S-Bandwidth as an Indicator of Perfect State Transfer on Quantum Networks

Nathaniel Johnston[a], Sarah Plosker[b], and **Luis M. B. Varona**[a]

[a]*Mount Allison University* [b]*Brandon University*

Atlantic Undergraduate Physics and Astronomy Conference 2025
Memorial University, St. John's, NL

February 1, 2025

Quantum Information Systems

Whereas classical computers use **bits** (0 or 1), quantum computers use superposed **qubits**. Quantum systems can solve *some* problems far faster—but they are highly susceptible to decoherence.

- A qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is a **superposition** of the classical basis states $|0\rangle$ and $|1\rangle$, where $|\alpha|^2 + |\beta|^2 = 1$ with $\alpha, \beta \in \mathbb{C}$
- Recall the famous thought experiment **Schrödinger's cat**:

Figure 1: My cat, **Ash**. (She may be both **awake** and **asleep**...)

Quantum Information Systems

Whereas classical computers use **bits** (0 or 1), quantum computers use superposed **qubits**. Quantum systems can solve *some* problems far faster—but they are highly susceptible to decoherence.

- A qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is a **superposition** of the classical basis states $|0\rangle$ and $|1\rangle$, where $|\alpha|^2 + |\beta|^2 = 1$ with $\alpha, \beta \in \mathbb{C}$
- Recall the famous thought experiment **Schrödinger's cat**:

Figure 1: My cat, **Ash**. (She may be both **awake** and **asleep**...)

Quantum Information Systems

Whereas classical computers use **bits** (0 or 1), quantum computers use superposed **qubits**. Quantum systems can solve *some* problems far faster—but they are highly susceptible to decoherence.

- A qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is a **superposition** of the classical basis states $|0\rangle$ and $|1\rangle$, where $|\alpha|^2 + |\beta|^2 = 1$ with $\alpha, \beta \in \mathbb{C}$
- Recall the famous thought experiment **Schrödinger's cat**:

Figure 1: My cat, **Ash**. (She may be both **awake** and **asleep**...)

Quantum Information Systems

We use undirected graphs to represent **quantum spin networks** of coupled qubits (typically realized by electrons or photons).

- **Vertices** represent qubit particles, **edges** represent couplings, and **edge weights** represent coupling constants/strengths
- Movement of information (contained in a particle's quantum state) from one vertex to another is called **state transfer**
- A network with quantum couplings between all but two qubits:

Figure 2: The complete multipartite graph $K_{1,1,1,2}$ on 5 vertices.

Quantum Information Systems

We use undirected graphs to represent **quantum spin networks** of coupled qubits (typically realized by electrons or photons).

- **Vertices** represent qubit particles, **edges** represent couplings, and **edge weights** represent coupling constants/strengths
- Movement of information (contained in a particle's quantum state) from one vertex to another is called **state transfer**
- A network with quantum couplings between all but two qubits:

Figure 2: The complete multipartite graph $K_{1,1,1,2}$ on 5 vertices.

Quantum Information Systems

We use undirected graphs to represent **quantum spin networks** of coupled qubits (typically realized by electrons or photons).

- **Vertices** represent qubit particles, **edges** represent couplings, and **edge weights** represent coupling constants/strengths
- Movement of information (contained in a particle's quantum state) from one vertex to another is called **state transfer**
- A network with quantum couplings between all but two qubits:

Figure 2: The complete multipartite graph $K_{1,1,1,2}$ on 5 vertices.

Quantum Information Systems

We use undirected graphs to represent **quantum spin networks** of coupled qubits (typically realized by electrons or photons).

- **Vertices** represent qubit particles, **edges** represent couplings, and **edge weights** represent coupling constants/strengths
- Movement of information (contained in a particle's quantum state) from one vertex to another is called **state transfer**
- A network with quantum couplings between all but two qubits:

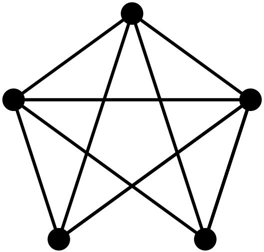

Figure 2: The complete multipartite graph $K_{1,1,1,2}$ on 5 vertices.

Perfect State Transfer

Definition (Perfect state transfer)

*Let G be a quantum network, and let $\Psi_t : V(G) \to \mathbb{C}$ be the wave function of $u \in V(G)$ after a period $t \geq 0$ of unitary evolution (so $\Psi_0(u) = 1$ and $\Psi_0(x) = 0\ \forall x \neq u$). We say there is **perfect state transfer (PST)** from u to $v \neq u$ if $\exists T > 0$ so that $|\Psi_T(v)| = 1$.*

- **i.e.,** when 100% of qubit u's initial information state is transferred to qubit v without physical particle motion
- To test for PST *specifically* from u to v, we can use the **discrete Schrödinger's equation** on G for u: $\frac{d}{dt}\Psi_t = iH\Psi_t$
- But using wave equations to determine whether PST occurs on G between *any* pair of qubits can get very, very messy...
- ...so we often turn to **matrix mechanics**!

Perfect State Transfer

Definition (Perfect state transfer)

Let G be a quantum network, and let $\Psi_t : V(G) \to \mathbb{C}$ be the wave function of $u \in V(G)$ after a period $t \geq 0$ of unitary evolution (so $\Psi_0(u) = 1$ and $\Psi_0(x) = 0\ \forall x \neq u$). We say there is ***perfect state transfer (PST)*** *from u to $v \neq u$ if $\exists T > 0$ so that $|\Psi_T(v)| = 1$.*

- **i.e.,** when 100% of qubit u's initial information state is transferred to qubit v without physical particle motion
- To test for PST *specifically* from u to v, we can use the **discrete Schrödinger's equation** on G for u: $\frac{d}{dt}\Psi_t = iH\Psi_t$
- But using wave equations to determine whether PST occurs on G between *any* pair of qubits can get very, very messy...
- ...so we often turn to **matrix mechanics**!

Perfect State Transfer

Definition (Perfect state transfer)

*Let G be a quantum network, and let $\Psi_t : V(G) \to \mathbb{C}$ be the wave function of $u \in V(G)$ after a period $t \geq 0$ of unitary evolution (so $\Psi_0(u) = 1$ and $\Psi_0(x) = 0 \; \forall x \neq u$). We say there is **perfect state transfer (PST)** from u to $v \neq u$ if $\exists T > 0$ so that $|\Psi_T(v)| = 1$.*

- **i.e.,** when 100% of qubit u's initial information state is transferred to qubit v without physical particle motion
- To test for PST *specifically* from u to v, we can use the **discrete Schrödinger's equation** on G for u: $\frac{d}{dt}\Psi_t = iH\Psi_t$
- But using wave equations to determine whether PST occurs on G between *any* pair of qubits can get very, very messy...
- ...so we often turn to **matrix mechanics**!

Perfect State Transfer

Definition (Perfect state transfer)

*Let G be a quantum network, and let $\Psi_t : V(G) \to \mathbb{C}$ be the wave function of $u \in V(G)$ after a period $t \geq 0$ of unitary evolution (so $\Psi_0(u) = 1$ and $\Psi_0(x) = 0\ \forall x \neq u$). We say there is **perfect state transfer (PST)** from u to $v \neq u$ if $\exists T > 0$ so that $|\Psi_T(v)| = 1$.*

- **i.e.,** when 100% of qubit u's initial information state is transferred to qubit v without physical particle motion
- To test for PST *specifically* from u to v, we can use the **discrete Schrödinger's equation** on G for u: $\frac{d}{dt}\Psi_t = iH\Psi_t$
- But using wave equations to determine whether PST occurs on G between *any* pair of qubits can get very, very messy...
- ...so we often turn to **matrix mechanics**!

Perfect State Transfer

Definition (Perfect state transfer)

*Let G be a quantum network, and let $\Psi_t : V(G) \to \mathbb{C}$ be the wave function of $u \in V(G)$ after a period $t \geq 0$ of unitary evolution (so $\Psi_0(u) = 1$ and $\Psi_0(x) = 0\ \forall x \neq u$). We say there is **perfect state transfer (PST)** from u to $v \neq u$ if $\exists T > 0$ so that $|\Psi_T(v)| = 1$.*

- **i.e.,** when 100% of qubit u's initial information state is transferred to qubit v without physical particle motion
- To test for PST *specifically* from u to v, we can use the **discrete Schrödinger's equation** on G for u: $\frac{d}{dt}\Psi_t = iH\Psi_t$
- But using wave equations to determine whether PST occurs on G between *any* pair of qubits can get very, very messy...
- ...so we often turn to **matrix mechanics**!

Perfect State Transfer

It is easier to search for PST between arbitrary $x, y \in V(G)$ with **unitary operators** rather than set up $|V(G)|$ wave equations.

- **e.g.,** the **transition matrix** $U_t = e^{itA(G)}$ models **unitary evolution** on G in the total absence of noise (where $A(G)$ is the adjacency matrix of G)
- There is PST on G iff $\exists T > 0$ and $x, y \in V(G)$ so that $|\langle x|U_T|y\rangle| = 1$, where $|x\rangle, |y\rangle$ are the **initial states** of x, y
- Now we only need the fixed initial states $\{|x\rangle : x \in V(G)\}$ instead of a new time-variant wave function for each particle
- Perfectly unitary evolution (and hence PST) is impossible in practice due to quantum noise, but we *can* achieve **pretty good state transfer (PGST)** (as high as $>97\%$ fidelity!)

Perfect State Transfer

It is easier to search for PST between arbitrary $x, y \in V(G)$ with **unitary operators** rather than set up $|V(G)|$ wave equations.

- **e.g.,** the **transition matrix** $U_t = e^{itA(G)}$ models **unitary evolution** on G in the total absence of noise (where $A(G)$ is the adjacency matrix of G)
- There is PST on G iff $\exists T > 0$ and $x, y \in V(G)$ so that $|\langle x|U_T|y\rangle| = 1$, where $|x\rangle, |y\rangle$ are the **initial states** of x, y
- Now we only need the fixed initial states $\{|x\rangle : x \in V(G)\}$ instead of a new time-variant wave function for each particle
- Perfectly unitary evolution (and hence PST) is impossible in practice due to quantum noise, but we *can* achieve **pretty good state transfer (PGST)** (as high as $>97\%$ fidelity!)

Perfect State Transfer

It is easier to search for PST between arbitrary $x, y \in V(G)$ with **unitary operators** rather than set up $|V(G)|$ wave equations.

- **e.g.,** the **transition matrix** $U_t = e^{itA(G)}$ models **unitary evolution** on G in the total absence of noise (where $A(G)$ is the adjacency matrix of G)
- There is PST on G iff $\exists T > 0$ and $x, y \in V(G)$ so that $|\langle x|U_T|y\rangle| = 1$, where $|x\rangle, |y\rangle$ are the **initial states** of x, y
- Now we only need the fixed initial states $\{|x\rangle : x \in V(G)\}$ instead of a new time-variant wave function for each particle
- Perfectly unitary evolution (and hence PST) is impossible in practice due to quantum noise, but we *can* achieve **pretty good state transfer (PGST)** (as high as $>97\%$ fidelity!)

Perfect State Transfer

It is easier to search for PST between arbitrary $x, y \in V(G)$ with **unitary operators** rather than set up $|V(G)|$ wave equations.

- **e.g.,** the **transition matrix** $U_t = e^{itA(G)}$ models **unitary evolution** on G in the total absence of noise (where $A(G)$ is the adjacency matrix of G)
- There is PST on G iff $\exists T > 0$ and $x, y \in V(G)$ so that $|\langle x|U_T|y\rangle| = 1$, where $|x\rangle, |y\rangle$ are the **initial states** of x, y
- Now we only need the fixed initial states $\{|x\rangle : x \in V(G)\}$ instead of a new time-variant wave function for each particle
- Perfectly unitary evolution (and hence PST) is impossible in practice due to quantum noise, but we *can* achieve **pretty good state transfer (PGST)** (as high as $>97\%$ fidelity!)

Perfect State Transfer

It is easier to search for PST between arbitrary $x, y \in V(G)$ with **unitary operators** rather than set up $|V(G)|$ wave equations.

- **e.g.,** the **transition matrix** $U_t = e^{itA(G)}$ models **unitary evolution** on G in the total absence of noise (where $A(G)$ is the adjacency matrix of G)
- There is PST on G iff $\exists T > 0$ and $x, y \in V(G)$ so that $|\langle x|U_T|y\rangle| = 1$, where $|x\rangle, |y\rangle$ are the **initial states** of x, y
- Now we only need the fixed initial states $\{|x\rangle : x \in V(G)\}$ instead of a new time-variant wave function for each particle
- Perfectly unitary evolution (and hence PST) is impossible in practice due to quantum noise, but we *can* achieve **pretty good state transfer (PGST)** (as high as $>97\%$ fidelity!)

Perfect State Transfer

The (unweighted) cycle graph on 4 vertices represents a quantum network with PST from node 1 to node 3. (We will see later that it is also something called **Hadamard diagonalizable**.)

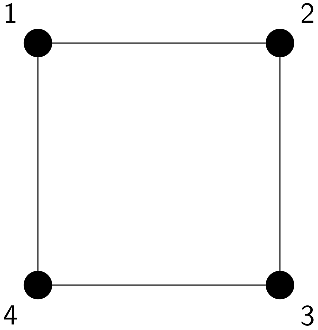

Figure 3: The cycle graph C_4 exhibits perfect state transfer.

$$\begin{pmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{pmatrix}$$

Figure 4: The **Laplacian matrix** $L(C_4) := D(C_4) - A(C_4)$.

Perfect State Transfer

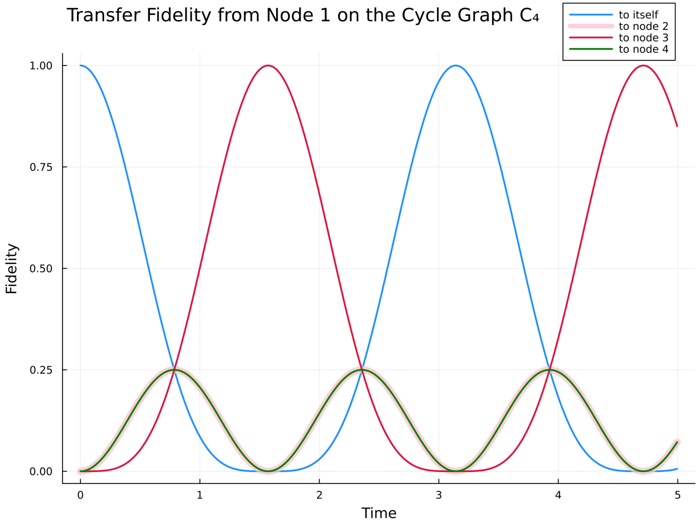

Figure 5: A visualization of PST on C_4 from node 1 to node 3.

Perfect State Transfer

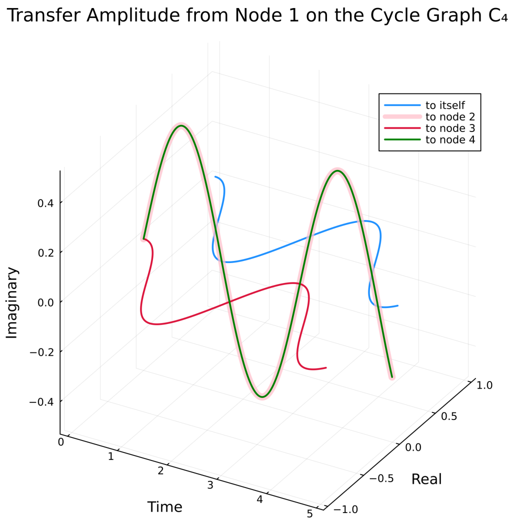

Figure 6: Probability amplitudes of node 1's wave function are **complex**...

S-Diagonalizability/S-Bandwidth

Many of the graphs confirmed to exhibit PST also happen to be $\{-1,1\}$- and $\{-1,0,1\}$-**diagonalizable** (for reasons yet unknown):

Definition (S-diagonalizability)

*A graph G on n vertices is called **S-diagonalizable** if its Laplacian $L(G)$ is diagonalizable by some matrix with entries from $S \subset \mathbb{Z}$—i.e., if $\exists P \in S^{n\times n}$ and diagonal $D \in \mathbb{R}^{n\times n}$ with $L(G) = PDP^{-1}$.*

In particular, PST graphs tend to have low (≤ 2) S-**bandwidths**:

Definition (S-bandwidth)

*Define $\beta : \mathcal{M} \to \mathbb{N}$ by $\beta(X) := \min\{k : |i-j| \geq k \implies x_{ij} = 0\}$ (some texts use $|i-j| > k$ instead). The **S-bandwidth** of a graph G on n vertices, denoted by $\beta_S(G)$, is then the minimum integer k so that $\exists P \in S^{n\times n}$ with $\beta(P^TP) = k$ and $L(G) = PDP^{-1}$.*

S-Diagonalizability/S-Bandwidth

Many of the graphs confirmed to exhibit PST also happen to be $\{-1,1\}$- and $\{-1,0,1\}$**-diagonalizable** (for reasons yet unknown):

Definition (S-diagonalizability)

*A graph G on n vertices is called S-**diagonalizable** if its Laplacian $L(G)$ is diagonalizable by some matrix with entries from $S \subset \mathbb{Z}$—i.e., if $\exists P \in S^{n\times n}$ and diagonal $D \in \mathbb{R}^{n\times n}$ with $L(G) = PDP^{-1}$.*

In particular, PST graphs tend to have low (≤ 2) S-**bandwidths**:

Definition (S-bandwidth)

*Define $\beta : \mathcal{M} \to \mathbb{N}$ by $\beta(X) := \min\{k : |i-j| \geq k \implies x_{ij} = 0\}$ (some texts use $|i-j| > k$ instead). The S-**bandwidth** of a graph G on n vertices, denoted by $\beta_S(G)$, is then the minimum integer k so that $\exists P \in S^{n\times n}$ with $\beta(P^TP) = k$ and $L(G) = PDP^{-1}$.*

S-Diagonalizability/S-Bandwidth

Many of the graphs confirmed to exhibit PST also happen to be $\{-1, 1\}$- and $\{-1, 0, 1\}$-**diagonalizable** (for reasons yet unknown):

Definition (S-diagonalizability)

*A graph G on n vertices is called **S-diagonalizable** if its Laplacian $L(G)$ is diagonalizable by some matrix with entries from $S \subset \mathbb{Z}$—i.e., if $\exists P \in S^{n\times n}$ and diagonal $D \in \mathbb{R}^{n\times n}$ with $L(G) = PDP^{-1}$.*

In particular, PST graphs tend to have low (≤ 2) **S-bandwidths**:

Definition (S-bandwidth)

*Define $\beta : \mathcal{M} \to \mathbb{N}$ by $\beta(X) := \min\{k : |i-j| \geq k \implies x_{ij} = 0\}$ (some texts use $|i-j| > k$ instead). The **S-bandwidth** of a graph G on n vertices, denoted by $\beta_S(G)$, is then the minimum integer k so that $\exists P \in S^{n\times n}$ with $\beta(P^TP) = k$ and $L(G) = PDP^{-1}$.*

S-Diagonalizability/S-Bandwidth

Many of the graphs confirmed to exhibit PST also happen to be $\{-1,1\}$- and $\{-1,0,1\}$-**diagonalizable** (for reasons yet unknown):

Definition (S-diagonalizability)

*A graph G on n vertices is called S-**diagonalizable** if its Laplacian $L(G)$ is diagonalizable by some matrix with entries from $S \subset \mathbb{Z}$—i.e., if $\exists P \in S^{n\times n}$ and diagonal $D \in \mathbb{R}^{n\times n}$ with $L(G) = PDP^{-1}$.*

In particular, PST graphs tend to have low (≤ 2) S-**bandwidths**:

Definition (S-bandwidth)

*Define $\beta : \mathcal{M} \to \mathbb{N}$ by $\beta(X) := \min\{k : |i-j| \geq k \implies x_{ij} = 0\}$ (some texts use $|i-j| > k$ instead). The S-**bandwidth** of a graph G on n vertices, denoted by $\beta_S(G)$, is then the minimum integer k so that $\exists P \in S^{n\times n}$ with $\beta(P^T P) = k$ and $L(G) = PDP^{-1}$.*

S-Diagonalizability/S-Bandwidth

That is, $\beta_S(G) = k$ if and only if k is the smallest integer for which the Laplacian matrix $L(G) := D(G) - A(G)$ has a full collection of eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in S^n$ with $|i - j| \geq k \implies \langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$.

- Of particular interest are **Hadamard** and **weak Hadamard diagonalizability** ($\beta_{\{-1,1\}}(G) = 1$ and $\beta_{\{-1,0,1\}}(G) \leq 2$)
- We are investigating *why* qubit couplings in HD/WHD graphs (such as C_4) are more conducive to high-fidelity transfer
- For now, we treat HD/WHD as a heuristic indicator of PST... motivating our **novel algorithm** to compute S-bandwidth (feasible for any graph on $n \leq 18$ vertices!)

S-Diagonalizability/S-Bandwidth

That is, $\beta_S(G) = k$ if and only if k is the smallest integer for which the Laplacian matrix $L(G) := D(G) - A(G)$ has a full collection of eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in S^n$ with $|i - j| \geq k \implies \langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$.

- Of particular interest are **Hadamard** and **weak Hadamard diagonalizability** ($\beta_{\{-1,1\}}(G) = 1$ and $\beta_{\{-1,0,1\}}(G) \leq 2$)
- We are investigating *why* qubit couplings in HD/WHD graphs (such as C_4) are more conducive to high-fidelity transfer
- For now, we treat HD/WHD as a heuristic indicator of PST... motivating our **novel algorithm** to compute S-bandwidth (feasible for any graph on $n \leq 18$ vertices!)

S-Diagonalizability/S-Bandwidth

That is, $\beta_S(G) = k$ if and only if k is the smallest integer for which the Laplacian matrix $L(G) := D(G) - A(G)$ has a full collection of eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n \in S^n$ with $|i - j| \geq k \implies \langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$.

- Of particular interest are **Hadamard** and **weak Hadamard diagonalizability** ($\beta_{\{-1,1\}}(G) = 1$ and $\beta_{\{-1,0,1\}}(G) \leq 2$)
- We are investigating *why* qubit couplings in HD/WHD graphs (such as C_4) are more conducive to high-fidelity transfer
- For now, we treat HD/WHD as a heuristic indicator of PST... motivating our **novel algorithm** to compute S-bandwidth (feasible for any graph on $n \leq 18$ vertices!)

S-Diagonalizability/S-Bandwidth

That is, $\beta_S(G) = k$ if and only if k is the smallest integer for which the Laplacian matrix $L(G) := D(G) - A(G)$ has a full collection of eigenvectors $\mathbf{v}_1, \dots, \mathbf{v}_n \in S^n$ with $|i - j| \geq k \implies \langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$.

- Of particular interest are **Hadamard** and **weak Hadamard diagonalizability** ($\beta_{\{-1,1\}}(G) = 1$ and $\beta_{\{-1,0,1\}}(G) \leq 2$)
- We are investigating *why* qubit couplings in HD/WHD graphs (such as C_4) are more conducive to high-fidelity transfer
- For now, we treat HD/WHD as a heuristic indicator of PST... motivating our **novel algorithm** to compute S-bandwidth (feasible for any graph on $n \leq 18$ vertices!)

Algorithm for S-Bandwidth

Given some undirected (and possibly weighted) graph G with Laplacian matrix $L \in \mathbb{R}^{n \times n}$, here is an overview of our algorithm:

- **First:** Validate that the unique eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$ of L are all integers. Iterate over all $\{-1, 0, 1\}$-vectors $\mathbf{v} \in \mathbb{R}^n$ (unique up to span) and **test $L\mathbf{v} = \lambda_i \mathbf{v}$** for $i \in \{1, 2, \ldots, k\}$.
- **Next:** With V_i denoting the matrix whose columns are all the $\{-1, 0, 1\}$-eigenvectors for λ_i, **use RREF on each V_i** to see if each eigenspace has a linearly independent $\{-1, 0, 1\}$-basis.
- **Third:** If G *is* diagonalizable, construct a basis with minimum Gramian bandwidth for each eigenspace by **recursively adding/eliminating vectors**. Identify $\beta_{\{-1,0,1\}}(G)$.
- **Last:** By tracking which $\{-1, 0, 1\}$-eigenvectors are also $\{-1, 1\}$-vectors, we can simultaneously determine $\beta_{\{-1,1\}}(G)$.

Algorithm for S-Bandwidth

Given some undirected (and possibly weighted) graph G with Laplacian matrix $L \in \mathbb{R}^{n \times n}$, here is an overview of our algorithm:

- **First:** Validate that the unique eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$ of L are all integers. Iterate over all $\{-1, 0, 1\}$-vectors $\mathbf{v} \in \mathbb{R}^n$ (unique up to span) and **test** $L\mathbf{v} = \lambda_i \mathbf{v}$ for $i \in \{1, 2, \ldots, k\}$.
- **Next:** With V_i denoting the matrix whose columns are all the $\{-1, 0, 1\}$-eigenvectors for λ_i, **use RREF on each** V_i to see if each eigenspace has a linearly independent $\{-1, 0, 1\}$-basis.
- **Third:** If G *is* diagonalizable, construct a basis with minimum Gramian bandwidth for each eigenspace by **recursively adding/eliminating vectors**. Identify $\beta_{\{-1,0,1\}}(G)$.
- **Last:** By tracking which $\{-1, 0, 1\}$-eigenvectors are also $\{-1, 1\}$-vectors, we can simultaneously determine $\beta_{\{-1,1\}}(G)$.

Algorithm for S-Bandwidth

Given some undirected (and possibly weighted) graph G with Laplacian matrix $L \in \mathbb{R}^{n \times n}$, here is an overview of our algorithm:

- **First:** Validate that the unique eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$ of L are all integers. Iterate over all $\{-1, 0, 1\}$-vectors $\mathbf{v} \in \mathbb{R}^n$ (unique up to span) and **test** $L\mathbf{v} = \lambda_i \mathbf{v}$ for $i \in \{1, 2, \ldots, k\}$.
- **Next:** With V_i denoting the matrix whose columns are all the $\{-1, 0, 1\}$-eigenvectors for λ_i, **use RREF on each** V_i to see if each eigenspace has a linearly independent $\{-1, 0, 1\}$-basis.
- **Third:** If G *is* diagonalizable, construct a basis with minimum Gramian bandwidth for each eigenspace by **recursively adding/eliminating vectors**. Identify $\beta_{\{-1,0,1\}}(G)$.
- **Last:** By tracking which $\{-1, 0, 1\}$-eigenvectors are also $\{-1, 1\}$-vectors, we can simultaneously determine $\beta_{\{-1,1\}}(G)$.

Algorithm for S-Bandwidth

Given some undirected (and possibly weighted) graph G with Laplacian matrix $L \in \mathbb{R}^{n \times n}$, here is an overview of our algorithm:

- **First:** Validate that the unique eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_k$ of L are all integers. Iterate over all $\{-1, 0, 1\}$-vectors $\mathbf{v} \in \mathbb{R}^n$ (unique up to span) and **test** $L\mathbf{v} = \lambda_i \mathbf{v}$ for $i \in \{1, 2, \ldots, k\}$.
- **Next:** With V_i denoting the matrix whose columns are all the $\{-1, 0, 1\}$-eigenvectors for λ_i, **use RREF on each** V_i to see if each eigenspace has a linearly independent $\{-1, 0, 1\}$-basis.
- **Third:** If G *is* diagonalizable, construct a basis with minimum Gramian bandwidth for each eigenspace by **recursively adding/eliminating vectors**. Identify $\beta_{\{-1,0,1\}}(G)$.
- **Last:** By tracking which $\{-1, 0, 1\}$-eigenvectors are also $\{-1, 1\}$-vectors, we can simultaneously determine $\beta_{\{-1,1\}}(G)$.

Algorithm for S-Bandwidth

Given some undirected (and possibly weighted) graph G with Laplacian matrix $L \in \mathbb{R}^{n\times n}$, here is an overview of our algorithm:

- **First:** Validate that the unique eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$ of L are all integers. Iterate over all $\{-1,0,1\}$-vectors $\mathbf{v} \in \mathbb{R}^n$ (unique up to span) and **test** $L\mathbf{v} = \lambda_i \mathbf{v}$ for $i \in \{1, 2, \dots, k\}$.
- **Next:** With V_i denoting the matrix whose columns are all the $\{-1,0,1\}$-eigenvectors for λ_i, **use RREF on each** V_i to see if each eigenspace has a linearly independent $\{-1,0,1\}$-basis.
- **Third:** If G *is* diagonalizable, construct a basis with minimum Gramian bandwidth for each eigenspace by **recursively adding/eliminating vectors**. Identify $\beta_{\{-1,0,1\}}(G)$.
- **Last:** By tracking which $\{-1,0,1\}$-eigenvectors are also $\{-1,1\}$-vectors, we can simultaneously determine $\beta_{\{-1,1\}}(G)$.

Available S-Bandwidth Data

We have tabulated the $\{-1, 1\}$- and $\{-1, 0, 1\}$-bandwidths of:

- All unweighted, connected $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 11$ vertices
- All unweighted, connected, regular/bipartite $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 14$ vertices
- All (non-uniformly) positively weighted, connected WHD graphs on $n \leq 5$ vertices – surprisingly, there are none!

With regards to pure graph theory – we are working on several theorems regarding **bipartiteness** and **graph compositions**. With

regards to numerics – we are using linear programming to investigate **WHD-inducing edge weights** for higher-order graphs.

Available S-Bandwidth Data

We have tabulated the $\{-1,1\}$- and $\{-1,0,1\}$-bandwidths of:

- All unweighted, connected $\{-1,0,1\}$-diagonalizable graphs on $n \leq 11$ vertices
- All unweighted, connected, regular/bipartite $\{-1,0,1\}$-diagonalizable graphs on $n \leq 14$ vertices
- All (non-uniformly) positively weighted, connected WHD graphs on $n \leq 5$ vertices – surprisingly, there are none!

With regards to pure graph theory – we are working on several theorems regarding **bipartiteness** and **graph compositions**. With

regards to numerics – we are using linear programming to investigate **WHD-inducing edge weights** for higher-order graphs.

Available S-Bandwidth Data

We have tabulated the $\{-1, 1\}$- and $\{-1, 0, 1\}$-bandwidths of:

- All unweighted, connected $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 11$ vertices
- All unweighted, connected, regular/bipartite $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 14$ vertices
- All (non-uniformly) positively weighted, connected WHD graphs on $n \leq 5$ vertices – surprisingly, there are none!

With regards to pure graph theory – we are working on several theorems regarding **bipartiteness** and **graph compositions**. With

regards to numerics – we are using linear programming to investigate **WHD-inducing edge weights** for higher-order graphs.

Available S-Bandwidth Data

We have tabulated the $\{-1, 1\}$- and $\{-1, 0, 1\}$-bandwidths of:

- All unweighted, connected $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 11$ vertices
- All unweighted, connected, regular/bipartite $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 14$ vertices
- All (non-uniformly) positively weighted, connected WHD graphs on $n \leq 5$ vertices – surprisingly, there are none!

With regards to pure graph theory – we are working on several theorems regarding **bipartiteness** and **graph compositions**. With regards to numerics – we are using linear programming to investigate **WHD-inducing edge weights** for higher-order graphs.

Available S-Bandwidth Data

We have tabulated the $\{-1,1\}$- and $\{-1,0,1\}$-bandwidths of:

- All unweighted, connected $\{-1,0,1\}$-diagonalizable graphs on $n \leq 11$ vertices
- All unweighted, connected, regular/bipartite $\{-1,0,1\}$-diagonalizable graphs on $n \leq 14$ vertices
- All (non-uniformly) positively weighted, connected WHD graphs on $n \leq 5$ vertices – surprisingly, there are none!

With regards to pure graph theory – we are working on several theorems regarding **bipartiteness** and **graph compositions**. With regards to numerics – we are using linear programming to investigate **WHD-inducing edge weights** for higher-order graphs.

Available S-Bandwidth Data

We have tabulated the $\{-1, 1\}$- and $\{-1, 0, 1\}$-bandwidths of:

- All unweighted, connected $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 11$ vertices
- All unweighted, connected, regular/bipartite $\{-1, 0, 1\}$-diagonalizable graphs on $n \leq 14$ vertices
- All (non-uniformly) positively weighted, connected WHD graphs on $n \leq 5$ vertices – surprisingly, there are none!

With regards to pure graph theory – we are working on several theorems regarding **bipartiteness** and **graph compositions**. With

regards to numerics – we are using linear programming to investigate **WHD-inducing edge weights** for higher-order graphs.

Available S-Bandwidth Data


```
SELECT "GraphOrder", "graph6", "Band01Neg", "Band1Neg", "Eigvals", "Eigvecs01Neg", "Eigvecs1Neg", "Regular"
FROM
    (
        SELECT * FROM "Order01" UNION ALL SELECT * FROM "Order02" UNION ALL
        SELECT * FROM "Order03" UNION ALL SELECT * FROM "Order04" UNION ALL
        SELECT * FROM "Order05" UNION ALL SELECT * FROM "Order06" UNION ALL
        SELECT * FROM "Order07" UNION ALL SELECT * FROM "Order08" UNION ALL
        SELECT * FROM "Order09" UNION ALL SELECT * FROM "Order10" UNION ALL
        SELECT * FROM "Order11"
    ) s
WHERE "Bipartite"='True' LIMIT 100
```

GraphOrder	graph6	Band01Neg	Band1Neg	Eigvals	Eigvecs01Neg	Eigvecs1Neg	Regular
1	@	1	1	93NUMPY	93NUMPY	93NUMPY	True
2	A_	1	1	93NUMPY	93NUMPY	93NUMPY	True
4	Cr	1	1	93NUMPY	93NUMPY	93NUMPY	True
6	EoSo	2	Inf	93NUMPY	93NUMPY	93NUMPY	True
6	Es\o	2	Inf	93NUMPY	93NUMPY	93NUMPY	True
8	Gs@ipo	1	1	93NUMPY	93NUMPY	93NUMPY	True
8	Gs`zro	1	1	93NUMPY	93NUMPY	93NUMPY	True
10	Is_BjX[N?	2	Inf	93NUMPY	93NUMPY	93NUMPY	True
10	IsaBzx{^?	2	Inf	93NUMPY	93NUMPY	93NUMPY	True

Figure 7: *Conjecture.* If an (unweighted and connected) **bipartite graph** G is $\{-1,0,1\}$-diagonalizable, then G **is regular** and $|G|$ **is even or** 1.

Thank you!

Figure 8: Pusheen the Cat <3