TWINS-GCN: SPECTRAL CONJUGATE FOR SPECTRAL GRAPH CONVOLUTIONAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Graph convolutional networks propagate information by repeated local aggregation through a graph shift operator; i.e., a K-layer network reaches K hops neighborhood. On the one hand, such spreading can lead to oversmoothing. On the other hand, long-range dependencies demand the depth. Transporting information on long distances and without attenuation requires the shift to distinguish a direction of flow, which a symmetric operator cannot perform but a directed one can fulfill. A natural way to extract pure-directionality is to take the skew-symmetric part of the shift operator through the Cartesian split, which, however, generally does not commute with the shift itself, meaning that the filters built on it are not shift-invariant. We instead use the spectral conjugate; i.e., the image of the operator under $\tau : z \mapsto \bar{z}$, which commutes with the shift and splits it into a dissipative and a non-dissipative part. Two filter families follow: a sum filter, whose non-dissipative component transports signal without energy loss, and a ratio filter, ratio in the pair of components rather than polynomial in the shift. Both arise from non-holomorphic kernels, placing them outside the holomorphic class underlying classical spectral convolution. On the directed cycle, the ratio filter becomes an IIR filter with global impulse response, for which we prove a long-range reach gap against every degree-K polynomial filter. Chebyshev reparameterization gives stable vertex-domain layers with real coefficients, yielding TWINS-GCN, which solves graph transfer tasks at reduced depth and is competitive with state-of-the-art graph convolutional networks on node classification benchmarks.

1 INTRODUCTION

Graph convolutional networks are among the most widely used architectures for learning on relational data, owing to their efficiency, being light weight and to their increased interpretability given the spectral theory that underpins them (Bruna et al., 2014; Defferrard et al., 2016; Kipf & Welling, 2017). However, two limitations are well documented: node representations tending to converge toward a constant as depth increases, known as oversmoothing (Li et al., 2018; Oono & Suzuki, 2020; Cai & Wang, 2020; Rusch et al., 2023) and a K-layer network aggregating over at most K hops neighborhood, which requires sufficient depth to capture long-range dependencies (Topping et al., 2022; Di Giovanni et al., 2023). Residual connections, normalization, and rewiring (Li et al., 2019; Alon & Yahav, 2021) serve as effective solutions for both obstacles, yet, because our goal is to address their common cause, we do not compare against this line of work.

The reasons for these limitations are largely related to the nature of the operator used in the convolutional networks. For a symmetric graph operator, high frequencies tend to be attenuated due to its real spectrum, leading to progressively less distinctive features since spectral convolution strengthens this attenuation when the number of convolution layers (depth) increases (Oono & Suzuki, 2020; Cai & Wang, 2020). Escaping this requires eigenvalues on the imaginary axis, an established criterion for stable non-dissipative dynamics (Haber & Ruthotto, 2018; Chang et al., 2019), which A-DGN and its successors obtain by imposing skew-symmetry on the learnable weights (Gravina et al., 2023; Heilig et al., 2025; Hariri et al., 2026). For directed graphs, the graph shift already possesses non-zero imaginary part as it is non-symmetric (Singh et al., 2016; Marques et al., 2020a).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

A natural way to extract skew-symmetry and thus imaginary spectrum is the Cartesian split, which separates the shift operator into its symmetric and skew-symmetric parts. We show, however, that neither part commutes with the shift unless the graph is normal—which directed graphs generally are not (Trefethen, 2005; Asllani et al., 2018)—so the induced filters are not shift-invariant. To overcome this limitation while still obtaining imaginary spectrum, we instead use the spectral conjugate (Nevanlinna, 2018), which commutes with the shift on any graph, coincides with the transpose when the shift is normal, and remains defined for non-diagonalizable shifts. On this decomposition, we build learnable graph convolutional layers from filters in (Chan et al., 2026a). Composing the resulting layers gives the TWINS-GCN framework. Importantly, since the underlying scalar functions (kernels) are not complex-differentiable, the construction falls outside the holomorphic functional calculus classical spectral filters rely on (Higham, 2008), introducing a new class of spectral graph convolutions.

Contributions. (1) We propose the spectral conjugate, rather than the transpose, to split the directed Laplacian Singh et al. (2016) into dissipative and non-dissipative components while preserving shift invariance. (2) We construct the sum and ratio filters from the resulting components, prove they are both non-holomorphic, and derive their vertex-domain forms via the Cauchy-Pompeiu formula. (3) We prove that on the directed cycle the ratio filter is an IIR filter with global impulse response, and establish a reach gap against every degree-K polynomial filter in the shift. (4) We reparameterize both filters in Chebyshev bases, yielding stable vertex-domain layers with real coefficients. (5) TWINS-GCN solves graph transfer tasks at a fraction of the depth conventional polynomial filters require, preserves feature distinctiveness with depth, and is competitive with state-of-the-art GCNs on general benchmarks for node classification.

2 DISSIPATIVE, ASYMMETRIC, AND SPECTRAL CONJUGATE FILTERS

2.1 WHY POLYNOMIAL SPECTRAL FILTERS ATTENUATE

Let $G = (\mathcal{V}, \mathcal{E}, \mathbf{W})$[1] be a weighted graph with $N = |\mathcal{V}|$ nodes and weight matrix $\mathbf{W} \in \mathbb{R}_{\geq 0}^{N \times N}$, and let $\mathbf{L} \in \mathbb{R}^{N \times N}$ be the corresponding graph shift operator (Ortega, 2022) with spectrum $\sigma(\mathbf{L}) = \{\lambda_n\} \subset \mathbb{C}$. Denote its Jordan decomposition as $\mathbf{L} = \mathbf{P}(\mathbf{\Lambda} + \mathbf{N})\mathbf{P}^{-1}$, where $\mathrm{diag}(\mathbf{\Lambda}) = \sigma(\mathbf{L})$ and $\mathbf{N}$ is the nilpotent part. The standard spectral filter is the matrix $h(\mathbf{L})$ induced by a scalar kernel h that is analytic on a neighborhood $\mathcal{D} \subset \mathbb{C}$ of $\sigma(\mathbf{L})$, on the Jordan block $\mathbf{J}_n$ of size m attached to λ_n,

$$h(\mathbf{J}_n) = \sum_{k=0}^{m-1} \frac{h^{(k)}(\lambda_n)}{k!} \mathbf{N}_n^k, \tag{1}$$

placing $h(\lambda_n)$ on the diagonal and its complex derivatives $h^{(k)}(\lambda_n)$ on the successive superdiagonals (Sandryhaila & Moura, 2014). Filtering a signal reads as $\mathbf{y} = h(\mathbf{L})\mathbf{x}$, and l layers apply $h(\mathbf{L})^l$. Since $\mathbf{N}_n$ is nilpotent, the sum terminates, so the gain of each spectral component is $|h(\lambda_n)|$ up to a factor polynomial in m. The standard GCN layer can be read as the degree one polynomial $h(\lambda) = 1 - t\lambda$ (Chamberlain et al., 2021), the first-order truncation of the heat kernel $h(\lambda) = e^{-t\lambda}$ whose gain is upper bounded by $e^{-t\Re(\lambda_n)}$ (Appendix C.5). Components with $\Re(\lambda_n) > 0$ thus decay geometrically with depth and only those with $\Re(\lambda_n) = 0$ survive: as $l \to \infty$ the representation collapses onto the $\Re(\lambda) = 0$ invariant subspace, which is the spectral form of oversmoothing. The following observation shows that under a symmetric shift this subspace is uninformative.

Symmetric shifts and non-dissipative propagation. A real symmetric $\mathbf{S}$ has $\sigma(\mathbf{S}) \subset \mathbb{R}$, so $\Re(\lambda_n) = \lambda_n$ for every n; and for a Laplacian on a connected graph the eigenvalue 0 is simple with eigenvector $\mathbf{1}$ (Fiedler, 1973; Chung, 1997). Under a symmetric shift, the constant signal is therefore the only mode with $\Re(\lambda) = 0$, and the only one preserved under arbitrary depth. Real symmetric operators therefore do not allow for non-dissipative propagation. For directed graphs, the operator is asymmetric, enables purely imaginary eigenvalues, and consequently allows non-dissipative propagation.

[1] Notations are provided in Appendix A

2.2 WHAT ABOUT THE CARTESIAN SPLIT

Specify $\mathcal{G}$ to be a directed graph, and consider the directed Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$ with real weight matrix $\mathbf{W}$ and $\mathbf{D} = \mathrm{diag}(\mathbf{W1})$ the in-degree matrix (Singh et al., 2016). $\mathbf{L}$ is non-symmetric, and its spectrum is complex with $\Im(\lambda) \neq 0$ in general. To isolate the asymmetric structure, a natural approach is the Cartesian split $\mathbf{L} = \frac{1}{2}(\mathbf{L} + \mathbf{L}^T) + \frac{1}{2}(\mathbf{L} - \mathbf{L}^T)$, whose second term is skew-symmetric and thus has purely imaginary spectrum. Besides pure imaginary spectrum, it is crucial for both terms to be linear shift-invariant (LSI) — to commute with the shift operator (Sandryhaila & Moura, 2014) — to enable the analysis of graph signals in the same frequency domain, and thereby the interpretation of the features. Additionally filters based on either components are LSI w.r.t. to $\mathbf{L}$ only when they the components themselves are.

Proposition 1 (Cartesian split breaks shift invariance)**.** The Cartesian components $\frac{1}{2}(\mathbf{L} \pm \mathbf{L}^T)$ commute with $\mathbf{L}$ if and only if $\mathbf{L}$ is normal.

Even when diagonalizable, directed Laplacians are usually non-normal (Trefethen, 2005; Asllani et al., 2018). Hence the Cartesian split yields components that are not LSI. Thereby, a different split satisfying commutation with $\mathbf{L}$ with summands respectively carrying spectrum on real $\sigma_R(\mathbf{L}) = \{\Re(\lambda_n)\}$ and imaginary axis $\sigma_I(\mathbf{L}) = \{j\Im(\lambda_n)\}$, is desirable. This leads us to naturally consider the spectral conjugate (Nevanlinna, 2018).

2.3 NON-HOLOMORPHIC FUNCTIONAL AND THE SPECTRAL CONJUGATE

The spectral conjugate of $\mathbf{L}$ is the matrix induced by the non-analytic kernel $\tau : z \mapsto \bar{z}$ (Nevanlinna, 2018), non-analytic since its Wirtinger derivative $\bar{\partial}\tau := \partial\tau/\partial\bar{z} = 1$ (Appendix C.4). Yet in equation 1, h is required to be analytic, equivalently holomorphic. Therefore to evaluate τ, we leverage non-holomorphic functional calculus. First recall that holomorphic functional calculus (Auscher et al., 1997) presents the evaluation of an analytic function h at $\lambda \in \Omega$ via the Cauchy integral

$$h(\lambda) = \frac{1}{2\pi i} \oint_\Gamma \frac{h(z)}{z - \lambda}\, dz, \tag{2}$$

where Γ is any contour enclosing λ in its interior. Lifting this to the matrix setting results in the standard graph filters (Appendix C.3) and classical spectral graph filters relying on canonical, Chebyshev, or Cayley polynomials falls within this framework (Appendix C.4). Evaluating τ instead follows from the Cauchy-Pompeiu formula (Bell, 2015), which extends the Cauchy integral to smooth but non-holomorphic functions:

$$h(\lambda) = \frac{1}{2\pi i} \oint_\Gamma \frac{h(z)}{z - \lambda}\, dz + \frac{1}{2\pi i} \iint_\Omega \frac{\bar{\partial} h(z)}{z - \lambda}\, dz \wedge d\bar{z}, \tag{3}$$

where Ω is the region enclosed by Γ. Evaluating τ at $\mathbf{L}$ through equation 3, as in Nevanlinna (2018), yields the subsequent definition of spectral conjugate.

Definition 2.1. (Nevanlinna, 2018, Def. 2.5 – Spectral conjugate) The spectral conjugate of $\mathbf{L}$ under the non-holomorphic function $\tau : z \mapsto \bar{z}$ is given by

$$\mathbf{L}^\sharp := \tau(\mathbf{L}) = \mathbf{P}\tau(\mathbf{J})\mathbf{P}^{-1} = \mathbf{P}\overline{\mathbf{\Lambda}}\mathbf{P}^{-1}, \tag{4}$$

where $\tau(\mathbf{J})$ is composed of the Jordan blocks $\tau(\mathbf{J}_n) = \overline{\lambda}_n \mathbf{I}$.

Theorem 1 (Conjugate properties)**.** $\mathbf{L}^\sharp$ *commutes with* $\mathbf{L}$*, is real, and coincides with* $\mathbf{L}^T$ *if and only if* $\mathbf{L}$ *is normal.*

Corollary 1 (Shift-invariant spectral split)**.** The decomposition

$$\mathbf{L} = \underbrace{\tfrac{1}{2}\big(\mathbf{L} + \mathbf{L}^\sharp\big)}_{\mathbf{L}^\circ} + \underbrace{\tfrac{1}{2}\big(\mathbf{L} - \mathbf{L}^\sharp\big)}_{\mathbf{L}^\uparrow} \tag{5}$$

is exact, both components commute with $\mathbf{L}$ and with each other, are real, and $\sigma(\mathbf{L}^\circ) = \sigma_R(\mathbf{L})$, $\sigma(\mathbf{L}^\uparrow) = \sigma_I(\mathbf{L})$. For normal $\mathbf{L}$, it reduces to the Cartesian split. Thereby we term $\mathbf{L}^\circ$ and $\mathbf{L}^\uparrow$ dissipative and asymmetric operators, respectively.

In practice, $\mathbf{L}^\sharp$ is computed with the Schur–Parlett algorithm (Parlett, 1976; Davies & Higham, 2003), which evaluates a matrix function on the Schur form instead of the (numerically unstable) Jordan form. Its cost is dominated by the Schur decomposition which scales as $O(N^3)$.

2.4 SPECTRAL FILTERS: THE SUM AND RATIO KERNELS

Recalling equation 5, one can design kernels that operate separately on $\sigma_R(\mathbf{L})$ and $\sigma_I(\mathbf{L})$. We consider the two kernels of (Chan et al., 2026a),

Sum kernel. Separate polynomial expansions in the real and imaginary parts,

$$h_{\text{sum}}(z) = \sum_{k=0}^{K_1} \mathbf{a}[k] \left(\frac{z+\bar{z}}{2}\right)^k + \sum_{k=0}^{K_2} \mathbf{b}[k] \left(\frac{z-\bar{z}}{2}\right)^k, \qquad \mathbf{a}[k], \mathbf{b}[k] \in \mathbb{C}. \tag{6}$$

Ratio kernel. Consider the polynomial in the ratio $(z-\bar{z})/(z+\bar{z})$,

$$h_{\text{rat}}(z) = \mathbf{c}[0] + \begin{cases} \displaystyle\sum_{k=1}^{K} \mathbf{c}[k] \left(\frac{z-\bar{z}}{z+\bar{z}}\right)^k, & \Re(z) \neq 0, \\ 0, & \Re(z) = 0, \end{cases} \qquad \mathbf{c}[k] \in \mathbb{C}, \tag{7}$$

The case $\Re(z) = 0$ arises only at the direct current (DC) eigenvalue $\lambda = 0$. Setting $h_{\text{rat}}(0) = \mathbf{c}[0]$ removes the division-by-zero instability, and allows filtering of the DC component. We next show that the sum and ratio (equation 6, 7) kernels span a new function class, being non-holomorphic.

Theorem 2 (Vertex-domain sum and ratio filters)**.** *The kernels h_{sum} and h_{rat} are non-holomorphic. Their corresponding graph filters admit the vertex-domain representations*

$$h_{sum}(\mathbf{L}) = \sum_{k=0}^{K_1} \mathbf{a}[k] \left(\mathbf{L}^{\circ}\right)^k + \sum_{k=0}^{K_2} \mathbf{b}[k] \left(\mathbf{L}^{\uparrow}\right)^k, \quad h_{rat}(\mathbf{L}) = \sum_{k=0}^{K} \mathbf{c}[k] \left(\mathbf{L}^{\hat{\Phi}}\right)^k, \tag{8}$$

where $\mathbf{L}^{\hat{\Phi}} = \mathbf{L}^{\uparrow}(\mathbf{L}^{\circ})^D$ and $(\mathbf{L}^{\circ})^D$ the Drazin inverse (Appendix C.2) of $\mathbf{L}^{\circ}$.

Theorem 2 has two consequences. First, it grounds the vertex-domain forms equation 8 for any $\mathbf{L}$, diagonalizable or not, whereas Chan et al. (2026a) define them only in the diagonalizable case. Second, no polynomial or convergent power series in z represents either kernel. On a fixed graph, $h(\mathbf{L})$ still equals a polynomial in $\mathbf{L}$ interpolating h on $\sigma(\mathbf{L})$, but one that is graph-specific, of degree up to N, and ill-conditioned (Appendix C.10). The new class is thus distinguished at the kernel level, and at the matrix level by degree and conditioning.

3 SPECTRAL GRAPH CONVOLUTION NETWORKS

3.1 GRAPH FILTER DESIGN

Canonical polynomial implementations of spectral filters are ill-conditioned in their coefficients, and Chebyshev bases are the typical choice to tackle this limitation (Boyd, 2001); however, they apply to real spectra. We show that equation 8 admits a Chebyshev reparameterization nonetheless. Throughout, T_k denotes the degree k Chebyshev polynomial, and is recursively defined as

$$T_k(x) = 2x\,T_{k-1}(x) - T_{k-2}(x), \quad k \geq 2, \qquad T_0(x) = 1,\ T_1(x) = x. \tag{9}$$

Sum filter. The operator $\mathbf{L}^{\circ}$ has real eigenvalues $\{\lambda_{\text{R}}[n]\}$, which we rescale to $[-1, 1]$:

$$\widetilde{\mathbf{L}}^{\circ} = \frac{\mathbf{L}^{\circ} - c_R\mathbf{I}}{R_R}, \qquad c_R = \tfrac{1}{2}\big(\max_n \lambda_{\text{R}}[n] + \min_n \lambda_{\text{R}}[n]\big), \qquad R_R = \max_n |\lambda_{\text{R}}[n] - c_R| + \varepsilon. \tag{10}$$

The operator $\mathbf{L}^{\uparrow}$ has purely imaginary eigenvalues $\{j\lambda_{\text{I}}[n]\}$, so $-j\mathbf{L}^{\uparrow}$ has real spectrum $\{\lambda_{\text{I}}[n]\}$. Because $\mathbf{L}^{\uparrow}$ is a real-valued matrix, its eigenvalues occur in conjugate pairs, giving $c_I = 0$; no recentering is needed and

$$\widetilde{\mathbf{L}}^{\uparrow} = -j\,\mathbf{L}^{\uparrow}/R_I, \qquad R_I = \max_n |\lambda_{\text{I}}[n]| + \varepsilon. \tag{11}$$

Combining degree-K_1 and degree-K_2 expansions in the two rescaled operators yields

$$h_{\text{sum}}(\mathbf{L}) = \sum_{k=0}^{K_1} \boldsymbol{\alpha}[k]\, T_k(\widetilde{\mathbf{L}}^{\circ}) + \sum_{k=0}^{K_2} \boldsymbol{\beta}[k]\, T_k(\widetilde{\mathbf{L}}^{\uparrow}), \tag{12}$$

with learnable $\boldsymbol{\alpha} \in \mathbb{C}^{K_1+1}$, $\boldsymbol{\beta} \in \mathbb{C}^{K_2+1}$.

Ratio filter. The quotient operator $\mathbf{L}^{\hat{\Phi}} = \mathbf{L}^{\uparrow}(\mathbf{L}^{\circ})^{D}$ likewise has purely imaginary eigenvalues $j\lambda_{\mathrm{I}}[n]/\lambda_{\mathrm{R}}[n]$ for $\lambda_{\mathrm{R}}[n] \neq 0$, so the same substitution applies. Setting $\mathbf{Z} = -j\mathbf{L}^{\hat{\Phi}}$ gives the real spectrum $\boldsymbol{z}[n] = \lambda_{\mathrm{I}}[n]/\lambda_{\mathrm{R}}[n]$, occurring in conjugate pairs, hence $c_Q = \frac{1}{2}\big(\max_n \boldsymbol{z}[n] + \min_n \boldsymbol{z}[n]\big) = 0$ and

$$\widetilde{\mathbf{Z}} = \mathbf{Z}/R_Q, \qquad R_Q = \max_n |\boldsymbol{z}[n]| + \varepsilon, \tag{13}$$

so that $h_{\text{rat}}(\mathbf{L}) = \sum_{k=0}^{K} \boldsymbol{\eta}[k]\, T_k(\widetilde{\mathbf{Z}})$ with learnable $\boldsymbol{\eta} \in \mathbb{C}^{K+1}$. To observe the effect of reparameterization, we provide the frequency response for each filter $h_{\text{sum}}(\mathbf{L}), h_{\text{rat}}(\mathbf{L})$ in Appendix B.4.

Theorem 3 (Real-valued twins filters)**.** *Real-valued filters expressible by $h_{sum}(\mathbf{L})$ are exactly those obtained with $\boldsymbol{\alpha}[k] \in \mathbb{R}$, $\boldsymbol{\beta}[2k] \in \mathbb{R}$ and $\boldsymbol{\beta}[2k+1] \in j\mathbb{R}$ for all k. Likewise, real-valued filters expressible by $h_{rat}(\mathbf{L})$ are exactly those obtained with $\boldsymbol{\eta}[2k] \in \mathbb{R}$ and $\boldsymbol{\eta}[2k+1] \in j\mathbb{R}$ for all k.*

Despite the odd-order coefficients being purely imaginary, factoring j out for these terms brings back the problem to a real-valued optimization. This effectively handles concerns of memory and compute overhead when optimizing over complex-valued parameter space (Krämer, 2024).

3.2 The TwinS-GCN architecture

In what follows, we describe the layer design for both the sum and ratio filters.

Sum filter based layer design. We propose a simple layer design that leverages the sum graph filter proposed in equation 8. Denote $\mathbf{X}^{(l)}$ the node features of dimension $N \times C$ (nodes times channels) at the l-th layer. The $(l+1)$ layer update reads as:

$$\mathbf{X}^{(l+1)} = \psi\left(\nu \sum_{k=0}^{K_1} T_k(\widetilde{\mathbf{L}}^{\circ})\mathbf{X}^{(l)}\boldsymbol{\Theta}_k^{(l,\circ)} + (1-\nu)\sum_{k=0}^{K_2} j^k\, T_k(\widetilde{\mathbf{L}}^{\uparrow})\mathbf{X}^{(l)}\boldsymbol{\Theta}_k^{(l,\uparrow)} + \mathbf{B}^{(l)}\right), \tag{14}$$

where ψ is a pointwise nonlinear function, $\boldsymbol{\Theta}_k^{(l,\circ)}$ and $\boldsymbol{\Theta}_k^{(l,\uparrow)}$ are learnable real weight matrices, $\mathbf{B}^{(l)}$ is a real learnable bias term and $\nu \in [0,1]$ is real learnable parameter as introduced in Rossi et al. (2024), controlling importance of dissipative and non-dissipative parts. Composing sum filter based layers, we build the TwinS (S) model.

Ratio filter based layer design. Similarly, the update rule for the $(l+1)$-th layer, based on the ratio graph filter proposed in equation 8 is given by:

$$\mathbf{X}^{(l+1)} = \psi\left(\sum_{k=0}^{K} j^k\, T_k(\widetilde{\mathbf{Z}})\mathbf{X}^{(l)}\boldsymbol{\Theta}_k^{(l)} + \mathbf{B}^{(l)}\right), \tag{15}$$

where $\boldsymbol{\Theta}_k^{(l)}$ is a learnable real weight matrix. Consecutive ratio filters constitute the TwinS (R) model. We also propose to combine both models leading to TwinS (C) (Appendix C.8).

3.3 Theoretical analysis

3.3.1 Mitigating oversmoothing

Oversmoothing arises when repeated graph convolution drives node features toward a constant vector, and is diagnosed by the decay across layers of the Dirichlet energy (Rusch et al., 2023)

$$\mathcal{S}(\mathbf{X}^{(l)}) = \tfrac{1}{2}\sum_{(n,m)\in\bar{\mathcal{E}}} \big\|\mathbf{X}_n^{(l)} - \mathbf{X}_m^{(l)}\big\|^2, \qquad \bar{\mathcal{E}} = \mathcal{E} \cup \{(m,n) : (n,m) \in \mathcal{E}\}. \tag{16}$$

As shown by the sum in equation 16, the Dirichlet energy sees only the symmetrized edge set $\bar{\mathcal{E}}$, or in another words the symmetric part of the shift, making it a measure of feature distinctiveness rather than of directed smoothness (Marques et al., 2020b). We retain it in that sense, following Choi et al. (2023), and verify the mitigation empirically in Section 5.1 by tracking $\mathcal{S}(\mathbf{X}^{(l)})$ across depth.

We analyze this in particular on the directed cycle $\mathcal{C}_N$, where $\mathbf{L}$ is normal and the graph Fourier basis is unitary, so for a scalar signal $\mathbf{x}^{(l)}$ Parseval identity gives

$$\mathcal{S}(\mathbf{x}^{(l)}) = \big(\mathbf{x}^{(l)}\big)^T \mathbf{L}^{\circ}\, \mathbf{x}^{(l)} = \sum_{n=0}^{N-1} \lambda_{\mathrm{R}}[n]\, \big|\hat{\mathbf{x}}^{(l)}[n]\big|^2. \tag{17}$$

Since $\mathbf{L}^{\uparrow}$ has purely imaginary spectrum $\sigma_I(\mathbf{L})$, any spectral function of unit modulus applied to it leaves every $|\hat{\mathbf{x}}[n]|$ unchanged and hence preserves equation 17 exactly. The filter $e^{\beta \mathbf{L}^{\uparrow}}$ relies on such function, which it acts as $\hat{\mathbf{x}}[n] \mapsto e^{j\beta\lambda_{\mathrm{I}}[n]}\hat{\mathbf{x}}[n]$, a pure phase rotation with $\mathcal{S}(e^{\beta \mathbf{L}^{\uparrow}}\mathbf{x}) = \mathcal{S}(\mathbf{x})$. A degree one sum filter with $\mathbf{a} = (0, 0)$ and $\mathbf{b} = (1, \beta)$ realizes its first-order truncation,

$$\mathbf{x}_{\text{sum}}^{(l+1)} = \big(\mathbf{I} + \beta \mathbf{L}^{\uparrow}\big)\mathbf{x}^{(l)} \implies \mathcal{S}\big(\mathbf{x}_{\text{sum}}^{(l+1)}\big) = \sum_{n} \lambda_{\mathrm{R}}[n]\big(1 + \beta^2 \lambda_{\mathrm{I}}[n]^2\big)\big|\hat{\mathbf{x}}^{(l)}[n]\big|^2, \tag{18}$$

so the truncation never contracts energy.

The mechanism is the complementary spectral action of the two components: $\mathbf{L}^{\circ}$ has real non-negative spectrum and attenuates high-frequency modes, whereas $\mathbf{L}^{\uparrow}$ transports features by phase alone and does not attenuate. Therefore increasing the depth does not necessarily imply smoothing. The argument extends beyond $\mathcal{C}_N$. On any directed graph, the eigenvalues of $e^{\beta \mathbf{L}^{\uparrow}}$ are $e^{j\beta\lambda_{\mathrm{I}}[n]}$, all of unit modulus, so no mode is attenuated exponentially: for diagonalizable $\mathbf{L}$, the filter is a pure phase rotation of each Fourier coefficient (no attenuation), and for defective $\mathbf{L}$, the nilpotent part adds at most a factor polynomial in β (Appendix C.5).

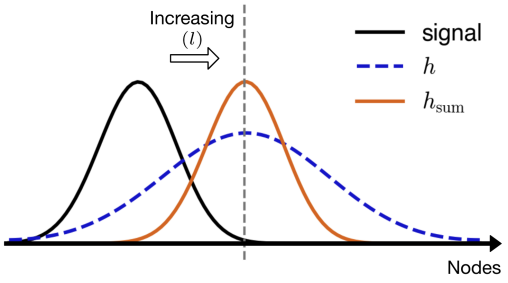


Figure 1: Isolating $\mathbf{L}^{\uparrow}$ allows h_{sum} to translate the signal while the polynomial kernel h inseparably moves and flattens it.

3.3.2 Expanding receptive field reach

Message-passing networks built on a local shift operator share a structural ceiling: a K-layer network moves information at most K hops (Theorem 5 (i)), so long-range aggregation often demands large depth. We show that the quotient operator $\mathbf{L}^{\hat{\Phi}}$ alleviates this ceiling, taking $\mathcal{C}_N$ with a pulse-translation task as a setting that isolates receptive field reach.

Graph ratio and IIR filters. The operator $\mathbf{L}^{\hat{\Phi}}$ shares the eigenbasis of the shift but has eigenvalues $\boldsymbol{\mu}[n] = j\lambda_{\mathrm{I}}[n]/\lambda_{\mathrm{R}}[n]$. On $\mathcal{C}_N$ these are a Möbius transform of the shift eigenvalues $\boldsymbol{z}[n] = e^{j\boldsymbol{\theta}[n]}$, which turns the graph ratio filter into a ratio function of the shift.

Theorem 4 (IIR correspondence). *On $\mathcal{C}_N$ with shift eigenvalues $\boldsymbol{z}[n] = e^{j\boldsymbol{\theta}[n]}$, the eigenvalues $\boldsymbol{\mu}[n]$ satisfy the relation $\boldsymbol{\mu}[n] = (\boldsymbol{z}[n] + 1)/(\boldsymbol{z}[n] - 1)$. Consequently, a degree-K polynomial filter in $\mathbf{L}^{\hat{\Phi}}$ has spectral response*

$$\sum_{k=0}^{K} \mathbf{c}[k]\, \boldsymbol{\mu}[n]^k \;=\; \frac{\sum_{k=0}^{K} \mathbf{c}[k]\, (\boldsymbol{z}[n] + 1)^k (\boldsymbol{z}[n] - 1)^{K-k}}{(\boldsymbol{z}[n] - 1)^K} \;=\; \frac{Q_K(\boldsymbol{z}[n])}{(\boldsymbol{z}[n] - 1)^K}, \tag{19}$$

i.e. a ratio or infinite impulse response (IIR) filter of type $[K/K]$ in the shift, with all K poles located at the DC mode $z = 1$. Its vertex-domain impulse response therefore has global support.

Although IIR for $\mathcal{C}_N$, on general graphs the ratio filter is no longer a ratio function of $\mathbf{L}$. Different from graph IIR filters (Isufi et al., 2016; Liu et al., 2018), it needs no iterative solver to perform filtering and inserts directly into any convolutional layer, retaining finite impulse response (FIR) efficiency while achieving IIR reach.

The reach of the ratio filter. We measure how far a filter propagates information. For $\mathbf{y} = h(\mathbf{L})\mathbf{x}$ with $\mathbf{x}$ an impulse at node 0, the influence of node 0 on node d (at distance d) is $h[d,0] := h(\mathbf{L})[d,0] = \partial \mathbf{y}[d]/\partial \mathbf{x}[0]$ (the sensitivity of Di Giovanni et al. (2023)), and we define the reach as the relative sensitivity

$$S_h(d) = \frac{|h[d,0]|}{\max_{v\in\mathcal{V}} |h[v,0]|} \in [0,1]. \tag{20}$$

Theorem 5 (Reach of shift, heat, and ratio filters). *On $\mathcal{C}_N$ with nodes $\mathcal{V} = \{0,\dots,N-1\}$, let $d \in \mathcal{V}\setminus\{0\}$, and assume $N \geq 3K+2$. Then:*

(i) (Shift filter) *If $h = \sum_{m=0}^{K} \gamma_m \mathbf{W}^m$, then $h[d,0] = \gamma_d$ for $d \leq K$ and*

$$S_h(d) = 0, \quad \textit{for every } d > K. \tag{21}$$

(ii) (Heat filter) *If $h = e^{-t\mathbf{L}^\circ}$ with $t > 0$, then*

$$\mathbf{S}_h(d) \leq \frac{(t/2)^d}{d!}. \tag{22}$$

(iii) (Ratio filter) *If $h = h_{rat} := \sum_{k=0}^{K} \mathbf{c}[k](\mathbf{L}^{\hat{\Phi}})^k$ then for $K < d < N-K$*

$$S_h(d) \approx \frac{|p(d/N)|}{\max_{v\in\mathcal{V}} |p(v/N)|}, \quad p(s) = \sum_{k=1}^{K} \mathbf{c}[k]\, \frac{(-1)^{k-1}2^k}{k!}\, N^{k-1} B_k(s), \tag{23}$$

where $B_k(s)$ is the Bernoulli polynomial of degree k.

The three regimes are qualitatively distinct. Reach of the shift polynomial (FIR) is identically zero beyond K hops (what we refer to as the FIR wall shown in Figure 2), the heat kernel (IIR) decays super-exponentially, and the ratio filter only polynomially. At $K = 1$, equation 23 gives $S_h(d) = |1 - 2d/N|$, nonzero at every distance except for $d = N/2$. Considering $\mathcal{G}(K) = \max_{h_{\text{rat}}} \min_{d\in\mathcal{V}} S_h(d)$, i.e., the worst-case reach at order K, we obtain $\mathcal{G}(K) > 0.95$ for $K \geq 4$ (Figure 4). While the $[K/K]$ characterization and bound on sensitivity are specific to $\mathcal{C}_N$, the escape from the K-hop class remains valid for general directed graph as the filters are not FIR. We empirically show this on general graphs through the graph transfer task (Bodnar et al., 2021; Di Giovanni et al., 2023) in Section 5.1, and further experiments reported in Appendix F.

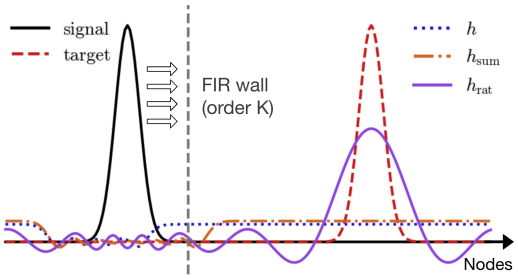


Figure 2: FIR models are obstructed by FIR wall whereas graph ratio filter overcomes it on $\mathcal{C}_N$.

4 Related Works

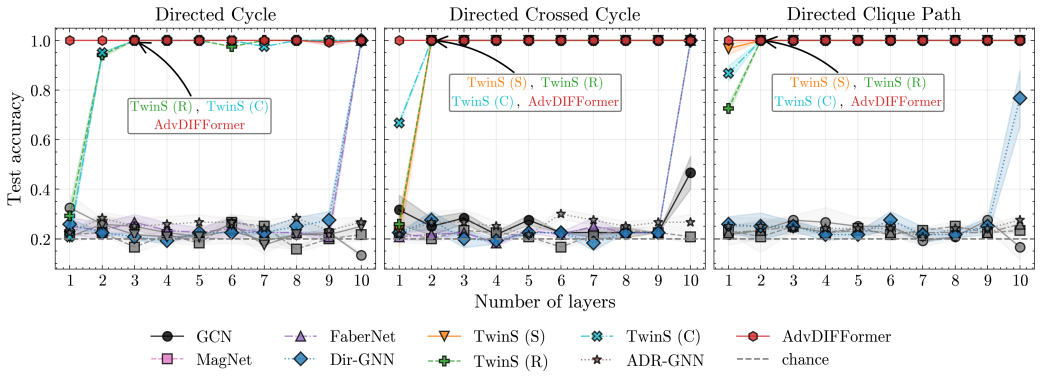
Spectral graph convolutional networks. FaberNet (HoloNet) (Koke & Cremers, 2024) performs spectral filtering on directed graphs by approximating holomorphic functions through Faber polynomials (Coleman & Smith, 1987; Ellacott, 1983), outperforming MagNet (Zhang et al., 2021), a directed GCN built on the Magnetic Laplacian (Furutani et al., 2020). Both constructs filter directly from a graph operator without separating the mechanisms governing propagation, and in particular without the dissipative and asymmetric split of equation 5. This is reflected in the function class they span: being holomorphic, neither can represent the kernels of Theorem 2. Dir-GNN (Rossi et al., 2024) aggregates separately over in- and out-neighborhoods with distinct weight matrices. Since $(\mathbf{L}, \mathbf{L}^T) \mapsto \left(\frac{1}{2}(\mathbf{L}+\mathbf{L}^T), \frac{1}{2}(\mathbf{L}-\mathbf{L}^T)\right)$ is bijective, this realises a Cartesian split, and Proposition 1 applies: the components do not commute with the shift, so the filters are not LSI.

Advection-diffusion graph neural networks. For diagonalizable shifts, equation 5 coincides with the graph diffusion and advection operators of Chan et al. (2026a) who study them as signal-processing tools, without learning or propagation analysis. This places our construction alongside architectures of similar concept (Eliasof et al., 2024; Wu et al., 2023), which we include as baselines. Further conceptual comparisons with these networks are provided in Appendix C.9.

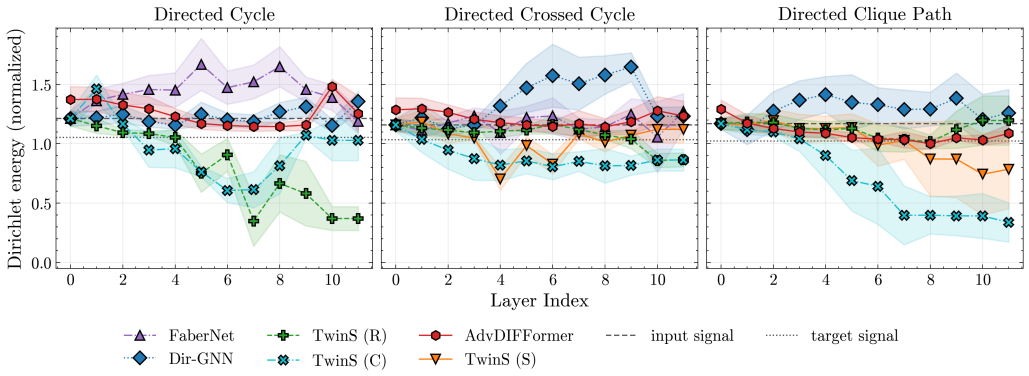
5 Experiments

Baselines include methods discussed in Section 4: MagNet, FaberNet, Dir-GNN, ADR-GNN and AdvDIFFormer. Methods that instead constrain the weight parametrization (Gravina et al., 2023; Heilig et al., 2025) (discussed in Section 1) act on a complementary axis and can be combined with ours (Appendix C.11), so we treat them as orthogonal rather than competing.

5.1 Receptive field reach and Oversmoothing



(a) Accuracy as a function of the number of layers



(b) Feature energy at each layer

Figure 3: a) Accuracy of different models on the graph transfer task. In the box are models with overlapping accuracy of 1. Models with performance lower than chance level (20%) at all depths are displayed in gray. b) Dirichlet energy computed after each convolution layer before activation, averaged within batch. Only models (10 layers) with accuracy beyond chance level are shown. Input and target signal's energy are also shown for each graph. In both case, error bars are over trials.

Settings. We evaluate reach and oversmoothing jointly on the graph transfer task (Bodnar et al., 2021; Di Giovanni et al., 2023), adapted to directed graphs (Appendix D). The task involves the propagation of a feature vector from a source node to a target node, which are separated by a certain number of hops in the graph. Following Choi et al. (2023), we track the Dirichlet energy (equation 16) of the node features across layers as a measure of feature distinctiveness.

Results. From Figure 3a, TWINS (R) and TWINS (C) transfer the feature accurately with only a few layers on all three graphs, and TWINS (S) does so on the crossed cycle and clique path, demonstrating TWINS-GCN's ability to capture long-range dependencies at minimal depth. AdvDIFFormer also solves the task at low depth, but at a higher cost in parameters and runtime (Table 7). The remaining polynomial methods (MagNet, FaberNet, Dir-GNN, GCN) and ADR-GNN instead require a depth comparable to the distance to target, or does not solve the task. Figure 3b shows that on each graph at least one TWINS-GCN variant maintains its Dirichlet energy across layers, preserving feature distinctness while solving the task. These results are consistent with reach and non-attenuation properties of Section 3.3 beyond directed cycle.

5.2 BENCHMARK DATASETS

Settings. We evaluate performances on directed node-classification benchmarks: Chameleon, Squirrel (filtered version as in Platonov et al. (2023)) representing links between Wikipedia pages related to chameleons and squirrels (Rozemberczki et al., 2021), Texas, Wisconsin, and Cornell which are datasets showing links between websites of different universities (Pei et al., 2020). We also compare our model against MLP (Rumelhart et al., 1986) and undirected graphs baselines — GCN (Kipf & Welling, 2017) and GAT (Brody et al., 2021).

Results. The results of the benchmark experiments are summarized in Table 1. Our proposed TWINS-GCN model matches the baseline methods across the datasets with best overall rank despite lower accuracy on Squirrel-F, demonstrating its effectiveness in capturing the underlying directed structures and improving node classification accuracy. Reasons for the increased performance could be attributed to the properties of long-range reach and mitigation of oversmoothing, as discussed in Section 5.1, along with the class of function our model leverages.

Table 1: Node classification accuracy (in percentage) across benchmark datasets. Best result per dataset in **bold**, second best underlined. Additional information in Appendix D.2.

	Model	Chameleon-F	Squirrel-F	Texas	Wisconsin	Cornell	Mean Rank
	MLP	36.73 ± 4.71	36.55 ± 1.82	82.16 ± 4.34	85.16 ± 4.95	$\underline{75.86} \pm \underline{3.94}$	6.70
Undir.	GCN	40.89 ± 4.12	39.47 ± 1.47	52.16 ± 5.16	48.92 ± 3.06	52.70 ± 5.30	10.20
Undir.	GAT	39.21 ± 3.08	35.62 ± 2.06	58.38 ± 6.63	49.41 ± 4.09	54.32 ± 5.05	10.20
Dir.	MagNet	42.36 ± 3.53	44.19 ± 1.39	$\underline{85.31} \pm \underline{4.56}$	83.40 ± 3.98	73.33 ± 4.17	$\underline{4.20}$
Dir.	Dir-GNN	44.13 ± 2.92	$\underline{44.28} \pm \underline{2.17}$	70.54 ± 6.52	67.39 ± 6.00	59.82 ± 6.90	6.40
Dir.	FaberNet	42.99 ± 3.33	$\mathbf{45.13 \pm 2.06}$	83.15 ± 5.30	79.54 ± 5.18	72.61 ± 4.72	4.60
Dir.	AdvDIFFormer	41.51 ± 3.63	41.41 ± 1.59	81.80 ± 6.36	76.54 ± 4.84	72.16 ± 5.50	7.20
Dir.	ADR-GNN	$\mathbf{46.07 \pm 4.07}$	44.09 ± 1.97	82.16 ± 5.16	84.51 ± 4.54	71.35 ± 4.39	4.90
Ours	TWINS (S)	42.33 ± 2.80	39.61 ± 1.55	84.59 ± 3.43	$\underline{87.12} \pm \underline{4.13}$	75.14 ± 3.15	4.60
Ours	TWINS (R)	$\underline{44.24} \pm \underline{3.23}$	40.15 ± 1.83	84.05 ± 3.91	$\mathbf{87.39 \pm 3.98}$	$\mathbf{76.76 \pm 3.01}$	**2.80**
Ours	TWINS (C)	41.64 ± 3.59	40.10 ± 1.58	$\mathbf{85.41 \pm 4.71}$	86.80 ± 3.79	75.68 ± 4.01	$\underline{4.20}$

6 CONCLUSION

We introduced TWINS-GCN, a model built upon the spectral conjugate split of the shift. We show that the resulting components and induced filters span a novel class of filters with kernels outside ones reachable by the holomorphic class underlying classical spectral convolutions. We illustrate TWINS-GCN's capability to preserve feature distinctiveness with far-reaching receptive fields through examples and the graph transfer task. TWINS-GCN also shows on real benchmarks competitive accuracy for node classification.

Outlooks and limitations. Although the transpose is not suitable for a shift-invariant split, is inexpensive, well conditioned, and its Cartesian components have spectra distinct from $\mathbf{L}$, which may add expressivity. Performing our splits on both $\mathbf{L}$ and $\mathbf{L}^T$ could thus improve accuracy at the cost of interpretability. Likewise, the antisymmetric weight parameterizations act on the channel-mixing matrices and are tangent to our operator-level construction. They can nevertheless be combined with our approach. Computationally, the spectral conjugate requires a Schur decomposition, a one-time cost per graph that nonetheless restricts the present implementation to moderate-size graphs. Krylov-Schur (Stewart, 2002) approximation of the leading modes is a route to scale.

AI USE STATEMENT

In this work, we used generative AI tools to assist with the writing of proofs, in phrasing and reorganization of text, tables and equations, and to improve experiment code, which we then reviewed, tested, and corrected. We have not used generative AI tools for research ideas, for the development of the proposed methods, nor for any conceptualization. We have reviewed all AI-assisted work by proofreading and verifying correctness of the code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

No conflict of interest to report nor ethical issues are foreseen in this work.

REPRODUCIBILITY STATEMENT

Here are listed the information to ensure reproducibility of our work:

- Mathematical concepts are either defined or referenced. Proofs of all Theorems are provided in Appendix B.
- Accompanying information for statements and concepts in the main text are provided in Appendix C.
- We exactly detail our newly introduced (TWINS-GCN) framework in the main body of our paper Section 3.2. Variants of the framework are discussed in Section 3.2
- The experimental setups for our experiments are described in Appendix D.
- The datasets used in our experiments are described in Appendix D.2.
- Hyperparameter settings of the methods used in the experiments are detailed in the Appendix D.1, D.2.
- Our code will be made publicly available conditionally on acceptance of the paper.

ACKNOWLEDGMENTS

Currently hidden for double-blind review.

REFERENCES

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021.

Malbor Asllani, Renaud Lambiotte, and Timoteo Carletti. Structure and dynamical behavior of non-normal networks. *Science advances*, 4(12):eaau9403, 2018.

Pascal Auscher, Alan McIntosh, and Andrea Nahmod. Holomorphic functional calculi of operators, quadratic estimates and interpolation. *Indiana University Mathematics Journal*, pp. 375–403, 1997.

Steven R Bell. *The Cauchy transform, potential theory and conformal mapping*. CRC press, 2015.

Pierre-Gabriel Berlureau, Ali Hariri, Victor Kawasaki-Borruat, Mia Zosso, and Pierre Vandergheynst. Advection-diffusion on graphs: A bakry-émery laplacian for spectral graph neural networks. *arXiv preprint arXiv:2602.18141*, 2026.

Cristian Bodnar, Fabrizio Frasca, Nina Otter, Yuguang Wang, Pietro Lio, Guido F Montufar, and Michael Bronstein. Weisfeiler and lehman go cellular: Cw networks. *Advances in neural information processing systems*, 34:2625–2640, 2021.

John P Boyd. *Chebyshev and Fourier spectral methods*. Courier Corporation, 2001.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. In *International Conference on Learning Representations*, 2014.

C. Cai and Y. Wang. A note on over-smoothing for graph neural networks. In *International Conference on Machine Learning*, 2020.

Stephen L. Campbell and Carl D. Meyer. *Generalized Inverses of Linear Transformations*. Classics in Applied Mathematics. SIAM, 2009. doi: 10.1137/1.9780898719048.

Ben Chamberlain, James Rowbottom, Maria I Gorinova, Michael Bronstein, Stefan Webb, and Emanuele Rossi. Grand: Graph neural diffusion. In *International conference on machine learning*, pp. 1407–1418. PMLR, 2021.

CHM Chan, A Cionca, V Škultéty, and D Van De Ville. Graph diffusion-advection operator for directed graph signal processing. *arXiv preprint arXiv:2606.16306*, 2026a.

Chun Hei Michael Chan, Alexandre Cionca, and Dimitri Van De Ville. Hilbert transform on graphs: Let there be phase. *IEEE Signal Processing Letters*, 32:1725–1729, 2025.

Chun Hei Michael Chan, Alexandre Cionca, and Dimitri Van De Ville. Statistical testing on directed graphs by surrogate data generation. *IEEE Transactions on Signal and Information Processing over Networks*, pp. 1–12, 2026b. doi: 10.1109/TSIPN.2026.3717613.

B. Chang, M. Chen, E. Haber, and E. H. Chi. AntisymmetricRNN: A dynamical system view on recurrent neural networks. In *International Conference on Learning Representations*, 2019.

Airlie Chapman. Advection on graphs. In *Semi-autonomous networks: Effective control of networked systems through protocols, design, and modeling*, pp. 3–16. Springer, 2015.

Jeongwhan Choi, Seoyoung Hong, Noseong Park, and Sung-Bae Cho. Gread: Graph neural reaction-diffusion networks. In *International conference on machine learning*, pp. 5722–5747. PMLR, 2023.

Fan RK Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

John P Coleman and Russell A Smith. The faber polynomials for circular sectors. *mathematics of computation*, 49(179):231–241, 1987.

Philip I. Davies and Nicholas J. Higham. A Schur–Parlett algorithm for computing matrix functions. *SIAM J. Matrix Anal. Appl.*, 25(2):464–485, 2003.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

Francesco Di Giovanni, Lorenzo Giusti, Federico Barbero, Giulia Luise, Pietro Lio, and Michael M Bronstein. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *International conference on machine learning*, pp. 7865–7885. PMLR, 2023.

Moshe Eliasof, Eldad Haber, and Eran Treister. Feature transportation improves graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 11874–11882, 2024.

SW Ellacott. Computation of faber series with application to numerical polynomial approximation in the complex plane. *Mathematics of Computation*, 40(162):575–587, 1983.

Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2): 298–305, 1973.

Satoshi Furutani, Toshiki Shibahara, Mitsuaki Akiyama, Kunio Hato, and Masaki Aida. Graph signal processing for directed graphs based on the hermitian laplacian. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pp. 447–463. Springer, 2020.

A. Gravina, D. Bacciu, and C. Gallicchio. Anti-symmetric DGN: A stable architecture for deep graph networks. In *International Conference on Learning Representations*, 2023.

Alessio Gravina, Moshe Eliasof, Claudio Gallicchio, Davide Bacciu, and Carola-Bibiane Schönlieb. On oversquashing in graph neural networks through the lens of dynamical systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 16906–16914, 2025.

Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse problems*, 34(1):014004, 2018.

Ali Hariri, Alvaro Arroyo, Alessio Gravina, Moshe Eliasof, Carola-Bibiane Schönlieb, Davide Bacciu, Xiaowen Dong, Kamyar Azizzadenesheli, and Pierre Vandergheynst. Return of chebnet: Understanding and improving an overlooked gnn on long range tasks. *Advances in Neural Information Processing Systems*, 38:136166–136196, 2026.

Simon Heilig, Alessio Gravina, Alessandro Trenta, Claudio Gallicchio, and Davide Bacciu. Port-hamiltonian architectural bias for long-range propagation in deep graph networks. In *International Conference on Learning Representations*, 2025.

Nicholas J. Higham. *Functions of Matrices: Theory and Computation*. SIAM, 2008.

Elvin Isufi, Andreas Loukas, Andrea Simonetto, and Geert Leus. Autoregressive moving average graph filtering. *IEEE Transactions on Signal Processing*, 65(2):274–288, 2016.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=SJU4ayYgl`.

Christian Koke and Daniel Cremers. Holonets: Spectral convolutions do extend to directed graphs. In *International Conference on Learning Representations*, volume 2024, pp. 49121–49154, 2024.

Nicholas Krämer. A tutorial on automatic differentiation with complex numbers. *arXiv preprint arXiv:2409.06752*, 2024.

Ron Levie, Federico Monti, Xavier Bresson, and Michael M Bronstein. Cayleynets: Graph convolutional neural networks with complex rational spectral filters. *IEEE Transactions on Signal Processing*, 67(1):97–109, 2018.

Guohao Li, Matthias Muller, Ali Thabet, and Bernard Ghanem. Deepgcns: Can gcns go as deep as cnns? In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9267–9276, 2019.

Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Jiani Liu, Elvin Isufi, and Geert Leus. Filter design for autoregressive moving average graph filters. *IEEE Transactions on Signal and Information Processing over Networks*, 5(1):47–60, 2018.

Antonio G Marques, Santiago Segarra, and Gonzalo Mateos. Signal processing on directed graphs: The role of edge directionality when processing and learning from network data. *IEEE Signal Processing Magazine*, 37(6):99–116, 2020a.

Antonio G. Marques, Santiago Segarra, and Gonzalo Mateos. Signal Processing on Directed Graphs: The Role of Edge Directionality When Processing and Learning From Network Data. *IEEE Signal Processing Magazine*, 37(6):99–116, November 2020b. ISSN 1053-5888, 1558-0792. doi: 10.1109/MSP.2020.3014597.

Olavi Nevanlinna. A non-holomorphic functional calculus and the complex conjugate of a matrix. *Linear Algebra and its Applications*, 537:191–220, 2018.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations*, 2020.

Antonio Ortega. *Introduction to Graph Signal Processing*. Cambridge University Press, 1 edition, June 2022. ISBN 978-1-108-55234-9 978-1-108-42813-2. doi: 10.1017/9781108552349.

Beresford N. Parlett. A recurrence among the elements of functions of triangular matrices. *Linear Algebra and its Applications*, 14(2):117–121, 1976.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-GCN: Geometric graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2020.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. geom-gcn: data release. `https://github.com/bingzhewei/geom-gcn`, 2022. Cornell node and edge files revised in commit `3cc5710` (17 July 2022); accessed 19 September 2026.

Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of gnns under heterophily: Are we really making progress? *arXiv preprint arXiv:2302.11640*, 2023.

Emanuele Rossi, Bertrand Charpentier, Francesco Di Giovanni, Fabrizio Frasca, Stephan Günnemann, and Michael M Bronstein. Edge directionality improves learning on heterophilic graphs. In *Learning on graphs conference*, pp. 25–1. PMLR, 2024.

Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

T Konstantin Rusch, Michael M Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

Aliaksei Sandryhaila and Jose M. F. Moura. Discrete Signal Processing on Graphs: Frequency Analysis. *IEEE Transactions on Signal Processing*, 62(12):3042–3054, June 2014. ISSN 1053-587X, 1941-0476. doi: 10.1109/TSP.2014.2321121.

Rahul Singh, Abhishek Chakraborty, and BS Manoj. Graph fourier transform based on directed laplacian. In *2016 International Conference on Signal Processing and Communications (SPCOM)*, pp. 1–5. IEEE, 2016.

Predrag S. Stanimirović, Dimitrios Pappas, and Vasilios N. Katsikis. Computing the drazin inverse using the schur decomposition. *Applied Mathematics and Computation*, 250:366–381, 2015.

Gilbert W Stewart. A krylov–schur algorithm for large eigenproblems. *SIAM Journal on Matrix Analysis and Applications*, 23(3):601–614, 2002.

Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M. Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. In *International Conference on Learning Representations*, 2022.

Lloyd N Trefethen. Spectra and pseudospectra: The behaviour of non-normal matrices and operators. In *The graduate student's guide to numerical analysis' 98: Lecture notes from the VIII EPSRC Summer School in Numerical Analysis*, pp. 217–250. Springer, 2005.

G. N. Watson. *A Treatise on the Theory of Bessel Functions*. Cambridge University Press, 2nd edition, 1944.

Qitian Wu, Chenxiao Yang, Kaipeng Zeng, and Michael Bronstein. Supercharging graph transformers with advective diffusion. *arXiv preprint arXiv:2310.06417*, 2023.

Xitong Zhang, Yixuan He, Nathan Brugnone, Michael Perlmutter, and Matthew Hirn. Magnet: A neural network for directed graphs. *Advances in neural information processing systems*, 34: 27003–27015, 2021.

A NOTATIONS

Table 2: Summary of notations used in the main text.

Notation	Description
Graph and signals	
$G = (\mathcal{V}, \mathcal{E}, \mathbf{W})$	Directed weighted graph with nodes $\mathcal{V}$, edges $\mathcal{E}$, weights $\mathbf{W}$
$N = \lvert\mathcal{V}\rvert$	Number of nodes
C	Number of feature channels
$\bar{\mathcal{E}}$	Symmetrized edge set, $\mathcal{E} \cup \{(m,n) : (n,m) \in \mathcal{E}\}$
$\mathbf{W} \in \mathbb{R}_{\geq 0}^{N \times N}$	Non-negative weight (adjacency) matrix
$\mathbf{D} = \mathrm{diag}(\mathbf{W}\mathbf{1})$	Diagonal in-degree matrix
$\mathbf{x} \in \mathbb{R}^N$	Graph signal (single channel)
$\mathbf{X}^{(l)} \in \mathbb{R}^{N \times C}$	Node feature matrix at layer l
$\hat{\mathbf{x}}[k]$	k-th graph Fourier coefficient of $\mathbf{x}$
$\mathbf{1}$	All-ones vector (constant signal)
Operators and spectral quantities	
$\mathbf{L} = \mathbf{D} - \mathbf{W}$	Directed graph Laplacian
$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}, \mathbf{L} = \mathbf{P}\mathbf{J}\mathbf{P}^{-1}$	Eigendecomposition and Jordan decomposition of $\mathbf{L}$
$(\cdot)^D$	Drazin inverse
$\mathbf{P},\ \mathbf{N}$	Generalized eigenvectors and nilpotent part (Jordan form)
$\mathbf{U}$	Eigenvector matrix (non-unitary for directed $\mathbf{L}$)
$\mathbf{\Lambda}$	Diagonal matrix of complex eigenvalues λ_k
$\lambda_{\mathrm{R}}[n],\ \lambda_{\mathrm{I}}[n]$	Real and imaginary parts of the n-th eigenvalue
$\mathbf{L}^{\circ}$	Dissipative operator, $\frac{1}{2}(\mathbf{L} + \mathbf{L}^{\sharp})$
$\mathbf{L}^{\uparrow}$	Asymmetric operator, $\frac{1}{2}(\mathbf{L} - \mathbf{L}^{\sharp})$
$\mathbf{L}^{\hat{\Phi}}$	Quotient operator, $\mathbf{L}^{\uparrow}(\mathbf{L}^{\circ})^D$
$\mathbf{L}^{\sharp}$	Spectral conjugate, $\mathbf{P}\overline{\mathbf{\Lambda}}\mathbf{P}^{-1}$
$\sigma(\mathbf{L})$	Spectrum of $\mathbf{L}$
$\sigma_R(\mathbf{L})$	Real axis spectrum of $\mathbf{L}$
$\sigma_I(\mathbf{L})$	Imaginary axis spectrum of $\mathbf{L}$
$\mathcal{S}(\cdot)$	Smoothness measure
Kernels, filters and reparameterization	
$\tau : \mathbb{C} \to \mathbb{C}$	Conjugate kernel
$h : \mathcal{D} \subset \mathbb{C} \to \mathbb{C}$	Spectral kernel
$h_{\mathrm{sum}},\ h_{\mathrm{rat}}$	Sum and ratio kernels
$K,\ K_1,\ K_2$	Polynomial degrees
$(\mathbf{a}[k],\ \mathbf{b}[k])$, $(\boldsymbol{\alpha}[k],\ \boldsymbol{\beta}[k])$	Sum filter coefficients (canonical and reparameterized)
$\mathbf{c}[k], \boldsymbol{\eta}[k]$	Ratio filter coefficients (canonical and reparameterized)
$\boldsymbol{\theta}[k]$	Standard polynomial filter coefficients
$T_k(\cdot)$	Chebyshev polynomial of the first kind, degree k
$\widetilde{\mathbf{L}}^{\circ},\ \widetilde{\mathbf{L}}^{\uparrow}$	Spectrally rescaled $\mathbf{L}^{\circ}$ and $\mathbf{L}^{\uparrow}$
$\mathbf{Z} = -j\,\mathbf{L}^{\hat{\Phi}},\ \widetilde{\mathbf{Z}}$	Real-spectrum quotient operator and its rescaling
$c_R,\ c_Q$	Spectral centering constants
$R_R,\ R_I,\ R_Q$	Spectral scaling radii
Architecture and analysis	
$\mathbf{\Theta}_k^{(l,\circ)},\ \mathbf{\Theta}_k^{(l,\uparrow)}$	Learnable weight matrices at layer l
$\mathbf{B}^{(l)}$	Learnable bias at layer l
ν, ρ	Learnable parameter shared across layers
$\psi(\cdot)$	Pointwise non-linear activation
$\mu_k,\ z_k$	Argument-ordering and shift eigenvalues on $\mathcal{C}_N$
$\mathcal{C}_N$	Directed cycle graph on N nodes
Complex analysis	
$\Gamma,\ \Omega$	Integration contour and enclosed region
$\bar{\partial} = \partial/\partial\bar{z}$	Wirtinger derivative (antiholomorphic)
$\partial = \partial/\partial z$	Wirtinger derivative (holomorphic)
$\overline{\mathcal{D}}$	Closure of domain of definition $\mathcal{D}$
j	Imaginary unit, $j^2 = -1$

B Proofs of Theorems and Propositions

B.1 Proof of Proposition 1 (Cartesian split breaks shift invariance)

Lemma 6. *Denoting the commutator* $\big[\mathbf{A}, \mathbf{B}\big] = \mathbf{AB} - \mathbf{BA}$*, we have that for any* $\mathbf{L} \in \mathbb{R}^{N\times N}$*,*

$$\big[\mathbf{L},\ \tfrac{1}{2}(\mathbf{L} \pm \mathbf{L}^T)\big] \ = \ \pm\tfrac{1}{2}\big[\mathbf{L},\ \mathbf{L}^T\big]. \tag{24}$$

Proof. For any $\mathbf{L} \in \mathbb{R}^{N\times N}$,

$$\big[\mathbf{L},\ \tfrac{1}{2}(\mathbf{L} \pm \mathbf{L}^T)\big] = \tfrac{1}{2}\left(\mathbf{L}(\mathbf{L} \pm \mathbf{L}^T) - (\mathbf{L} \pm \mathbf{L}^T)\mathbf{L}\right) = \pm\tfrac{1}{2}(\mathbf{L}\mathbf{L}^T - \mathbf{L}^T\mathbf{L}) = \pm\tfrac{1}{2}\big[\mathbf{L},\ \mathbf{L}^T\big]. \tag{25}$$

□

If $\mathbf{L}$ is normal, then $\mathbf{L}\mathbf{L}^T = \mathbf{L}^T\mathbf{L}$ and hence $\big[\mathbf{L},\ \tfrac{1}{2}(\mathbf{L} \pm \mathbf{L}^T)\big] = 0$, so the Cartesian components commute with $\mathbf{L}$. Conversely, if the Cartesian components commute with $\mathbf{L}$, then $[\mathbf{L},\ \mathbf{L}^T] = 0$ and $\mathbf{L}$ is normal. Hence the Cartesian components commute with $\mathbf{L}$ if and only if $\mathbf{L}$ is normal. □

B.2 Proof of Theorem 1 (Conjugate properties)

Proof of main theorem. We refer to Nevanlinna (2018)[Proposition 5.1] for the commutativity of $\mathbf{L}$ and $\mathbf{L}^\sharp$. If $\mathbf{L}$ is normal, then $\mathbf{L}^\sharp = \mathbf{U}\overline{\mathbf{\Lambda}}\mathbf{U}^H = \mathbf{L}^T$. Conversely, if $\mathbf{L}^\sharp = \mathbf{L}^T$, then as $\mathbf{L}^\sharp$ commutes with $\mathbf{L}$, $\mathbf{L}$ is normal.

What remains is the proof of the realness of $\mathbf{L}^\sharp$ when $\mathbf{L}$ is real.

Lemma 7. *(Nevanlinna, 2018, Eq. 1.9 – Block action) Let* $h \in \mathcal{C}^{N-1}(\overline{\mathcal{D}})$ *and let* $\mathbf{J}_n = \lambda_n\mathbf{I} + \mathbf{N}_n$ *be a Jordan block of size* m*. Then*

$$h(\mathbf{J}_n) = \sum_{k=0}^{m-1} \frac{(\partial^k h)(\lambda_n)}{k!}\,\mathbf{N}_n^k, \qquad \partial = \frac{\partial}{\partial z}, \tag{26}$$

where ∂ *denotes the Wirtinger derivative instead of the complex derivative. This is a generalization of equation 1 to non-holomorphic functions.*

Only ∂ appears in equation 26 and given that the conjugate kernel $\tau(z) = \bar{z}$ satisfies $\partial\tau \equiv 0$, so every term with $k \geq 1$ vanishes and

$$\tau(\mathbf{J}_n) = \overline{\lambda_n}\,\mathbf{I}, \tag{27}$$

as defined in Definition 2.1.

Lemma 8 (Conjugation)**.** *Let* $h \in \mathcal{C}^{N-1}(\overline{\mathcal{D}})$*. Setting* $h^*(z) := \overline{h(\bar{z})}$*, we have* $h^* \in \mathcal{C}^{N-1}(\overline{\mathcal{D}})$ *and*

$$\overline{h(\mathbf{A})} = h^*(\overline{\mathbf{A}}) \qquad \textit{for every } \mathbf{A} \in \mathbb{C}^{N\times N}. \tag{28}$$

Proof. Conjugation exchanges the Wirtinger derivatives, $\partial\,\overline{F} = \overline{\bar{\partial}F}$, so $(\partial h^*)(z) = \overline{(\bar{\partial}h)(\bar{z})}$ and, inductively, $(\partial^k h^*)(\bar{\lambda}) = \overline{(\bar{\partial}^k h)(\lambda)}$. If $\mathbf{A} = \mathbf{P}\mathbf{J}\mathbf{P}^{-1}$ then $\overline{\mathbf{A}} = \overline{\mathbf{P}}\,\overline{\mathbf{J}}\,\overline{\mathbf{P}}^{-1}$, and $\overline{\mathbf{J}}$ is a Jordan form whose blocks are $\overline{\lambda}_n\mathbf{I} + \mathbf{N}_n$, the nilpotent parts being real. Applying equation 26 blockwise,

$$\overline{h(\mathbf{J}_n)} = \sum_{k=0}^{m-1} \frac{\overline{(\partial^k h)(\lambda_n)}}{k!}\,\mathbf{N}_n^k = \sum_{k=0}^{m-1} \frac{(\partial^k h^*)(\overline{\lambda_n})}{k!}\,\mathbf{N}_n^k = h^*(\overline{\mathbf{J}_n}), \tag{29}$$

and conjugating $\mathbf{P}\,h(\mathbf{J})\,\mathbf{P}^{-1}$ gives equation 28. □

Applying Lemma 8 to the conjugate kernel: $\tau^*(z) = \overline{\tau(\bar{z})} = \overline{\bar{z}} = z = \tau(z)$. Since $\mathbf{L}$ is real, $\overline{\mathbf{L}} = \mathbf{L}$, and equation 28 gives

$$\overline{\mathbf{L}^\sharp} = \overline{\tau(\mathbf{L})} = \tau^*(\overline{\mathbf{L}}) = \tau(\mathbf{L}) = \mathbf{L}^\sharp. \tag{30}$$

A matrix equal to its own conjugate is real. □

Proof of corollary. The decomposition $\mathbf{L} = \frac{1}{2}\big(\mathbf{L} + \mathbf{L}^\sharp\big) + \frac{1}{2}\big(\mathbf{L} - \mathbf{L}^\sharp\big)$ is exact. Both components commute with $\mathbf{L}$ and with each other since $\mathbf{L}^\sharp$ commutes with $\mathbf{L}$. As $\mathbf{L}^\sharp$ is real, both components of the decomposition are also real. Also by definition, $\sigma(\mathbf{L}^\circ) = \sigma_R(\mathbf{L})$, $\sigma(\mathbf{L}^\uparrow) = \sigma_I(\mathbf{L})$. Following, when $\mathbf{L}$ is normal, this decomposition reduces to the Cartesian split. □

B.3 Proof of Theorem 2 (Vertex-domain sum and ratio filters)

Non-holomorphicity. Decomposing $\Re(z) = (z + \bar{z})/2$ and $\Im(z) = (z - \bar{z})/(2j)$, the Wirtinger derivative $\bar{\partial} h_{\text{sum}}, \bar{\partial} h_{\text{rat}}$ simplifies to:

$$\bar{\partial} h_{\text{sum}}(z) = \sum_{k=1}^{K_1} \frac{k\,\mathbf{a}[k]}{2^k}(z + \bar{z})^{k-1} + \sum_{k=1}^{K_2} \frac{-k\,\mathbf{b}[k]}{(2j)^k}(z - \bar{z})^{k-1} \neq 0, \tag{31}$$

$$\bar{\partial} h_{\text{rat}}(z) = \sum_{k=1}^{K} \mathbf{c}[k] \frac{-2kz(z - \bar{z})^{k-1}}{j^k (z + \bar{z})^{k+1}} \neq 0, \tag{32}$$

for non-zero $\mathbf{a}, \mathbf{b}, \mathbf{c}$. This shows that both kernels are not holomorphic, as their Wirtinger derivative is non-zero.

Vertex-domain expressions. Both filters follow from the functional calculus of Nevanlinna (2018) applied to kernels built from the identity and the conjugation $\tau(z) = \bar{z}$.

Lemma 9 ((Nevanlinna, 2018, Cor. 3.5)). *Let $\mathbf{L} \in \mathbb{C}^{N \times N}$ and $\mathcal{D} \subset \mathbb{C}$ be open with $\mathcal{D} \supset \sigma(\mathbf{L})$. The map $h \mapsto h(\mathbf{L})$ from $\mathcal{C}^{N-1}(\overline{\mathcal{D}})$ to $\mathbb{C}^{N \times N}$ is a homomorphism.*

Write $\kappa^\circ = \frac{1}{2}(\mathrm{id} + \tau)$ and $\kappa^\uparrow = \frac{1}{2}(\mathrm{id} - \tau)$, so that linearity of Lemma 9 together with $\tau(\mathbf{L}) = \mathbf{L}^\sharp$ gives $\kappa^\circ(\mathbf{L}) = \mathbf{L}^\circ$ and $\kappa^\uparrow(\mathbf{L}) = \mathbf{L}^\uparrow$, and multiplicativity gives $(\kappa^\circ)^k(\mathbf{L}) = (\mathbf{L}^\circ)^k$ and $(\kappa^\uparrow)^k(\mathbf{L}) = (\mathbf{L}^\uparrow)^k$ for every $k \in \mathbb{N}$. Since $h_{\text{sum}} = \sum_{k=0}^{K_1} \mathbf{a}[k](\kappa^\circ)^k + \sum_{k=0}^{K_2} \mathbf{b}[k](\kappa^\uparrow)^k$, linearity yields equation 8 directly:

$$h_{\text{sum}}(\mathbf{L}) = \sum_{k=0}^{K_1} \mathbf{a}[k]\,(\mathbf{L}^\circ)^k + \sum_{k=0}^{K_2} \mathbf{b}[k]\,(\mathbf{L}^\uparrow)^k. \tag{33}$$

The ratio kernel requires more care, since it involves a quotient and is discontinuous where $\Re(z) = 0$. We assume throughout that $\lambda = 0$ is the only eigenvalue of $\mathbf{L}$ on the imaginary axis, which holds for the Laplacian of any connected graph. As $\sigma(\mathbf{L})$ is finite, fix $\varepsilon > 0$ small enough that the discs $B(\lambda_n, \varepsilon)$ are pairwise disjoint and κ° does not vanish on any $B(\lambda_n, \varepsilon)$ with $\lambda_n \neq 0$, and set $\mathcal{D} = \bigcup_n B(\lambda_n, \varepsilon)$. Being open and disconnected, $\mathcal{D}$ admits kernels prescribed independently on each set, which lets us define

$$\xi(z) = \begin{cases} 1/\kappa^\circ(z), & z \in B(\lambda_n, \varepsilon),\ \lambda_n \neq 0, \\ 0, & z \in B(0, \varepsilon). \end{cases} \tag{34}$$

On each set ξ is either constant or the reciprocal of a non-vanishing polynomial in $(z, \bar{z})$, so $\xi \in \mathcal{C}^{N-1}(\overline{\mathcal{D}})$ and Lemma 9 applies. By construction ξ inverts the nonzero part of $\sigma(\mathbf{L}^\circ)$ and sets the rest to 0, which is the spectral characterisation of the Drazin inverse (Campbell & Meyer, 2009), thence $\xi(\mathbf{L}) = (\mathbf{L}^\circ)^D$. Multiplicativity then gives

$$(\kappa^\uparrow \xi)(\mathbf{L}) = \kappa^\uparrow(\mathbf{L})\,\xi(\mathbf{L}) = \mathbf{L}^\uparrow(\mathbf{L}^\circ)^D = \mathbf{L}^{\hat{\Phi}}, \tag{35}$$

and, applied repeatedly, $(\kappa^\uparrow \xi)^k(\mathbf{L}) = (\mathbf{L}^{\hat{\Phi}})^k$ for all $k \geq 0$. Finally $h_{\text{rat}} = \mathbf{c}[0] + \sum_{k=1}^{K} \mathbf{c}[k](\kappa^\uparrow \xi)^k$ on $\mathcal{D}$, where on the points with $\lambda_n \neq 0$ we have $\kappa^\uparrow \xi = \kappa^\uparrow / \kappa^\circ$, recovering the quotient of equation 7, while on $B(0, \varepsilon)$ every term with $k \geq 1$ vanishes and the kernel reduces to $\mathbf{c}[0]$, matching h_{rat} at $\Re(z) = 0$. Linearity then gives

$$h_{\text{rat}}(\mathbf{L}) = \sum_{k=0}^{K} \mathbf{c}[k]\,\big(\mathbf{L}^{\hat{\Phi}}\big)^k. \tag{36}$$

□

B.4 Proof of Theorem 3 (Real-valued twins filters)

As in Chan et al. (2026a), leveraging the Vandermonde matrix, any real-valued filters expressible by $h_{\text{sum}}(\mathbf{L})$ and $h_{\text{rat}}(\mathbf{L})$ i.e. elements of respectively:

$$\{h_{\text{sum}}(\mathbf{L}) : (\mathbf{a}, \mathbf{b}) \in \mathbb{C}^{K_1+1} \times \mathbb{C}^{K_2+1}\} \cap \mathbb{R}^{N\times N} \quad \text{and} \quad \{h_{\text{rat}}(\mathbf{L}) : \mathbf{c} \in \mathbb{C}^{K+1}\} \cap \mathbb{R}^{N\times N} \tag{37}$$

as defined in equation 8, can be expressed as elements of $\{h_{\text{sum}}(\mathbf{L}) : (\mathbf{a}, \mathbf{b}) \in \mathbb{R}^{K_1+1} \times \mathbb{R}^{K_2+1}\}$ and $\{h_{\text{rat}}(\mathbf{L}) : \mathbf{c} \in \mathbb{R}^{K+1}\}$ guaranteeing their equality — this holds for both diagonalizable and non-diagonalizable $\mathbf{L}$ thanks to conjugate eigenvalues occurrence in the Jordan decomposition. Since we need $h_{\text{sum}}(\mathbf{L}), h_{\text{rat}}(\mathbf{L})$ to be real, we start with $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}$. By identifying the unique coefficients for the Chebyshev expansions of $\widetilde{\mathbf{L}}^{\circ}, \widetilde{\mathbf{L}}^{\uparrow}, \widetilde{\mathbf{Z}}$, starting with maximal order K

$$\boldsymbol{\alpha}[K]T_K(\widetilde{\mathbf{L}}^{\circ}) = \boldsymbol{\alpha}[K]\sum_{k=0}^{K}\mathbf{a}_K[k](\widetilde{\mathbf{L}}^{\circ})^k = \boldsymbol{\alpha}[K]\sum_{k=0}^{K}\left(\frac{\mathbf{a}_K[k]}{R_R}\sum_{m=0}^{k}\binom{k}{m}(\mathbf{L}^{\circ})^m(c_R\mathbf{I})^{k-m}\right), \tag{38}$$

$$\boldsymbol{\beta}[K]T_K(\widetilde{\mathbf{L}}^{\uparrow}) = \boldsymbol{\beta}[K]\sum_{k=0}^{K}\mathbf{b}_K[k](\widetilde{\mathbf{L}}^{\uparrow})^k = \boldsymbol{\beta}[K]\sum_{k=0}^{K}(-j)^k\frac{\mathbf{b}_K[k]}{R_I}(\mathbf{L}^{\uparrow})^k, \tag{39}$$

$$\boldsymbol{\eta}[K]T_K(\widetilde{\mathbf{Z}}) = \boldsymbol{\eta}[K]\sum_{k=0}^{K}\mathbf{c}_K[k](\widetilde{\mathbf{Z}})^k = \boldsymbol{\eta}[K]\sum_{k=0}^{K}(-j)^k\frac{\mathbf{c}_K[k]}{R_Q}(\mathbf{L}^{\hat{\Phi}})^k. \tag{40}$$

Given T_K is of maximal order K, no other terms except for $\boldsymbol{\alpha}[K]$ contribute to the expansion in $\mathbf{a}[K]$, therefore we have exactly $\boldsymbol{\alpha}[K]\frac{\mathbf{a}_K[K]}{R_R} = \mathbf{a}[K]$. Similarly, considering the two remaining expressions $\boldsymbol{\beta}[K](-j)^K\frac{\mathbf{b}_K[K]}{R_I} = \mathbf{b}[K]$, and $\boldsymbol{\eta}[K](-j)^K\frac{\mathbf{c}_K[K]}{R_Q} = \mathbf{c}[K]$. Hence the following holds:

- $\boldsymbol{\alpha}[K]$ is real,
- $\boldsymbol{\beta}[K]$ is real for even K and purely imaginary for odd K,
- $\boldsymbol{\eta}[K]$ is real for even K and purely imaginary for odd K.

For lower order coefficients, we can apply the same argument recursively (while subtracting the previous orders for which we have already determined the coefficients) to show that $\boldsymbol{\alpha}[k]$ is real for all k and $\boldsymbol{\beta}[k], \boldsymbol{\eta}[k]$ are real for even k and purely imaginary for odd k. As this procedure can be performed for all $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}$,

$$\begin{aligned}&\{h_{\text{sum}}(\mathbf{L}) : (\boldsymbol{\alpha}, \boldsymbol{\beta}) \in \mathbb{C}^{K_1+1} \times \mathbb{C}^{K_2+1}\} \cap \mathbb{R}^{N\times N}\\ &\quad = \{h_{\text{sum}}(\mathbf{L}) : \boldsymbol{\alpha}[k] \in \mathbb{R},\ \boldsymbol{\beta}[2k] \in \mathbb{R},\ \boldsymbol{\beta}[2k+1] \in j\mathbb{R}\},\end{aligned} \tag{41}$$

$$\begin{aligned}&\{h_{\text{rat}}(\mathbf{L}) : \boldsymbol{\eta} \in \mathbb{C}^{K+1}\} \cap \mathbb{R}^{N\times N}\\ &\quad = \{h_{\text{rat}}(\mathbf{L}) : \boldsymbol{\eta}[2k] \in \mathbb{R},\ \boldsymbol{\eta}[2k+1] \in j\mathbb{R}\}.\end{aligned} \tag{42}$$

This completes the proof of Theorem 3. □

Further intuition on reparameterized matrices. Additional intuition on the nature of the real-matrix condition is provided in the following lemma and expressions, which in one hand show how the Chebyshev polynomial parity interacts with the matrix parity of $\widetilde{\mathbf{L}}^{\uparrow}$ and $\widetilde{\mathbf{Z}}$ to yield a real-matrix filter and in the other hand show how the spectral response is decomposed into real and imaginary parts that are even and odd in λ_{I} respectively.

Lemma 10 (Parity of Chebyshev polynomials)**.** *Let $x \in [-1, 1]$, the k-th degree Chebyshev polynomials T_k satisfies*

$$T_k(-x) = (-1)^k T_k(x) \Rightarrow \begin{cases}\textit{even function} & k \textit{ even},\\ \textit{odd function} & k \textit{ odd}.\end{cases} \tag{43}$$

Proof. For $x \in [-1, 1]$, $T_k(-x) = \cos(k\arccos(-x)) = \cos(k(\pi - \arccos(x))) = (-1)^k T_k(x)$. □

Lemma 11 (Matrix parity of Chebyshev polynomials)**.** *Let* $\mathbf{Q}$ *be a matrix with purely imaginary entries, giving*

$$\mathbf{Q}^i \in \begin{cases} \mathbb{R}^{N\times N} & i \text{ even,} \\ j\mathbb{R}^{N\times N} & i \text{ odd.} \end{cases} \tag{44}$$

Parallely, from Lemma 10 the Chebyshev polynomial T_k *has the same parity as* k*, so it contains only monomials* x^i *with* $i \equiv k \pmod 2$*. Combining with equation 44:*

$$A_k = T_k(\mathbf{Q}) \in \begin{cases} \mathbb{R}^{N\times N} & k \text{ even,} \\ j\,\mathbb{R}^{N\times N} & k \text{ odd.} \end{cases} \tag{45}$$

Proof. Write $T_k(x) = \sum_{i\equiv k\,(2)} a_i x^i$ with $a_i \in \mathbb{R}$. Then $A_k = \sum_{i\equiv k\,(2)} a_i \mathbf{Q}^i$. Each summand $a_i\mathbf{Q}^i$ lies in $\mathbb{R}^{N\times N}$ if i even, in $j\mathbb{R}^{N\times N}$ if i odd, and all summands share the same parity as k, so the whole sum is either in $\mathbb{R}^{N\times N}$ or $j\mathbb{R}^{N\times N}$. □

Frequency response as a function of Chebyshev polynomials. The spectral response at eigenvalue $\lambda[n] = \lambda_{\mathrm{R}}[n] + j\lambda_{\mathrm{I}}[n]$ of the sum filter $h_{\text{sum}}(\lambda[n])$ can be expressed in terms of Chebyshev polynomials as

$$\Re\big(h_{\text{sum}}(\lambda[n])\big) = \sum_k \boldsymbol{\alpha}[k] T_k(x_{R,n}) + \sum_i \boldsymbol{\beta}[2i]\, T_{2i}(x_{I,n}), \tag{46}$$

$$\Im\big(h_{\text{sum}}(\lambda[n])\big) = \sum_i \boldsymbol{\beta}[2i+1]\, T_{2i+1}(x_{I,n}), \tag{47}$$

where $x_{R,n} = (\lambda_{\mathrm{R}}[n] - c_R)/R_R$ and $x_{I,n} = \lambda_{\mathrm{I}}[n]/R_I$. For the ratio filter $h_{\text{rat}}(\lambda[n])$ we have

$$\Re\big(h_{\text{rat}}(\lambda[n])\big) = \sum_i \boldsymbol{\eta}[2i]\, T_{2i}(x_{Q,n}), \tag{48}$$

$$\Im\big(h_{\text{rat}}(\lambda[n])\big) = \sum_i \boldsymbol{\eta}[2i+1]\, T_{2i+1}(x_{Q,n}), \tag{49}$$

where $x_{Q,n} = \left(-j\frac{j\lambda_{\mathrm{I}}[n]}{\lambda_{\mathrm{R}}[n]}\right)/R_Q$.

B.5 Proof of Theorem 4 (IIR correspondence)

Möbius identity. On the cycle, $\lambda_{\mathrm{R}}[n] = 1 - \cos\boldsymbol{\theta}[n]$ and $\lambda_{\mathrm{I}}[n] = -\sin\boldsymbol{\theta}[n]$, so

$$\boldsymbol{\mu}[n] = \frac{j\lambda_{\mathrm{I}}[n]}{\lambda_{\mathrm{R}}[n]} = \frac{-j\sin\boldsymbol{\theta}[n]}{1-\cos\boldsymbol{\theta}[n]} = \frac{-2j\sin(\boldsymbol{\theta}[n]/2)\cos(\boldsymbol{\theta}[n]/2)}{2\sin^2(\boldsymbol{\theta}[n]/2)} = -j\cot\left(\frac{\boldsymbol{\theta}[n]}{2}\right). \tag{50}$$

Writing $\boldsymbol{z}[k] = e^{j\boldsymbol{\theta}[n]}$ and using $\cot(\theta/2) = j(z+1)/(z-1)$,

$$\boldsymbol{\mu}[n] = -j\cdot j\,\frac{\boldsymbol{z}[n]+1}{\boldsymbol{z}[n]-1} = \frac{\boldsymbol{z}[n]+1}{\boldsymbol{z}[n]-1}, \tag{51}$$

which is equation 50. This is a Möbius transform, projecting from the unit circle onto the imaginary axis.

Rational structure equation 19. Substituting equation 50 into a degree-K polynomial,

$$\sum_{k=0}^{K} \mathbf{c}[k]\boldsymbol{\mu}[n]^k = \frac{\sum_{k=0}^{K}\mathbf{c}[k](\boldsymbol{z}[n]+1)^k(\boldsymbol{z}[n]-1)^{K-k}}{(\boldsymbol{z}[n]-1)^K} = \frac{Q_K(\boldsymbol{z}[n])}{(\boldsymbol{z}[n]-1)^K}, \tag{52}$$

where Q_K is a polynomial of degree $\le K$ obtained by multiplying the denominator. This is a rational function of $\boldsymbol{z}[n]$ of type $[K/K]$ whose denominator $(\boldsymbol{z}[n]-1)^K$ places all K poles at $\boldsymbol{z} = 1$, the DC mode. A rational spectral response corresponds to an IIR (ARMA) vertex-domain filter and hence has non-compact support. □

B.6 Proof of Theorem 5 (Reachability gap)

Starting notation. $\Delta u[\ell] = u[\ell+1] - 2u[\ell] + u[\ell-1]$ is the discrete Laplacian and $\mathbf{g}_k := (\mathbf{L}^{\hat{\Phi}})^k \delta_0$ the k-th kernel, so that $\mathbf{g}_k[d]$ is the impulse response of $(\mathbf{L}^{\hat{\Phi}})^k$ at distance d. Since $\mathbf{L}$ is normal on C_N, the spectral split gives

$$\mathbf{L}^{\circ} = \mathbf{I} - \tfrac{1}{2}(\mathbf{W} + \mathbf{W}^T), \qquad \mathbf{L}^{\uparrow} = \tfrac{1}{2}(\mathbf{W}^T - \mathbf{W}), \tag{53}$$

so that, acting on a signal $u \in \mathbb{R}^N$,

$$(\mathbf{L}^{\circ} u)[\ell] = -\tfrac{1}{2}(\Delta u)[\ell], \qquad (\mathbf{L}^{\uparrow} u)[\ell] = \tfrac{1}{2}\big(u[\ell+1] - u[\ell-1]\big) : \tag{54}$$

a second difference and a centered first difference.

Proposition 2 (Identity on ratio operator)**.** $\mathbf{L}^{\circ}\mathbf{L}^{\hat{\Phi}} = \mathbf{L}^{\uparrow}$, and $\mathbf{L}^{\hat{\Phi}}\mathbf{1} = \mathbf{0}$, $\mathbf{1}^T\mathbf{L}^{\hat{\Phi}} = \mathbf{0}^T$.

Proof. All operators commute, so it suffices to observe their spectral definition. □

Lemma 12 (Linear form of $\mathbf{g}_1$)**.** $\mathbf{g}_1[\ell] = 2\ell/N - 1$ *for* $1 \le \ell \le N-1$*, and* $\mathbf{g}_1[0] = 0$*.*

Proof. Apply Proposition 2 to δ_0 and use equation 54:

$$\mathbf{L}^{\circ}\mathbf{g}_1 = \mathbf{L}^{\uparrow}\delta_0 = \tfrac{1}{2}(\delta_{-1} - \delta_1) \qquad \Longleftrightarrow \qquad \Delta\mathbf{g}_1 = \delta_1 - \delta_{-1}. \tag{55}$$

By equation 55 the second difference of $\mathbf{g}_1$ vanishes at every vertex other than $\ell = \pm 1$. A sequence with vanishing second difference on a discrete interval is affine on that interval, so $\mathbf{g}_1[\ell] = a + b\ell$ for $2 \le \ell \le N-2$. Evaluating $\Delta\mathbf{g}_1$ at $\ell = 0$ and inserting the affine form gives $\mathbf{g}_1[0] = a + \frac{bN}{2}$. An additional condition comes from the fact that $\mathbf{L}^{\hat{\Phi}}$ sets the zero-th frequency to 0, i.e., $\hat{\mathbf{g}}_1[0] = \mu[0] = 0$ and therefore $\sum_\ell \mathbf{g}_1[\ell] = 0$. Combining these two conditions gives $a = -\frac{bN}{2}$ and $\mathbf{g}_1[0] = 0$. Once again evaluating $\Delta\mathbf{g}_1$ at $\ell = 1$ gives $\mathbf{g}_1[2] - 2\mathbf{g}_1[1] + \mathbf{g}_1[0] = 1$, which with the affine form gives $b = 2/N$. Hence $\mathbf{g}_1[\ell] = 2\ell/N - 1$ for $1 \le \ell \le N-1$. □

Lemma 13 (Degree and leading coefficient)**.** *For* $N \ge 3k+2$ *there is a polynomial* P_k *of degree exactly* k *with* $\mathbf{g}_k[d] = P_k(d)$ *on the interval* $k < d < N-k$*, whose leading coefficient obeys*

$$a_k = -\frac{2a_{k-1}}{k}, \qquad a_1 = \frac{2}{N}, \qquad \Longrightarrow \qquad a_k = \frac{(-1)^{k-1}2^k}{N\,k!}. \tag{56}$$

Proof. We proceed by induction on k,

Base case $k = 1$: Lemma 12 gives $P_1(d) = 2d/N - 1$, of degree 1 with $a_1 = 2/N$.

Inductive hypothesis: Suppose the statement holds at order $k-1$, so that $\mathbf{g}_{k-1} = P_{k-1}$ on $k-1 < d < N-k+1$ with $\deg P_{k-1} = k-1$ and leading coefficient $a_{k-1} \neq 0$.

Applying Proposition 2 to $\mathbf{g}_k = \mathbf{L}^{\hat{\Phi}}\mathbf{g}_{k-1}$ and using equation 54,

$$-\tfrac{1}{2}\,\Delta\mathbf{g}_k = \mathbf{L}^{\uparrow}\mathbf{g}_{k-1}. \tag{57}$$

Let d lie in the interval $k < d < N-k$. Then both $d-1$ and $d+1$ lie in $k-1 < d < N-k+1$, where the inductive hypothesis applies, so by equation 54 the right-hand side of equation 57 equals

$$Q(d) := \tfrac{1}{2}\big(P_{k-1}(d+1) - P_{k-1}(d-1)\big). \tag{58}$$

Expanding, the terms of degree $k-1$ cancel and Q has degree exactly $k-2$ with leading coefficient $(k-1)a_{k-1}$.

On the interval, equation 57 reads $\Delta\mathbf{g}_k = -2Q$ and to it we seek a particular solution that is a polynomial of degree k with leading coefficient b. Since $\Delta(b\,d^k) = b\,k(k-1)\,d^{k-2} + O(d^{k-3})$, matching the coefficients of d^{k-2} gives

$$b\,k(k-1) = -2(k-1)a_{k-1} \qquad \Longrightarrow \qquad b = -\frac{2a_{k-1}}{k} \neq 0. \tag{59}$$

The homogeneous solutions of $\Delta u = 0$ on a discrete interval are the affine sequences, which for $k \ge 2$ cannot cancel the term of degree k, so $a_k \neq 0$. Hence $\mathbf{g}_k$ agrees on the interval with a polynomial of degree exactly k, and $a_k = b$, which is equation 56. The hypothesis $N \ge 3k+2$ guarantees that the interval contains more than $k+1$ vertices, so the polynomial is not degenerate. Unrolling the recursion leads to equation 56. □

Lemma 14 (Bernoulli form)**.** *Let $s = d/N$ and $c_k = (-1)^{k-1}2^k/k!$. Then, uniformly on $k < d < N - k$,*

$$\mathbf{g}_k[d] = N^{k-1} c_k \, B_k(s) + O\big(N^{k-3}\big)\,, \tag{60}$$

with equality for $k = 1$.

Proof. Set $\tilde{P}_k(s) := P_k(Ns)/N^{k-1}$, the k-th kernel expressed in the fraction distance $s = d/N$. Its leading term is $a_k(Ns)^k/N^{k-1} = Na_k s^k$, so by equation 56 its leading coefficient is $Na_k = (-1)^{k-1}2^k/k! =: c_k$, which is independent of N. This normalization is what allows a limit to be taken when computing derivatives.

Under $d = Ns$, the second difference acts as $N^{-2}\,\mathrm{d}^2/\mathrm{d}s^2$ and the centered first difference as $N^{-1}\,\mathrm{d}/\mathrm{d}s$, each with relative error $O(N^{-2})$ on polynomials of fixed degree. Hence differentiating over s, equation 57 becomes $P_k'' = -2N\,P_{k-1}'$, and substituting $P_k = N^{k-1}\tilde{P}_k$ cancels every power of N, leaving

$$\tilde{P}_k'' = -2\,\tilde{P}_{k-1}', \qquad \tilde{P}_1(s) = 2s - 1 = 2B_1(s). \tag{61}$$

Try $\tilde{P}_k = c_k B_k$. The Bernoulli polynomials satisfy $B_k' = k\,B_{k-1}$, hence $B_k'' = k(k-1)B_{k-2}$, so equation 61 holds if and only if

$$c_k\,k(k-1) = -2\,c_{k-1}(k-1) \qquad \Longleftrightarrow \qquad c_k = -\frac{2c_{k-1}}{k}, \tag{62}$$

which is the recursion equation 56 of Lemma 13. With $c_1 = 2$ from equation 61, this gives $c_k = (-1)^{k-1}2^k/k!$.

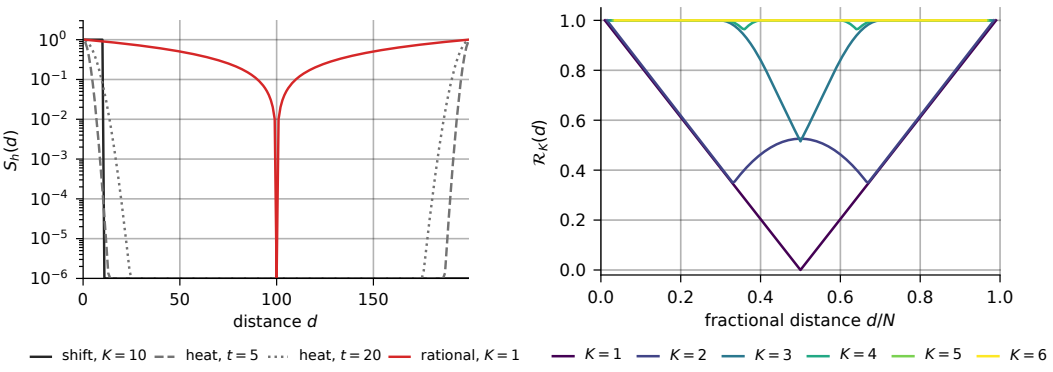
To fully determine solution of the second-order equation 61 it first satisfies, $\int_0^1 \tilde{P}_k = 0$, which follows from $\hat{\mathbf{g}}_k[0] = \mu[0]^k = 0$. The second is $\tilde{P}_k(1-s) = (-1)^k\tilde{P}_k(s)$, which follows from $\mathbf{g}_k[N-\ell] = (-1)^k\mathbf{g}_k[\ell]$. This equality is surmised from the expression in Lemma 12 where $\mathbf{g}_1$ is an odd sequence, furthermore $\mathbf{g}_k = \mathbf{g}_1^{*k}$, and the convolution of two odd sequences is even, so parity alternates with k. Both conditions are satisfied for B_k, by its vanishing mean and by the reflection formula $B_k(1-s) = (-1)^k B_k(s)$. The ansatz of Bernoulli polynomials therefore holds.

The only approximation is the replacement of differences by derivatives when considering polynomials, leading to $O(N^{-2})$ term. At k steps relative error remains $O(N^{-2})$, hence the recursion gives equation 60. For $k = 1$ no approximation is made, since Lemma 12 is exact. □

Proof of Theorem 5. (i): Since $\mathbf{W}^m\delta_0 = \delta_m$, the kernel of $h = \sum_{m\le K}\gamma_m\mathbf{W}^m$ is $\sum_{m\le K}\gamma_m\delta_m$. It therefore equals γ_d at $d \le K$ and vanishes identically for $d > K$, hence $S_h(d) = 0$ there. The coefficients γ_m are unconstrained, so the response on the first K hops is arbitrary.

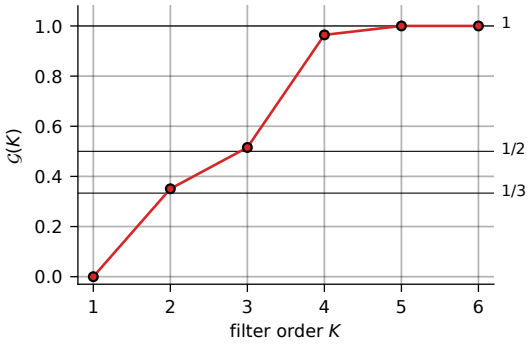
(ii): By equation 53, $e^{-t\mathbf{L}^\circ} = e^{-t}\exp\big(\frac{t}{2}(\mathbf{W} + \mathbf{W}^T)\big)$. The generating identity $e^{\frac{t}{2}(z+z^{-1})} = \sum_d I_d(t)z^d$, where I_d is the modified Bessel function of first kind (Watson, 1944), leads to the impules response $h[d,0]$ as $e^{-t}I_d(t)$, and the peak is at $d = 0$ because $I_0 \ge I_d$ for all d. For the bound of $S_h(d) = I_d(t)/I_0(t)$, compare the defining series termwise: the m-th term of $I_d(t)$ is $(t/2)^{d+2m}/\big(m!\,(d+m)!\big)$ and that of $\frac{(t/2)^d}{d!}I_0(t)$ is $(t/2)^{d+2m}/\big(d!\,(m!)^2\big)$, so their ratio is $d!\,m!/(d+m)! \le 1$. Summing over m gives $I_d(t) \le \frac{(t/2)^d}{d!}I_0(t)$, and $S_h(d) \le \frac{(t/2)^d}{d!}$, which is the claim.

(iii): The term $\boldsymbol{\eta}[0]\mathbf{I}_N$ has kernel $\boldsymbol{\eta}[0]\delta_0$, which vanishes on $K < d < N - K$; only the terms with $k \ge 1$ contribute there. On the same interval $K < d < N - K$, Lemma 14 applies to each $\mathbf{g}_k$ with $k \le K$. A weighted sum $\mathbf{g}_k$ with weights $\boldsymbol{\eta}[k]$ gives $p(s)$ in equation 23. Substituting $p(s) \approx h[d,0]$ into $S_h(d)$, first yields $|p(s)|/\max_{q\in[0,1]}|p(q)|$. The remaining discrepancy is between the maximum over the vertices $v \in \mathcal{V}$ and the supremum over the continuum $[0,1]$, which for polynomials of fixed degree is $O(N^{-2})$.



(a) Sensitivity profiles

(b) Best filter sensitivity across distances



(c) $\mathcal{G}(K)$ function

Figure 4: a) Sensitivity profiles computed for shift, heat ($t = 5$), heat ($t = 20$), and ratio ($K = 1$) filters. b) Plot of the optimal filter sensitivity across distances. c) Plot of the $\mathcal{G}(K)$ function. Throughout N is set to 200.

C ACCOMPANYING INFORMATION

C.1 SPECTRAL CONJUGATE DEFINITION

Definition C.1 (Spectral conjugate)**.** Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ and consider its Jordan canonical form $\mathbf{A} = \mathbf{P}(\mathbf{\Lambda} + \mathbf{N})\mathbf{P}^{-1}$, where $\mathbf{\Lambda}$ is diagonal and $\mathbf{N}$ is nilpotent, i.e., there exists $k \in \mathbb{N}$ s.t. $\mathbf{N}^k = \mathbf{0}$. Then spectral conjugate $\mathbf{A}^\sharp$ is defined as

$$\mathbf{A}^\sharp = \mathbf{P}\overline{\mathbf{\Lambda}}\mathbf{P}^{-1}, \tag{63}$$

where $\overline{\mathbf{\Lambda}}$ is the complex conjugate of $\mathbf{\Lambda}$.

Interestingly enough, the spectral conjugate $\mathbf{A}^\sharp$ is different from the Hermitian conjugate $\mathbf{A}^H = \overline{\mathbf{A}^T}$, which is defined as the complex conjugate of the transpose of $\mathbf{A}$. The spectral conjugate $\mathbf{A}^\sharp$ preserves the eigenvectors and generalized eigenvectors of $\mathbf{A}$, while only taking the complex conjugate of its eigenvalues. In contrast, the Hermitian conjugate $\mathbf{A}^H$ does not preserve the eigenvectors of $\mathbf{A}$ in general.

Satisfying properties. Let $\tau : \mathbb{C} \to \mathbb{C}$ be the map $\tau(z) = \overline{z}$, and functional extension to $\mathbf{A} \in \mathbb{C}^{n\times n}$ as $\tau(\mathbf{A}) = \mathbf{A}^\sharp$. Then the spectral conjugate satisfies the following properties:

- $\tau(\mathbf{A} + \mathbf{B}) = \tau(\mathbf{A}) + \tau(\mathbf{B})$, if $\mathbf{A}$ and $\mathbf{B}$ commute,
- $\tau(\mathbf{A})B = \mathbf{B}\,\tau(\mathbf{A})$, if $\mathbf{A}$ and $\mathbf{B}$ commute,
- $\tau(\alpha\mathbf{A}) = \overline{\alpha}\,\tau(\mathbf{A})$, for any $\alpha \in \mathbb{C}$,
- $\tau(\mathbf{P}\mathbf{A}\mathbf{P}^{-1}) = \mathbf{P}\,\tau(\mathbf{A})\,\mathbf{P}^{-1}$, for any invertible matrix $\mathbf{P}$.

Computation. In practice $\mathbf{A}^\sharp$ is computed from the complex Schur decomposition $\mathbf{A} = \mathbf{Q}\mathbf{T}\mathbf{Q}^H$, with $\mathbf{T}$ upper triangular. The diagonal is set by conjugating the eigenvalues, $\mathbf{F}_{ii} = \overline{\mathbf{T}_{ii}}$, and the strictly upper-triangular entries are filled in super-diagonal by super-diagonal using the Parlett recurrence, which enforces the commutation $\mathbf{F}\mathbf{T} = \mathbf{T}\mathbf{F}$ (Parlett, 1976). For clustered or repeated eigenvalues the scalar recurrence is ill-conditioned and the blocked variant is used instead, with each diagonal block set to $\overline{\mu}\mathbf{I}$ for the cluster mean μ and the off-diagonal blocks recovered from Sylvester equations (Davies & Higham, 2003; Higham, 2008). Recomposing, $\mathbf{A}^\sharp = \mathbf{Q}\mathbf{F}\mathbf{Q}^H$, yields the spectral conjugate without forming the Jordan decomposition explicitly.

C.2 DRAZIN INVERSE DEFINITION

Definition C.2 (Drazin inverse)**.** Let $\mathbf{A} \in \mathbb{C}^{n\times n}$ and let $k = \mathrm{ind}(\mathbf{A})$ denote its index, i.e. the smallest non-negative integer such that $\mathrm{rank}(\mathbf{A}^k) = \mathrm{rank}(\mathbf{A}^{k+1})$, equivalently the size of the largest Jordan block associated with the eigenvalue 0. The Drazin inverse $\mathbf{A}^D$ is the unique matrix satisfying

$$\mathbf{A}^D\mathbf{A}\,\mathbf{A}^D = \mathbf{A}^D, \qquad \mathbf{A}\,\mathbf{A}^D = \mathbf{A}^D\mathbf{A}, \qquad \mathbf{A}^{k+1}\mathbf{A}^D = \mathbf{A}^k. \tag{64}$$

The commutativity condition $\mathbf{A}\mathbf{A}^D = \mathbf{A}^D\mathbf{A}$ is what distinguishes the Drazin inverse from the Moore–Penrose pseudoinverse $\mathbf{A}^\dagger$, which satisfies $\mathbf{A}\mathbf{A}^\dagger\mathbf{A} = \mathbf{A}$ but does not in general commute with $\mathbf{A}$.

Spectral characterisation. Writing the Jordan decomposition of $\mathbf{A}$ with its invertible and nilpotent parts separated,

$$\mathbf{A} = \mathbf{P}\begin{pmatrix}\mathbf{C} & \mathbf{0}\\ \mathbf{0} & \mathbf{N}\end{pmatrix}\mathbf{P}^{-1}, \qquad \mathbf{C}\text{ invertible}, \quad \mathbf{N}^k = \mathbf{0}, \tag{65}$$

the Drazin inverse admits the closed form

$$\mathbf{A}^D = \mathbf{P}\begin{pmatrix}\mathbf{C}^{-1} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{pmatrix}\mathbf{P}^{-1}. \tag{66}$$

Equivalently, $\mathbf{A}^D$ acts on the spectrum of $\mathbf{A}$ through

$$\lambda \;\longmapsto\; \begin{cases}\lambda^{-1}, & \lambda \neq 0,\\ 0, & \lambda = 0,\end{cases} \tag{67}$$

inverting the non-zero part of the spectrum while annihilating the nilpotent part. In particular $\mathbf{A}^D$ shares the eigenvectors and generalized eigenvectors of $\mathbf{A}$, which is precisely why the two operators commute.

Computation. In practice $\mathbf{A}^D$ is obtained from a Schur decomposition $\mathbf{A} = \mathbf{Q}\mathbf{T}\mathbf{Q}^H$ whose eigenvalues are reordered so that the non-zero ones occupy the leading diagonal block: inverting that block, zeroing the trailing nilpotent block, and recovering the coupling term from a Sylvester equation yields $\mathbf{A}^D$ without ever forming the Jordan decomposition explicitly (Stanimirović et al., 2015; Campbell & Meyer, 2009). When combined with the computation of the spectral conjugate (Appendix C.1), the Schur decomposition is already available and hence only the pointwise scalar inversion and computation of coupling terms is required. Ultimately the computation of $\mathbf{L}^\circ, \mathbf{L}^\uparrow, \mathbf{L}^\uparrow(\mathbf{L}^\circ)^D$ is dominated by the Schur decomposition.

C.3 Lifting holomorphic function to matrix

The evaluation of a holomorphic function h at a matrix $\mathbf{L}$ follows from equation 2 where $\frac{1}{z-\lambda}$ is replaced by the resolvent $(z\mathbf{I} - \mathbf{L})^{-1}$ (Auscher et al., 1997)

$$h(\mathbf{L}) = \frac{1}{2\pi i} \oint_\Gamma h(z)\,(z\mathbf{I} - \mathbf{L})^{-1}\,dz, \tag{68}$$

where Γ encloses the spectrum $\sigma(\mathbf{L})$ and $(z\mathbf{I} - \mathbf{L})$ is invertible for all $z \in \Gamma$.

C.4 Wirtinger derivative and holomorphicity of polynomial and rational kernels

Wirtinger derivative. Consider a complex number $z = x + jy$ and its conjugate $\bar{z} = x - jy$, the Wirtinger derivatives are defined as follows:

$$\partial_z = \frac{1}{2}(\partial_x - j\partial_y), \quad \partial_{\bar{z}} = \frac{1}{2}(\partial_x + j\partial_y). \tag{69}$$

For notation simplicity, we write $\partial := \partial_z$ and $\bar{\partial} := \partial_{\bar{z}}$. In particular the derivatives satisfy $\bar{\partial} z = 0$ and $\partial \bar{z} = 0$.

Holomorphicity. Define a degree K polynomial kernel as:

$$p_K : \sigma(\mathbf{L}) \to \mathbb{C}, \quad z \mapsto \sum_{k=0}^{K} c_k z^k, \tag{70}$$

where $c_k \in \mathbb{C}$ are the coefficients. Define a $[K_1, K_2]$ rational kernel as:

$$r_{K_1,K_2} : \sigma(\mathbf{L}) \to \mathbb{C}, \quad z \mapsto \frac{p_{K_1}(z)}{p_{K_2}(z)}. \tag{71}$$

Since no instance of $\bar{z}$ is in the expressions of p_K and r_{K_1,K_2}, the Wirtinger derivatives $\bar{\partial} p_K, \bar{\partial} r_{K_1,K_2}$ are null. Thereby p_K and r_{K_1,K_2} are holomorphic. Consequently, as canonical and Chebyshev polynomials can be expressed in the form p_K, they are holomorphic. Similarly, Cayley polynomials Levie et al. (2018) are special case of the rational kernels, hence they also are holomorphic.

C.5 Kernels gain for non-diagonalizable operator

Upper bound on gain of heat kernel Consider the heat kernel $h(\lambda) = e^{-t\lambda}$, which is holomorphic since $\bar{\partial} h = 0$, so that $\partial^k h = h^{(k)}$ and $h^{(k)}(\lambda) = (-t)^k e^{-t\lambda}$. Substituting into equation 1 for a Jordan block $\mathbf{J}_n$ of size m gives

$$h(\mathbf{J}_n) = e^{-t\lambda_n} \sum_{k=0}^{m-1} \frac{(-t)^k}{k!} \mathbf{N}_n^k = e^{-t\lambda_n} \begin{pmatrix} 1 & -t & \frac{t^2}{2!} & \cdots & \frac{(-t)^{m-1}}{(m-1)!} \\ 0 & 1 & -t & \cdots & \frac{(-t)^{m-2}}{(m-2)!} \\ 0 & 0 & 1 & \cdots & \frac{(-t)^{m-3}}{(m-3)!} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}. \tag{72}$$

Since $\|\mathbf{N}_n^k\|_2 = 1$ (with $\|\cdot\|_2 = \sigma_{\max}(\cdot)$ the operator norm) for $k \leq m-1$, the gain of the block obeys

$$\left\|h(\mathbf{J}_n)\right\|_2 \;\leq\; e^{-t\Re(\lambda_n)} \sum_{k=0}^{m-1} \frac{t^k}{k!} \;=\; e^{-t\Re(\lambda_n)}\, O\big(t^{m-1}\big)\,, \tag{73}$$

so defectiveness contributes at most a polynomial factor in t, dominated by the exponential whenever $\Re(\lambda_n) > 0$. □

Lower bound on the gain of $e^{\beta\mathbf{L}^\uparrow}$. In the same way as for the heat kernel, on a Jordan block $\mathbf{J}_n$ of size m, $\mathbf{L}^\uparrow$ acts as $j\lambda_I[n]\mathbf{I} + \frac{1}{2}\mathbf{N}_n$, so $e^{\beta\mathbf{L}^\uparrow}\big|_{\mathbf{J}_n} = e^{j\beta\lambda_I[n]}\, e^{\beta\mathbf{N}_n/2}$, with inverse $e^{-j\beta\lambda_I[n]}\, e^{-\beta\mathbf{N}_n/2}$. Since $\|\mathbf{N}_n^k\|_2 = 1$ for $k \leq m-1$, its smallest singular value satisfies

$$\min_{\|\hat{\mathbf{x}}\|_2=1} \left\|e^{\beta\mathbf{L}^\uparrow}\big|_{\mathbf{J}_n}\hat{\mathbf{x}}\right\|_2 = \left\|e^{-\beta\mathbf{N}_n/2}\right\|_2^{-1} \;\geq\; \left(\sum_{k=0}^{m-1} \frac{(|\beta|/2)^k}{k!}\right)^{-1} \;\geq\; \left(1 + \tfrac{|\beta|}{2}\right)^{-(m-1)}. \tag{74}$$

The gain is therefore lower bounded by a factor in β. For frequencies (eigenvalues) in Jordan blocks of size $m = 1$, the gain is unitary. For diagonalizable $\mathbf{L}$ (i.e. $m = 1$ for all blocks), the gain is unitary for all frequencies. □

C.6 Minimal polynomial

This subsection recalls few propositions and corollaries of Chan et al. (2026a) applicable for diagonalizable $\mathbf{L}$ on the minimal polynomial of the dissipative (diffusion) and asymmetric (advection) operators. This determines the maximal layer depth, providing additional guidance for the design of the TwinS-GCN architecture.

Proposition 3. Let N_C be the number of pairs of conjugate eigenvectors, the degree of the minimal polynomial of $\mathbf{L}^\circ$ is $\deg(m_{\mathbf{L}^\circ}(\lambda)) = N - N_C$.

Corollary 2. There exists a set of coefficients $c_k \in \mathbb{C}$ such that the matrix polynomial

$$m_{\mathbf{L}^\circ}(\mathbf{L}^\circ) = (\mathbf{L}^\circ)^{N-N_C} + \sum_{k=0}^{N-N_C-1} c_k(\mathbf{L}^\circ)^k = 0. \tag{75}$$

Proposition 4. Let N_Z be the number of eigenvalues with zero imaginary part, then it holds that the degree of the minimal polynomial of $\mathbf{L}^\uparrow$ is $\deg(m_{\mathbf{L}^\uparrow}(\lambda)) = N - N_Z + 1$.

Corollary 3. There exists a set of coefficients $c_k \in \mathbb{C}$ such that the matrix polynomial

$$m_{\mathbf{L}^\uparrow}(\mathbf{L}^\uparrow) = (\mathbf{L}^\uparrow)^{N-N_Z+1} + \sum_{k=0}^{N-N_Z} c_k(\mathbf{L}^\uparrow)^k = 0. \tag{76}$$

C.7 Justification on having both sum and ratio Graph Filters

In Chan et al. (2026a), the sum and ratio graph kernels are designed to perform different types of filtering. It is important to first note that the image of h_{sum} and h_{rat} are not mutually inclusive, and span different subspaces. Additionally, authors of Chan et al. (2026a) have shown that h_{sum} are better suited for diffusive low-pass filtering, while h_{rat} are better suited for advective low-pass filtering and phase manipulation Chan et al. (2025; 2026b). Finally they can be composed disregarding the order of composition, as the applications of both kernels to $\mathbf{L}$ commute.

C.8 TwinS-GCN Combined architecture: TwinS (C)

Following our sum and ratio graph filters proposed in equation 8, one can simply combine both graph filters as they span different subspaces. Denote $\mathbf{X}^{(l)}$ the node features of dimension $N \times C$

(nodes times channels) at the l-th layer. The $(l+1)$ layer update reads as:

$$\mathbf{X}^{(l+1)} = \psi\Big(\nu \sum_{k=0}^{K_1} T_k(\widetilde{\mathbf{L}}^{\circ})\mathbf{X}^{(l)}\mathbf{\Theta}_k^{(l,\circ)} + (1-\nu)\sum_{k=0}^{K_2} j^k\, T_k(\widetilde{\mathbf{L}}^{\uparrow})\mathbf{X}^{(l)}\mathbf{\Theta}_k^{(l,\uparrow)} + \rho \sum_{k=0}^{K} j^k\, T_k(\widetilde{\mathbf{Z}})\mathbf{X}^{(l)}\mathbf{\Theta}_k^{(l)} + \mathbf{B}^{(l)}\Big), \tag{77}$$

where ψ is a nonlinear function, $\mathbf{\Theta}_k^{(l,\circ)}$, $\mathbf{\Theta}_k^{(l,\uparrow)}$ and $\mathbf{\Theta}_k^{(l)}$ are respectively learnable real weight matrices, $\mathbf{B}^{(l)}$ is a learnable bias term and learnable parameters $\nu, \rho \in [0,1]$.

C.9 Conceptual comparison with advection diffusion graph neural networks

As discussed in Section 4, for diagonalizable shifts, equation 5 coincides with the graph diffusion and advection operators of Chan et al. (2026a) (separate discussion in following paragraph). Hence, we compare conceptually the graph diffusion and advection operators obtained through our construct for diagonalizable operators with three related works. Berlureau et al. (2026) reweight the undirected Laplacian by a learned potential field, which remains symmetric and therefore anisotropically diffusive. ADR-GNN (Eliasof et al., 2024) learns the edges of its advection operator, following Chapman (2015), whose construction conserves mass rather than emulating transport and is neither skew-symmetric nor read off the input graph. AdvDIFFormer (Wu et al., 2023) models diffusion by global attention and advection by message passing along directed edges. In each case the directional term is an architectural addition to a dissipative model and carries no purely imaginary spectrum.

While similar to the work from Chan et al. (2026a), crucial differences include that our framework applies to non-diagonalizable operators, provide foundations in non-holomorphic calculus, and address problems of oversmoothing and receptive field reach. It is also in practice more stable to use thanks to our contribution in its Chebyshev reparameterization. Finally Chan et al. (2026a) belongs to directed GSP literature and does not tackle topics of neural networks.

C.10 On polynomial interpolation of the spectral conjugate

On a finite and fixed graph, any spectral window is exactly realizable by a polynomial in the shift operator. If $\mathbf{L}$ has distinct eigenvalues $\lambda_1, \ldots, \lambda_K$ and minimal polynomial of degree m, then $h(\mathbf{L}) = p_{\mathbf{L}}(\mathbf{L})$ for the unique interpolant of degree $\leq m-1$ matching the data $\partial^m h(\lambda_j)$ (Nevanlinna, 2018, Prop. 5.1, 5.3). Non-holomorphicity of h supposedly could be overcome. We show by explicit computation that the resulting coefficients are nonetheless unusable, whereas for holomorphic they remain stable.

In both experiments below we fit a quadratic $p(z) = a_0 + a_1 z + a_2 z^2$ through three eigenvalues by solving the Vandermonde system, and compare two target kernels: the holomorphic $h(z) = z^2 + 1$ and the conjugation map $\tau(z) = \bar{z}$, for which $\bar{\partial}\tau = 1 \neq 0$. We write $z_0 = 1 + i$ and let $\varepsilon > 0$.

Experiment 1: two close eigenvalues. Take the eigenvalues $\lambda_1 = 0$, $\lambda_2 = z_0$ and $\lambda_3 = z_0 + \varepsilon d$ with $|d| = 1$, so that λ_3 lies at distance ε from λ_2 in the direction d. As $\varepsilon \to 0$ the node set converges to $\{0, z_0, z_0\}$ for every choice of d.

For the holomorphic target kernel the interpolant is exact and independent of both ε and d,

$$p(z) = z^2 + 1, \qquad (a_0, a_1, a_2) = (1, 0, 1). \tag{78}$$

For τ one obtains $a_0 = 0$ and

$$d = 1: \quad a_1 = \frac{i(-\varepsilon - 3 - i)}{\varepsilon + 1 + i}, \quad a_2 = \frac{1+i}{\varepsilon + 1 + i}; \tag{79}$$

$$d = i: \quad a_1 = \frac{i(-\varepsilon - 3 + i)}{\varepsilon + 1 - i}, \quad a_2 = \frac{1+i}{\varepsilon + 1 - i}. \tag{80}$$

These coefficients are bounded for small ε, but the denominators differ depending on the direction d

$$\lim_{\varepsilon \to 0}(a_1, a_2) = \begin{cases} (-1-2i,\ 1), & d = 1, \\ (\ 1-2i,\ i), & d = i. \end{cases} \tag{81}$$

From this example, we see that small numerical inaccuracies from eigensolvers may lead to two completely different filters, caused by the direction of approach for which, smaller distance is ineffective.

Experiment 2: three close eigenvalues. Take the eigenvalues $\lambda_1 = z_0$, $\lambda_2 = z_0+\varepsilon$, $\lambda_3 = z_0+i\varepsilon$, so all three nodes lie within ε of z_0. Again the holomorphic target gives $(a_0, a_1, a_2) = (1, 0, 1)$ for every ε, while for τ

$$a_0(\varepsilon) = \frac{2(-1+i)}{\varepsilon}, \qquad a_1(\varepsilon) = -\frac{i(\varepsilon+4)}{\varepsilon}, \qquad a_2(\varepsilon) = \frac{1+i}{\varepsilon}, \tag{82}$$

so that $\max_k |a_k| = (\varepsilon+4)/\varepsilon \sim 4\varepsilon^{-1}$.

	$\varepsilon = 10^{-1}$	10^{-2}	10^{-3}	10^{-4}
$\max_k \lvert a_k\rvert$, target z^2+1	1.0	1.0	1.0	1.0
$\max_k \lvert a_k\rvert$, target $\bar{z}$	4.1×10^1	4.0×10^2	4.0×10^3	4.0×10^4

Table 3: Coefficient magnitude under a cluster of three.

Beyond the non-uniquely determined interpolation from Experiment.1, the interpolating coefficients may also suffer from arbitrarily large values, making it numerically unstable. While we show ill-defined interpolation and numerical instability for the case of τ (spectral conjugate), it surmised to be extended to general non-holomorphic kernels. Either of these examples are realistic especially for non-diagonalizable operators where there necessarily exist an eigenvalue with algebraic multiplicity greater than 1.

C.11 Combine with anti-symmetric weight

A-DGN (Gravina et al., 2023) models each node feature $\mathbf{x}_n(t) \in \mathbb{R}^C$ as the state of an ODE whose channel-mixing matrix is constrained to be anti-symmetric (their equation 4):

$$\partial_t \mathbf{x}_n(t) = \sigma\big((\mathbf{V} - \mathbf{V}^\top)\,\mathbf{x}_n(t) + \Phi(\mathbf{X}(t), \mathcal{N}_n) + \mathbf{b}\big), \tag{83}$$

where $\mathbf{V} \in \mathbb{R}^{C\times C}$ is learnable and Φ aggregates over the neighborhood $\mathcal{N}_n$. Anti-symmetry places the eigenvalues of $\mathbf{V} - \mathbf{V}^\top$ on the imaginary axis, which is the source of A-DGN's non-dissipative behavior. A forward-Euler discretization with step ϵ and a small damping $\gamma > 0$ for numerical stability (their equation 5) gives, in matrix form,

$$\mathbf{X}^{(l+1)} = \mathbf{X}^{(l)} + \epsilon\,\psi\big(\mathbf{X}^{(l)}(\mathbf{V} - \mathbf{V}^\top - \gamma\mathbf{I}) + \Phi(\mathbf{X}^{(l)}) + \mathbf{B}\big). \tag{84}$$

The two constructions act on different axes: A-DGN constrains how channels are mixed, whereas TwinS (R) constrains how information moves across nodes. They therefore combine by using the ratio filter of equation 15 as the aggregation Φ:

$$\mathbf{X}^{(l+1)} = \mathbf{X}^{(l)} + \epsilon\,\psi\Big(\mathbf{X}^{(l)}(\mathbf{V} - \mathbf{V}^\top - \gamma\mathbf{I}) + \sum_{k=0}^{K} j^k\, T_k(\widetilde{\mathbf{Z}})\,\mathbf{X}^{(l)}\mathbf{\Theta}_k^{(l)} + \mathbf{B}^{(l)}\Big), \tag{85}$$

where $\mathbf{V}$ is the anti-symmetric channel-mixing weight and $\mathbf{\Theta}_k^{(l)}$ are the filter weights of TwinS (R). TwinS (S) and TwinS (C) combine in the same way, by substituting equation 14 or equation 77 for Φ. Whether the non-dissipativity guarantees of both components hold jointly is not immediate. We leave this analysis, and the empirical evaluation of the combination, to future work.

D EXPERIMENTAL DETAILS

D.1 GRAPH TRANSFER TASK

General settings. The task requires the model to learn a mapping from a source node to a target node on a graph of size N. Denote $\mathbf{x}[n] \in \mathbb{R}^C$ to be the feature vector at node n. Following Bodnar et al. (2021) the node features are set as follows:

$$\mathbf{x}[n] = \begin{cases} \mathbf{e}_k, & \text{if } n \text{ is source node,} \\ \mathbf{0}, & \text{if } n \text{ is target node,} \\ \mathbf{1} + \boldsymbol{\epsilon}, & \text{otherwise.} \end{cases} \tag{86}$$

where $\mathbf{e}_k$ is set to be a one-hot vector such that $\boldsymbol{e}_k[n] = \delta_{k=n}$ with integer $k \in \{1, \dots, C\}$ being randomly selected and $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, 0.1\mathbf{I}_N)$. The source node and target node being selected such that the distance between them is $d(n_{\text{source}}, n_{\text{target}}) = r$. The ground truth for the target node's feature is set to be the same as the source node's. The model is evaluated through mean squared error (MSE) between the predicted target node feature and the ground truth target node feature. The predicted class is the index with highest value (since all others should be receiving less energy) and it is compared with the ground truth class k, leading to an accuracy for each sample. The graphs used for the task are shown in Figure 5. Without loss of generality, in our experiments we select node feature of dimension $C = 5$, consider 200 samples, split into (60%) training, (20%) validation and (20%) testing. The source node is randomly selected, while the target node is selected such that the distance between them is $r = 10$ which is the graph diameter.

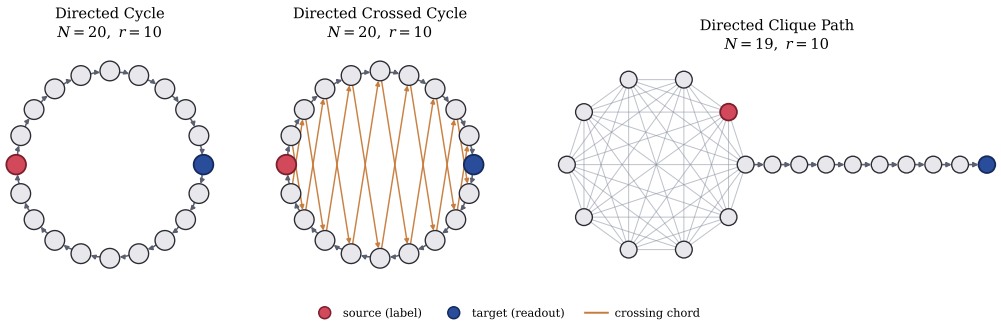


Figure 5: Graphs used for the task (same as in Gravina et al. (2025)).

Details on Section 5.1. For depth experiment: source and target are held a fixed distance of 10 apart, and the sweep varies only network depth. For every topology, model and depth $\mathbf{K} \in \{1, \dots, 10\}$ we run an independent hyperparameter search (Appendix D.1), so no depth is handicapped by settings tuned at another. Each configuration is trained on 200 sampled graphs split $60\%/20\%/20\%$ into 120 training, 40 validation and 40 test samples under a fixed data seed, with the one-hot label placed on the source node and all other nodes carrying an uninformative constant perturbed by Gaussian noise ($\sigma = 0.1$). Training minimizes the mean squared error between the target node's output and the one-hot label, plus a suppression term penalizing the squared output of every non-target node, which prevents the network from spreading the label rather than transporting it. We report accuracy: a prediction counts as correct when its largest coordinate matches the label's. Runs are capped at 300 epochs and evaluated every 8 epochs, stopping early after 30 consecutive evaluations (i.e. 240 epochs) without an improvement in validation MSE. Each configuration is scored at its lowest-validation-MSE epoch and the test accuracy is read at that same epoch, so the test split never informs selection. Every reported figure is the mean and standard deviation over 10 random initializations.

Hyperparameter search. The search uses Optuna's Tree-structured Parzen Estimator (Akiba et al., 2019), run independently for each (topology, model, depth) triple at 48 trials, and selects on validation MSE. Depth is swept exhaustively over $\{1, \dots, 10\}$ rather than sampled, so each model is tuned 480 times per topology. The searched ranges are: hidden dimension $\in \{4, 8, 16, 32\}$,

non-linearity $\in \{\text{ReLU}, \text{None}\}$, normalization $\in \{\text{True}, \text{False}\}$, bias $\in \{\text{True}, \text{False}\}$, and learning rate $\in \{0.001, 0.005\}$. Dropout and weight decay are held at 0 throughout: on a task this small, regularization only slows transport without improving generalization. Three further axes are searched only by the architectures that read them — whether the branch-mixing weights α and β are learned or fixed at 0.5 (employed in Dir-GNN, FaberNet, TwinS-GCN), and MagNet's charge $q \in \{0, 0.05, 0.10, 0.15, 0.20, 0.25\}$. While introduced as the standard combinatorial directed Laplacian, options for operator include $\{\text{left}, \text{combinatorial}, \text{right}, \text{symmetric}\}$ normalized directed Laplacian. Cluster tolerance (Appendix C.1) was set to $1e^{-9}$, as we expect no distinct eigenvalue clusters. A dash in the table (Table 4) means that the hyperparameter is not relevant to the current architecture. Accuracies are the mean and standard deviation over random initializations. For the same model, when several depths achieve perfect accuracy, the table reports the shallowest, since on a task that many models saturate the meaningful winner is the cheapest one. Corresponding run-times for each of these models are shown in Table 7. For Dir-GNN, FaberNet, MagNet TWINS-GCN, we consider only the degree of each layer to be 1 (to fully isolate the effect of depth), meaning that we set $K = 1$ for Dir-GNN, FaberNet, MagNet, while for TWINS-GCN, which start at index $K = 1$ (programming convention) for the identity matrix, we set $K = 2$ to represent degree 1 polynomial.

Table 4: Best hyperparameter configuration per topology for the **graph transfer task**, selected over depth and all searched axes by validation MSE. A dash marks an axis the architecture does not take. α/β report whether the branch-mixing weights are learned; q is MagNet's charge.

Model	depth	#hidden	non-lin.	norm.	bias	lr	α	β	q	Accuracy
Directed Cycle										
GCN	1	8	ReLU	No	No	0.005	–	–	–	0.325 ± 0.054
MagNet	6	4	–	–	Yes	0.005	–	–	0.05	0.267 ± 0.031
FaberNet	10	32	–	–	Yes	0.005	Yes	–	–	1.000 ± 0.000
Dir-GNN	10	16	None	Yes	No	0.005	Yes	–	–	1.000 ± 0.000
ADR-GNN	2	4	–	–	–	0.005	–	–	–	0.283 ± 0.024
AdvDIFFormer	1	16	–	–	–	0.005	–	–	–	1.000 ± 0.000
TWINS (S)	2	4	None	Yes	No	0.001	Yes	–	–	0.258 ± 0.042
TWINS (R)	3	8	None	Yes	No	0.001	–	–	–	1.000 ± 0.000
TWINS (C)	3	16	None	Yes	Yes	0.005	No	Yes	–	1.000 ± 0.000
Directed Crossed Cycle										
GCN	10	32	ReLU	No	Yes	0.005	–	–	–	0.467 ± 0.066
MagNet	3	8	–	–	No	0.001	–	–	0.1	0.233 ± 0.012
FaberNet	10	16	–	–	Yes	0.005	Yes	–	–	1.000 ± 0.000
Dir-GNN	10	32	ReLU	No	No	0.005	Yes	–	–	1.000 ± 0.000
ADR-GNN	6	8	–	–	–	0.005	–	–	–	0.300 ± 0.020
AdvDIFFormer	1	32	–	–	–	0.005	–	–	–	1.000 ± 0.000
TWINS (S)	2	32	ReLU	Yes	Yes	0.005	Yes	–	–	1.000 ± 0.000
TWINS (R)	2	16	None	Yes	No	0.001	–	–	–	1.000 ± 0.000
TWINS (C)	2	32	None	Yes	Yes	0.001	No	No	–	1.000 ± 0.000
Directed Clique Path										
GCN	3	16	None	Yes	Yes	0.001	–	–	–	0.275 ± 0.020
MagNet	3	4	–	–	Yes	0.005	–	–	0.0	0.250 ± 0.020
FaberNet	10	16	–	–	No	0.001	Yes	–	–	0.258 ± 0.031
Dir-GNN	10	16	None	Yes	No	0.005	Yes	–	–	0.767 ± 0.112
ADR-GNN	10	8	–	–	–	0.005	–	–	–	0.275 ± 0.020
AdvDIFFormer	1	32	–	–	–	0.005	–	–	–	1.000 ± 0.000
TWINS (S)	2	32	None	Yes	No	0.005	Yes	–	–	1.000 ± 0.000
TWINS (R)	2	16	ReLU	Yes	Yes	0.005	–	–	–	1.000 ± 0.000
TWINS (C)	2	16	None	Yes	Yes	0.001	Yes	Yes	–	1.000 ± 0.000

D.2 DATASETS BENCHMARKING

General settings. The benchmarking datasets used in the experiments are summarized in Table 5. We evaluate on transductive node classification over directed graphs. Cornell, Texas and Wisconsin are small web-page networks that are strongly heterophilic (edge homophily 0.12, 0.06 and 0.17 respectively), so neighboring nodes usually carry different labels and edge direction is informative.

We use the provided ten train/validation/test splits distributed with the datasets. Every configuration is trained on the training nodes for at most 300 epochs, evaluated every 5 epochs, with early stopping after 20 evaluations without validation improvement. Following standard practice, we score each run at the epoch of highest validation accuracy and report the test accuracy at that same epoch, so the test nodes never influence model selection. Each reported figure is the mean and standard deviation of classification accuracy over 30 runs — the ten splits crossed with three random initializations.

Table 5: Statistics of the node classification benchmarks.

Dataset	**#Nodes**	**#Edges**	**#Feat.**	**#Cls.**
Cornell	183	295	1,703	5
Texas	183	309	1,703	5
Wisconsin	251	499	1,703	5
Chameleon†	2,277	36,101	2,325	5
Squirrel†	5,201	217,073	2,089	5

† Original versions. Filtered variants (Platonov et al., 2023): Chameleon (890 / 13,584), Squirrel (2,223 / 46,998).

Among the results provided in Table 1 here are the ones retrieved from literature:

- Results of GCN, GAT on Wisconsin, Cornell, Texas are taken from Eliasof et al. (2024) and on Chameleon-F, Squirrel-F, comes from Platonov et al. (2023).
- Results of MLP on Chameleon-F, Squirrel-F, comes from Platonov et al. (2023) (ResNet).

All remaining entries are computed from re-implementation following original repository. Disclaimer: the results reproduced for MLP, MagNet, ADR-GNN on Wisconsin, Cornell, Texas may differ from accuracies found in their respective original paper Zhang et al. (2021), Eliasof et al. (2024). We have noticed in their open repository that the employed Wisconsin, Cornell, Texas datasets were from the pre-2022 geom-gcn (Pei et al., 2020) release while in our experiments we employ the latest release (Pei et al., 2022), hence we reproduce the results on these datasets.

Hyperparameter search. Selected hyperparameters for the benchmarking datasets are reported in Table 6. The search uses Optuna's TPE sampler (multivariate), and configurations are selected on mean validation accuracy. The shared search space is: depth $\in \{2, 3, 4, 5\}$, hidden dimension $\in \{32, 64, 128, 256\}$, dropout $\in \{0, 0.2, 0.4, 0.5, 0.6\}$, normalization and bias $\in \{\text{True}, \text{False}\}$, learning rate $\in \{0.001, 0.005, 0.01, 0.05\}$, weight decay $\in \{0, 5.10^{-4}, 5.10^{-3}\}$, input feature scaling $\in \{\text{raw}, \text{row}, \text{std}\}$, jumping knowledge $\in \{\text{None}, \max, \text{cat}\}$ and, for spectral models, order $K \in \{1, 2, 3, 4\}$. The non-linearity is fixed to ReLU. Each model also searches its own parameters: MagNet's charge $q \in \{0, 0.05, \ldots, 0.25\}$; the Dir-GNN and FaberNet mixing weight $\alpha \in \{0, 0.5, 1\}$ and whether it is learned; FaberNet's zero-order term; the ADR-GNN step size $\in \{0.02, 0.05, 0.1, 0.2\}$; and the AdvDIFFormer heads $\in \{1, 2, 4\}$ and order $K_{\text{ord}} \in \{1, 2, 3\}$. For TwinS-GCN, the search covers the branch-mixing weights α, β, the Laplacian normalization $\in \{\text{left}, \text{combinatorial}, \text{right}, \text{symmetric}\}$, and the eigenvalue-clustering tolerance $\in \{\text{default}, 10^{-6}, 10^{-3}\}$. Operators are rescaled by their maximum absolute entry. Next we discuss few differences in protocol: on WebKB, TwinS-GCN uses 72 trials, against 96 for MLP, MagNet, Dir-GNN and FaberNet and 72 for ADR-GNN and AdvDIFFormer to find hyperparameters. On Chameleon-F and Squirrel-F, TwinS-GCN uses 32 trials, against 248 and 48 respectively. TwinS-GCN thus has the smallest budget on every dataset. Its validation score, however, averages over all 10 splits, whereas the baselines average over 3 (5 for the attention-based baselines on Chameleon-F and Squirrel-F). Finally, for Texas and Squirrel-F the TwinS-GCN search was widened to depth 1, $K \in \{5, \ldots, 8\}$, weight decay 10^{-2} and patience 50, which accounts for the settings in Table 6 that are not within the shared search space.

Table 6: Hyperparameter search results for **benchmarking datasets**. A dash marks an axis the architecture does not take. α/β report whether the branch-mixing weights are learned; q is MagNet's charge. Cluster tol are $1e^{-9}$ unless specified.

Model	depth	#hidden	dropout	norm.	bias	lr	w.d.	feat.	JK	K	learn α	learn β	model-specific	Accuracy
Chameleon-F														
MagNet	3	128	0.2	–	Yes	0.005	0.005	raw	–	1	–	–	q=0	0.424 ± 0.035
Dir-GNN	5	128	0.5	No	Yes	0.005	0.0005	row	cat	–	No	–	α=0.5	0.441 ± 0.029
FaberNet	4	128	0.6	–	Yes	0.01	0.0005	row	max	1	No	–	α=0.5, zero-ord.: No	0.430 ± 0.033
AdvDIFFormer	4	128	0.4	–	–	0.05	0	std	–	–	–	–	heads 2, K_{ord}=1	0.415 ± 0.036
ADR-GNN	4	64	0.6	–	–	0.01	0.005	std	–	–	–	–	step 0.05	0.461 ± 0.041
TWINS (S)	2	256	0.6	Yes	No	0.001	0	std	cat	2	Yes	–	right	0.423 ± 0.028
TWINS (R)	2	256	0.5	No	Yes	0.001	0.005	std	cat	4	–	–	none, tol 10^{-6}	0.442 ± 0.032
TWINS (C)	2	128	0.4	Yes	No	0.01	0	std	none	4	No	Yes	none, tol 10^{-3}	0.416 ± 0.036
Squirrel-F														
MagNet	5	128	0.2	–	Yes	0.05	0	raw	–	2	–	–	q=0.05	0.442 ± 0.014
Dir-GNN	4	64	0.5	No	Yes	0.005	0.0005	row	cat	–	No	–	α=0.5	0.443 ± 0.022
FaberNet	3	64	0.6	–	Yes	0.001	0.005	raw	None	1	No	–	α=0.5, zero-ord.: No	0.451 ± 0.021
AdvDIFFormer	2	64	0.6	–	–	0.005	0	std	–	–	–	–	heads 1, K_{ord}=1	0.414 ± 0.016
ADR-GNN	5	64	0.4	–	–	0.05	0.005	std	–	–	–	–	step 0.2	0.441 ± 0.020
TWINS (S)	3	64	0.6	Yes	No	0.001	0.0005	std	cat	4	Yes	–	left, tol 10^{-3}	0.396 ± 0.015
TWINS (R)	2	32	0.6	Yes	No	0.005	0.005	std	None	2	–	–	left, tol 10^{-3}	0.402 ± 0.018
TWINS (C)	2	128	0.6	Yes	Yes	0.005	0	std	none	5	Yes	Yes	left	0.401 ± 0.016
Texas														
MLP	4	32	0.4	–	Yes	0.01	0.005	raw	–	–	–	–	–	0.822 ± 0.043
MagNet	2	256	0.2	–	Yes	0.01	0.005	raw	–	1	–	–	q=0.1	0.853 ± 0.046
Dir-GNN	2	64	0.6	Yes	No	0.05	0.005	raw	cat	–	Yes	–	α=0	0.705 ± 0.065
FaberNet	2	64	0.5	–	Yes	0.005	0.005	raw	cat	2	No	–	α=1, zero-ord.: Yes	0.832 ± 0.053
AdvDIFFormer	3	32	0.2	–	–	0.05	0.0005	raw	–	–	–	–	heads 4, K_{ord}=1	0.818 ± 0.064
ADR-GNN	2	64	0.2	–	–	0.05	0.005	raw	–	–	–	–	step 0.02	0.822 ± 0.052
TWINS (S)	2	64	0.5	Yes	Yes	0.01	0.0005	raw	max	3	No	–	left	0.846 ± 0.034
TWINS (R)	1	128	0.5	Yes	No	0.01	0.0005	row	cat	8	–	–	comb.	0.841 ± 0.039
TWINS (C)	2	256	0.2	Yes	Yes	0.01	0.01	raw	cat	8	Yes	No	left, tol 10^{-3}	0.854 ± 0.047
Wisconsin														
MLP	4	64	0.4	–	Yes	0.01	0.005	raw	–	–	–	–	–	0.852 ± 0.050
MagNet	2	32	0.4	–	Yes	0.01	0.005	raw	–	1	–	–	q=0.15	0.834 ± 0.040
Dir-GNN	5	64	0.4	Yes	No	0.05	0.0005	raw	max	–	No	–	α=0.5	0.674 ± 0.060
FaberNet	2	256	0.5	–	Yes	0.001	0.005	raw	cat	2	No	–	α=1, zero-ord.: Yes	0.795 ± 0.052
AdvDIFFormer	2	256	0.4	–	–	0.01	0.0005	raw	–	–	–	–	heads 1, K_{ord}=2	0.765 ± 0.048
ADR-GNN	3	64	0.2	–	–	0.05	0.005	raw	–	–	–	–	step 0.02	0.845 ± 0.045
TWINS (S)	2	64	0.2	Yes	No	0.005	0.0005	row	cat	2	Yes	–	right	0.871 ± 0.041
TWINS (R)	2	256	0.2	Yes	Yes	0.005	0.005	raw	cat	3	–	–	right	0.874 ± 0.040
TWINS (C)	2	64	0	Yes	No	0.01	0.005	raw	max	2	Yes	No	right	0.868 ± 0.038
Cornell														
MLP	3	64	0.2	–	Yes	0.01	0.005	raw	–	–	–	–	–	0.759 ± 0.039
MagNet	2	32	0.2	–	Yes	0.01	0.005	raw	–	1	–	–	q=0.25	0.733 ± 0.042
Dir-GNN	2	128	0.2	Yes	No	0.05	0.005	std	None	–	No	–	α=0.5	0.598 ± 0.069
FaberNet	3	32	0.2	–	No	0.01	0.005	raw	cat	4	No	–	α=0, zero-ord.: Yes	0.726 ± 0.047
AdvDIFFormer	2	128	0.4	–	–	0.01	0.0005	raw	–	–	–	–	heads 2, K_{ord}=1	0.722 ± 0.055
ADR-GNN	2	128	0	–	–	0.005	0.005	raw	–	–	–	–	step 0.02	0.714 ± 0.044
TWINS (S)	2	64	0.5	Yes	Yes	0.01	0.0005	raw	max	3	No	–	sym.	0.751 ± 0.032
TWINS (R)	2	64	0.5	Yes	Yes	0.01	0.0005	raw	max	3	–	–	right	0.768 ± 0.030
TWINS (C)	2	64	0.5	Yes	Yes	0.01	0.0005	raw	max	3	No	No	right	0.757 ± 0.040

E MODELS RUNTIME

Graph transfer task. We report the runtime of the models used in the graph transfer task. The runtime are provided for all different graphs. The runtime are measured on Apple M1 Max. Table 7 shows the minimal needed runtime/parameters of each model to solve the graph transfer task. Table 8 shows for comparable configuration, the models runtime/parameters.

Table 7: Runtime and parameter count per topology for the **graph transfer task** as specified in Appendix D.1, for the **best configuration** selected in Table 4. Train time is per-epoch, averaged over 100 epochs (train repeats is 5) and inference time (infer repeats is 50) is per forward pass over the full graph (40 graphs/batch). Build time is provided specifically to TWINS-GCN models which require a one-time precomputing, per graph, of the operators. Accuracy is reported for reference.

Model	#Params	Train (ms/epoch)	Inference (ms)	Build (ms)	Accuracy
Directed Cycle					
GCN	25	0.67 ± 0.34	0.09 ± 0.00	–	0.325 ± 0.054
MagNet	269	9.12 ± 0.28	1.80 ± 0.01	–	0.267 ± 0.031
FaberNet	39,110	26.94 ± 0.78	4.44 ± 0.00	–	1.000 ± 0.000
Dir-GNN	4,417	10.58 ± 0.06	1.96 ± 0.00	–	1.000 ± 0.000
ADR-GNN	345	11.67 ± 0.52	1.61 ± 0.01	–	0.283 ± 0.024
AdvDIFFormer	1,317	3.66 ± 0.23	0.48 ± 0.14	–	1.000 ± 0.000
TWINS (S)	161	2.88 ± 0.43	0.43 ± 0.04	0.0	0.258 ± 0.042
TWINS (R)	288	2.60 ± 0.14	0.34 ± 0.00	0.0	1.000 ± 0.000
TWINS (C)	2,719	6.40 ± 0.71	1.17 ± 0.07	0.0	1.000 ± 0.000
Directed Crossed Cycle					
GCN	8,805	14.49 ± 0.54	2.89 ± 0.16	–	0.467 ± 0.066
MagNet	421	4.92 ± 0.09	0.98 ± 0.00	–	0.233 ± 0.012
FaberNet	10,342	23.55 ± 0.17	4.39 ± 0.14	–	1.000 ± 0.000
Dir-GNN	17,025	10.47 ± 0.26	1.80 ± 0.14	–	1.000 ± 0.000
ADR-GNN	3,213	53.38 ± 0.69	9.21 ± 0.20	–	0.300 ± 0.020
AdvDIFFormer	4,677	4.45 ± 0.08	0.63 ± 0.00	–	1.000 ± 0.000
TWINS (S)	1,429	2.98 ± 0.01	0.52 ± 0.06	0.5	1.000 ± 0.000
TWINS (R)	320	1.73 ± 0.04	0.23 ± 0.00	0.9	1.000 ± 0.000
TWINS (C)	2,142	3.88 ± 0.05	0.79 ± 0.06	0.9	1.000 ± 0.000
Directed Clique Path					
GCN	453	3.79 ± 0.09	0.69 ± 0.00	–	0.275 ± 0.020
MagNet	161	4.84 ± 0.18	0.96 ± 0.00	–	0.250 ± 0.020
FaberNet	9,702	21.28 ± 0.15	4.01 ± 0.00	–	0.258 ± 0.031
Dir-GNN	4,417	10.55 ± 0.08	1.95 ± 0.00	–	0.767 ± 0.112
ADR-GNN	5,293	113.09 ± 0.73	21.52 ± 0.85	–	0.275 ± 0.020
AdvDIFFormer	4,677	4.92 ± 0.11	0.65 ± 0.01	–	1.000 ± 0.000
TWINS (S)	1,281	2.81 ± 0.04	0.52 ± 0.06	0.2	1.000 ± 0.000
TWINS (R)	362	1.71 ± 0.03	0.27 ± 0.03	0.4	1.000 ± 0.000
TWINS (C)	1,088	4.30 ± 0.04	0.78 ± 0.06	0.4	1.000 ± 0.000

Table 8: Runtime and parameter count per topology for the **graph transfer task** as specified in Appendix D.1. The count is under an **equivalent configuration** (hidden=32, depth=4, polynomial degree (K)=1, dropout=0, bias=True, non-linearity=ReLU). Remaining information is as in Table 7.

Model	#Params	Train (ms/epoch)	Inference (ms)
Directed Cycle			
GCN	2,469	8.93 ± 1.80	1.02 ± 0.24
MagNet	6,917	17.34 ± 1.30	3.01 ± 0.68
FaberNet	27,205	38.95 ± 0.34	6.77 ± 0.60
Dir-GNN	4,938	9.33 ± 0.21	1.40 ± 0.12
ADR-GNN	30,181	134.74 ± 3.63	22.85 ± 4.63
AdvDIFFormer	17,445	34.32 ± 0.39	5.04 ± 1.11
TWINS (S)	9,876	11.96 ± 0.25	2.05 ± 0.72
TWINS (R)	4,938	10.03 ± 4.00	1.61 ± 0.97
TWINS (C)	14,814	22.52 ± 2.21	3.22 ± 1.56
Directed Crossed Cycle			
GCN	2,469	12.64 ± 0.43	2.20 ± 0.89
MagNet	6,917	19.03 ± 1.85	4.64 ± 1.57
FaberNet	27,205	49.51 ± 3.24	7.91 ± 1.93
Dir-GNN	4,938	10.00 ± 0.33	1.79 ± 0.81
ADR-GNN	30,181	193.44 ± 16.22	35.31 ± 7.38
AdvDIFFormer	17,445	41.97 ± 2.84	5.76 ± 2.45
TWINS (S)	9,876	13.75 ± 0.83	2.21 ± 0.37
TWINS (R)	4,938	8.47 ± 0.47	1.11 ± 0.34
TWINS (C)	14,814	20.97 ± 1.74	3.28 ± 0.91
Directed Clique Path			
GCN	2,469	17.46 ± 0.99	2.81 ± 1.18
MagNet	6,917	20.83 ± 3.74	4.27 ± 1.24
FaberNet	27,205	47.63 ± 6.61	7.50 ± 1.42
Dir-GNN	4,938	9.92 ± 0.80	1.56 ± 0.41
ADR-GNN	30,181	303.24 ± 15.23	53.27 ± 23.50
AdvDIFFormer	17,445	33.62 ± 1.16	4.62 ± 1.47
TWINS (S)	9,876	14.47 ± 0.36	2.58 ± 0.95
TWINS (R)	4,938	9.20 ± 0.43	1.71 ± 0.34
TWINS (C)	14,814	18.87 ± 0.24	3.49 ± 0.94

Benchmarking datasets. We report the runtime of the models used in the benchmarking of datasets. The runtime are measured on Apple M1 Max. Table 9 shows the runtime/parameters used by each model to solve the benchmarking task. Table 11 shows for comparable configuration, the models runtime/parameters.

Table 9: Runtime and parameter count, illustrated with **Wisconsin** dataset **benchmarking of datasets**, for the **best configuration** selected in Table 6. Train time is per-epoch, averaged over 100 epochs (train repeats is 5) and inference time (infer repeats is 50) is per forward pass over the full graph. Accuracy is reported to distinguish relevance across models.

Model	#Params	Train (ms/epoch)	Inference (ms)	Accuracy
MLP	117,701	1.44 ± 0.26	0.19 ± 0.05	0.852 ± 0.050
MagNet	111,429	6.78 ± 0.25	3.76 ± 0.74	0.834 ± 0.040
Dir-GNN	251,072	4.15 ± 0.12	1.49 ± 0.20	0.674 ± 0.060
FaberNet	6,029,317	55.83 ± 2.13	23.56 ± 6.79	0.795 ± 0.052
AdvDIFFormer	1,095,941	13.45 ± 0.50	2.67 ± 0.36	0.765 ± 0.048
ADR-GNN	197,125	18.80 ± 0.58	8.50 ± 0.29	0.845 ± 0.045
TWINS (S)	452,993	4.19 ± 0.81	1.17 ± 0.37	0.871 ± 0.041
TWINS (R)	1,508,613	16.27 ± 0.16	8.10 ± 1.72	0.874 ± 0.040
TWINS (C)	678,849	5.23 ± 0.11	1.80 ± 0.36	0.868 ± 0.038

Table 10: Build time (ms) per TWINS-GCN model, per graph.

Model	Cornell	Texas	Wisconsin	Chameleon-F	Squirrel-F
TWINS (S)	20	10	50.0	1,230	22,540
TWINS (R)	50	40	90.0	3,810	58,540
TWINS (C)	50	40	90.0	3,810	58,540

Table 11: Runtime and parameter count, illustrated with **Wisconsin** dataset **benchmarking of datasets**, for **equivalent configuration** (hidden=32, depth=4, dropout=0, K=2, bias=True). Train time is per-epoch, averaged over 50 epochs (train repeats is 3) and inference time (infer repeats is 50) is per forward pass over the full graph.

Model	#Params	Train (ms/epoch)	Inference (ms)
MLP	56,805	1.27 ± 0.44	0.17 ± 0.04
Dir-GNN	113,610	1.72 ± 0.09	0.39 ± 0.00
MagNet	173,157	10.67 ± 0.28	5.85 ± 0.30
FaberNet	692,677	14.61 ± 0.27	6.29 ± 0.12
ADR-GNN	84,517	15.03 ± 0.14	6.03 ± 0.67
AdvDIFFormer	79,973	6.93 ± 0.36	1.54 ± 0.00
TWINS (S)	227,220	2.97 ± 0.07	0.68 ± 0.00
TWINS (R)	113,610	1.58 ± 0.08	0.41 ± 0.03
TWINS (C)	340,830	4.16 ± 0.02	1.07 ± 0.03

F ADDITIONAL EXPERIMENT FOR RECEPTIVE FIELD REACH

F.1 GRAPH SIGNAL TRANSFER ON CYCLE GRAPH

Settings. To gain further insight into the results of Section 5.1, we conduct a long-range dependency experiment on the directed cycle graph $\mathcal{C}_N$ with $N = 200$ nodes. The goal of this task is to find the filter coefficients to transfer a localized signal from a source node to a target node at various distances. In particular, we study the target reconstruction error of the different filters (equation 8) underlying TWINS-GCN, as their behavior determines the receptive field of the model.

We start with a localized signal defined as a gaussian centered at an arbitrarily chosen source node, and we select a target node such that the number of hops (shortest path distance) between them is r. The starting signal and the ground truth target signal are thus respectively defined as

$$\mathbf{x}_{\text{source}}[n] = e^{-\frac{(n-n_{\text{source}})^2}{2\sigma^2}}, \qquad \mathbf{x}_{\text{target}}[n] = e^{-\frac{(n-n_{\text{target}})^2}{2\sigma^2}}, \qquad \text{with } d(n_{\text{source}}, n_{\text{target}}) = r, \tag{87}$$

where σ is the bump's width, and d shortest path distance from n_{source} to n_{target}.

First, we generate different pairs of source and target nodes, spanning all distances up to the graph radius $r = 100$ (half the graph), and construct signals with $\sigma = 2$. Then, for each fixed order $k \in \{10, 20, 50\}$, we estimate the coefficients of the sum and ratio filters h_{sum} and h_{rat} introduced in Section 2.4 and the polynomial filter h of the same order. The coefficients are optimized for the normalized MSE (NMSE) between the filtered source signal and the ground truth target signal, where

$$\text{NMSE}(\mathbf{x}_{\text{target}}, \hat{\mathbf{x}}_{\text{source},k}) = \frac{\left\|\mathbf{x}_{\text{target}} - \hat{\mathbf{x}}_{\text{source},k}\right\|_2^2}{\left\|\mathbf{x}_{\text{target}}\right\|_2^2} \tag{88}$$

where $\hat{\mathbf{x}}_{\text{source},k} = \hat{h}_{i,k}(\mathbf{L})\,\mathbf{x}_{\text{source}}$, and $\hat{h}_{i,k}$ is the estimated k-th order filter of class $h_i \in \{h, h_{\text{sum}}, h_{\text{rat}}\}$. The results are shown in Figure 6.

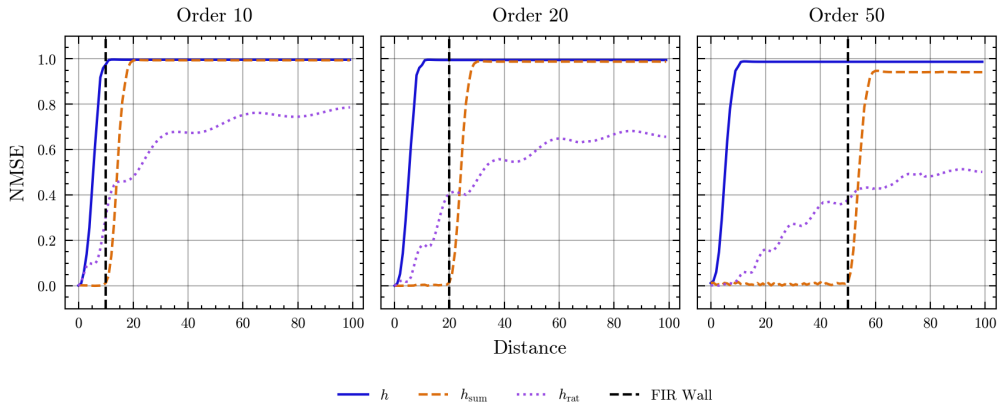


Figure 6: Long-range dependency experiment on cycle graph with increasing orders.

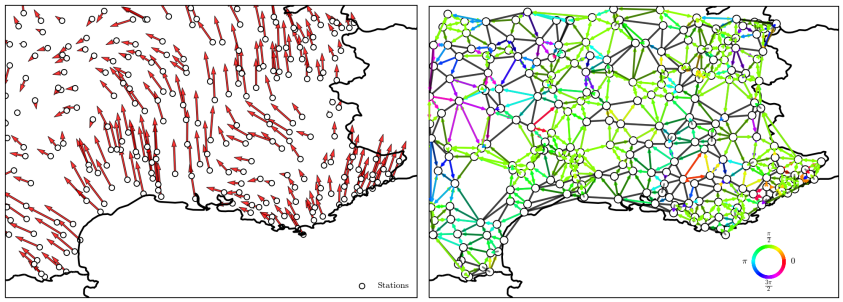
Results. We clearly observe the effect of the FIR wall in Figure 6, where the kernel h generally struggles to reach 0 NMSE (perfect transfer) even when its order (degree) is higher than the distance between the source and the target, which in the Figure is shown as percentage of the graph radius). The kernel h_{sum} also performs marginally better than h, consistent with the capacity of leveraging $\mathbf{L}^{\uparrow}$ stated in Section 3.3.1, preventing oversmoothing. Hence in contrast to h, h_{sum} keeps the initial signal localized, reducing NMSE. Finally, h_{rat} overcomes this wall as even when its order is largely lower than the distance, the signal transfer still happen (NMSE < 1). It is however performing slightly worse than polynomial kernels when its order is larger than the distance, as expected from Theorem 5.

F.2 GRAPH SIGNAL TRANSFER ON TEMPERATURE GRAPH

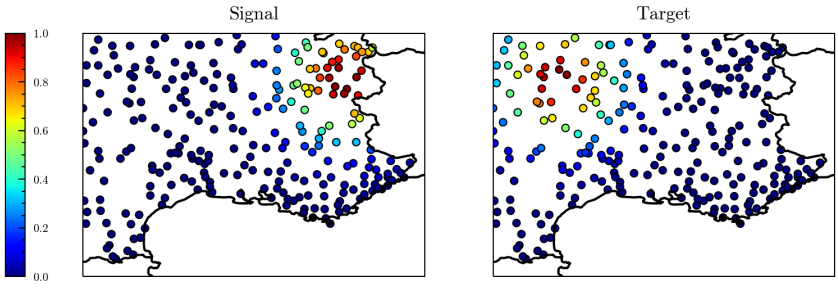
Settings. Similarly to the experiment in Section F.1, we conduct a long-range dependency experiment on a temperature graph with $N = 212$ nodes, built as described in Figure 7b (Chan et al., 2026a). This time around, we set the source node randomly on the east side of the graph (source regions as the wind goes from east to west) and a target node randomly on the west side. The starting signal and the ground truth target signal are thus respectively defined as

$$\mathbf{x}_{\text{source}}[n] = e^{-\frac{(\mathbf{c}[n]-\mathbf{c}[n_{\text{source}}])^2}{2\sigma^2}}, \quad \mathbf{x}_{\text{target}}[n] = e^{-\frac{(\mathbf{c}[n]-\mathbf{c}[n_{\text{target}}])^2}{2\sigma^2}}, \qquad \text{with } d(n_{\text{source}}, n_{\text{target}}) = r, \tag{89}$$

with $\mathbf{c}[n]$ being the coordinates of node n, σ the bump's width, and d the shortest path distance from n_{source} to n_{target}. We generate 10 different pairs of source and target nodes, leading to 10 pairs of signals with $\sigma = 1$. An example of source-target pair is shown in Figure 7c. The coefficients are optimized for the NMSE between the filtered source signal and the ground truth target signal (equation 88). The results are shown in Figure 8a and 8b. We observe in particular that h_{rat} breaks the FIR wall and is able to better reach the target node despite a similar order than other kernels. With increasing order, the performance of h_{rat} improves, but is surpassed by h_{sum}.

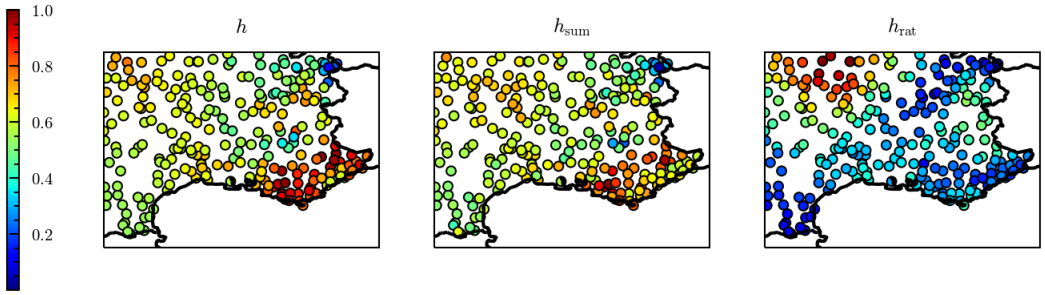


(a) Measured wind velocity field (b) Derived temperature graph

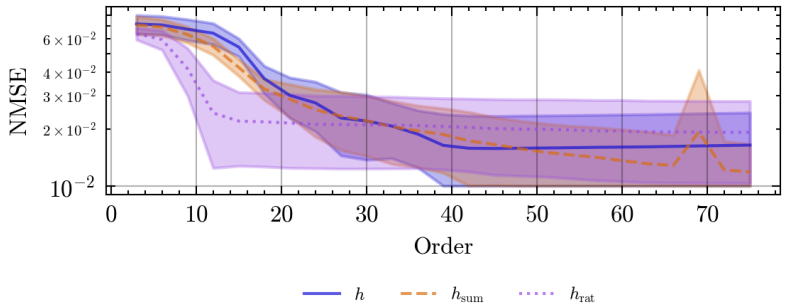


(c) Signal centered on source and target nodes used for the temperature displacement task

Figure 7: a) Measured wind velocity field. b) Derived temperature graph. c) Signal centered on source and target nodes used for the temperature displacement task.



(a) Estimated target signal (filter order: $K = 10$)



(b) NMSE for target reconstruction as a function of filter order

Figure 8: a) Estimated target signal using kernels $h, h_{\text{sum}}, h_{\text{rat}}$ with $K = 10$. b) Mean NMSE across the 10 (source,target) pairs for estimated target signal as a function of filter order.

Results. In this experiment we show similar takeaway as in Appendix F.1 but instead for a real graph, that is a temperature graph with direction given by the wind. We see once more a better NMSE performance for h_{rat} on low filter order regime w.r.t. h and h_{sum}. On the other hand, h_{sum} is able to achieve an overall better NMSE than h.