

We conclude:

- We proposed a parametrised graph shift operator (PGSO) that encodes graph structures.
- We showed that the PGSO has real eigenvalues and a set of real eigenvectors. In addition, we proved spectral bounds on the PGSO.

Conclusions

We conclude:

- We proposed a parametrised graph shift operator (PGSO) that encodes graph structures.
- We showed that the PGSO has real eigenvalues and a set of real eigenvectors. In addition, we proved spectral bounds on the PGSO.
- We demonstrated that the PGSO can be included in the GNN model training and improves their performance on real world datasets.

Conclusions

We conclude:

- We proposed a parametrised graph shift operator (PGSO) that encodes graph structures.
- We showed that the PGSO has real eigenvalues and a set of real eigenvectors. In addition, we proved spectral bounds on the PGSO.
- We demonstrated that the PGSO can be included in the GNN model training and improves their performance on real world datasets.
- A study on stochastic blockmodel graphs demonstrated the ability of the PGSO to automatically adapt to networks with varying sparsity.

ArXiv: https://arxiv.org/abs/2101.10050
Code: https://github.com/gdasoulas/PGSO

The two research questions

In answer to our two research questions:

The two research questions

In answer to our two research questions:

- Our experimental results have shown that the optimal representation of graph structures is task- and data-dependent.

The two research questions

In answer to our two research questions:

- Our experimental results have shown that the optimal representation of graph structures is task- and data-dependent.
- We have furthermore found that PGSO parameters can be incorporated in the training of GNNs and leads to numerically stable learning and message passing operators.

Thank you for your attention!

@GDasoulas, @JLutzeyer & @mvazirg

References

L. Dall'Amico, R. Couillet & N. Tremblay, "Optimal Laplacian regularization for sparse spectral community detection," *ICASSP*, 2020.

S. Butler & F. Chung, "Spectral graph theory," In: L. Hogben (ed) *Handbook of linear algebra (2nd edition)*, Boca Raton, FL: CRC Press, pp. 47/1—47/14, 2017.

D. Cvetkovic, R. Rowlinson & S. Simic, *Eigenspaces of graphs*, Cambridge, UK: Cambridge University Press, 1997.

J. A. Deri & J. M. F. Moura, "Spectral projector-based graph Fourier transforms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, pp. 785–795, 2017.

V.P. Dwivedi, C.K. Joshi, T. Laurent, Y. Bengio, X. Bresson, "Benchmarking Graph Neural Networks," *arXiv:2003.00982*, 2020.

Thomas N. Kipf & M. Welling. "Semi-supervised classification with graph convolutional networks" *ICLR*, 2017.

J. Lutzeyer, *Network Representation Matrices and their Eigenproperties: A Comparative Study,* PhD thesis: Imperial College London, 2020.

A. Ortega, P. Frossard, J. Kovacevic, J. M. F. Moura & P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, pp. 808–828, 2018.

T. Qin & K. Rohe, "Regularized Spectral Clustering under the Degree-Corrected Stochastic Blockmodel," *Advances in neural information processing systems (NIPS)*, pp. 3120–3128, 2013.

A. Sandryhaila & J. M. F. Moura "Discrete signal processing on graphs," *IEEE Transactions on Signal Processing*, vol. 61, pp. 1644–1656, 2013.

K. Xu, W. Hu, J. Leskovec & S. Jegelka, "How powerful are graph neural networks?," *ICLR*, 2019.