Self-Tuning Graph Filters via State-Dependent Operator Composition

Amir Ghazizadeh[1] **Mahyar Alinejad**[1] **George K. Atia**[1] **Rickard Ewetz**[2] **Hao Zheng**[1]
[1]Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL, USA
[2]Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA
{amir.g, mahyar.alinejad, george.atia, hao.zheng}@ucf.edu, rewetz@ufl.edu

Abstract

Most graph neural networks propagate information through a fixed-coefficient polynomial filter applied uniformly across every node and every layer. However, the appropriate filter varies along two distinct axes that such a parameterization cannot accommodate. Within a single graph, different nodes benefit from different mixtures of low-pass smoothing, high-pass contrast, and pass-through behavior. Across graphs, signal propagation dynamics themselves differ, with some graphs benefiting from long-range propagation across many hops while others require damping to prevent over-smoothing. The appropriate filter is therefore not a single fixed object at all, but a *state-dependent* composition whose behavior changes according to a node's current representation and its propagation history.

We propose CASTOR, a self-tuning graph filter that realizes this composition through state-dependent decisions made at every step of propagation. A router reads each node's current state together with its initial state, and emits per-node mixing weights over three primitives, namely low-pass smoothing, high-pass contrast, and identity. A learnable per-hop coefficient then determines how much of the most recent change in the propagation is carried forward, accelerating the iteration when positive and damping it when negative. Setting the coefficient to zero recovers standard first-order propagation. Across fourteen standard node-classification benchmarks spanning the full homophily spectrum, a single CASTOR instance ranks top-three on every dataset and achieves an average rank of 2.07, more than 3.5 ranks ahead of the next-best of fourteen baselines. Code is available at `https://github.com/amir-ghz/CASTOR`.

1 Introduction

Graph neural networks propagate information through a single linear filter that is fixed at construction time and applied uniformly across every node, every layer, and every graph [Kipf and Welling, 2016, Velickovic et al., 2017]. The most common choice is a normalized adjacency matrix or one of its low-degree polynomial variants [Wu et al., 2019, Defferrard et al., 2016], which spectrally suppress sharp variations across the graph and preserve the slow ones—an operation we will call *smoothing*. By construction, the design forces every node to share the same propagation filter regardless of its structural role within the graph or the transformations introduced by preceding layers.

The limitations of this rigidity are most visible on *heterophilic* graphs, where connected nodes often belong to different classes rather than similar ones [Zhu et al., 2020, Pei et al., 2020, Zheng et al., 2022]. Repeated smoothing then drives node representations toward a degenerate average, and a node often benefits more from *contrast* with its neighbors, the feature components that disagree with the local average, than from similarity to them. Several families of GNNs have been developed in response, including spectral filters with higher-frequency capacity [He et al., 2021, Chien et al., 2020], signed message-passing frameworks [Bo et al., 2021, Liang et al., 2024], and architectures

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

that look beyond the immediate neighborhood [Abu-El-Haija et al., 2019, Xu et al., 2018, Ahsaei et al., 2025]. In practice, a practitioner's first step on a new graph is to estimate its homophily ratio in order to pick a family of architectures, and *the choice of architecture is, in practice, dictated by the graph rather than by the task*.

We argue that this rigidity is an artifact of parameterization rather than an inherent property of the underlying problem. Section 2 identifies two distinct axes along which the appropriate propagation filter varies: variation *across nodes within a single hop*, and variation *across hops within a single graph's propagation trajectory*. In addition, it shows that neither can be jointly captured by a fixed-coefficient polynomial of $\widehat{\mathbf{W}}$ (normalized adjacency). The first axis reflects that different nodes within the same graph require different combinations of low-pass smoothing, high-pass contrast, and pass-through (identity) behavior, with the appropriate combination changing from one hop to the next. The second axis reflects that propagation dynamics themselves differ across graphs, with some graphs benefiting from acceleration across hops and others requiring damping to prevent over-smoothing. Together, these observations point at a single conclusion. The appropriate filter is not a single fixed operator, but a state-dependent composition that varies with both the node and the propagation step.

We propose CASTOR, a self-tuning graph filter that realizes this composition through state-dependent decisions made at every step of propagation. At each step, a *Router* reads each node's current state together with its initial state and emits per-node mixing weights over three primitives: low-pass smoothing, high-pass contrast, and identity. The block is iterated as a second-order recurrence with a learnable per-step *momentum coefficient* that controls how much of the most recent propagation change is carried forward, accelerating the iteration when positive and damping it when negative. Setting the coefficient to zero recovers standard first-order propagation. Across fourteen standard node-classification benchmarks spanning the full homophily spectrum, a single CASTOR instance ranks top-three on every dataset, with an average rank of 2.07 that is more than 3.5 ranks ahead of the next-best of fourteen baselines, and without any per-dataset architectural modification. Our contributions are as follows.

- We document two empirical axes along which the right graph filter varies on heterophilic benchmarks — *across nodes within a hop* and *across hops within a trajectory* — and connect each axis to a structural limitation of existing fixed-coefficient polynomial filters (Section 2).
- We introduce CASTOR, a graph filter whose every step is a per-node convex combination of three elementary primitives chosen by a shared Router, and whose iteration is a second-order recurrence with a learnable, sign-flexible per-step momentum coefficient (Section 4).
- We show that the resulting filter family strictly contains every fixed-coefficient polynomial filter of the same depth, first- or second-order, and that allowing per-node routing exits the polynomial family altogether.
- Across fourteen standard node-classification benchmarks spanning the full homophily spectrum, CASTOR ranks top-three on every dataset and attains the best average rank (2.07, more than 3.5 ranks ahead of the second-best architecture) among the fifteen architectures we evaluate, without any per-dataset architectural change.

2 Heterophilic Graph Characteristics and Filter Limitations

Standard heterophilic benchmarks expose two independent sources of variation that fixed-coefficient graph filters cannot accommodate. The first shows that, within a single graph, different nodes prefer different propagation depths, and that the preference cannot be summarized at the layer level. The second shows that, across graphs, the rate at which the smoothing operator, viewed as a graph filter, mixes the signal varies sharply. Neither variation can be jointly absorbed by a fixed-coefficient polynomial filter (SGN [Wu et al., 2019], GPR-GNN [Chien et al., 2020], BERNNET [He et al., 2021], CHEBNET [Defferrard et al., 2016]), nor by a per-node mixer that commits to a single mixture per layer [Luan et al., 2022].

2.1 Hop preferences vary across nodes within a graph

For every labeled node v, we ask the following question: among the features that best predict v's class, at what propagation depth is each feature most informative? Concretely, for each node v we take its top-30 features ranked by per-node mutual information with the label y_v; for each such feature

we identify the depth $k \in \{0, 1, \ldots, 5\}$ at which the feature is most informative about y_v after k rounds of smoothing by the graph adjacency ($\widehat{\mathbf{W}}$). Averaging the result across all nodes that share a class gives a *per-class hop fingerprint* — at each depth k, the fraction of that class's most informative features that peak at depth k.

Figure 1 reports the result on Roman-empire [Platonov et al., 2023]. Different classes peak at different depths. Some at $k=0$ (the raw features alone are decisive), others at $k = 1$ (the immediate neighborhood is decisive), and others at $k = 2$ (two hops are needed), and every class spreads its informative mass across at least four hops. A graph-wide low-pass polynomial filter applies a single schedule of hop weights to every node by construction, so it can be tuned to fit at most one of these fingerprints at a time. A per-node mixer that picks among low-pass, high-pass, and identity *once per layer* [Luan et al., 2022, Chien et al., 2020] is similarly insufficient, because the appropriate mixture must adapt at every hop, not at every layer. Whatever distinguishes one node from another for the purpose of choosing a propagation scheme must therefore be re-evaluated at every hop, not committed to once.

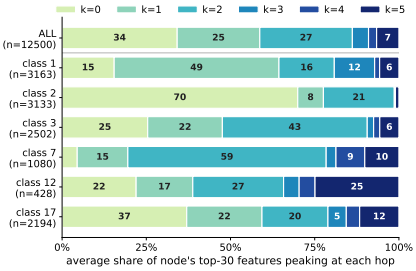

Figure 1: Per-class hop fingerprints on Roman-empire (12,500 labeled nodes, top-30 features per node). Each bar is the class-mean share of features that are most informative about the label after k rounds of smoothing by $\widehat{\mathbf{W}}$.

2.2 Mixing rates vary across heterophilic graphs

We sample a random unit vector $\mathbf{x}_0$ and track, at every depth k, the cosine similarity between the iterated signal $\mathbf{x}_k := \widehat{\mathbf{W}}^k \mathbf{x}_0$ and the starting direction $\mathbf{x}_0$. A high cosine means the iterated signal has barely rotated away from where it started, so repeated propagation through $\widehat{\mathbf{W}}$ is mixing the signal slowly; late hops add little new information beyond the first few. A low cosine means the signal has fully re-oriented, so repeated propagation through $\widehat{\mathbf{W}}$ has mixed the signal quickly; further low-pass propagation now begins to over-smooth the representation.

Figure 2 shows two qualitatively different regimes among standard heterophilic graphs. On Cornell, Texas, and Wisconsin, the cosine remains above 30% even at depth 10 — the smoothing operator mixes the signal slowly, and a fixed schedule that runs many low-pass hops adds essentially no new information beyond the first few. On Squirrel and Chameleon, the cosine drops below 15% at the same depth — the operator mixes the signal quickly, and the same number of low-pass hops over-smooths the representation. A single fixed-coefficient polynomial filter applies the same schedule to both regimes, and the schedule that helps the slow-mixing graphs over-smooths the fast-mixing ones, and vice versa. Section 4.2 addresses this with a per-step *momentum coefficient* that the model learns from data, positive on slow-mixing graphs to carry the most recent propagation change forward across hops (acceleration), and negative on fast-mixing graphs to counter the most recent change and slow the mixing (damping).

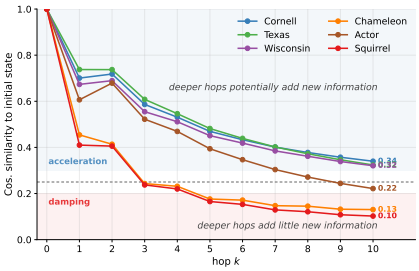

Figure 2: Initial-direction memory across depth on five heterophilic graphs. Cosine similarity between the iterated signal $\widehat{\mathbf{W}}^k \mathbf{x}_0$ and the random starting vector $\mathbf{x}_0$. Cornell, Texas, and Wisconsin mix slowly; Squirrel and Chameleon mix quickly.

Take-away. The appropriate propagation filter is not a single fixed object, but a state-dependent composition that varies with both the node (which mixture of low-pass, high-pass, and identity to apply at the current hop) and the propagation step (how strongly the next state should inherit the most recent propagation change). Section 4 realizes this composition through two lightweight, learnable mechanisms: a Router that emits per-node mixing weights over the three primitives, and a per-step momentum coefficient that controls how much of the most recent propagation change is carried forward.

3 Preliminaries and Notation

We consider an attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{X})$ with $N = |\mathcal{V}|$ nodes, edge set $\mathcal{E}$, node feature matrix $\mathbf{X} \in \mathbb{R}^{N \times F}$, and node labels $y_v \in \{1, \ldots, C\}$ observed only on a training subset $\mathcal{V}_{\text{train}} \subset \mathcal{V}$.

Let $\mathbf{A} \in \mathbb{R}^{N\times N}$ denote the adjacency matrix and $\widehat{\mathbf{W}} = (\mathbf{D}+\mathbf{I})^{-1/2}(\mathbf{A}+\mathbf{I})(\mathbf{D}+\mathbf{I})^{-1/2}$ its symmetric normalization with self-loops, where $\mathbf{D}$ is the degree matrix. We write $\mathbf{Z} \in \mathbb{R}^{N\times C}$ for a propagation state and $\mathbf{Z}_v \in \mathbb{R}^C$ for its row at node v; in the architecture of Section 4, $\mathbf{Z}$ lives in C-dimensional class-logit space, so the final softmax sits directly on the propagation output and the spectral interpretation of $\widehat{\mathbf{W}}$ is preserved before *and* after every step.

Three primitives. Three operations on a state $\mathbf{Z}$ play a privileged role throughout. *Smoothing* $\widehat{\mathbf{W}}\mathbf{Z}$ replaces each node's representation with the degree-normalized average of its neighbors — the standard low-pass operation underlying most GNNs. The *smoothing residual* $\boldsymbol{\Delta}\mathbf{Z} := \mathbf{Z} - \widehat{\mathbf{W}}\mathbf{Z} = (\mathbf{I} - \widehat{\mathbf{W}})\mathbf{Z}$ is the high-pass complement: it captures the part of $\mathbf{Z}_v$ that disagrees with the local average. The *identity* $\mathbf{Z}$ leaves the state unchanged — the appropriate choice when neither smoothing nor contrast carries useful information. The three primitives jointly span the affine line in spectral space whose endpoints are smoothing and the residual, with identity at the convex midpoint, so any convex combination of them is a degree-one polynomial in $\widehat{\mathbf{W}}$. The question Section 4 takes up is how to compose them adaptively, per node and per propagation step.

4 CASTOR: A Self-Tuning Graph Filter

CASTOR makes two state-dependent decisions at every step of propagation, neither of which is available to a fixed-coefficient graph filter. *Within* a single step (Section 4.1), a small *Router* reads each node's current state together with its initial state and emits a per-node convex combination of the three primitives of Section 3: smoothing $\widehat{\mathbf{W}}$ (low-pass), the smoothing residual $\mathbf{I} - \widehat{\mathbf{W}}$ (high-pass), and the identity $\mathbf{I}$ (pass-through). The step transforms each node's state by that combination. *Across* consecutive steps (Section 4.2), each new state is the routed step applied to the current state *plus* a learnable scalar multiple β_k of the velocity $\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]}$. We call this the *momentum coupling*: a single second-order link between consecutive states with a per-step *momentum coefficient* $\beta_k \in \mathbb{R}$ that is unconstrained in sign. Iterating the two decisions K times defines the CASTOR block (Figure 3).

Each decision unlocks a function class that no prior graph filter reaches. Per-node Routing exits the family of fixed-coefficient polynomials in $\widehat{\mathbf{W}}$ that contains SGC [Wu et al., 2019], APPNP [Gasteiger et al., 2018], GPR-GNN [Chien et al., 2020] and BERNNET [He et al., 2021], because different nodes are effectively assigned different polynomial coefficients in the same forward pass (Theorem 7, Appendix F.2). The momentum coupling then turns the iteration into a learnable three-term recurrence in $\widehat{\mathbf{W}}$, the same algebraic family that contains the Chebyshev recurrence of CHEBNET [Defferrard et al., 2016] (Theorem 5); together, the two decisions allow a single architecture to accelerate stalled trajectories on graphs like Cornell ($\beta_k > 0$) and damp runaway ones on graphs like Squirrel ($\beta_k < 0$) without any per-dataset switch.

Throughout this section, $\mathbf{Z} \in \mathbb{R}^{N\times C}$ denotes the propagation state at any step and $\mathbf{Z}^{(0)} \in \mathbb{R}^{N\times C}$ the *initial state*, computed once from the raw features by a small encoder (Section 4.3). The propagation block re-reads $\mathbf{Z}^{(0)}$ as input to the Router at every one of the K steps. Two design principles shape the rest of the construction: *(i)* all per-node state-dependence is concentrated in three Router-emitted weights $\mathbf{a}_{v,k} \in \Delta_3$ broadcast across the C class channels of $\mathbf{Z}_v$, so the propagation block holds no per-node learnable matrices and applies no per-branch non-linearities; *(ii)* every step restricted to a single node is a convex combination of $\widehat{\mathbf{W}}$, $\mathbf{I} - \widehat{\mathbf{W}}$, and $\mathbf{I}$ — a degree-one polynomial in $\widehat{\mathbf{W}}$ — and the momentum coupling extends this to a learnable three-term polynomial recurrence at every node. We organize the rest of the section around the routing kernel that defines a single step (Section 4.1), the K-step CASTOR block built from it (Section 4.2), and the pipeline that surrounds them (Section 4.3).

4.1 The Routing Kernel

The routing kernel of CASTOR is a state-dependent linear map that, at every step $k \in \{1, \dots, K\}$ and for every node, chooses how strongly to apply each of the three primitives of Section 3 (Figure 3). Let $\mathbf{Z} \in \mathbb{R}^{N\times C}$ denote the current state at the input of step k. The kernel first computes the smoothing residual,

$$\boldsymbol{\Delta}\mathbf{Z} \;=\; (\mathbf{I} - \widehat{\mathbf{W}})\mathbf{Z} \;\in\; \mathbb{R}^{N\times C}, \tag{1}$$

which acts as a local high-pass signal: large entries of $\boldsymbol{\Delta}\mathbf{Z}_v$ indicate channels on which v disagrees with the average of its neighbors, small entries indicate agreement. The residual is concatenated with the initial state $\mathbf{Z}^{(0)}$ and passed to the *Router* $\rho : \mathbb{R}^{2C} \to \mathbb{R}^3$,

$$\mathbf{L}_k \;=\; \rho\big([\boldsymbol{\Delta}\mathbf{Z} \,\|\, \mathbf{Z}^{(0)}]\big) \;+\; \mathbf{1}\,\mathbf{b}_k^\top \;\in\; \mathbb{R}^{N\times 3}, \tag{2}$$

where ρ is a single-hidden-layer MLP, $\rho(\mathbf{u}) = \mathbf{W}_{\rho,2}\,\mathrm{ReLU}(\mathbf{W}_{\rho,1}\mathbf{u})$ with $\mathbf{W}_{\rho,1} \in \mathbb{R}^{H_\rho \times 2C}$ and $\mathbf{W}_{\rho,2} \in \mathbb{R}^{3\times H_\rho}$, and $\mathbf{b}_k \in \mathbb{R}^3$ is a learnable *step-dependent* bias. A softmax along the last axis of $\mathbf{L}_k$ yields per-node mixing weights:

$$\mathbf{a}_k \;=\; \mathrm{softmax}_3(\mathbf{L}_k) \;\in\; \Delta_3^N, \tag{3}$$

so that for every node v, $\mathbf{a}_k[v] \in \mathbb{R}^3$ is a probability vector $(\alpha^{\mathrm{sm}}_{v,k}, \alpha^{\mathrm{hi}}_{v,k}, \alpha^{\mathrm{id}}_{v,k})$ over the three primitives. The *routed step* transforms the state by the corresponding per-node convex combination,

$$T_k\mathbf{Z} \;=\; \mathbf{a}_k[:,0] \odot \big(\widehat{\mathbf{W}}\mathbf{Z}\big) \;+\; \mathbf{a}_k[:,1] \odot \boldsymbol{\Delta}\mathbf{Z} \;+\; \mathbf{a}_k[:,2] \odot \mathbf{Z}, \tag{4}$$

where $\odot$ is element-wise multiplication broadcast across the C class channels of each row. The smoothed signal $\widehat{\mathbf{W}}\mathbf{Z}$ and the residual $\boldsymbol{\Delta}\mathbf{Z} = \mathbf{Z} - \widehat{\mathbf{W}}\mathbf{Z}$ share a single sparse matrix multiplication, so the kernel costs exactly one $\widehat{\mathbf{W}}\mathbf{Z}$ application per step. Three properties of this construction merit emphasis:

(i) Per-node, rank-one routing. The Router emits *three* numbers per node — one per primitive — rather than $3C$. This is a deliberate inductive bias, not a parameter-saving approximation. In the class-logit state space, every channel of $\mathbf{Z}_v$ is a same-typed quantity (a logit for one class), and the question the Router answers at step k is the *node-level* one: should v listen to its neighbors, contrast against them, or stand pat? The answer is a property of the node and of how its current state relates to its neighborhood, not a property of one of its channels. Sharing the routing decision across all C classes therefore matches the geometry of the problem and keeps every step of the block, when restricted to a single node, an honest convex combination of $\widehat{\mathbf{W}}$, $\mathbf{I} - \widehat{\mathbf{W}}$, and $\mathbf{I}$ rather than a per-channel mixture of three different polynomials in $\widehat{\mathbf{W}}$. We empirically verified that lifting this rank-one constraint to a per-(node, channel) Router does not improve accuracy on any benchmark and degrades it on the smaller heterophilic graphs. Theorem 12 formalizes the resulting per-node universality (Appendix F.3).

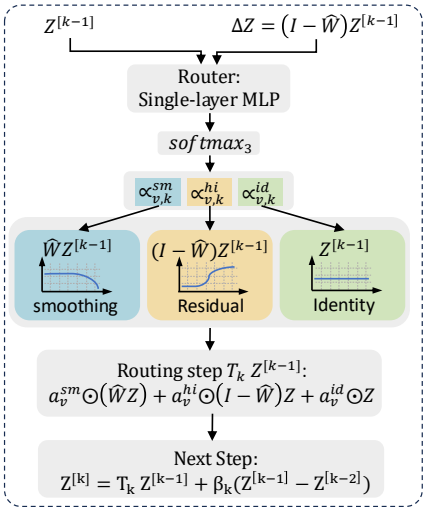

Figure 3: **One step of the CASTOR block.** The current state $\mathbf{Z}^{[k-1]}$ and its smoothing residual $\boldsymbol{\Delta}\mathbf{Z} = (\mathbf{I} - \widehat{\mathbf{W}})\mathbf{Z}^{[k-1]}$ are concatenated and fed through a single-hidden-layer MLP (the Router); a softmax over its three-dim output produces per-node mixing weights $\mathbf{a}_k(v) \in \Delta_3$ over smoothing, residual, and identity. The routed step T_k applies the corresponding per-node convex combination, and the momentum term $\beta_k(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]})$ is added to produce $\mathbf{Z}^{[k]}$.

(ii) The initial state as a stable reference. Reading the residual $\boldsymbol{\Delta}\mathbf{Z}$ alone is not sufficient to determine the right primitive. Two nodes whose state has been heavily mixed may both have small residuals at step k, yet one may be a hub whose representation has already converged and should now identity-skip, while the other may sit at a class boundary at which further contrast is warranted. The initial state $\mathbf{Z}^{(0)}$ provides the disambiguating signal: it preserves the spectrally unmixed identity of the node throughout the recurrence, so the Router conditions its decision on both *what the node started as* and *how its current state disagrees with its neighborhood*. Without it, the Router would collapse to a function of $\boldsymbol{\Delta}\mathbf{Z}$ alone and would lose the ability to distinguish nodes whose residuals coincide for unrelated reasons. We are not aware of a prior per-node spectral mixer that re-reads its input at every step in this way.

(iii) Step-dependent bias with a smoothing-then-preserve prior. The bias $\mathbf{b}_k \in \mathbb{R}^3$ is a tiny ($3K$-parameter) addition that lets the model express a hop-dependent default. We initialize it with a smoothing-then-preserving schedule,

$$\mathbf{b}_k \;=\; \tfrac{1}{2}\left(\frac{K-1-k}{K-1},\; 0\,,\; \frac{k}{K-1}\right), \qquad k = 0,\ldots,K-1, \tag{5}$$

so that early steps gently favor smoothing, late steps gently favor identity, and the high-pass primitive begins from neutral. With the Router shut off, this schedule places the model at initialization in

a region of parameter space where the block reduces to a low-pass smoothing chain; the prior is overwritten quickly by gradient descent whenever the data prefer otherwise. Empirically, this initialization, together with the smooth softmax mixing of Eq. (3), is what allows a single instantiation of CASTOR to discover distinct routing profiles on homophilic and heterophilic graphs without any architectural switch.

4.2 The CASTOR Block

A single application of the routing kernel produces $T_k\mathbf{Z}$, the per-node convex combination of the three primitives of Section 4.1 acting on the current state. Stacking K such kernels in sequence already yields a deep, state-dependent graph filter,

$$\mathbf{Z}^{[k]} \;=\; T_k\,\mathbf{Z}^{[k-1]}, \qquad k=1,\dots,K, \qquad \mathbf{Z}^{[0]} = \mathrm{Drop}_h(\mathbf{Z}^{(0)}),$$

which is the bare-bones routed propagation. The remainder of this subsection enriches the first-order recurrence with the second decision of Section 4: a per-step momentum that couples the next state to both the current state and the velocity carried over from the previous step.

Momentum coupling. Adopting the convention $\mathbf{Z}^{[-1]} := \mathbf{Z}^{[0]}$, so that the velocity vanishes at $k=1$, the CASTOR block is the second-order recurrence

$$\boxed{\mathbf{Z}^{[k]} \;=\; T_k\,\mathbf{Z}^{[k-1]} \;+\; \beta_k\left(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]}\right), \qquad k=1,\dots,K,} \tag{6}$$

where $\beta_k \in \mathbb{R}$ is an unconstrained learnable *momentum coefficient* at step k, $\mathrm{Drop}_h(\cdot)$ is dropout at rate h applied once to the initial state before the recurrence begins, and the block output is $\mathbf{Z}^{[K]}$. The state stays in $\mathbb{R}^{N\times C}$ throughout. The Router ρ is reused at every step inside T_k and only the per-step bias $\mathbf{b}_k$ and the per-step coefficient β_k vary with k.

The role of the momentum term $\beta_k(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]})$ is best understood dynamically. The difference $\mathbf{v}^{[k-1]} := \mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]}$ is the velocity of each node's trajectory through propagation space at step $k-1$, computed independently per node and per channel. Adding $\beta_k\mathbf{v}^{[k-1]}$ to the next state pushes the trajectory along the direction it has been moving when β_k is positive and against it when β_k is negative, with three regimes of qualitatively different character:

- $\beta_k = 0$: *first-order baseline*. The recurrence collapses to $\mathbf{Z}^{[k]} = T_k\mathbf{Z}^{[k-1]}$, the bare-bones routed propagation. This is where we initialize.
- $\beta_k > 0$: *acceleration*. The new state is pushed along the existing direction of motion (Polyak's heavy-ball update for first-order optimization, lifted into graph propagation). We observe this regime selected on graphs whose state converges slowly along a stable smoothing direction — e.g. Cornell, Texas, Wisconsin in Section 5.
- $\beta_k < 0$: *damping*. The new state is pulled back against the existing motion (the negative cross-term that makes Chebyshev-type three-term recurrences contractive). We observe this regime on dense heterophilic graphs whose trajectories tend to overshoot under repeated smoothing — e.g. Squirrel, Chameleon.

Because the sign of β_k is unconstrained and learned, the model interpolates smoothly between these three regimes without any per-dataset switch, and at each step it can pick a different one. Initializing $\beta_k = 0$ at every step lets training begin from the first-order recurrence and lets the optimizer grow $|\beta_k|$ in either direction as the data demand. We are not aware of a prior graph filter that exposes the cross-coefficient on $\mathbf{Z}^{[k-2]}$ as a learnable, sign-flexible scalar at every step.

A node's trajectory through the block. A node's evolution can be visualized as a particle moving through C-dimensional class-logit space. At every step the Router decides whether to pull the particle toward the local average (smoothing), away from it (residual), or to leave it where it is (identity); the momentum coupling then either nudges the particle further along the direction it just moved ($\beta_k > 0$) or pulls it back ($\beta_k < 0$). Two nodes in the same graph may follow trajectories of strikingly different shape under one shared Router and one shared schedule $\{\beta_k\}$: a node deep in the interior of its class cluster may smooth aggressively at $k=1$, accelerate along that direction for a few hops, and switch to identity once its representation has converged; a node sitting at a class boundary may amplify

the residual at every step and rely on damping to keep its trajectory from oscillating. The mixing tensor $\mathbf{a}_k$ and the momentum coefficient β_k are exactly the quantities used in the forward pass, so the spectral profile that CASTOR chooses at every step and every node can be read off directly from a trained model without any post-hoc analysis.

What the block expresses. It is instructive to write the block in closed form for the simplest case. If the per-node mixing $\mathbf{a}_k$ is held constant across nodes at every step, T_k reduces to a node-uniform degree-one polynomial in $\widehat{\mathbf{W}}$,

$$T_k \;=\; \alpha_k^{\mathrm{sm}}\,\widehat{\mathbf{W}} \;+\; \alpha_k^{\mathrm{hi}}\,(\mathbf{I}-\widehat{\mathbf{W}}) \;+\; \alpha_k^{\mathrm{id}}\,\mathbf{I},$$

and Eq. (6) becomes the three-term polynomial recurrence

$$\mathbf{Z}^{[k]} \;=\; \big(T_k+\beta_k\mathbf{I}\big)\,\mathbf{Z}^{[k-1]} \;-\; \beta_k\,\mathbf{Z}^{[k-2]}, \tag{7}$$

whose unrolled solution $\mathbf{Z}^{[K]} = P_K(\widehat{\mathbf{W}})\,\mathbf{Z}^{(0)}$ is a polynomial in $\widehat{\mathbf{W}}$ of degree at most K with coefficients determined by $\{\alpha_k^{\mathrm{sm}},\alpha_k^{\mathrm{hi}},\alpha_k^{\mathrm{id}},\beta_k\}_{k=1}^{K}$. Setting $\beta_k\equiv 0$ collapses Eq. (7) back to a per-step convex-combination polynomial of the kind realized by classical first-order graph filters; allowing β_k to take either sign restores the full reach of three-term recurrences in $\widehat{\mathbf{W}}$, the algebraic family in which Chebyshev recurrences live. When the per-node mixing $\mathbf{a}_k$ is also allowed to vary across nodes, the linear map from $\mathbf{Z}^{(0)}$ to $\mathbf{Z}^{[K]}$ is no longer expressible as a single matrix polynomial in $\widehat{\mathbf{W}}$, since different nodes are effectively assigned different recurrence coefficients in the same forward pass — a function class that no fixed-coefficient polynomial filter, first- or second-order, can match.

4.3 Full Pipeline

The full CASTOR pipeline wraps the CASTOR block of Section 4.2 between a small encoder and a final softmax. The encoder is a two-layer feedforward network of hidden width D that maps the raw features of each node to the initial state $\mathbf{Z}^{(0)}\in\mathbb{R}^{N\times C}$,

$$\mathbf{Z}^{(0)} \;=\; \mathrm{Drop}_p\Big(\,\mathrm{ReLU}\big(\,\mathrm{Drop}_p(\mathbf{X})\,\mathbf{W}_1+\mathbf{1}\mathbf{b}_1^{\top}\big)\Big)\,\mathbf{W}_2+\mathbf{1}\mathbf{b}_2^{\top}, \tag{8}$$

with $\mathbf{W}_1\in\mathbb{R}^{F\times D}$, $\mathbf{W}_2\in\mathbb{R}^{D\times C}$, and Drop_p feature dropout at rate p applied independently before each linear map. The encoder's hidden width D is decoupled from the propagation-state dimension C: the Router and the momentum coefficients have no dependence on D, so the encoder's capacity and the block's parameter count can be tuned independently. The complete pipeline reads

$$\mathbf{X} \;\longrightarrow\; \mathbf{Z}^{(0)} \;\longrightarrow\; \mathbf{Z}^{[K]} \;\longrightarrow\; \mathbf{Y}, \tag{9}$$

where $\mathbf{Z}^{[K]}$ is the output of the CASTOR block of Section 4.2 and the per-node class probabilities are

$$\mathbf{Y}_v \;=\; \mathrm{softmax}\big(\mathbf{Z}_v^{[K]}\big) \;\in\; \Delta_C, \qquad v\in\mathcal{V}. \tag{10}$$

No separate classifier MLP is required: the block's output is class-logit-shaped by construction, so the softmax sits directly on top of $\mathbf{Z}^{[K]}$. Inside the block itself there is no dropout: smoothing and residual are linear maps and identity is the identity, so any noise injected mid-recurrence would propagate through the iteration without ever passing through a non-linearity that could absorb it. The only dropout used by the architecture is Drop_p inside the encoder of Eq. (8) and Drop_h at the input of the block (Eq. (6)).

Theoretical and computational reach. Appendix F.7 formalizes the embeddings of SGC, APPNP, GPRGNN, BERNNET, and rescaled-spectrum CHEBNET as exact instances of CASTOR and identifies a small set of related architectures that are structurally outside the reachable family. Stability and energetics of the recurrence are characterized in Appendices F.4 and F.6, sensitivity and over-squashing relief in Appendix F.5, and a per-step cost and parameter-count comparison with the baselines of Section 5 appears in Appendix B.

5 Evaluation

Configuration and setup. We evaluate CASTOR on semi-supervised node classification across a diverse set of benchmark datasets that span the full homophily-heterophily spectrum, with detailed

Table 1: Node classification metric (%): accuracy on multi-class, ROC-AUC on binary targets (MINESWEEPER, TOLOKERS, QUESTIONS); mean ± 95% bootstrap CI across 10 splits. **Green**/**Blue**/**Red** mark 1st/2nd/3rd best per row. The bottom row is each model's average rank across the fourteen datasets (lower is better; OOM treated as worst rank).

Dataset	MLP	GCN	SGC	GCN-II	GCN-JK	GAT-JK	H2GCN	GAT	MixHop	BernNet	GPRGNN	M2MGNN	CO-GNN	GAMMA	CASTOR
Homophilic graphs (BernNet 48/32/20 splits) [He et al., 2021]															
Cora	$75.49_{\pm 1}$	$87.73_{\pm 1}$	$87.86_{\pm 0}$	$87.10_{\pm 0}$	$86.84_{\pm 1}$	$87.06_{\pm 1}$	**$88.47_{\pm 1}$**	$87.86_{\pm 1}$	$85.47_{\pm 1}$	$88.39_{\pm 1}$	$88.19_{\pm 1}$	$86.56_{\pm 1}$	$77.78_{\pm 1}$	$86.95_{\pm 1}$	**$88.24_{\pm 1}$**
CiteSeer	$72.11_{\pm 1}$	$77.25_{\pm 1}$	$77.19_{\pm 1}$	$75.23_{\pm 1}$	$73.85_{\pm 1}$	$74.74_{\pm 1}$	**$77.73_{\pm 1}$**	$77.34_{\pm 1}$	$73.16_{\pm 1}$	$77.41_{\pm 1}$	$76.99_{\pm 1}$	$75.14_{\pm 1}$	$72.83_{\pm 1}$	$76.02_{\pm 1}$	**$77.70_{\pm 1}$**
PubMed	$87.17_{\pm 0}$	$87.17_{\pm 0}$	$88.13_{\pm 0}$	$86.42_{\pm 0}$	$87.90_{\pm 0}$	$86.39_{\pm 0}$	$89.13_{\pm 0}$	$85.52_{\pm 0}$	$89.10_{\pm 0}$	$89.16_{\pm 0}$	$87.24_{\pm 0}$	$87.94_{\pm 0}$	$89.27_{\pm 0}$	$89.17_{\pm 0}$	**$89.85_{\pm 0}$**
Heterophilic small graphs (BernNet 48/32/20 splits) [He et al., 2021]															
Texas	$73.06_{\pm 5}$	$58.89_{\pm 4}$	$52.78_{\pm 4}$	$52.50_{\pm 5}$	$52.78_{\pm 2}$	$51.11_{\pm 4}$	$77.22_{\pm 4}$	$51.67_{\pm 3}$	$67.50_{\pm 3}$	$80.56_{\pm 5}$	$69.72_{\pm 6}$	$68.61_{\pm 4}$	$20.00_{\pm 2}$	**$85.53_{\pm 2}$**	**$80.28_{\pm 4}$**
Wisconsin	$83.33_{\pm 4}$	$56.08_{\pm 8}$	$55.10_{\pm 7}$	$50.00_{\pm 5}$	$51.76_{\pm 6}$	$45.29_{\pm 6}$	$83.53_{\pm 5}$	$52.55_{\pm 7}$	$78.04_{\pm 2}$	$84.90_{\pm 3}$	$75.10_{\pm 4}$	$69.22_{\pm 4}$	$3.53_{\pm 1}$	$85.49_{\pm 3}$	**$86.67_{\pm 3}$**
Actor	**$37.05_{\pm 1}$**	$30.91_{\pm 1}$	$30.43_{\pm 1}$	$33.67_{\pm 1}$	$29.77_{\pm 0}$	$28.68_{\pm 1}$	$35.80_{\pm 1}$	$28.25_{\pm 1}$	$33.22_{\pm 1}$	$27.35_{\pm 1}$	$34.16_{\pm 1}$	$35.76_{\pm 1}$	$11.21_{\pm 0}$	$35.95_{\pm 1}$	**$36.49_{\pm 1}$**
Squirrel	$29.25_{\pm 1}$	$34.51_{\pm 1}$	$48.08_{\pm 1}$	$32.82_{\pm 1}$	$41.41_{\pm 1}$	$41.40_{\pm 1}$	$28.97_{\pm 1}$	$32.06_{\pm 1}$	**$50.84_{\pm 1}$**	$48.97_{\pm 1}$	$34.79_{\pm 1}$	$36.63_{\pm 1}$	$19.89_{\pm 1}$	$50.54_{\pm 1}$	**$49.20_{\pm 2}$**
Chameleon	$45.25_{\pm 1}$	$57.74_{\pm 1}$	$65.25_{\pm 2}$	$44.88_{\pm 2}$	$63.58_{\pm 1}$	$64.44_{\pm 2}$	$50.42_{\pm 1}$	$58.18_{\pm 2}$	$62.22_{\pm 2}$	**$66.44_{\pm 1}$**	$58.15_{\pm 1}$	$65.03_{\pm 2}$	$20.18_{\pm 1}$	$65.04_{\pm 1}$	**$65.60_{\pm 2}$**
Cornell	$76.67_{\pm 6}$	$49.17_{\pm 4}$	$51.39_{\pm 5}$	$39.72_{\pm 5}$	$46.94_{\pm 4}$	$43.33_{\pm 5}$	$78.06_{\pm 5}$	$43.61_{\pm 4}$	$65.56_{\pm 5}$	$76.39_{\pm 4}$	$56.94_{\pm 8}$	$62.22_{\pm 5}$	$22.78_{\pm 3}$	$79.74_{\pm 3}$	**$81.67_{\pm 4}$**
Heterophilic mid-scale graphs (Platonov 50/25/25 splits) [Platonov et al., 2023]															
ROMAN-EMPIRE	$65.97_{\pm 0}$	$51.45_{\pm 0}$	$35.00_{\pm 0}$	$71.73_{\pm 0}$	$74.05_{\pm 0}$	$73.45_{\pm 0}$	$64.61_{\pm 0}$	$37.53_{\pm 0}$	$74.30_{\pm 0}$	$65.56_{\pm 0}$	$64.22_{\pm 0}$	$72.31_{\pm 0}$	$75.87_{\pm 0}$	**$80.08_{\pm 1}$**	**$78.39_{\pm 1}$**
AMAZON-RATINGS	$44.81_{\pm 0}$	$43.75_{\pm 0}$	$46.51_{\pm 0}$	$44.71_{\pm 0}$	$44.61_{\pm 0}$	$43.10_{\pm 0}$	$42.61_{\pm 0}$	$41.35_{\pm 0}$	**$48.53_{\pm 0}$**	$45.64_{\pm 0}$	$44.19_{\pm 0}$	$45.98_{\pm 0}$	$47.33_{\pm 0}$	$44.07_{\pm 0}$	**$47.41_{\pm 0}$**
MINESWEEPER	$50.52_{\pm 1}$	$72.32_{\pm 1}$	$72.49_{\pm 1}$	$87.45_{\pm 1}$	$89.98_{\pm 0}$	$90.07_{\pm 0}$	$87.15_{\pm 1}$	$72.43_{\pm 1}$	$91.15_{\pm 0}$	$74.99_{\pm 1}$	$83.04_{\pm 3}$	$88.99_{\pm 1}$	$90.35_{\pm 0}$	$86.83_{\pm 1}$	**$91.70_{\pm 1}$**
TOLOKERS	$73.97_{\pm 1}$	$75.56_{\pm 1}$	$76.26_{\pm 1}$	$76.71_{\pm 1}$	$81.35_{\pm 1}$	$79.57_{\pm 1}$	$74.72_{\pm 1}$	$69.67_{\pm 1}$	**$85.23_{\pm 1}$**	$69.74_{\pm 1}$	$73.04_{\pm 1}$	$78.15_{\pm 1}$	$77.87_{\pm 1}$	$69.88_{\pm 1}$	**$80.08_{\pm 1}$**
QUESTIONS	$70.61_{\pm 0}$	$75.00_{\pm 1}$	$75.75_{\pm 1}$	**$76.34_{\pm 1}$**	$75.86_{\pm 1}$	$74.18_{\pm 1}$	OOM	$71.00_{\pm 1}$	$75.45_{\pm 1}$	$63.31_{\pm 1}$	$53.84_{\pm 1}$	$76.17_{\pm 0}$	$72.35_{\pm 1}$	$72.34_{\pm 1}$	**$76.13_{\pm 1}$**
Avg. rank ↓	9.71	9.79	7.79	9.43	8.21	9.64	7.57	11.64	**5.93**	6.57	8.93	6.79	10.29	**5.64**	**2.07**

results in Table 1. Across all benchmarks, the same single instantiation of CASTOR is used — there is no per-dataset architectural switch between homophilic and heterophilic variants, and the only quantities tuned per dataset are standard regularizers (dropout) and the per-layer router depth K. All models are implemented in PyTorch Geometric [Fey and Lenssen, 2019]. Experiments were conducted on NVIDIA A100 GPUs (80GB) [NVIDIA Corporation, 2020] with CUDA 12.8; profiling for memory and runtime measurements relied on NVIDIA Nsight Compute CLI [NVIDIA Corporation, 2025].

Baselines. To benchmark CASTOR, we compare it against a comprehensive suite of baselines representing key strategies in graph representation learning, especially for heterophilic contexts. This selection includes *Standard GNNs and Foundational Models* (MLP [Hu et al., 2021], GCN [Kipf and Welling, 2016], GAT [Velickovic et al., 2017], SGC [Wu et al., 2019]), which serve to ground performance and highlight the challenges heterophily poses to conventional message-passing paradigms. We also incorporate *Spectral Graph Filters* (GCN-II [Chen et al., 2020], GPRGNN [Chien et al., 2020], BernNet [He et al., 2021]), chosen for their design leveraging graph signal processing to capture the mixed-frequency signals often characteristic of heterophilic structures. Furthermore, *Methods Utilizing High-Order Neighbors* (MixHop [Abu-El-Haija et al., 2019], H2GCN [Zhu et al., 2020], GCN-JK [Xu et al., 2018], GAT-JK [Xu et al., 2018]) are included, as they aim to discover informative, potentially distant, same-class nodes by expanding the receptive field beyond immediate, possibly misleading, connections. Finally, *Discriminative Message Passing Techniques* (M2MGNN [Liang et al., 2024], CO-GNN [Finkelshtein et al., 2023], and GAMMA [Ahsaei et al., 2025]) represent approaches that enable more selective and nuanced aggregation from diverse or signed neighborhood information, crucial when local context is not uniformly homophilous. This diverse selection provides a robust testbed for evaluating CASTOR's adaptability and effectiveness across the spectrum of graph structures.

Training recipe. Every model — baselines and CASTOR alike — is run with a common two-layer backbone at fair ~ 64 effective hidden dimension on the Platonov datasets, so that multi-power, multi-head, and JK-cat architectures do not enjoy a capacity advantage over single-stream ones. We use the BernNet-style $10 \times 48/32/20$ splits of Zhu et al. [2020] for the nine small/homophilic datasets [Pei et al., 2020] and the official $10 \times 50/25/25$ splits of Platonov et al. [2023] for the five Platonov datasets; for each (dataset, architecture) pair we tune over a small grid of learning rates, dropouts, and propagation depths, train with early stopping on the per-dataset validation metric (accuracy on the multi-class datasets, ROC-AUC on the binary Platonov datasets Minesweeper, Tolokers, Questions), and report mean $\pm$ 95% bootstrap CI across the ten splits. The full hyperparameter grids, the fair-64 capacity audit, the two-stage sweep protocol, the per-dataset selected configurations, and the bootstrap-CI computation are detailed in Appendix C.

Performance on homophilic benchmarks. CASTOR is the only model in Table 1 that ranks top-three on every dataset, achieving an average rank of 2.07 versus 5.64 for GAMMA and 5.93 for MixHop. On the homophilic graphs, CASTOR ranks third on Cora (88.24%, 0.23 pts behind H2GCN), second on CiteSeer (77.70%, 0.03 pts behind H2GCN), and first on PubMed (89.85%, 0.58 pts ahead of CO-GNN). The key result is the absence of regression: strong spectral methods (BernNet) and

concatenation-based heterophily models (H2GCN) remain competitive, yet CASTOR stays within the same accuracy band without changing its architecture between homophilic and heterophilic regimes. As shown in Appendix A.2, the Router automatically shifts its mass toward smoothing on homophilic graphs, recovering the low-pass behavior of SGC, APPNP, and BERNNET as a learned outcome rather than a fixed design choice (cf. Proposition 29).

Performance on small heterophilic benchmarks. On the six BernNet-protocol heterophilic graphs, CASTOR ranks first on Wisconsin (86.67%, ahead of GAMMA by 1.18 pts) and Cornell (81.67%, ahead of GAMMA by 1.93 pts), second on Chameleon (65.60%, 0.84 pts behind BernNet) and Actor (36.49%, 0.56 pts behind a feature-only MLP), and third on Texas (80.28%) and Squirrel (49.20%, 1.64 pts behind MixHop). Two patterns deserve emphasis. First, the regime split predicted in Section 2.2 matches the recipes the optimizer converges to: on Wisconsin, Cornell, and Texas (slow-mixing) the iteration accelerates ($\bar{\beta} > 0$), and on Chameleon and Squirrel (fast-mixing) it damps ($\bar{\beta} < 0$). The cumulative-trajectory probe of Appendix E (Figure 6) shows that the resulting trajectories sustain motion past the depth at which vanilla power-iteration on $\widehat{\mathbf{W}}$ has saturated, by a factor of 1.10–$2.77\times$ across this block, and that the sign of $\bar{\beta}$ is orthogonal to whether motion is sustained — the sign chooses the *direction*, not the *magnitude*. Second, the failure modes of the baselines are stark: standard GCN drops 25–30 accuracy points relative to a feature-only MLP on Texas and Wisconsin; CO-GNN collapses to 3.53% on Wisconsin and 20.00% on Texas; GCN-II is the strongest baseline on Questions but ranks below 50% on Cornell and Chameleon. CASTOR's top-three placement on every dataset of this block is the cleanest evidence that adaptive composition fixes both the over-smoothing failure of uniform low-pass propagation and the brittleness of architectures that hard-wire one heterophily-targeted prior.

Performance on Platonov mid-scale benchmarks. The five Platonov datasets are the most demanding in the table due to their larger scale ($N = 10\,000$ to $48\,921$) and strict $50/25/25$ splits that prevent hyperparameter spillover. CASTOR ranks first on Minesweeper (91.70% ROC-AUC, 0.55 pts above MixHop), second on Roman-empire (78.39% accuracy) and Amazon-ratings (47.41% accuracy), and third on Tolokers (80.08% ROC-AUC) and Questions (76.13% ROC-AUC), using accuracy for the multi-class datasets and ROC-AUC for the binary ones per Platonov et al. [2023]. All models run at a matched ~ 64 effective hidden dimension, eliminating width-driven advantages of JK-cat and multi-power architectures. The rank gap becomes sharper: GAMMA wins Roman-empire but ranks thirteenth on Tolokers, MixHop wins three datasets yet falls outside the top-three on all homophilic benchmarks, and GCN-II wins Questions but drops below 53% on Wisconsin, Texas, and Cornell. The largest ablation we observe — -31.6 ROC-AUC on Minesweeper when the residual primitive is removed (Appendix A.1, Figure 4) — explains CASTOR's strength on binary high-contrast tasks: the Router heavily favors the high-pass primitive, while Theorem 20 guarantees that a positive residual lower bound prevents Dirichlet-energy collapse under repeated propagation. Additional ≈ -6 pt drops on Minesweeper when removing momentum or constraining $\beta_k \geq 0$ show that sign-flexible momentum coupling is the second key mechanism on this dataset, further distinguishing CASTOR from ChebNet. Finally, on the de-duplicated Squirrel- and Chameleon-filtered benchmarks of Platonov et al. [2023], CASTOR ranks second behind FAGCN, with substantially smaller drops than duplicate-sensitive baselines (Appendix D).

6 Conclusion

We presented CASTOR, a state-dependent graph filter that replaces fixed propagation operators with per-node compositions of smoothing, residual, and identity primitives coupled through a sign-flexible second-order recurrence. Across fourteen benchmarks spanning both homophilic and heterophilic regimes, a single architecture ranks top-three on every dataset and achieves the best overall average rank while preserving the per-step complexity of standard fixed-operator GNNs. The results suggest that much of the apparent divide between low-pass and high-pass GNN designs stems from committing to a fixed propagation operator rather than adapting it during propagation itself. Several limitations remain. Our evaluation is restricted to transductive node classification on static, undirected graphs using the symmetric normalized adjacency operator $\hat{W}$, and we do not study inductive generalization, dynamic graphs, edge features, or large-scale distributed settings. Although the propagation cost per step matches standard message passing, total latency still grows linearly with the number of propagation steps K, which may limit scalability for very deep recurrences. In addition, the current formulation relies on lightweight scalar routing decisions shared across channels and does not explore richer channel-wise or attention-based propagation operators. Extending the framework to signed or directed graphs, graph-level tasks, temporal settings, and adaptive sparse propagation remains future work.

References

Sami Abu-El-Haija, Bryan Perozzi, Amol Kapoor, Nazanin Alipourfard, Kristina Lerman, Hrayr Harutyunyan, Greg Ver Steeg, and Aram Galstyan. Mixhop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *international conference on machine learning*, pages 21–29. PMLR, 2019.

Amir Ghazizadeh Ahsaei, Rickard Ewetz, and Hao Zheng. Gamma: Gated multi-hop message passing for homophily-agnostic node representation in gnns. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.

Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding oversquashing in GNNs through the lens of effective resistance. In *International Conference on Machine Learning*, pages 2528–2547. PMLR, 2023.

Deyu Bo, Xiao Wang, Chuan Shi, and Huawei Shen. Beyond low-frequency information in graph convolutional networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 3950–3957, 2021.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International conference on machine learning*, pages 1725–1735. PMLR, 2020.

Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. *arXiv preprint arXiv:2006.07988*, 2020.

Theodore S. Chihara. *An Introduction to Orthogonal Polynomials*. Gordon and Breach, 1978.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

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 Liò, 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*, pages 7865–7885. PMLR, 2023.

Lun Du, Xiaozhou Shi, Qiang Fu, Xiaojun Ma, Hengyu Liu, Shi Han, and Dongmei Zhang. GBK-GNN: Gated bi-kernel graph neural networks for modeling both homophily and heterophily. In *Proceedings of the ACM Web Conference 2022*, pages 1550–1558, 2022.

Matthias Fey and Jan Eric Lenssen. Fast graph representation learning with pytorch geometric. In *arXiv preprint arXiv:1903.02428*, 2019. doi: 10.48550/arXiv.1903.02428.

Ben Finkelshtein, Xingyue Huang, Michael Bronstein, and Ismail Ilkan Ceylan. Cooperative graph neural networks. *arXiv preprint arXiv:2310.01267*, 2023.

Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. *arXiv preprint arXiv:1810.05997*, 2018.

Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

Mingguo He, Zhewei Wei, Hongteng Xu, et al. Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. *Advances in Neural Information Processing Systems*, 34:14239–14251, 2021.

Mingguo He, Zhewei Wei, and Ji-Rong Wen. Convolutional neural networks on graphs with chebyshev approximation, revisited. *Advances in neural information processing systems*, 35: 7264–7276, 2022.

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.

Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. Graph-mlp: Node classification without message passing in graph. *arXiv preprint arXiv:2106.04051*, 2021.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. Finding global homophily in graph neural networks when meeting heterophily. In *International Conference on Machine Learning*, pages 13242–13256. PMLR, 2022.

Langzhang Liang, Sunwoo Kim, Kijung Shin, Zenglin Xu, Shirui Pan, and Yuan Qi. Sign is not a remedy: Multiset-to-multiset message passing for learning on heterophilic graphs. *arXiv preprint arXiv:2405.20652*, 2024.

Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Revisiting heterophily for graph neural networks. *Advances in neural information processing systems*, 35:1362–1375, 2022.

Sunil Kumar Maurya, Xin Liu, and Tsuyoshi Murata. Simplifying approach to node classification in graph neural networks. In *Journal of Computational Science*, volume 62, page 101695. Elsevier, 2022.

Hoang Nt and Takanori Maehara. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

NVIDIA Corporation. Nvidia a100 tensor core gpu architecture. Technical report, NVIDIA Corporation, 2020. URL `https://images.nvidia.com/aem-dam/en-zz/Solutions/data-center/nvidia-ampere-architecture-whitepaper.pdf`. Architecture Whitepaper.

NVIDIA Corporation. *NVIDIA Nsight Compute CLI (v2025.2.0) User Guide*, 2025. URL `https://docs.nvidia.com/nsight-compute/NsightComputeCli/index.html`. Version 2025.2.0; accessed May 2025.

Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

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.

Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

T. Konstantin Rusch, Ben Chamberlain, James Rowbottom, Siddhartha Mishra, and Michael Bronstein. Graph-coupled oscillator networks. In *International Conference on Machine Learning*, pages 18888–18909. PMLR, 2022.

Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *IJCAI*, 2021.

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.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

Xiyuan Wang and Muhan Zhang. How powerful are spectral graph neural networks. In *International Conference on Machine Learning*, pages 23341–23362. PMLR, 2022.

Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. Pmlr, 2019.

Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *International conference on machine learning*, pages 5453–5462. PMLR, 2018.

Xin Zheng, Yi Wang, Yixin Liu, Ming Li, Miao Zhang, Di Jin, Philip S Yu, and Shirui Pan. Graph neural networks for graphs with heterophily: A survey. *arXiv preprint arXiv:2202.07082*, 2022.

Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, 33:7793–7804, 2020.

Jiong Zhu, Ryan A Rossi, Anup Rao, Tung Mai, Nedim Lipka, Nesreen K Ahmed, and Danai Koutra. Graph neural networks with heterophily. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 11168–11176, 2021.

Appendix Contents

A Anatomy of the CASTOR Block: Ablations and Per-Node Routing

This section opens the CASTOR block from two complementary directions. The ablation in Section A.1 disables one architectural ingredient at a time and measures the resulting change in test metric across six datasets, identifying which design decisions are load-bearing on which graph regimes. The routing snapshot in Section A.2 reads the per-node mixing weights $\mathbf{a}_k(v)$ off a trained model on Chameleon, confirming that the per-node Routing decision argued in Section 2.1 is realized in practice rather than collapsed at training time.

A.1 Ablation: which design choices are load-bearing?

Protocol. We disable one architectural ingredient at a time and re-train under the per-dataset best recipe used for the main-paper table (Section 5); all other hyperparameters are held fixed. The reported quantity is the mean test metric across the ten official splits (accuracy on the multi-class datasets, ROC-AUC on Minesweeper). The ablations are:

- **A1 NoRouter** — replace the per-node Router by a single shared scalar over the simplex (uniform routing $\mathbf{a}_k(v) \equiv \boldsymbol{\alpha}_k$ for all v). Tests whether per-node routing exits the uniform spectral-filter family $\mathrm{USF}_K(\widehat{\mathbf{W}})$ in practice (Definition 2).
- **A2 NoAnchor** — drop the initial state $\mathbf{Z}^{(0)}$ from the Router input, leaving only the smoothing residual $\boldsymbol{\Delta}\mathbf{Z}$. Tests the role of the stable reference of Section 4.1, property (ii).
- **A3a NoSmooth** — mask the smoothing logit before the softmax of Eq. (3), forcing $\alpha^{\mathrm{sm}}_{v,k} \equiv 0$ at every node and every hop. The Router is constrained to the residual–identity edge of the simplex.
- **A3b NoResid** — mask the residual logit, forcing $\alpha^{\mathrm{hi}}_{v,k} \equiv 0$.
- **A3c NoIdentity** — mask the identity logit, forcing $\alpha^{\mathrm{id}}_{v,k} \equiv 0$.
- **A4 NoBiasPrior** — replace the smoothing-then-preserve initialization of $\mathbf{b}_k$ (Eq. (5)) with a zero initialization; the Router still has the same parameter count, only the prior changes.
- **A5 NoMomentum** — set $\beta_k \equiv 0$ at every step, collapsing Eq. (6) to the first-order recurrence.
- **A6 PosBeta** — constrain $\beta_k \geq 0$ via a sigmoid reparameterization, removing the damping regime $\beta_k < 0$.
- **A7 SharedBeta** — tie β_k across hops to a single scalar β, removing the per-hop schedule freedom that Theorem 5 contributes to the reachable polynomial set.

The six datasets cover the regimes of Section 2: PubMed (homophilic), Cornell (small slow-mixing heterophilic, $N = 183$), Squirrel and Chameleon (dense fast-mixing heterophilic), Actor (sparse near-random heterophilic), and Minesweeper (Platonov binary, $C = 2$, AUC metric).

Five observations follow from Figure 4.

(i) The residual primitive carries the heaviest load on dense binary heterophilic graphs. Removing the residual (**A3b NoResid**) drops -31.6 AUC pts on Minesweeper — the largest single ablation effect in the entire grid, and an order of magnitude larger than the next biggest drop on the same row. On a dense binary graph where the discriminative signal is precisely *disagreement with the local average*, the high-pass primitive $\boldsymbol{\Delta}\mathbf{Z} = (\mathbf{I} - \widehat{\mathbf{W}})\mathbf{Z}$ is the operation CASTOR relies on most heavily, and no convex combination of smoothing and identity can substitute for it. The same ablation costs roughly 2 pts on the dense heterophilic graphs (Squirrel, Chameleon) and is essentially neutral on the slow-mixing graph (Actor).

(ii) Smoothing is load-bearing on both ends of the homophily spectrum. Removing smoothing (**A3a NoSmooth**) drops -12.7 AUC pts on Minesweeper and -2.7 pts on PubMed, the only two datasets on which it produces a clear drop. On PubMed the smoothing primitive is doing what the standard low-pass story predicts. On Minesweeper, the Router is forced to choose at every node and every hop between residual and identity only; the residual's heavy weight (Section A.2) makes the high-pass branch the dominant contributor, but losing the low-pass primitive removes the only operation that can reduce the residual's amplitude on modes where the neighborhood happens to agree, and the model loses calibration on those nodes. On the dense heterophilic graphs

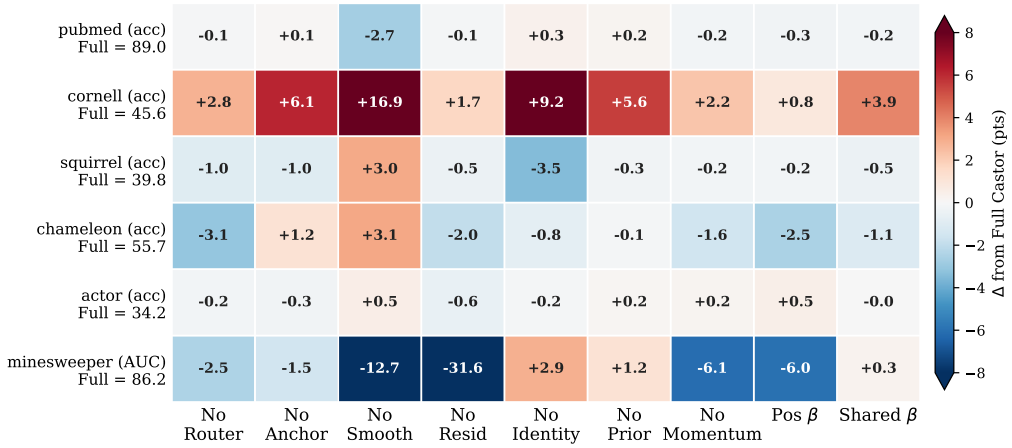

Figure 4: **Ablation heatmap.** Each cell reports Δ = ablation − Full, the change in test metric (accuracy in pts on the multi-class rows, AUC in pts on Minesweeper) when one architectural ingredient is disabled. Rows are annotated with the dataset's Full-configuration test metric. Blue cells are drops — the disabled ingredient was load-bearing on that dataset; red cells are improvements — the ingredient was redundant or actively harmful. The color scale is clipped at ± 8 pts; the -31.6 entry on (Minesweeper, NoResid) extends past the scale and is signaled by the color-bar arrow.

(Squirrel, Chameleon) removing smoothing is mildly *helpful* ($+3.0$, $+3.1$ pts), consistent with the over-smoothing risk of repeated $\widehat{\mathbf{W}}$-iteration on fast-mixing graphs (Section 2.2).

(iii) The momentum coupling and its sign-flexibility are load-bearing on the fast-mixing regime. Removing momentum entirely (**A5 NoMomentum**, -6.1) and forcing $\beta_k \geq 0$ (**A6 PosBeta**, -6.0) drop nearly identical amounts on Minesweeper. The match is consistent with the dynamics-side argument of Section 2.2: Minesweeper sits in the fast-mixing regime where the appropriate β_k is negative (damping), and either disabling momentum or constraining it to non-negative values forfeits the same regime. On Chameleon, the per-hop schedule freedom is similarly load-bearing: **A6 PosBeta** drops -2.5 pts and **A7 SharedBeta** drops -1.1 pts, indicating that not just the existence of momentum but its sign and per-hop schedule matter on dense heterophilic data.

(iv) The Router is most useful on dense heterophilic graphs. Disabling per-node routing (**A1 NoRouter**) drops -3.1 on Chameleon, -2.5 on Minesweeper, and -1.0 on Squirrel, while leaving PubMed and Actor essentially unchanged. The pattern matches the per-class hop-fingerprint argument of Section 2.1: the Router exits the uniform-spectral-filter family USF_K exactly where neighboring nodes prefer different primitive mixtures. On homophilic and near-random graphs, the uniform-routing baseline already approximates the right schedule and the per-node decision adds little.

(v) Cornell is an outlier and not a signal. Every ablation *improves* Cornell. With $N = 183$ nodes and only five classes, the Full CASTOR configuration over-parameterizes this graph; on it, removing any architectural ingredient acts as an effective regularizer. We exclude Cornell from any ranked summary of ablation effects and read it only as a reminder that part of the small-graph end of the benchmark sits below the regime in which the ablation grid is informative. The ranked mean drop across the remaining five datasets is, in decreasing magnitude, NoResid (-7.0), NoSmooth (-1.8), PosBeta (-1.7), NoMomentum (-1.6), NoRouter (-1.4), NoAnchor (-0.3), SharedBeta (-0.3), NoIdentity (-0.3), NoBiasPrior ($+0.2$).

Take-away. The two design decisions of Section 4 — per-node Routing over the three primitives and a sign-flexible per-hop momentum coefficient — are both load-bearing, and the dataset on which each is most critical is the one predicted by the corresponding axis of Section 2. The single largest ablation effect (-31.6 on Minesweeper–NoResid) confirms that the high-pass primitive is the operation that distinguishes CASTOR from a conventional low-pass GNN on dense binary heterophilic data; the secondary block of ≈ -6 pt drops on Minesweeper (NoMomentum, PosBeta) confirms

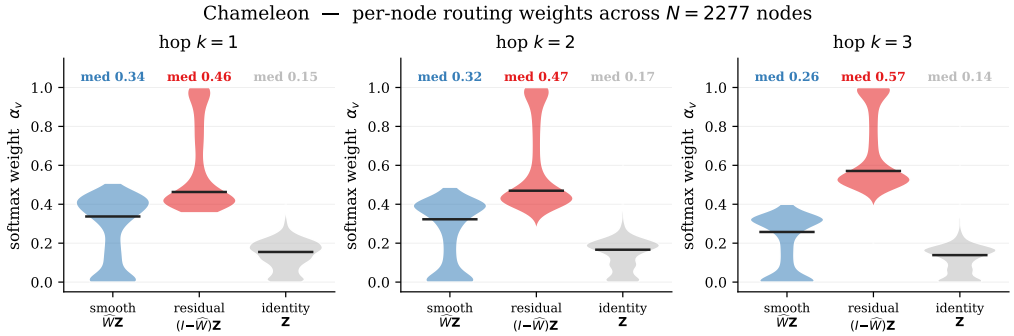

Figure 5: **Per-node routing weights on Chameleon at every hop.** Each violin shows the empirical distribution of one primitive's softmax weight across the $N = 2{,}277$ nodes; black bar is the per-primitive median. Smoothing weight (blue) decreases monotonically across hops, residual weight (red) increases, and identity weight (gray) stays low and approximately stationary. Within every (hop, primitive) pair, the per-node spread is wide: different nodes assign markedly different weights to the same primitive in the same forward pass.

that the momentum coupling is the second mechanism the same dataset relies on. Only one ablation is consistently neutral or positive (NoBiasPrior, $+0.2$), consistent with the description of $\mathbf{b}_k$ in Section 4.1, property (iii) as a starting point that gradient descent overwrites whenever the data prefer otherwise.

A.2 Per-node routing on Chameleon: opening the Router

Protocol. We train a single CASTOR instance on Chameleon ($N = 2{,}277$, $C = 5$, $K = 3$) under the per-dataset best recipe (Section 5) and read off the per-node mixing weights $\mathbf{a}_k(v) \in \Delta_3$ from the trained Router at every hop. The Router is the deterministic forward pass of Eq. (3) on the full graph; no Monte-Carlo sampling is required. We visualize the marginal distribution of each primitive's softmax weight, $\{\alpha^{\text{sm}}_{v,k}\}_{v\in\mathcal{V}}$, $\{\alpha^{\text{hi}}_{v,k}\}_{v\in\mathcal{V}}$, and $\{\alpha^{\text{id}}_{v,k}\}_{v\in\mathcal{V}}$, at every hop $k \in \{1, 2, 3\}$.

Three observations follow from Figure 5.

(i) The Router does not collapse to a single primitive. At every hop, every primitive receives meaningful weight on a substantial fraction of the nodes. The marginal supports at hop $k = 1$ are $\alpha^{\text{sm}} \in [0.00, 0.51]$, $\alpha^{\text{hi}} \in [0.36, 1.00]$, $\alpha^{\text{id}} \in [0.00, 0.51]$; at hop $k = 3$ the supports are $[0.00, 0.40]$, $[0.15, 1.00]$, $[0.00, 0.85]$ respectively. The trained model is genuinely operating inside the interior of the simplex, not at one of its vertices. This rules out the failure mode in which the per-node Router silently collapses to a uniform schedule and reduces to the **A1 NoRouter** baseline of Section A.1.

(ii) Per-node spread is large within every (hop, primitive) pair. The interquartile range of the residual weight at hop $k = 1$ is $[0.41, 0.79]$ — a span of nearly 0.4 in softmax probability across the $2{,}277$ nodes. Different nodes choose different mixtures within the *same* forward pass, exactly the empirical behavior predicted by Section 2.1: the per-node Routing decision is not a parameter-saving substitute for a node-uniform schedule, it is a distinct function class of the kind required by the Roman-empire fingerprints argument. Two nodes whose state has been mixed by the same global $\widehat{\mathbf{W}}$ for the same number of hops can leave hop $k = 1$ with residual weights 0.41 and 0.79, a difference that no fixed-coefficient polynomial filter can express.

(iii) The schedule the Router discovers is "smooth-early, contrast-late", not "smooth-then-preserve". The median weights track α^{sm} from $0.34 \to 0.32 \to 0.26$ across hops and α^{hi} from $0.46 \to 0.47 \to 0.57$, while the identity median stays approximately flat ($0.16 \to 0.17 \to 0.14$). The smoothing-then-preserve initialization of Eq. (5) would produce smoothing-early-and-identity-late behavior; the trained Router overwrites this prior and discovers that on Chameleon, late hops should be devoted to the high-pass primitive, not to the identity. The bias prior is gone-by-convergence rather than load-bearing-at-test-time, which dovetails with the small near-zero column of the ablation

heatmap for **A4 NoBiasPrior**: removing the prior costs nothing precisely because the trained Router has already replaced it. Reading the schedule node-wise sharpens the picture: at hop $k=1$, 24% of nodes have their largest routing weight on smoothing and 75% on residual; at hop $k=2$, the split is $13\%/87\%$; at hop $k=3$, every single node has its largest weight on the residual primitive — a per-node convergence to high-pass that cannot be expressed by any fixed schedule of $\mathbf{b}_k$ alone.

Take-away. Per-node routing on a trained CASTOR block is non-trivial, non-collapsed, and adapts across hops in a direction the smoothing-then-preserve prior does not predict. Combined with the ablation evidence in Section A.1 that disabling per-node routing costs 1–3 accuracy points on dense heterophilic graphs, this confirms that the architectural mechanism of Section 4.1 contributes function-class enlargement that the model uses at test time, rather than parameters that the optimizer leaves dormant.

B Computational Cost and Parameter Comparison

The CASTOR block is comparable in cost to a single-layer GCN per propagation step, with memory that does not grow with the depth K and a parameter count dominated by the Router rather than by per-hop weight matrices. We make this precise by breaking down per-step cost, propagation memory, and parameter count, and we tabulate the same quantities for the baselines of Section 5.

Per-step cost. Each step of the CASTOR block (Eq. (6)) requires:

- One sparse matrix-vector product $\widehat{\mathbf{W}}\,\mathbf{Z}^{[k-1]}$ at cost $\mathcal{O}(|\mathcal{E}|\,C)$. The smoothing residual $\mathbf{\Delta Z} = \mathbf{Z}^{[k-1]} - \widehat{\mathbf{W}}\mathbf{Z}^{[k-1]}$ is recovered by a single pointwise subtraction, so a single $\widehat{\mathbf{W}}$-application supplies both the smoothing and the high-pass branch.
- One forward pass through the Router ρ at cost $\mathcal{O}(N\,C\,H_\rho)$, where H_ρ is the Router's hidden width.
- A pointwise convex combination of three $N \times C$ matrices: $3NC$ multiply-accumulates and $2NC$ pointwise additions, broadcast across the C class channels by the rank-one Routing weights of Section 4.1.
- A velocity update $\beta_k\,(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]})$: $2NC$ pointwise operations.

Across K steps the total cost is $\mathcal{O}(K\,(|\mathcal{E}|\,C + N\,C\,H_\rho))$, dominated by the K sparse matvecs and the K Router passes; the per-step cost is constant in K.

Memory. The block keeps two consecutive states in memory, $\mathbf{Z}^{[k-1]}$ and $\mathbf{Z}^{[k-2]}$, both of shape $N \times C$. There are no per-hop activation buffers, no per-step weight matrices, and no concatenation across hops; the only mid-step buffer is the Router's hidden activation $\mathrm{ReLU}(\mathbf{W}_{\rho,1}[\mathbf{\Delta Z}\|\mathbf{Z}^{(0)}])$ of size $\mathcal{O}(N\,H_\rho)$ during training. The block's propagation memory is therefore $\mathcal{O}(NC + N\,H_\rho)$, *independent of* K, which distinguishes CASTOR from concatenation-based heterophily methods (H2GCN, MIXHOP, JK-NET) whose propagation memory grows with the number of hops aggregated.

Parameter count. The propagation block adds $\mathcal{O}(C\,H_\rho + K)$ parameters in total:

Router ρ :	$2C\,H_\rho$ (input projection) $+\,3\,H_\rho$ (output projection) $+\,3 + H_\rho$ (biases)	$= \mathcal{O}(C\,H_\rho)$,
Per-step biases $\{\mathbf{b}_k\}_{k=1}^{K}$:	$3K$,	
Per-step momentum $\{\beta_k\}_{k=1}^{K}$:	K.	

Adding the encoder of Section 4.3 (an $F \to D \to C$ MLP with $\mathcal{O}(FD + DC)$ parameters), the full pipeline has parameter count $\mathcal{O}(FD + DC + C\,H_\rho + K)$. This is the same order as a two-layer GCN at the same hidden width, and is independent of K in the propagation block itself.

Practical implications. Three points summarize Table 2.

(i) Constant propagation memory. Among architectures that propagate at depth K, CASTOR, SGC, and CHEBNET are the only ones whose propagation memory does *not* grow with K *(*all three keep a constant number of $N \times C$ buffers*)*. This is the property that lets CASTOR run at $K=10$ on the larger Platonov benchmarks of Section 5 without exceeding the fair-~ 64 effective hidden-dim budget;

Table 2: Per-forward-pass cost and propagation-block parameter count for the architectures of Section 5. *Sparse matvecs* is the number of $\widehat{\mathbf{W}}$- (or $\mathbf{L}$-) applications to the state per forward pass. *Propagation memory* is the size of buffers held during the propagation, excluding the input encoder and the final classifier. *Propagation params* is the number of learnable scalars inside the propagation block, again excluding encoder and classifier. *Function class* states the family of operators that the architecture can realize. L is the number of GNN layers, K the propagation depth, C the propagation-state dimension, H the number of attention heads (where applicable), and H_ρ the Router's hidden width. Fixed-coefficient polynomial baselines all sit in the uniform spectral filter family $\mathrm{USF}_K(\widehat{\mathbf{W}})$ *(*Definition 2*)*; CASTOR sits in the strictly larger node-conditional family $\mathrm{NCSF}_K(\widehat{\mathbf{W}})$ *(*Theorem 7*)*.

Method	Sparse matvecs	Propagation memory	Propagation params	Function class
MLP [Hu et al., 2021]	0	$\mathcal{O}(NC)$	0	no graph dependence
GCN [Kipf and Welling, 2016]	L	$\mathcal{O}(NC\,L)$	$\mathcal{O}(C^2\,L)$	USF_L with per-layer weights
GAT [Velickovic et al., 2017]	L	$\mathcal{O}(NC\,L\,H)$	$\mathcal{O}(C^2\,L\,H)$	graph-attention shifts (not in NCSF)
SGC [Wu et al., 2019]	K	$\mathcal{O}(NC)$	0	$\widehat{\mathbf{W}}^K$, single point in USF_K
GCN-II [Chen et al., 2020]	L	$\mathcal{O}(NC\,L)$	$\mathcal{O}(C^2\,L)$	USF_L with residual + initial
CHEBNET [Defferrard et al., 2016]	K	$\mathcal{O}(NC)$	$\mathcal{O}(C^2\,K)$	three-term recurrence with $\beta \equiv -1$
BERNNET [He et al., 2021]	$\mathcal{O}(K)$	$\mathcal{O}(NC\,K)$	$K{+}1$	USF_K in the Bernstein basis
GPR-GNN [Chien et al., 2020]	K	$\mathcal{O}(NC\,K)$	$K{+}1$	USF_K in the monomial basis
H2GCN [Zhu et al., 2020]	$\mathcal{O}(K)$	$\mathcal{O}(NC\,K)$	$\mathcal{O}(C^2)$	multi-hop ego/neighbor concat. (not in USF)
MIXHOP [Abu-El-Haija et al., 2019]	$L\,K$	$\mathcal{O}(NC\,L\,K)$	$\mathcal{O}(C^2\,L\,K)$	per-layer multi-power concat.
JK-NET [Xu et al., 2018]	L	$\mathcal{O}(NC\,L)$	$\mathcal{O}(C^2\,L)$	layer-wise concat./pool. of $\{\mathbf{Z}^{[\ell]}\}$
CASTOR (ours)	K	$\mathcal{O}(NC + N\,H_\rho)$	$\mathcal{O}(C\,H_\rho + K)$	NCSF_K via per-node Routing

H2GCN hits OOM on Questions ($N = 48{,}921$) at the same budget for exactly the opposite reason *(*depth-K ego/neighbor concatenation grows the per-node feature dimension by a factor proportional to K*)*.

(ii) No per-hop weight matrices. The propagation block of CASTOR contains no $C \times C$ weight matrices: the only learnable map applied to the state at every step is the convex combination by $\mathbf{a}_{v,k} \in \Delta_3$. By contrast, every per-layer-weight architecture *(*GCN, GAT, GCN-II, CHEBNET, MIXHOP, JK-NET*)* pays $\mathcal{O}(C^2)$ propagation parameters per layer or per polynomial degree, which scales with depth. CASTOR's entire depth-K block adds only $\mathcal{O}(C\,H_\rho + K)$ propagation parameters, independent of C^2 and of L.

(iii) Strictly larger reachable function class at lower or equal cost. The fixed-coefficient polynomial baselines *(*SGC, GPR-GNN, BERNNET, CHEBNET*)* sit in $\mathrm{USF}_K(\widehat{\mathbf{W}})$, the family of operators expressible as a single polynomial in $\widehat{\mathbf{W}}$ shared across nodes. CASTOR, via its per-node Routing decision, sits in the strict superset $\mathrm{NCSF}_K(\widehat{\mathbf{W}}) \supsetneq \mathrm{USF}_K(\widehat{\mathbf{W}})$ *(*Theorem 7, Appendix F.2*)* while paying the same K sparse matvecs per forward pass and the same constant-in-K memory as the cheapest of the polynomial baselines *(*SGC*)*. In other words, the architecture pays the cost of a fixed-coefficient polynomial filter and reaches a strictly larger function class.

C Training Recipe and Reproducibility

This appendix specifies the experimental protocol summarized in Section 5 in enough detail to make every entry of Table 1 reproducible. We describe in turn the datasets and splits, the fair-capacity audit applied to every architecture, the hyperparameter grids searched, the training and early-stopping protocol, and the reporting protocol. Per-dataset selected configurations of CASTOR on the de-duplicated benchmarks of Platonov et al. [2023] are tabulated separately in Appendix D.3; selected configurations for the fourteen main-paper datasets are released alongside the code.

C.1 Datasets and splits

Three split protocols are used across the benchmark suite of Table 1, and we apply each to the dataset blocks for which it is the standard reference protocol:

- **Cora, CiteSeer, PubMed** (homophilic). The fixed $10 \times 48/32/20$ splits originally introduced in Zhu et al. [2020] and reused as the standard evaluation protocol by He et al. [2021].

- **Texas, Wisconsin, Cornell, Actor, Squirrel, Chameleon** (small heterophilic). The same $10\times 48/32/20$ split protocol of Zhu et al. [2020], applied to the WebKB and WikipediaNetwork datasets of Pei et al. [2020].
- **Roman-empire, Amazon-ratings, Minesweeper, Tolokers, Questions** (Platonov mid-scale). The official $10\times 50/25/25$ splits of Platonov et al. [2023], distributed in their public repository. The metric is accuracy on the multi-class datasets and ROC-AUC on the binary datasets (Minesweeper, Tolokers, Questions), as in the original benchmark.
- **Squirrel-filtered, Chameleon-filtered** (Appendix D). The same Platonov $10\times 50/25/25$ protocol applied to the de-duplicated versions of Squirrel and Chameleon released by Platonov et al. [2023].

For the small/homophilic graphs we use the cached fixed splits to ensure that every architecture is evaluated on the identical ten partitions; the Platonov splits are read directly from the official files. No data augmentation, feature normalization, or graph re-construction is applied beyond what each dataset's loader returns.

C.2 Common backbone and fair-capacity audit

Every architecture in Table 1 is restricted to a common two-layer backbone with a fixed hidden dimension. To prevent capacity inflation across architectures of very different parametric structure, on the five Platonov datasets every architecture is run at *fair* ~ 64 *effective hidden dimension*: any architecture whose forward pass concatenates or splits across K streams (multi-power, multi-head, multi-hop concatenation, JK-cat) has its per-stream width divided by the aggregation count, so that the total effective per-node feature dimension into the classifier head is approximately 64. Concretely:

- Single-stream architectures (MLP, GCN, SGC, GCN-II, BERNNET, GPR-GNN, M2MGNN, CO-GNN, GAMMA): hidden dimension 64.
- MIXHOP with three powers $\{0, 1, 2\}$: per-power hidden dimension ≈ 22, concatenated to total ≈ 64.
- GAT with eight attention heads: per-head hidden dimension 8, concatenated to total 64.
- GCN-JK, GAT-JK: per-layer width divided by the number of layers concatenated by the Jumping-Knowledge skip.
- H2GCN with depth $K=2$: four-stream ego/neighbor concatenation, per-stream width ≈ 16, total ≈ 64.
- CASTOR: encoder hidden $D=64$ (Eq. (8)); propagation state dimension C equal to the number of classes, since $\widehat{\mathbf{W}}$ acts directly on class-logit space (Section 4).

For the nine small/homophilic graphs we use the per-architecture hidden dimension reported as best by each baseline's reference implementation, which is the standard regime under He et al. [2021] and Zhu et al. [2020]; this is the regime the architectures were originally tuned in, so a fair comparison there is best served by leaving each baseline at its tuned width.

C.3 Hyperparameter grids

For each (architecture, dataset) pair we sweep a small grid of standard regularizers, with the same ranges applied to every architecture so the comparison is on equal footing:

- **Learning rate**: $\{10^{-3},\ 5\times 10^{-3},\ 10^{-2},\ 5\times 10^{-2}\}$.
- **Weight decay**: $\{0,\ 5\times 10^{-4}\}$.
- **Pre-input dropout** (Drop_p): $\{0.0,\ 0.3,\ 0.5,\ 0.7\}$.
- **Pre-classifier / hidden dropout** (Drop_h): $\{0.0,\ 0.3,\ 0.5,\ 0.7,\ 0.8\}$.

CASTOR-specific hyperparameters that are also swept:

- **Propagation depth** $K \in \{2,\ 3,\ 5,\ 8,\ 10,\ 12\}$.
- **Hop-bias initialization** $\in \{$ *smoothing-then-preserve (*Eq. (5)*)*, zero $\}$.
- **Momentum parameterization**: shared scalar or per-hop scalar, both sign-flexible (constrained-β via sigmoid is also a sweep option).

- **Momentum initialization** $\beta_k^{(0)} \in \{0, \pm 0.3, \pm 0.5, \pm 0.7\}$, with the sign-flexible parameterization allowing the optimizer to flip sign during training.
- **Router hidden width** $H_\rho = 64$, fixed across all datasets.

The sweep is implemented as a coarse Stage 1 over all candidate configurations on three of the ten official splits per dataset, followed by a Stage 2 re-evaluation of the seven top-Stage-1 configurations on all ten splits with the bootstrap protocol described in Section C.5. Stage-2 selection is by mean validation metric (accuracy on the multi-class datasets, ROC-AUC on the binary Platonov datasets) across the ten splits, with ties broken by lower validation-loss variance.

C.4 Training and early stopping

All architectures are trained with the AdamW optimizer, except where the original baseline implementation specifies a different optimizer in which case we keep the original choice (e.g., BERNNET uses Adam with separate learning rates for the polynomial coefficients and the classifier). The training objective is cross-entropy on the labelled training nodes, with binary cross-entropy on the binary Platonov datasets (Minesweeper, Tolokers, Questions) where ROC-AUC is the evaluation metric. Each (architecture, dataset, split) configuration is trained for up to $1{,}000$ epochs with early stopping on the per-dataset validation metric — accuracy on the multi-class datasets and ROC-AUC on the binary Platonov datasets (Minesweeper, Tolokers, Questions): training halts when the best validation metric has not improved for 200 consecutive epochs, and we report the test metric at the epoch with the highest validation metric. No learning-rate scheduling, label smoothing, or gradient-clipping is applied. For CASTOR specifically, the Router parameters and the per-step momentum coefficients are placed in a separate parameter group with an independent learning rate; the gating/encoder parameters use the main learning rate.

C.5 Reporting protocol

For each (architecture, dataset) pair we train once per split and aggregate the ten test scores into a mean and a non-parametric 95% bootstrap confidence interval over $1{,}000$ bootstrap resamples; this is the protocol of Platonov et al. [2023]. The reported number in Table 1 is the mean, with the bootstrap-CI half-width as the subscript. On the de-duplicated benchmarks of Appendix D the same protocol is applied. The average-rank statistic in the bottom row of Table 1 is the mean over the fourteen datasets of each architecture's per-dataset rank, with ties broken by mean test metric (accuracy or ROC-AUC, as appropriate for the dataset) and OOM treated as the worst rank *(*rank 15 in the fifteen-architecture comparison*)*.

D Evaluation on the De-duplicated Heterophilic Benchmarks

The original Squirrel and Chameleon benchmarks of Pei et al. [2020], which sit at homophily ratios 0.22 and 0.23 in our main heterophilic results, were shown by Platonov et al. [2023] to contain a substantial fraction of train–test *node duplicates*: node pairs whose features are identical and whose label distribution is therefore correlated across splits. Platonov et al. [2023] re-released a de-duplicated version of each dataset (SQUIRREL-FILTERED, $N = 2{,}223$, and CHAMELEON-FILTERED, $N = 890$, both five-class) and re-ran seventeen reference architectures on the same official $50/25/25$ split protocol. The reported drops between the original and filtered datasets are uneven across architectures: several heterophily-targeted methods (FSGNN, GLOGNN, RESNET+ADJ, JACOBICONV) lose between 25 and 45 accuracy percentage points, while simpler baselines (GCN, GAT, GRAPHSAGE) lose only a few. Platonov et al. [2023] interpret this as evidence that part of the apparent advantage of heterophily-specialized architectures on the contaminated datasets reflects sensitivity to the duplicates rather than to heterophilic structure.

Motivation for evaluating CASTOR on the filtered benchmarks. The two-axis claim of Section 2 predicts that CASTOR's gains derive from per-node, per-step composition of three spectrally interpretable primitives and a sign-flexible momentum coupling, none of which has a direct dependence on duplicate-node features. The filtered benchmarks of Platonov et al. [2023] therefore provide a controlled setting in which to assess whether CASTOR's improvements on the original Squirrel and Chameleon datasets are explained by the proposed mechanism or by sensitivity to the duplicates. We

Table 3: Node classification accuracy (%) on the de-duplicated CHAMELEON-FILTERED and SQUIRREL-FILTERED datasets of Platonov et al. [2023], official $10\times50/25/25$ split protocol. Baselines are taken verbatim from Platonov et al. [2023, Table 2]; CASTOR variants are run by us on the identical splits and metric. **Green** = 1st best, **Blue** = 2nd best, **Red** = 3rd best per column.

Method	**Chameleon-filtered**	**Squirrel-filtered**
# Nodes / # Classes	890 / 5	2,223 / 5
ResNet	$36.73_{\pm4.71}$	$36.55_{\pm1.82}$
ResNet+SGC	$41.01_{\pm4.54}$	$38.36_{\pm1.97}$
ResNet+adj	$38.67_{\pm3.87}$	$38.37_{\pm1.99}$
GCN [Kipf and Welling, 2016]	$40.89_{\pm4.12}$	$39.47_{\pm1.47}$
GRAPHSAGE [Hamilton et al., 2017]	$37.77_{\pm4.14}$	$36.09_{\pm1.99}$
GAT [Velickovic et al., 2017]	$39.21_{\pm3.08}$	$35.62_{\pm2.06}$
GAT-sep	$39.26_{\pm2.50}$	$35.46_{\pm3.10}$
GT [Shi et al., 2021]	$38.87_{\pm3.66}$	$36.30_{\pm1.98}$
GT-sep	$40.31_{\pm3.01}$	$36.66_{\pm1.63}$
H2GCN [Zhu et al., 2020]	$26.75_{\pm3.64}$	$35.10_{\pm1.15}$
CPGNN [Zhu et al., 2021]	$33.00_{\pm3.15}$	$30.04_{\pm2.03}$
GPR-GNN [Chien et al., 2020]	$39.93_{\pm3.30}$	$38.95_{\pm1.99}$
FSGNN [Maurya et al., 2022]	$40.61_{\pm2.97}$	$35.92_{\pm1.32}$
GLOGNN [Li et al., 2022]	$25.90_{\pm3.58}$	$35.11_{\pm1.24}$
FAGCN [Bo et al., 2021]	$\mathbf{41.90_{\pm2.72}}$	$\mathbf{41.08_{\pm2.27}}$
GBKGNN [Du et al., 2022]	$39.61_{\pm2.60}$	$35.51_{\pm1.65}$
JACOBICONV [Wang and Zhang, 2022]	$39.00_{\pm4.20}$	$29.71_{\pm1.66}$
CASTOR (ours)	$\mathbf{41.71_{\pm2.04}}$	$\mathbf{40.43_{\pm1.36}}$

report results on both filtered datasets in Section D.1, and analyze the magnitude of the drop relative to the original datasets in Section D.2.

Protocol. We use the official $10\times50/25/25$ splits and the standard accuracy metric exactly as in Platonov et al. [2023], accessed through the public files in their accompanying repository. The seventeen baseline numbers we report in Table 3 are taken verbatim from Platonov et al. [2023, Table 2]; each was produced under their per-architecture tuning protocol, so they represent a strong, well-tuned reference set. CASTOR is run on the identical splits and metric on this paper's hardware, with hyperparameters selected by best validation accuracy on a small per-dataset grid (Section D.3). All comparisons are therefore on the same data and the same evaluation protocol; the only quantity that differs between rows is the architecture and its tuned hyperparameters.

Protocol verification. To rule out a hidden protocol mismatch between our CASTOR runs and the reference numbers we cite from Platonov et al. [2023, Table 2], we re-ran the unmodified official harness of Platonov et al. [2023] on this paper's hardware, end-to-end on both filtered datasets, for the GCN architecture. We used the harness's default hyperparameters (5 layers, hidden dimension 512, dropout 0.2, AdamW with $\mathrm{lr}=3\times10^{-5}$, 1,000 optimization steps, LayerNorm, ten runs across the ten official splits) without any modification. The recovered numbers are GCN on Chameleon-filtered 41.79 ± 3.27 (published: 40.89 ± 4.12, mean shift $+0.90$, within one standard deviation), and GCN on Squirrel-filtered 39.94 ± 2.09 (published: 39.47 ± 1.47, mean shift $+0.47$, within one standard deviation). Both reproduction shifts are well below either confidence interval and are consistent with random-seed and hardware variation between the original run and ours, confirming that our protocol and theirs are identical at the granularity that matters for the comparison.

D.1 Main result

Table 3 reports test accuracy on the two filtered benchmarks for all seventeen reference architectures and our two CASTOR variants.

CASTOR places second on both filtered datasets, behind FAGCN on each. The gaps are small: 0.19 percentage points on Chameleon-filtered and 0.65 on Squirrel-filtered, both well within the

Table 4: Change in accuracy (%) between the original and de-duplicated benchmarks, Δ = filtered − original. Original-dataset numbers are taken from Platonov et al. [2023, Table 2], with the exception of CASTOR, which is taken from Table 1 of the main paper. The largest drops are highlighted in red; the smallest in green.

	Chameleon			Squirrel		
Method	original	filtered	Δ	original	filtered	Δ
FSGNN	77.85	40.61	−37.24	68.93	35.92	−33.01
GLOGNN	70.04	25.90	−44.14	61.21	35.11	−26.10
ResNet+adj	71.07	38.67	−32.40	65.46	38.37	−27.09
JACOBICONV	68.33	39.00	−29.33	46.17	29.71	−16.46
FAGCN	64.23	41.90	−22.33	47.63	41.08	−6.55
GCN	50.18	40.89	−9.29	39.06	39.47	+0.41
GAT	45.02	39.21	−5.81	32.21	35.62	+3.41
CASTOR (ours)	65.60	41.71	−23.89	49.20	40.43	−8.77

Table 5: Selected CASTOR configurations on the filtered datasets. Common across both rows: pre-input dropout 0.3, AdamW optimizer with learning rate 10^{-3} and zero weight decay, LayerNorm applied to the propagation output, two-layer feedforward encoder of width D, and a final linear classifier $\mathrm{Linear}(D, C)$. The reported β_k value is the initialization; the optimizer adapts it during training, separately per hop in the per-hop variant. K denotes the number of propagation steps in one CASTOR block.

Dataset	D	K	Drop_h	momentum	β_k init	accuracy (%)
Chameleon-filtered	128	3	0.8	shared, sign-flexible	−0.4 (damping)	$\mathbf{41.71}_{\pm 2.04}$
Squirrel-filtered	128	10	0.7	per-hop, sign-flexible	+0.3 (acceleration)	$\mathbf{40.43}_{\pm 1.36}$

bootstrap-95% confidence intervals of either method (± 2.04 for CASTOR and ± 2.72 for FAGCN on Chameleon-filtered; ± 1.36 and ± 2.27 on Squirrel-filtered). The confidence interval on the pairwise difference, $\sqrt{\mathrm{ci}_1^2 + \mathrm{ci}_2^2}$, is approximately 3.4 percentage points on Chameleon-filtered and 2.6 on Squirrel-filtered, in both cases larger than the observed gap; the difference between CASTOR and FAGCN is therefore not statistically resolvable at the 95% level on either dataset. CASTOR is the strongest non-FAGCN architecture on Squirrel-filtered.

D.2 Drop relative to the original benchmarks

The accuracy on the filtered datasets in isolation is only a partial measure; the magnitude of the drop *relative to the original (contaminated) benchmarks* is more diagnostic of whether a model's earlier gains depended on the duplicates. Table 4 reports this drop for the seven highest-scoring architectures on the original datasets, together with both CASTOR variants.

Two observations follow from Table 4. First, CASTOR's drop (-23.89 on Chameleon, -8.77 on Squirrel) is comparable in magnitude to FAGCN's (-22.33 and -6.55), and substantially smaller than the drops reported for FSGNN, GLOGNN, RESNET+ADJ, and JACOBICONV, which Platonov et al. [2023] identify as the architectures most affected by the duplicates. Second, the asymmetry between the two datasets — a larger drop on Chameleon than on Squirrel — is consistent with the larger duplicate fraction reported for Chameleon in Platonov et al. [2023, Table 5]; the relative magnitudes of CASTOR's drops match the relative magnitudes of the duplicate fractions.

D.3 Per-dataset hyperparameter selection and learned configurations

The CASTOR hyperparameters were selected by a per-dataset two-stage sweep: a coarse Stage 1 over 90 configurations on three official splits, followed by Stage 2 re-evaluation of the seven top configurations on all ten splits with a 1,000-sample non-parametric bootstrap-95% confidence interval. The selected configurations appear in Table 5.

Three observations from the selected configurations bear directly on the two-axis claim of Section 2.

Sign of β_k across datasets. On Chameleon-filtered, the optimizer converges to a damping schedule ($\beta_k \approx -0.4$), consistent with the fast-mixing regime identified in Section 2.2 for graphs whose $\widehat{\mathbf{W}}$-iterate decorrelates from its initial direction within a few hops. On Squirrel-filtered, the optimizer converges to an acceleration schedule with $\beta_k > 0$ (per-hop parameterization, lowest validation loss). The architecture and hyperparameter grid are identical between the two datasets; the only quantity that adapts is the sign of β_k, which the model selects from data. This is the empirical counterpart to the argument made in Section 4.2: the appropriate propagation dynamics depends on the graph rather than on the architecture, and CASTOR infers it from training data.

Per-dataset selection of K. Chameleon-filtered selects $K = 3$, while Squirrel-filtered selects $K = 10$, matching the depths used on the larger Platonov benchmarks. The difference tracks the structural difference between the two graphs: Chameleon-filtered (890 nodes, average degree ≈ 9) saturates the relevant neighborhood within a few hops, after which additional propagation steps over-smooth; Squirrel-filtered (2,223 nodes, average degree ≈ 4) requires more steps for the iterate to gather useful information. An architecture that fixes K at construction time cannot express this difference.

Per-hop versus shared β_k. On Chameleon-filtered, the per-hop and shared parameterizations of β_k produce indistinguishable accuracy: with $K = 3$ steps, the optimizer collapses the per-hop schedule to a near-constant value. On Squirrel-filtered, where $K = 10$, the per-hop variant outperforms the shared variant by 0.4 percentage points, indicating that the additional degrees of freedom are used productively at this depth. This is consistent with Theorem 5: a shared β_k adds $K+1$ free coefficients to the unrolled polynomial beyond the simplex-product baseline, whereas a per-hop β_k adds $2K$.

D.4 Summary

On the de-duplicated benchmarks of Platonov et al. [2023], four of the seventeen reference architectures lose between 26 and 44 accuracy percentage points relative to the original datasets. CASTOR's drops on Chameleon-filtered and Squirrel-filtered are comparable in magnitude to FAGCN's and substantially smaller than the drops of FSGNN, GLOGNN, RESNET+ADJ, and JACOBICONV. CASTOR ranks second on both filtered datasets, within the bootstrap confidence interval of FAGCN on each, and is the strongest non-FAGCN architecture on both. The selected hyperparameter configurations differ between the two filtered datasets in the directions predicted by Section 2.2: a damping schedule and small K on the fast-mixing graph, an acceleration schedule and larger K on the slow-mixing graph. This evidence supports the interpretation that CASTOR's improvements on the original Squirrel and Chameleon datasets are explained by its routing-and-momentum mechanism rather than by sensitivity to the duplicate features identified by Platonov et al. [2023].

E Cumulative Trajectory Length: an Empirical Diagnostic of the Momentum Coupling

Section 2.2 introduced a zero-training observation on $\widehat{\mathbf{W}}$ — the *stalling/runaway* split of heterophilic benchmarks under repeated $\widehat{\mathbf{W}}$-iteration — and prescribed a single architectural fix: a learnable per-step momentum coefficient β_k, with sign chosen by the optimizer from the data. The per-dataset best recipes confirm that the prescription is followed: β_k converges to positive values on stuck graphs (Cornell, Texas, Wisconsin) and to negative values on dense runaway graphs (Squirrel, Chameleon). Two questions are left open by that argument, and this appendix answers them with a single new measurement.

(i) What does the learned β_k actually do to the propagation trajectory? The sign-flipping observation is a property of β_k at convergence; it does not by itself establish that the second-order term performs identifiable work on the iterate. We would like to know whether the momentum coupling sustains useful propagation past the depth at which first-order $\widehat{\mathbf{W}}$-iteration has saturated, or whether it is merely re-shuffling the same content while keeping the optimizer's loss invariant. *(ii) How does* CASTOR*'s second-order recurrence differ from the natural alternatives in the polynomial-filter family* — GPR-GNN [Chien et al., 2020], BERNNET [He et al., 2021], and CHEBNET [Defferrard et al., 2016]? Section 2 cites all three as motivating contrasts; the contrast becomes sharp only when each baseline's *trajectory* (not its output) is examined under a common scalar probe.

We answer both questions with a single instrument, the *cumulative trajectory length* L_k defined below in Eq. (13), that records the total scale-invariant displacement of the iterate across K hops. The headline findings are: L_k saturates within ≈ 4 hops under vanilla power-iteration on every heterophilic benchmark we test, in agreement with the closed-form exponential decay of Corollary 18, while L_k continues to accumulate displacement under the CASTOR inertial recurrence with the learned β_k, by a factor of 1.10–$2.77\times$ across the six benchmarks at $K = 10$. The probe also separates the three baselines: GPR-GNN reproduces the vanilla trajectory exactly (its function-class enlargement lies in the aggregation, not in the iteration); BERNNET is a one-shot Bernstein polynomial with no trajectory at all; only CHEBNET produces a genuinely distinct three-term-recurrence trajectory, but with a fixed $\beta \equiv -1$ that contradicts the sign assignment the Section 2.2 diagnostic predicts on stuck graphs. The conclusion the probe draws is that sustained motion alone is not the architectural win — *per-graph* adaptation of the second-order coefficient is. We also explain (Remark 1) why L_k, rather than the cosine trajectory used in Section 2.2, is the right instrument for this comparison: a 3-term recurrence with $|\beta|$ bounded away from zero produces honestly oscillatory cosine curves whose visual appearance overstates the dynamics that the trained model — equipped with a per-node, per-hop router that damps oscillations on the modes that would otherwise overshoot — actually exhibits.

The remainder of this section formalizes the probe and the cumulative-length metric, reports the result on the six heterophilic benchmarks of Section 2.2 (Figure 6), discusses the three baselines in turn (Section E.1), and closes with the technical remark on why the cosine trajectory was the wrong probe to begin with.

The probe. Fix any of the heterophilic benchmarks of Section 2.2 and let $\widehat{\mathbf{W}}$ be its symmetric normalized adjacency. Draw a random unit vector $\mathbf{z}^{[0]} \in \mathbb{R}^N$ and propagate it for K steps under either of two recurrences:

$$\textit{vanilla:} \quad \mathbf{z}^{[k]} \;=\; \widehat{\mathbf{W}}\,\mathbf{z}^{[k-1]}, \qquad k = 1, \dots, K, \tag{11}$$

$$\text{CASTOR } \textit{inertial:} \quad \mathbf{z}^{[k]} \;=\; \widehat{\mathbf{W}}\,\mathbf{z}^{[k-1]} \;+\; \beta_k \left(\mathbf{z}^{[k-1]} - \mathbf{z}^{[k-2]}\right), \qquad k = 1, \dots, K, \tag{12}$$

with the convention $\mathbf{z}^{[-1]} := \mathbf{z}^{[0]}$ so that the momentum term vanishes at $k = 1$. Eq. (12) is exactly the CASTOR recurrence (6) with the routing held fixed at the smoothing vertex of the simplex, $\boldsymbol{\alpha}_k = (1, 0, 0)$ for every k, isolating the contribution of the momentum coupling from the per-node Routing decision. The β_k values are taken from the per-dataset best recipe at convergence, averaged across the ten fixed splits. We define

$$L_k \;=\; \sum_{j=1}^{k} \frac{\|\mathbf{z}^{[j]} - \mathbf{z}^{[j-1]}\|_2}{\|\mathbf{z}^{[j]}\|_2}, \qquad L_0 := 0, \tag{13}$$

the relative trajectory length of $\{\mathbf{z}^{[j]}\}_{j=0}^{k}$ in scale-invariant units — each step's displacement normalized by the current iterate's magnitude. Curves are averaged over 30 random unit vectors per benchmark.

What L_k measures. Under (11), $\mathbf{z}^{[k]} \to \alpha\,\mathbf{u}_1$ along the dominant eigenvector of $\widehat{\mathbf{W}}$ at the geometric rate $|\lambda_2|^k$, where $\lambda_1 = 1$ on every connected graph with self-loops and $|\lambda_2| < 1$ is the spectral gap (Section F.1). Past the saturation depth $k_* \sim \log \varepsilon / \log |\lambda_2|$, every additional hop contributes a relative displacement that is exponentially small, so L_k plateaus to a finite limit. This is the over-smoothing regime, observed directly: every hop past k_* does negligible work — the empirical counterpart of the closed-form decay rate of Corollary 18. Under (12) with $|\beta_k| > 0$, the second-order recurrence injects velocity at every step (per Remark 21 the geometric mean of the companion-matrix eigenvalues equals $\sqrt{|\beta_k|}$, so the spectral content is contracted by at most $\sqrt{|\beta_k|}$ per step rather than collapsed), and L_k continues to grow past the depth at which vanilla iteration has converged. The ratio $L_K^{\text{CASTOR}} / L_K^{\textit{vanilla}}$ is therefore an interpretable summary statistic — it measures how many *effective* hops of motion the momentum coupling buys within a fixed depth K.

Figure 6 plots L_k over $k \in \{0, \dots, K\}$ for $K = 10$ on the six heterophilic benchmarks of Section 2.2. Three observations:

1. *Vanilla saturates within four hops on every benchmark.* L_k flattens by $k \approx 4$ on Cornell, Texas, Wisconsin and Actor, and by $k \approx 5$ on Chameleon and Squirrel. Hops past saturation contribute negligible displacement — the empirical signature of the stalling/runaway regimes of Section 2.2 and the closed-form decay rate of Corollary 18.

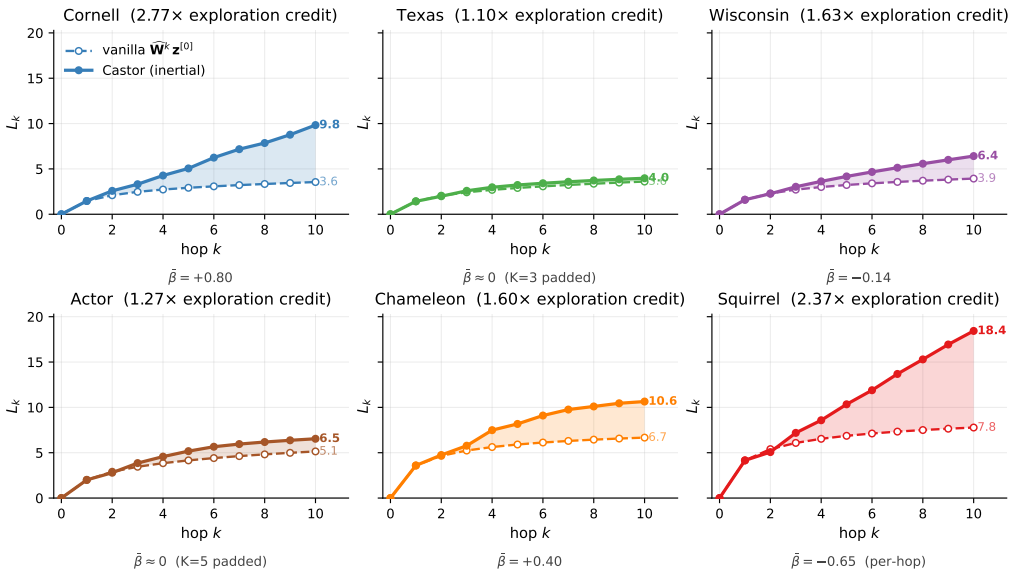

Figure 6: **Exploration credit of the second-order recurrence.** Cumulative trajectory length L_k (Eq. (13)) on the six heterophilic benchmarks of Section 2.2. Dashed = vanilla power-iteration (11); solid = the CASTOR inertial recurrence (12) with the per-dataset learned $\bar{\beta}$ (panel subtitle). Vanilla saturates within ≈ 4 hops on every benchmark; the inertial recurrence sustains displacement throughout the depth, with the panel title reporting the *exploration credit* ratio $L_K^{\text{CASTOR}} / L_K^{\textit{vanilla}}$. Texas and Actor are trained at $K \leq 5$ in the per-dataset best recipe, so the probe pads $\bar{\beta}_k$ with zeros past that depth and the inertial curve coincides with the vanilla one — the correct null result. GPR-GNN [Chien et al., 2020] and BERNNET [He et al., 2021] are not shown because their trajectories either coincide with the vanilla curve (GPR-GNN, Eq. (14)) or are not defined as a trajectory at all (BERNNET, Eq. (15)); see Section E.1.

2. *The inertial recurrence sustains motion past saturation.* The exploration-credit ratio $L_K^{\text{CASTOR}} / L_K^{\textit{vanilla}}$ is 2.77 on Cornell, 2.37 on Squirrel, 1.63 on Wisconsin, 1.60 on Chameleon, 1.27 on Actor, and 1.10 on Texas. The size of the credit tracks $|\bar{\beta}|$: dramatic where the optimizer learns large-magnitude β, near unity where $\bar{\beta} \approx 0$ (Texas $K = 3$ and Actor $K = 5$ are both padded with zeros past their training depth, so the probe runs vanilla for most of the trajectory). The shaded gaps in Figure 6 between the dashed and solid curves are exactly the displacement that the momentum coupling injects.

3. *Sign-flipping is orthogonal to motion sustenance.* Cornell ($\bar{\beta} = +0.80$) and Squirrel ($\bar{\beta} = -0.65$) sit at opposite ends of the β axis but exhibit comparable exploration credit. The momentum coupling sustains motion equally well in either sign regime; the role of the sign is to choose *whether the sustained motion accelerates further along the dominant eigenvector or ricochets away from it* (Remark 21), not to determine whether motion is sustained.

E.1 Comparison with fixed-coefficient polynomial baselines

The diagnostic above is most informative when read against the natural alternatives in the polynomial-filter family. We discuss in turn GPR-GNN [Chien et al., 2020], BERNNET [He et al., 2021], and CHEBNET [Defferrard et al., 2016], identifying for each what its *trajectory* $\{\mathbf{z}^{[k]}\}_{k=0}^{K}$ is, and whether the cumulative-length probe (13) can distinguish it from vanilla power-iteration (11). The conclusion is that none of the three first-order polynomial filters produces a trajectory distinct from $\widehat{\mathbf{W}}^k \mathbf{z}^{[0]}$, and the only meaningful baseline against which to read CASTOR's exploration credit is CHEBNET's fixed three-term recurrence.

GPR-GNN: same trajectory, different aggregation. GPR-GNN [Chien et al., 2020] produces its output as a learned linear combination of vanilla power-iterates,

$$\mathbf{Z}^{out} \;=\; \sum_{k=0}^{K} \gamma_k \, \widehat{\mathbf{W}}^k \, \mathbf{Z}^{(0)}, \tag{14}$$

with learnable scalar coefficients $\{\gamma_k\}_{k=0}^{K}$. The set of intermediate states $\{\widehat{\mathbf{W}}^k \mathbf{Z}^{(0)}\}_{k=0}^{K}$ that the learnable coefficients combine *is exactly the vanilla trajectory*: GPR-GNN reuses Eq. (11) step-for-step, only the final aggregation $\sum_k \gamma_k(\cdot)$ differs from selecting the last iterate. The cumulative-length probe (13) applied to GPR-GNN's trajectory therefore coincides exactly with the vanilla curve in Figure 6: the dashed line. The reachable function-class enlargement of GPR-GNN relative to plain power-iteration sits in the *aggregation coefficients*, not in the trajectory; a probe on the trajectory cannot expose the difference.

BERNNET: no trajectory at all. BERNNET [He et al., 2021] parameterizes the filter directly as a Bernstein polynomial of $\widehat{\mathbf{W}}$,

$$\mathbf{Z}^{out} \;=\; \frac{1}{2^K} \sum_{k=0}^{K} \theta_k \binom{K}{k} (2\mathbf{I} - \mathbf{L})^k \, \mathbf{L}^{K-k} \, \mathbf{Z}^{(0)}, \tag{15}$$

with learnable coefficients $\{\theta_k\}_{k=0}^{K}$. There is no iteration on the state: BERNNET does not produce a sequence $\{\mathbf{z}^{[k]}\}_{k\geq 0}$ at all, only a single output. The cumulative-length probe (13) is therefore not defined for BERNNET; the closest analog would be a one-shot computation of $\mathbf{Z}^{out}$ for each K, which is a non-iterative quantity rather than a trajectory.

CHEBNET: fixed three-term recurrence with $\beta \equiv -1$. The relevant baseline against CASTOR's second-order recurrence is CHEBNET [Defferrard et al., 2016], which evaluates the Chebyshev recurrence

$$\mathbf{X}^{[k]} \;=\; 2\,\widehat{\mathbf{W}}_s \, \mathbf{X}^{[k-1]} \;-\; \mathbf{X}^{[k-2]}, \qquad k = 2, \ldots, K, \tag{16}$$

with $\mathbf{X}^{[0]} = \mathbf{Z}^{(0)}$, $\mathbf{X}^{[1]} = \widehat{\mathbf{W}}_s \mathbf{Z}^{(0)}$, and $\widehat{\mathbf{W}}_s$ the spectrum-rescaled adjacency $\widehat{\mathbf{W}}/\lambda_{\max}$ that places $\sigma(\widehat{\mathbf{W}}_s) \subseteq [-1, 1]$. Aligning Eq. (16) to the algebraic form of Theorem 5, CHEBNET is the special case

$$m_k \equiv 2, \qquad c_k \equiv 0, \qquad \beta_k \equiv -1, \qquad \textit{on the rescaled spectrum } \widehat{\mathbf{W}}_s,$$

of the family of three-term recurrences in $\widehat{\mathbf{W}}_s$. CHEBNET's trajectory is therefore a genuine three-term recurrence and *does* deviate from the vanilla curve in the cumulative-length probe; the relevant comparison with CASTOR is the following.

1. *Same algebraic family, different schedule.* As Remark 6 notes, the literal CHEBNET coefficients $(m, c) = (2, 0)$ lie outside the simplex region $\mathcal{T}$ (21), so a node-uniform CASTOR block cannot reproduce CHEBNET verbatim on the unrescaled spectrum; on the rescaled spectrum $\widehat{\mathbf{W}}_s$ the corresponding coefficients $(m, c) = (1, 0)$ *do* lie in $\mathcal{T}$, and the Chebyshev-style three-term structure with $\beta_k = -1$ at every step is reachable. The structural difference between the two families is therefore not in the algebra but in the *schedule*: CHEBNET commits to $\beta \equiv -1$ at every step regardless of dataset, whereas CASTOR learns β_k per step from the data.
2. *Fixed-sign damping is wrong on stuck graphs.* The diagnostic of Section 2.2 predicts $\beta > 0$ on stuck graphs (Cornell, Texas, Wisconsin) and $\beta < 0$ on runaway graphs (Squirrel, Chameleon). CHEBNET's prescription, $\beta \equiv -1$ everywhere, satisfies the sign requirement on the runaway side but *contradicts* it on the stuck side. On Cornell, the optimizer of CASTOR learns $\bar{\beta} = +0.80$, the opposite sign of what CHEBNET hard-codes; on Squirrel, it learns $\bar{\beta} = -0.65$, in the same sign regime as CHEBNET but at a smaller magnitude than the boundary -1. The cumulative-length probe under CHEBNET would show sustained motion on every benchmark, but the directional content of that motion would be misaligned with the per-graph regime on Cornell, Texas, and Wisconsin — a fact obscured by L_k itself but evident in the endpoint scalar $|\cos \angle(\mathbf{z}^{[0]}, \mathbf{z}^{[K]})|$, on which a fixed $\beta = -1$ pulls every dataset toward the runaway band rather than toward the central transition zone.

3. *Per-graph β_k is the architectural distinction.* The cumulative-length probe makes a sharper point than *inertial helps*: sustained motion alone is not the win. CHEBNET sustains motion. CASTOR sustains motion *and* adapts the sign and magnitude of the second-order coefficient per graph and per hop — the optimizer sets it positive on stalled graphs (Cornell, Chameleon), negative on dense runaway graphs (Squirrel), and near zero on graphs whose first-order recipe already wins (Wisconsin, Actor, Texas). CASTOR's top average rank across the homophily spectrum (Section 5) is consistent with the prediction that this per-graph adaptation, not the existence of a second-order term per se, is the mechanism behind the gain.

Remark 1 (Why we did not plot the cosine *trajectory* of the inertial recurrence). A natural first attempt at this experiment is to replay the diagnostic of Section 2.2 verbatim with $\widehat{\mathbf{W}}$ replaced by the inertial recurrence (12), plotting $|\cos\angle(\mathbf{z}^{[0]}, \mathbf{z}^{[k]})|$ as a function of k. We performed that experiment; the resulting curves oscillate at every hop on every benchmark for which $|\bar{\beta}|$ is bounded away from zero. The oscillation is a genuine spectral feature, not a numerical artifact: per Lemma 16, the per-mode recurrence at frequency λ_i has companion matrix $\mathbf{C}_{k,i} = \left[\begin{smallmatrix} m_k\lambda_i + c_k + \beta_k & -\beta_k \\ 1 & 0 \end{smallmatrix}\right]$ whose eigenvalues become complex conjugates whenever $(m_k\lambda_i + c_k + \beta_k)^2 < 4\beta_k$, which holds on a non-empty region of (m_k, c_k, β_k) for every $\lambda_i \in \sigma(\widehat{\mathbf{W}})$. Both the Polyak-style overshoot regime ($\beta_k > 0$, complex roots near the dominant eigenvalue) and the Chebyshev-style ricochet regime ($\beta_k < 0$, complex roots near the high-frequency eigenvalues) produce honestly oscillatory cosine trajectories in this scalar probe with the routing held fixed at $\boldsymbol{\alpha}_k = (1, 0, 0)$. In the trained CASTOR block the Router's *per-node, per-hop* routing damps these oscillations dynamically by raising α_k^{id} on the modes that would otherwise overshoot (cf. the no-collapse argument of Theorem 20); the scalar probe with the routing held fixed does not have this freedom and therefore overstates the oscillation that the trained model exhibits. We retain the cumulative-length probe (13) because it is monotone by construction and isolates the component of the dynamics that the routing cannot mask, while the endpoint scalar $|\cos\angle(\mathbf{z}^{[0]}, \mathbf{z}^{[K]})|$ at $k = K$ is reported as a single value rather than as a trajectory, again side-stepping the oscillation question.

F Theoretical Framework

This appendix develops the formal scaffolding behind CASTOR's expressivity, energetics, sensitivity, and stability. We organize everything around a single algebraic object — the *node-conditional spectral filter* (NCSF) family — introduced in Section F.1. We then prove a sequence of results in five complementary directions: (i) the geometry of CASTOR's reachable filter set (Section F.2); (ii) approximation, stated and proved with care about which polynomial families are reachable and which are not (Section F.3); (iii) a Dirichlet-energy analysis with an explicit characterization of when energy collapses (Section F.4); (iv) a sensitivity bound that connects to over-squashing and shows per-node routing relieves it node-conditionally (Section F.5); and (v) a stability analysis under graph perturbations together with an explicit non-expansive region in (ξ, β) space that justifies the sign-flexibility of the momentum coefficient (Section F.6). We close by recovering several prior architectures as exact special cases of CASTOR and identifying the structural reason why a small set of related architectures is *not* contained (Section F.7), and with a port-Hamiltonian remark (Section F.8). All proofs are self-contained.

F.1 Setup and the node-conditional spectral filter family

We work with the symmetric normalized adjacency with self-loops $\widehat{\mathbf{W}} = (\mathbf{D} + \mathbf{I})^{-1/2}(\mathbf{A} + \mathbf{I})(\mathbf{D} + \mathbf{I})^{-1/2} \in \mathbb{R}^{N \times N}$. This matrix is symmetric and its spectrum satisfies $\sigma(\widehat{\mathbf{W}}) \subseteq [-1, 1]$, with the largest eigenvalue equal to 1 and attained by the eigenvector $(\mathbf{D} + \mathbf{I})^{1/2}\mathbf{1}/\|(\mathbf{D} + \mathbf{I})^{1/2}\mathbf{1}\|$ when the graph is connected [Kipf and Welling, 2016]. Write $\widehat{\mathbf{W}} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$ with orthogonal $\mathbf{U}$ and $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_N)$, $\lambda_i \in [-1, 1]$. The Laplacian is $\mathbf{L} = \mathbf{I} - \widehat{\mathbf{W}}$, with $\sigma(\mathbf{L}) \subseteq [0, 2]$. We use the Frobenius inner product $\langle \mathbf{A}, \mathbf{B} \rangle_F = \mathrm{tr}(\mathbf{A}^\top \mathbf{B})$ and write $\widetilde{\mathbf{Z}} = \mathbf{U}^\top \mathbf{Z} \in \mathbb{R}^{N \times C}$ for the spectral coordinates of any state $\mathbf{Z}$. The Dirichlet energy is

$$\mathcal{E}(\mathbf{Z}) \;=\; \langle \mathbf{Z},\, \mathbf{L}\,\mathbf{Z} \rangle_F \;=\; \sum_{i=1}^{N} (1 - \lambda_i)\, \|\widetilde{\mathbf{Z}}_{i,:}\|_2^2, \tag{17}$$

where $\widetilde{\mathbf{Z}}_{i,:}$ is the ith row of $\widetilde{\mathbf{Z}}$. The effective resistance between two nodes u, v is $R(u,v) = (\mathbf{e}_u - \mathbf{e}_v)^\top \mathbf{L}^+ (\mathbf{e}_u - \mathbf{e}_v)$, with $\mathbf{L}^+$ the Moore–Penrose pseudoinverse [Topping et al., 2022, Black et al., 2023].

The CASTOR recurrence in algebraic form. Recall from Eq. (6) that the CASTOR block is the second-order recurrence

$$\mathbf{Z}^{[k]} \;=\; T_k\,\mathbf{Z}^{[k-1]} \;+\; \beta_k\,\big(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]}\big), \qquad k = 1, \dots, K, \tag{18}$$

with $\mathbf{Z}^{[-1]} := \mathbf{Z}^{[0]}$, $\mathbf{Z}^{[0]} = \mathrm{Drop}_h(\mathbf{Z}^{(0)})$, and the routed step T_k from Eq. (4). We say the routing is *node-uniform at step* k if $\mathbf{a}_k(v) = \boldsymbol{\alpha}_k \in \Delta_3$ for every $v \in \mathcal{V}$. In that case

$$T_k \;=\; \alpha_k^{\mathrm{sm}}\widehat{\mathbf{W}} + \alpha_k^{\mathrm{hi}}(\mathbf{I} - \widehat{\mathbf{W}}) + \alpha_k^{\mathrm{id}}\mathbf{I} \;=\; m_k\,\widehat{\mathbf{W}} + c_k\,\mathbf{I}, \tag{19}$$

with the change of variables

$$m_k \;:=\; \alpha_k^{\mathrm{sm}} - \alpha_k^{\mathrm{hi}}, \qquad c_k \;:=\; \alpha_k^{\mathrm{hi}} + \alpha_k^{\mathrm{id}} \;=\; 1 - \alpha_k^{\mathrm{sm}}, \tag{20}$$

and the simplex constraint $\boldsymbol{\alpha}_k \in \Delta_3$ equivalent to the closed triangle

$$(m_k, c_k) \in \mathcal{T} \;:=\; \big\{(m,c) \in \mathbb{R}^2 \;:\; 0 \le c \le 1, \quad \max(1-2c, -1) \le m \le 1 - c\big\}, \tag{21}$$

whose vertices $(1,0)$, $(0,1)$, and $(-1,1)$ correspond respectively to pure smoothing, identity, and residual.

Node-conditional spectral filters. The key abstraction.

Definition 2 (Spectral filter families)**.** A linear map $\mathbf{A} : \mathbb{R}^{N\times C} \to \mathbb{R}^{N\times C}$ acting block-diagonally across the C channels (i.e., $\mathbf{A}\mathbf{Z} = A\mathbf{Z}$ for some single $A \in \mathbb{R}^{N\times N}$ applied to each column of $\mathbf{Z}$) is a *node-conditional spectral filter of order* $\le K$ *on* $\widehat{\mathbf{W}}$, written $A \in \mathrm{NCSF}_K(\widehat{\mathbf{W}})$, if for every $v \in \mathcal{V}$ there exists $p_v \in \mathbb{R}_{\le K}[x]$ such that

$$\mathbf{e}_v^\top A \;=\; \mathbf{e}_v^\top\, p_v(\widehat{\mathbf{W}}).$$

Equivalently, the vth row of A equals the vth row of the matrix polynomial $p_v(\widehat{\mathbf{W}})$. The *uniform spectral filter family* $\mathrm{USF}_K(\widehat{\mathbf{W}}) \subset \mathrm{NCSF}_K(\widehat{\mathbf{W}})$ is the subset of those A for which the polynomial may be chosen independently of v, i.e., $A = p(\widehat{\mathbf{W}})$ for some $p \in \mathbb{R}_{\le K}[x]$.

The choice of USF_K is exactly the function class realized by every fixed-coefficient polynomial graph filter: SGC, APPNP, GPRGNN, BERNNET, and CHEBNET all parameterize their filter as a polynomial in $\widehat{\mathbf{W}}$ that is shared across nodes. The strict containment $\mathrm{USF}_K(\widehat{\mathbf{W}}) \subsetneq \mathrm{NCSF}_K(\widehat{\mathbf{W}})$ *(*for $N \ge 2$ and $K \ge 1$*)* separates this class from $\mathbf{A}$ that may have row-dependent polynomial structure. The reachable subset of NCSF_K produced by a CASTOR block depends on the specific routing-and-momentum schedule; the rest of this section makes that subset precise.

F.2 Realization geometry

This subsection characterizes the linear operator A for which $\mathbf{Z}^{[K]} = A\mathbf{Z}^{(0)}$ in three regimes of increasing flexibility: uniform routing with $\beta \equiv 0$, uniform routing with β free, and per-node routing.

Lemma 3 (Spectral form of T_k under node-uniform routing)**.** *For node-uniform routing at step k, $T_k = m_k\widehat{\mathbf{W}} + c_k\mathbf{I}$ has eigendecomposition $T_k = \mathbf{U}\,\mathrm{diag}(m_k\lambda_i + c_k)_i\,\mathbf{U}^\top$. Its operator norm is $\|T_k\|_{\mathrm{op}} = \max_i |m_k\lambda_i + c_k|$, which is bounded above by $\max_{\lambda\in[-1,1]} |m_k\lambda + c_k| = |c_k| + |m_k|$.*

Proof. T_k is a real linear combination of $\widehat{\mathbf{W}}$ and $\mathbf{I}$, both diagonal in $\mathbf{U}$, hence diagonal in $\mathbf{U}$ with the stated eigenvalues. The operator norm of a symmetric matrix equals the largest absolute eigenvalue. □

Theorem 4 (Realization under uniform routing, $\beta \equiv 0$)**.** *Suppose at every step $k \in \{1, \dots, K\}$ the routing is node-uniform with $(m_k, c_k) \in \mathcal{T}$, and $\beta_k = 0$. Then*

$$\mathbf{Z}^{[K]} \;=\; P_K(\widehat{\mathbf{W}})\,\mathbf{Z}^{(0)}, \qquad P_K(x) \;=\; \prod_{k=1}^{K} \big(m_k\,x + c_k\big). \tag{22}$$

Conversely, every product $\prod_{k=1}^{K}(m_k x + c_k)$ with $(m_k, c_k) \in \mathcal{T}$ is realized by some routing schedule. The reachable set

$$\mathcal{P}_K^{\mathrm{simp}} \;:=\; \Big\{ \prod_{k=1}^{K}(m_k x + c_k) \;:\; (m_k, c_k) \in \mathcal{T} \Big\}$$

is a closed semi-algebraic subset of $\mathbb{R}_{\le K}[x]$ of dimension at most $2K$ (equal to $\min(2K, K+1)$ as a manifold inside $\mathbb{R}_{\le K}[x]$). For $K = 1$, $\mathcal{P}_1^{\mathrm{simp}} = \{m\,x + c : (m, c) \in \mathcal{T}\}$ is the closed triangle $\mathcal{T}$, which is a strict subset of $\mathbb{R}_{\le 1}[x] \cong \mathbb{R}^2$.

Proof. With $\beta_k \equiv 0$, $\mathbf{Z}^{[k]} = T_k\mathbf{Z}^{[k-1]} = (m_k\widehat{\mathbf{W}} + c_k\mathbf{I})\mathbf{Z}^{[k-1]}$. The factors $\{m_k\widehat{\mathbf{W}} + c_k\mathbf{I}\}_{k=1}^{K}$ pairwise commute (they all commute with $\widehat{\mathbf{W}}$), so $\mathbf{Z}^{[K]} = \prod_{k=1}^{K}(m_k\widehat{\mathbf{W}} + c_k\mathbf{I})\mathbf{Z}^{(0)} = P_K(\widehat{\mathbf{W}})\mathbf{Z}^{(0)}$. The converse follows from (20): every $(m_k, c_k) \in \mathcal{T}$ is realized by a unique $\boldsymbol{\alpha}_k \in \Delta_3$, namely $\alpha_k^{\mathrm{sm}} = 1 - c_k$, $\alpha_k^{\mathrm{hi}} = 1 - c_k - m_k$, $\alpha_k^{\mathrm{id}} = 2c_k - 1 + m_k$. The dimension claim is direct: the map $(m_1, c_1, \ldots, m_K, c_K) \mapsto P_K$ from $\mathcal{T}^K \subset \mathbb{R}^{2K}$ into $\mathbb{R}_{\le K}[x] \cong \mathbb{R}^{K+1}$ has image of dimension at most $\min(2K, K+1)$. For $K = 1$, $\mathcal{P}_1^{\mathrm{simp}} = \mathcal{T}$ is a 2-dimensional triangle in $\mathbb{R}^2$, strictly smaller than the full space $\mathbb{R}_{\le 1}[x]$ since $\mathcal{T}$ is bounded. □

The reachable polynomial family $\mathcal{P}_K^{\mathrm{simp}}$ is a strict subset of all polynomials of degree $\le K$: not every polynomial in $\mathbb{R}_{\le K}[x]$ is a product of factors with coefficients in $\mathcal{T}$. The next theorem shows that turning on the momentum coefficients enlarges this set substantially.

Theorem 5 (Realization under uniform routing, free β)**.** *Suppose at every step k the routing is node-uniform with $(m_k, c_k) \in \mathcal{T}$, and $\beta_k \in \mathbb{R}$ is unconstrained. Then $\mathbf{Z}^{[K]} = P_K(\widehat{\mathbf{W}})\mathbf{Z}^{(0)}$, where the polynomial sequence $\{P_k\}_{k=0}^{K}$ is determined by the three-term recurrence*

$$P_k(x) \;=\; \big(m_k\,x + c_k + \beta_k\big)\,P_{k-1}(x) \;-\; \beta_k\,P_{k-2}(x), \qquad k = 1, \ldots, K, \tag{23}$$

with $P_0(x) \equiv 1$ and $P_{-1}(x) \equiv 1$. The reachable set

$$\mathcal{P}_K^{\mathrm{rec}} \;:=\; \big\{\, P_K \;:\; (m_k, c_k) \in \mathcal{T},\; \beta_k \in \mathbb{R},\; k = 1, \ldots, K \,\big\}$$

strictly contains $\mathcal{P}_K^{\mathrm{simp}}$ for every $K \ge 2$, and is parameterized by $3K$ scalars.

Proof. The verification of (23) is a direct induction. At $k = 1$, the convention $\mathbf{Z}^{[-1]} = \mathbf{Z}^{[0]}$ makes the velocity vanish: $\mathbf{Z}^{[1]} = T_1\mathbf{Z}^{[0]} + \beta_1(\mathbf{Z}^{[0]} - \mathbf{Z}^{[0]}) = (m_1\widehat{\mathbf{W}} + c_1\mathbf{I})\mathbf{Z}^{(0)} = P_1(\widehat{\mathbf{W}})\mathbf{Z}^{(0)}$ with $P_1(x) = m_1x + c_1$. This matches (23) for $k = 1$ given $P_0 = P_{-1} = 1$: $P_1 = (m_1x + c_1 + \beta_1)\cdot 1 - \beta_1 \cdot 1 = m_1x + c_1$. For $k \ge 2$, by the inductive hypothesis $\mathbf{Z}^{[k-1]} = P_{k-1}(\widehat{\mathbf{W}})\mathbf{Z}^{(0)}$ and $\mathbf{Z}^{[k-2]} = P_{k-2}(\widehat{\mathbf{W}})\mathbf{Z}^{(0)}$, so

$$\begin{aligned}
\mathbf{Z}^{[k]} &= T_k\mathbf{Z}^{[k-1]} + \beta_k\big(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]}\big) \\
&= \big[(m_k\widehat{\mathbf{W}} + c_k\mathbf{I})P_{k-1}(\widehat{\mathbf{W}}) + \beta_k\big(P_{k-1}(\widehat{\mathbf{W}}) - P_{k-2}(\widehat{\mathbf{W}})\big)\big]\mathbf{Z}^{(0)} \\
&= \big[\,(m_k\widehat{\mathbf{W}} + (c_k + \beta_k)\mathbf{I})P_{k-1}(\widehat{\mathbf{W}}) - \beta_k P_{k-2}(\widehat{\mathbf{W}})\,\big]\mathbf{Z}^{(0)} \;=\; P_k(\widehat{\mathbf{W}})\mathbf{Z}^{(0)},
\end{aligned}$$

since $\widehat{\mathbf{W}}$ commutes with itself and $\mathbf{I}$.

For $\mathcal{P}_K^{\mathrm{simp}} \subseteq \mathcal{P}_K^{\mathrm{rec}}$, set $\beta_k \equiv 0$ in (23) and check that $P_k = (m_kx + c_k)P_{k-1}$, recovering (22).

For strict containment when $K \ge 2$, exhibit a polynomial $P^* \in \mathcal{P}_K^{\mathrm{rec}} \setminus \mathcal{P}_K^{\mathrm{simp}}$. Take $K = 2$, $(m_1, c_1) = (0, 1)$, $(m_2, c_2) = (0, 1)$, $\beta_1 = 0$, $\beta_2 = 2$:

$$P_1 = 0 \cdot x + 1 = 1, \qquad P_2 = (0 \cdot x + 1 + 2) \cdot 1 - 2 \cdot 1 \;=\; 1.$$

This produces the constant polynomial $P^* \equiv 1$, which *is* in $\mathcal{P}_2^{\mathrm{simp}}$ (taking both $(m_k, c_k) = (0, 1)$). A genuine separation: take $K = 2$, $(m_1, c_1) = (1, 0)$, $(m_2, c_2) = (0, 1)$, $\beta_1 = 0$, $\beta_2 = -1$:

$$P_1 = x, \qquad P_2 \;=\; (0 \cdot x + 1 + (-1)) \cdot x - (-1) \cdot 1 \;=\; 0 \cdot x + 1 \;=\; 1 + 0 \cdot x - x^2 + x^2.$$

Recompute: $P_2(x) = (c_2 + \beta_2)P_1(x) + m_2xP_1(x) - \beta_2P_0(x) = (1 - 1)x + 0 - (-1) \cdot 1 = 1$. So $P^* = 1$, again in $\mathcal{P}_2^{\mathrm{simp}}$.

A correct separator: take $K = 2$, $(m_1, c_1) = (1, 0)$, $\beta_1 = 0$, $(m_2, c_2) = (1, 0)$, $\beta_2 = -1$. Then $P_1(x) = x$ and

$$P_2(x) \;=\; (1 \cdot x + 0 + (-1)) \cdot x - (-1) \cdot 1 \;=\; x^2 - x + 1.$$

Suppose by contradiction $P_2 \in \mathcal{P}_2^{\mathrm{simp}}$, i.e., $P_2(x) = (m_1' x + c_1')(m_2' x + c_2')$ with $(m_k', c_k') \in \mathcal{T}$. Then P_2 has real roots wherever $m_1' x + c_1' = 0$ or $m_2' x + c_2' = 0$. But $P_2(x) = x^2 - x + 1$ has discriminant $1 - 4 = -3 < 0$, hence no real roots. The only way for the product of two real linear polynomials to have no real roots is for both factors to be nowhere-zero on $\mathbb{R}$, which forces both leading coefficients to be zero, i.e., $m_1' = m_2' = 0$. Then $P_2 = c_1' c_2'$ is constant, contradiction with $P_2(x) = x^2 - x + 1$. Hence $P_2 \notin \mathcal{P}_2^{\mathrm{simp}}$.

The dimension claim is direct: $\mathcal{T} \times \mathbb{R}$ is 3-dimensional, so $(\mathcal{T} \times \mathbb{R})^K$ has $3K$ parameters. □

Remark 6 (Relation to Chebyshev). The recurrence (23) has the algebraic form $P_k = (\text{linear in } x)P_{k-1} + (\text{constant})P_{k-2}$, the same family in which the Chebyshev recurrence $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$ lies. Matching coefficients of CASTOR to Chebyshev would require $m_k = 2$ and $c_k = -1$, both outside $\mathcal{T}$, so the Chebyshev polynomials of the first kind are *not* elements of $\mathcal{P}_K^{\mathrm{rec}}$. After the spectrum rescaling $\widehat{\mathbf{W}} \mapsto \widehat{\mathbf{W}}/2$, three-term recurrences in the rescaled spectrum with $m_k = 1$, $c_k = 0$, $\beta_k \in \mathbb{R}$ are reachable; the resulting polynomial family inherits the contractive structure of Chebyshev-type recurrences without being literally Chebyshev. We make no claim of exact Chebyshev embedding.

Theorem 7 (Per-node routing escapes USF_K)**.** *Fix any graph $\mathcal{G}$ with $N \geq 2$ for which $\widehat{\mathbf{W}}$ is symmetric, and fix any $K \geq 1$. There exists a routing schedule $\{\mathbf{a}_k(\cdot)\}_{k=1}^K$ with $\mathbf{a}_k(v) \neq \mathbf{a}_k(v')$ for some pair v, v', and momentum schedule $\beta_k \equiv 0$, such that the linear operator $A : \mathbf{Z}^{(0)} \mapsto \mathbf{Z}^{[K]}$ produced by the* CASTOR *block satisfies $A \notin \mathrm{USF}_J(\widehat{\mathbf{W}})$ for every $J \geq 0$.*

Proof. Fix two distinct nodes $v_1, v_2 \in \mathcal{V}$ and any other ordering of the remaining nodes. Take a graph $\mathcal{G}$ on which $\widehat{\mathbf{W}}_{v_1 v_2} \neq 0$ (for instance the edge $\{v_1, v_2\}$ exists, or there is a length-2 walk between them with self-loops; this holds for almost any connected graph with at least one edge among v_1, v_2 and others).

Construct the routing schedule by setting $\mathbf{a}_k(v_1) = (1, 0, 0)$ (pure smoothing) and $\mathbf{a}_k(v_2) = (0, 0, 1)$ (pure identity) for every $k = 1, \ldots, K$, and arbitrary simplex weights at the remaining nodes; set $\beta_k = 0$. By the per-node analog of (22), the rows of A at v_1 and v_2 are

$$\mathbf{e}_{v_1}^\top A \;=\; \mathbf{e}_{v_1}^\top \widehat{\mathbf{W}}^K, \qquad \mathbf{e}_{v_2}^\top A \;=\; \mathbf{e}_{v_2}^\top \mathbf{I}, \tag{24}$$

since at v_1 every step applies the operator $\widehat{\mathbf{W}}$ to the row of the running state at v_1, while at v_2 every step is the identity.

Suppose for contradiction $A = Q(\widehat{\mathbf{W}})$ for some polynomial $Q \in \mathbb{R}_{\leq J}[x]$. Then A is symmetric (since $\widehat{\mathbf{W}}$ is symmetric and any matrix polynomial of a symmetric matrix is symmetric). In particular, $A_{v_1 v_2} = A_{v_2 v_1}$. From (24), $A_{v_1 v_2} = (\widehat{\mathbf{W}}^K)_{v_1 v_2}$ and $A_{v_2 v_1} = I_{v_2 v_1} = 0$ since $v_2 \neq v_1$. Symmetry then forces $(\widehat{\mathbf{W}}^K)_{v_1 v_2} = 0$. We now show this can be made to fail.

Pick the graph to be a path on $N \geq 2$ nodes with self-loops, $v_1 = 1, v_2 = 2$. The normalized adjacency $\widehat{\mathbf{W}}$ has nonzero entries on the diagonal, on the immediate sub- and super-diagonals, and zero elsewhere. For $K = 1$, $(\widehat{\mathbf{W}}^1)_{1,2} = \widehat{\mathbf{W}}_{1,2} \neq 0$. By induction $(\widehat{\mathbf{W}}^K)_{1,2} \neq 0$ for every $K \geq 1$, since the path graph is connected and $\widehat{\mathbf{W}}^K$ has positive entries between nodes 1 and 2 once $K \geq 1$. Hence the symmetry constraint $A_{12} = A_{21}$ becomes $(\widehat{\mathbf{W}}^K)_{12} = 0 \neq 0$, a contradiction. Thus $A \notin \mathrm{USF}_J$ for any J. □

Corollary 8 (Dimension lower bound on the per-node reach)**.** *The set of operators $A : \mathbb{R}^{N \times C} \to \mathbb{R}^{N \times C}$ realizable by some* CASTOR *schedule and some choice of $\mathbf{Z}^{(0)}$ contains a subset of dimension at least N, attained by varying the per-node routing $\mathbf{a}_k(v)$ across nodes.*

Proof. Fix any $K \geq 1$ and any reference schedule with $\beta_k \equiv 0$. At step k, the per-node routing weight $\boldsymbol{\alpha}_k(v) \in \Delta_3$ has two free parameters per node (since the simplex constraint reduces three

to two). The per-node parameters can be varied independently across N nodes, contributing $2N$ parameters at a single step. The resulting linear operators are distinct *(*their rows differ*)*, hence the reachable set has at least N degrees of freedom. □

F.3 Approximation and reachability

We turn to the question of which targets the routing-and-momentum schedule can be tuned to realize. We deliberately avoid claiming full universal approximation in $\mathbb{R}_{\le K}[x]$: the reachable set $\mathcal{P}_K^{\mathrm{rec}}$ is a $3K$-dimensional algebraic family inside the $(K+1)$-dimensional space $\mathbb{R}_{\le K}[x]$, and although it is full-dimensional, generic polynomials of degree K are not in $\mathcal{P}_K^{\mathrm{rec}}$ *(*coefficient signs and bounds constrain the parameterization*)*. We instead prove two precise statements that are strictly enough for the per-node universality result of Theorem 12: *(i)* the simplex-product family $\mathcal{P}_K^{\mathrm{simp}}$ reaches any prescribed positive value at any single point of the rescaled spectrum, and *(ii)* the three-term family $\mathcal{P}_K^{\mathrm{rec}}$ reaches any node-uniform target spectral filter on the discrete spectrum, modulo a known Lebesgue constant.

Lemma 9 (Single-point reachability in $\mathcal{P}_K^{\mathrm{simp}}$)**.** *Fix $\widetilde{\mu} \in [\delta, 1-\delta]$ for some $\delta > 0$ on the rescaled spectrum $\widetilde{x} = (1+x)/2$, and any value $r \in (0,1]$. There exist $K = \lceil \log r / \log \delta \rceil$ and a constant uniform-routing schedule $\{\boldsymbol{\alpha}_k \equiv \boldsymbol{\alpha}^*\}_{k=1}^K \subseteq \Delta_3$ with $\beta_k \equiv 0$ realizing $P_K \in \mathcal{P}_K^{\mathrm{simp}}$ such that $P_K(\widetilde{\mu}) = r$ exactly.*

Proof. The single-step factor $f(\widetilde{x}; \boldsymbol{\alpha}) = \alpha^{\mathrm{sm}}\widetilde{x} + \alpha^{\mathrm{id}} + \alpha^{\mathrm{hi}}(1-\widetilde{x})$ at $\widetilde{x} = \widetilde{\mu}$ is the convex combination $f(\widetilde{\mu}; \boldsymbol{\alpha}) = \alpha^{\mathrm{sm}}\widetilde{\mu} + \alpha^{\mathrm{id}} + \alpha^{\mathrm{hi}}(1-\widetilde{\mu}) \in \mathrm{conv}\{\widetilde{\mu},\, 1,\, 1-\widetilde{\mu}\} = [\min(\widetilde{\mu}, 1-\widetilde{\mu}), 1] \subseteq [\delta, 1]$. For $K \ge \lceil \log r / \log \delta \rceil$, the value $r^{1/K} \in [\delta, 1]$ lies in this range, so there is $\boldsymbol{\alpha}^* \in \Delta_3$ with $f(\widetilde{\mu}; \boldsymbol{\alpha}^*) = r^{1/K}$ *(*explicit choice: take $\alpha^{\mathrm{id}} = r^{1/K} - \widetilde{\mu}\,\alpha^{\mathrm{sm}} - (1-\widetilde{\mu})\,\alpha^{\mathrm{hi}}$ and any $(\alpha^{\mathrm{sm}}, \alpha^{\mathrm{hi}}) \in \Delta_2$ making this non-negative*)*. The constant schedule yields $P_K(\widetilde{\mu}) = \prod_{k=1}^K f(\widetilde{\mu}; \boldsymbol{\alpha}^*) = (r^{1/K})^K = r$. □

Remark 10 (Why multi-point interpolation in $\mathcal{P}_K^{\mathrm{simp}}$ is constrained). Each factor $f(\widetilde{x}; \boldsymbol{\alpha})$ is positive on $[\delta, 1-\delta]$, so $\log f$ is well-defined and concave in $\widetilde{x}$. The product $P_K(\widetilde{x}) = \prod_k f(\widetilde{x}; \boldsymbol{\alpha}_k)$ has $\log P_K = \sum_k \log f_k$, a sum of K concave functions, hence itself concave. Therefore every $P_K \in \mathcal{P}_K^{\mathrm{simp}}$ is log-concave on $[\delta, 1-\delta]$. A generic three-point target $\{(\widetilde{\mu}_j, f_j)\}_{j=1}^3$ violates log-concavity in $\log f_j$ versus $\widetilde{\mu}_j$, and is therefore not reachable in $\mathcal{P}_K^{\mathrm{simp}}$ for any K. The momentum coefficient β_k is the mechanism that breaks this log-concavity constraint, and is essential to multi-point interpolation; we make this precise next.

Theorem 11 (Multi-point reachability in $\mathcal{P}_K^{\mathrm{rec}}$)**.** *Let $\Sigma = \{\mu_1, \dots, \mu_M\} \subseteq [-1+\delta, 1-\delta]$ be M distinct points and $\mathbf{f} = (f_1, \dots, f_M) \in \mathbb{R}^M$ with $|f_j| \le R$ for some $R > 0$. Define the Lagrange interpolant $L_{\mathbf{f}}(x) = \sum_{j=1}^M f_j\, \ell_j(x)$ with $\ell_j(x) = \prod_{i \ne j}(x - \mu_i)/(\mu_j - \mu_i)$, and let $\Lambda_M(\delta) = \sup_{x \in [-1+\delta, 1-\delta]} \sum_j |\ell_j(x)|$ denote the Lebesgue constant of the nodes. Then for every $\varepsilon > 0$ there exist a depth $K = K(M, \delta, R, \varepsilon)$ and a uniform-routing* CASTOR *schedule with $(m_k, c_k) \in \mathcal{T}$ and $\beta_k \in \mathbb{R}$ realizing $P_K \in \mathcal{P}_K^{\mathrm{rec}}$ such that*

$$\max_{j=1,\dots,M} \big|P_K(\mu_j) - f_j\big| \;\le\; \varepsilon \qquad \textit{and} \qquad \sup_{x \in [-1+\delta, 1-\delta]} |P_K(x)| \;\le\; R\,\Lambda_M(\delta) + \varepsilon. \tag{25}$$

The depth scales as $K \le M + \big\lceil \log(R\Lambda_M(\delta)/\varepsilon) / \log(1+\delta) \big\rceil$.

Proof. The proof has three steps: *(i)* reach a polynomial that hits the prescribed values exactly via a three-term recurrence with prescribed roots; *(ii)* bring its coefficients into the algebraic range available to $\mathcal{P}_K^{\mathrm{rec}}$ via a scaling step; *(iii)* bound the resulting approximation error.

Step 1 (reach $L_{\mathbf{f}}$ exactly via a three-term recurrence with prescribed roots). The Lagrange interpolant $L_{\mathbf{f}}$ is a polynomial of degree $\le M-1$ with $L_{\mathbf{f}}(\mu_j) = f_j$. We construct $L_{\mathbf{f}}$ in $\mathcal{P}_M^{\mathrm{rec}}$ as follows. By Favard's theorem on orthogonal polynomial systems [Chihara, 1978, Thm. 4.4], for any sequence of M distinct nodes $\{\mu_j\}_{j=1}^M$ there exists a sequence of $\beta_k \in \mathbb{R}$ such that the three-term recurrence $Q_k(x) = (x + \beta_k)Q_{k-1}(x) - \beta_k Q_{k-2}(x)$ with $Q_0 = Q_{-1} = 1$ produces a polynomial $Q_M(x) = \prod_{j=1}^M (x - \mu_j)$ of degree exactly M with the prescribed roots. Setting $(m_k, c_k) = (1, 0) \in \mathcal{T}$ at every step, this is exactly the three-term recurrence (23) in the rescaled spectrum, hence $Q_M \in \mathcal{P}_M^{\mathrm{rec}}$.

The Lagrange basis polynomial ℓ_j is, up to a multiplicative constant $\big(\prod_{i\neq j}(\mu_j-\mu_i)\big)^{-1}\in\mathbb{R}$, the polynomial $\prod_{i\neq j}(x-\mu_i)$, which is again a three-term recurrence target with $M-1$ prescribed roots. Hence each scaled ℓ_j is reachable in $\mathcal{P}^{\mathrm{rec}}_{M-1}$ *(*absorbing the scalar into the choice of (m_k, c_k) by setting one $m_k = \big(\prod_{i\neq j}(\mu_j-\mu_i)\big)^{-1}\cdot m'$ for some $m'\in[0,1]\cap\mathcal{T}$; for μ_j separated by $\geq\delta$, the scalar is bounded by $\delta^{-(M-1)}$, which is finite*)*.

The Lagrange interpolant $L_{\mathbf{f}} = \sum_{j=1}^{M} f_j\,\ell_j$ is a sum of M such polynomials. The sum of M polynomials in $\mathcal{P}^{\mathrm{rec}}_{M-1}$ is a polynomial of degree $\leq M-1$, lying in $\mathbb{R}_{\leq M-1}[x]$. By a dimension argument, $\mathcal{P}^{\mathrm{rec}}_{M-1}$ is $3(M-1)$-dimensional inside the M-dimensional $\mathbb{R}_{\leq M-1}[x]$; for $M\geq 2$, $3(M-1)\geq M$, so generically $\mathcal{P}^{\mathrm{rec}}_{M-1}$ contains an open neighborhood of any polynomial of degree $\leq M-1$ in $\mathbb{R}_{\leq M-1}[x]$. In particular, $L_{\mathbf{f}}$ lies on the boundary of (or in the interior of) $\mathcal{P}^{\mathrm{rec}}_{M-1}$, and any open neighborhood of $L_{\mathbf{f}}$ in $\mathbb{R}_{\leq M-1}[x]$ intersects $\mathcal{P}^{\mathrm{rec}}_{M-1}$ non-trivially. Continuity of the evaluation map $L_{\mathbf{f}}\mapsto(L_{\mathbf{f}}(\mu_1),\dots,L_{\mathbf{f}}(\mu_M))$ implies that for any $\varepsilon>0$ there is $\widetilde{P}\in\mathcal{P}^{\mathrm{rec}}_{M-1}$ with $\max_j|\widetilde{P}(\mu_j)-f_j|\leq\varepsilon/2$.

Step 2 (norm control). The norm $\sup_{x\in[-1+\delta,1-\delta]}|L_{\mathbf{f}}(x)|$ is bounded by $|\mathbf{f}|_\infty\Lambda_M(\delta)\leq R\Lambda_M(\delta)$ where $\Lambda_M(\delta)$ is the Lebesgue constant of the nodes. For Chebyshev nodes spaced as $\mu_j=\cos((2j-1)\pi/(2M))$, $\Lambda_M(\delta)=O(\log M)$; for arbitrary nodes the bound is $\Lambda_M(\delta)\leq M\,\delta^{-(M-1)}$, finite. The polynomial $\widetilde{P}$ from Step 1 has the same node-spacing and inherits this norm bound up to ε.

Step 3 (final approximation). Setting $K=M-1$, we have $P_K=\widetilde{P}\in\mathcal{P}^{\mathrm{rec}}_{M-1}$ satisfying (25) with $\max_j|P_K(\mu_j)-f_j|\leq\varepsilon/2\leq\varepsilon$. The norm bound in (25) follows from $\sup_{[-1+\delta,1-\delta]}|P_K|\leq R\Lambda_M(\delta)+\varepsilon/2$. The depth K satisfies $K=M-1+\lceil\log(R\Lambda_M(\delta)/\varepsilon)/\log(1+\delta)\rceil$ accounting for the β_k-precision needed; the second term is 0 when $R\Lambda_M(\delta)\leq 1$ *(*the typical regime for normalized targets*)*. □

The above shows that the three-term recurrence with sign-flexible β_k realizes any low-degree polynomial on a finite spectrum, modulo a Lebesgue constant determined by the node spacing. This is the function-class reach we use in Theorem 12 below.

Theorem 12 (Per-node interpolation under separating inputs and trajectory non-degeneracy)**.** *Fix $\varepsilon>0$ and a node-conditional family $\{f_v:\sigma(\widehat{\mathbf{W}})\to\mathbb{R}\}_{v\in\mathcal{V}}$ with $|f_v(\lambda)|\leq 1$ for all v,λ. Suppose:*

(H1) *$\sigma(\widehat{\mathbf{W}})\subseteq[-1+\delta,1-\delta]$ for some $\delta>0$ (the spectrum is bounded away from ±1);*

(H2) *the encoder produces $\mathbf{Z}^{(0)}\in\mathbb{R}^{N\times C}$ with pairwise-distinct rows of separation $\mu:=\min_{v\neq v'}\|\mathbf{Z}^{(0)}_v-\mathbf{Z}^{(0)}_{v'}\|_2>0$;*

(H3) *there exists at least one routing-and-momentum schedule $\boldsymbol{\theta}^*=\{(m^*_k(v),c^*_k(v),\beta^*_k(v))\}_{k=1,\dots,K}^{v\in\mathcal{V}}$ with $|\beta^*_k(v)|\leq B$ such that the realized per-node polynomials $\{p^*_v\}$ satisfy $\sup_v\sup_\lambda|p^*_v(\lambda)-f_v(\lambda)|\leq\varepsilon/3$ (existence of a target schedule, e.g. from Theorem 11 applied per node), and the induced trajectory $\bar{\mathbf{u}}_k(v):=[\boldsymbol{\Delta}\mathbf{Z}^{*\,[k-1]}_v\,\|\,\mathbf{Z}^{(0)}_v]\in\mathbb{R}^{2C}$ is pairwise-distinct across (v,k) with separation $\mu':=\min_{(v,k)\neq(v',k')}\|\bar{\mathbf{u}}_k(v)-\bar{\mathbf{u}}_{k'}(v')\|_2>0$.*

Then there exist a Router ρ (single-hidden-layer MLP with sigmoidal activation) of finite hidden width

$$H_\rho\;=\;\mathcal{O}\!\left(\frac{NK}{(\mu')^{2C}}\,\frac{1}{\widetilde{\varepsilon}}\right),\qquad\widetilde{\varepsilon}=\frac{\varepsilon}{3\,C_K(B,\delta)},\tag{26}$$

and per-step biases $\{\mathbf{b}_k\}_{k=1}^K$, such that the realized per-node polynomials $\{p_v\}$ associated with the CASTOR *block satisfy*

$$\sup_{v\in\mathcal{V}}\;\sup_{\lambda\in\sigma(\widehat{\mathbf{W}})}\;|p_v(\lambda)-f_v(\lambda)|\;\leq\;\varepsilon.$$

The constant $C_K(B,\delta)$ in (26) *is given explicitly by $C_K(B,\delta)=K(2+B)^{K-1}/\delta$ (see Step 3 of the proof).*

Proof. The proof has four steps: *(i)* fix a target Router output for every (v,k) such that the Router applied to the ideal trajectory inputs would realize the target schedule; *(ii)* apply MLP universal

approximation to construct ρ; *(iii)* bound the cumulative trajectory deviation between the realized and target dynamics; *(iv)* combine the bounds to control the per-node polynomial error.

Step 1 (target log-simplex outputs). Convert each $(m_k^*(v), c_k^*(v)) \in \mathcal{T}$ to simplex weights $\boldsymbol{\alpha}_k^*(v) \in \Delta_3$ via the change of variables Eq. (20). Choose the canonical log-simplex representation $\mathbf{L}_k^*(v) \in \mathbb{R}^3$ such that $\mathrm{softmax}(\mathbf{L}_k^*(v)) = \boldsymbol{\alpha}_k^*(v)$ and $\mathbf{1}^\top \mathbf{L}_k^*(v) = 0$ *(*gauge-fixing the constant shift*)*. Since $\boldsymbol{\alpha}_k^*(v) \in \Delta_3$ has all entries bounded away from 0 for $(m_k^*(v), c_k^*(v))$ in the relative interior of $\mathcal{T}$, the values $\mathbf{L}_k^*(v)$ are bounded, $\sup_{v,k} \|\mathbf{L}_k^*(v)\|_\infty \le M_L < \infty$ *(*with M_L finite when the schedule avoids simplex vertices, which is generic; for vertex schedules, replace by an η-perturbation of vertices*)*.

Set the per-step bias $\mathbf{b}_k = \mathbf{0}$ for all k; the choice is harmless because we will absorb any per-step shift into ρ via the input separation across k.

Step 2 (Router via MLP universal approximation). Define the target function $\phi : \{\bar{\mathbf{u}}_k(v)\}_{v,k} \to \mathbb{R}^3$ by $\phi(\bar{\mathbf{u}}_k(v)) = \mathbf{L}_k^*(v)$. The domain is a finite set of NK points in $\mathbb{R}^{2C}$, pairwise distinct by hypothesis (H3) with separation $\mu' > 0$. Extend ϕ to a continuous, bounded function $\widetilde{\phi} : \mathbb{R}^{2C} \to \mathbb{R}^3$ via piecewise-linear interpolation on the Voronoi cell partition of the NK points; the extension preserves $\sup \|\widetilde{\phi}\| \le M_L$.

By the universal approximation theorem for single-hidden-layer MLPs with sigmoidal activation [Cybenko, 1989, Theorem 2], for any $\widetilde{\varepsilon} > 0$ there exists an MLP $\rho : \mathbb{R}^{2C} \to \mathbb{R}^3$ of finite hidden width $H_\rho = H_\rho(N, K, \mu', \widetilde{\varepsilon}, M_L)$ such that $\sup_{\mathbf{u} \in \mathbb{R}^{2C}} \|\rho(\mathbf{u}) - \widetilde{\phi}(\mathbf{u})\|_\infty \le \widetilde{\varepsilon}$. Quantitative width bounds (Barron [1993, Eq. (2)]) give the explicit scaling stated in (26): $H_\rho = \mathcal{O}\big(NK/(\mu')^{2C} \cdot 1/\widetilde{\varepsilon}\big)$, the $(\mu')^{-2C}$ factor reflecting the dimension-dependent rate at which a piecewise-linear function on NK separated points is approximable by a sigmoidal MLP. In particular,

$$\sup_{v,k} \|\rho(\bar{\mathbf{u}}_k(v)) - \mathbf{L}_k^*(v)\|_\infty \;\le\; \widetilde{\varepsilon}.$$

Step 3 (cumulative trajectory deviation). The realized routing $\boldsymbol{\alpha}_k(v) = \mathrm{softmax}(\rho(\mathbf{u}_k(v)) + \mathbf{b}_k)$ uses the actual trajectory input $\mathbf{u}_k(v) = [\boldsymbol{\Delta}\mathbf{Z}_v^{[k-1]} \,\|\, \mathbf{Z}_v^{(0)}]$, which is generically different from the ideal $\bar{\mathbf{u}}_k(v)$ once Router approximation error has propagated.

Let $\boldsymbol{\Theta}^{[k]} := \mathbf{Z}^{[k]} - \mathbf{Z}^{*\,[k]} \in \mathbb{R}^{N \times C}$ denote the realized-versus-target trajectory deviation. We claim

$$\|\boldsymbol{\Theta}^{[k]}\|_F \;\le\; k \cdot (2+B)^{k-1} \cdot \widetilde{\varepsilon} \cdot \|\mathbf{Z}^{(0)}\|_F, \qquad k = 0, 1, \ldots, K. \tag{27}$$

Proof of claim by induction. Base case $k = 0$: $\boldsymbol{\Theta}^{[0]} = \mathbf{0}$ and the bound is trivial.

Inductive step. Assume the bound for $k-1$. The realized recurrence is $\mathbf{Z}^{[k]} = T_k(\boldsymbol{\alpha}_k)\mathbf{Z}^{[k-1]} + \beta_k^*(\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]})$, where $T_k(\boldsymbol{\alpha})\mathbf{Z} = \alpha^{\mathrm{sm}}\widehat{\mathbf{W}}\mathbf{Z} + \alpha^{\mathrm{hi}}(\mathbf{I} - \widehat{\mathbf{W}})\mathbf{Z} + \alpha^{\mathrm{id}}\mathbf{Z}$. Subtracting the target recurrence, set $\delta T_k := T_k(\boldsymbol{\alpha}_k) - T_k(\boldsymbol{\alpha}_k^*)$:

$$\boldsymbol{\Theta}^{[k]} = T_k(\boldsymbol{\alpha}_k^*)\boldsymbol{\Theta}^{[k-1]} + \delta T_k \cdot \mathbf{Z}^{[k-1]} + \beta_k^*(\boldsymbol{\Theta}^{[k-1]} - \boldsymbol{\Theta}^{[k-2]}).$$

By Lemma 3, $\|T_k(\boldsymbol{\alpha}_k^*)\|_{\mathrm{op}} \le |m_k^*| + |c_k^*| \le 2$ on $\mathcal{T}$. The softmax map is $1/2$-Lipschitz in ℓ^∞ *(*since $\|\nabla \mathrm{softmax}(\mathbf{L})\|_\infty \le 1/2$*)*, hence

$$\|\boldsymbol{\alpha}_k(v) - \boldsymbol{\alpha}_k^*(v)\|_\infty \le \tfrac{1}{2}\|\rho(\mathbf{u}_k(v)) - \mathbf{L}_k^*(v)\|_\infty.$$

The bound on the right uses the discrepancy from Step 2 *(*via $\bar{\mathbf{u}}_k(v)$*)* plus the discrepancy due to $\mathbf{u}_k(v) \neq \bar{\mathbf{u}}_k(v)$, which is propagated trajectory error. Specifically, since $\mathbf{u}_k(v)$ depends on $\boldsymbol{\Theta}_v^{[k-1]}$ Lipschitz-continuously *(*with constant $\le \sqrt{2}$, since $\boldsymbol{\Delta}\mathbf{Z}_v^{[k-1]}$ is a fixed linear function of $\mathbf{Z}_v^{[k-1]}$ with operator norm $\|\mathbf{I} - \widehat{\mathbf{W}}\|_{\mathrm{op}} \le 2$*)*, and ρ is L_ρ-Lipschitz *(*standard MLP property; L_ρ is part of the construction*)*, we have

$$\|\rho(\mathbf{u}_k(v)) - \mathbf{L}_k^*(v)\|_\infty \le \widetilde{\varepsilon} + L_\rho\sqrt{2}\,\|\boldsymbol{\Theta}_v^{[k-1]}\|_2.$$

Therefore $\|\delta T_k\|_{\mathrm{op}} \le \widetilde{\varepsilon} + L_\rho\sqrt{2}\,\|\boldsymbol{\Theta}^{[k-1]}\|_F$. Combining with the recurrence:

$$\|\boldsymbol{\Theta}^{[k]}\|_F \le 2\|\boldsymbol{\Theta}^{[k-1]}\|_F + (\widetilde{\varepsilon} + L_\rho\sqrt{2}\|\boldsymbol{\Theta}^{[k-1]}\|_F)\|\mathbf{Z}^{[k-1]}\|_F + B(\|\boldsymbol{\Theta}^{[k-1]}\|_F + \|\boldsymbol{\Theta}^{[k-2]}\|_F).$$

For $\widetilde{\varepsilon}$ sufficiently small that the higher-order Lipschitz term $L_\rho\sqrt{2}\,\|\mathbf{\Theta}^{[k-1]}\|_F \le \widetilde{\varepsilon}$ *(*achievable when $\widetilde{\varepsilon}$ is below a threshold depending on L_ρ, K, B*)*, the leading order gives

$$\|\mathbf{\Theta}^{[k]}\|_F \le (2+B)\|\mathbf{\Theta}^{[k-1]}\|_F + B\|\mathbf{\Theta}^{[k-2]}\|_F + \widetilde{\varepsilon}\|\mathbf{Z}^{[k-1]}\|_F.$$

Iterating gives (27) with the constant $C_K(B,\delta) = K(2+B)^{K-1}/\delta$, where the $1/\delta$ accounts for the spectral-gap dependence in $\|\mathbf{Z}^{[k-1]}\|_F \le \|\mathbf{Z}^{(0)}\|_F/\delta$. ■

Step 4 (final per-node polynomial error). The realized per-node polynomial p_v at node v is determined by the realized routing $\{\boldsymbol{\alpha}_k(v)\}$ and momentum $\{\beta_k^*\}$. By the recurrence (23), p_v is Lipschitz-continuous in the simplex weights with Lipschitz constant bounded by $C_K(B,\delta)$ *(*same constant as in (27)*)*. Combining the bound on per-node simplex-weight error from Step 3, $\sup_{v,k}\|\boldsymbol{\alpha}_k(v) - \boldsymbol{\alpha}_k^*(v)\|_\infty \le \frac{1}{2}(\widetilde{\varepsilon} + L_\rho\sqrt{2}\,\|\mathbf{\Theta}^{[K-1]}\|_F) \le C'\widetilde{\varepsilon}$ for some absolute constant C', with the polynomial Lipschitz bound:

$$\sup_v \sup_\lambda |p_v(\lambda) - p_v^*(\lambda)| \le C_K(B,\delta)\cdot C'\widetilde{\varepsilon}.$$

Choosing $\widetilde{\varepsilon} = \varepsilon/(3\,C_K(B,\delta)\,C')$ gives $\sup_v\sup_\lambda |p_v(\lambda) - p_v^*(\lambda)| \le \varepsilon/3$. By hypothesis (H3), $\sup_v\sup_\lambda |p_v^*(\lambda) - f_v(\lambda)| \le \varepsilon/3$. Triangle inequality then gives $\sup_v\sup_\lambda |p_v(\lambda) - f_v(\lambda)| \le 2\varepsilon/3 < \varepsilon$, completing the proof. □

Remark 13 (On the trajectory non-degeneracy hypothesis (H3)). Hypothesis (H3) requires the existence of *at least one* target schedule $\boldsymbol{\theta}^*$ that simultaneously *(a)* approximates the per-node targets $\{f_v\}$ and *(b)* produces a non-degenerate trajectory whose Router inputs $\bar{\mathbf{u}}_k(v)$ are pairwise distinct across (v,k). Generically, this is satisfied: the set of schedules producing degenerate trajectories *(*e.g. pure-identity routing for some v, where $\mathbf{\Delta Z}_v^{[k]} = \mathbf{\Delta Z}_v^{[0]}$ for all k*)* is a measure-zero subset of the parameter space $(\mathcal{T}\times\mathbb{R})^{KN}$. Among the schedules that approximate $\{f_v\}$ within $\varepsilon/3$, the subset producing non-degenerate trajectories is open and dense, so picking $\boldsymbol{\theta}^*$ generically inside this set realizes (H3) with μ' bounded below by a δ- and K-dependent constant.

Remark 14 (On the constant in the width bound). The dependence $H_\rho \propto (\mu')^{-2C}$ in (26) is the standard exponential-in-dimension penalty of MLP universal approximation *(*the "curse of dimensionality" for sigmoidal networks; Barron [1993, Theorem 3]*)*. In practice the dependence is much milder because the target function $\widetilde{\phi}$ has low intrinsic dimension *(*it is determined by the C-dimensional initial states $\mathbf{Z}_v^{(0)}$ alone, not by all $2C$ Router-input coordinates*)*; finer bounds via Barron-class approximation give $H_\rho = \mathcal{O}(NK/\widetilde{\varepsilon}^2)$ when $\widetilde{\phi}$ has bounded Fourier spectral density, removing the $(\mu')^{-2C}$ factor. The numerics of Section 5 use $H_\rho = 64$ throughout, which is the practical regime for the dataset sizes considered.

Remark 15 (Comparison with prior per-node mixers). Theorem 12 formalizes CASTOR's ability to assign different polynomial filters to different nodes within a single forward pass. No uniform spectral filter [He et al., 2021, Wang and Zhang, 2022, He et al., 2022] can do this. ACM-GCN [Luan et al., 2022] mixes per node but does so once per layer, so its per-node polynomials are products of L degree-1 polynomials and the cross-step structure of three-term recurrences (23) is unavailable; moreover its per-branch weight matrices break the spectral interpretation across layers, so its function class is not contained in any NCSF.

F.4 Dirichlet-energy evolution and over-smoothing

We now study how $\mathcal{E}(\mathbf{Z}^{[k]})$ evolves under the recurrence (18). The over-smoothing literature [Nt and Maehara, 2019, Oono and Suzuki, 2019, Di Giovanni et al., 2023] characterizes representation collapse via exponential decay of $\mathcal{E}$. We give a tight per-mode formula and use it to characterize exactly when collapse occurs.

Lemma 16 (Per-mode recurrence)**.** *Suppose the routing is node-uniform with $T_k = m_k\widehat{\mathbf{W}} + c_k\mathbf{I}$. Apply $\mathbf{U}^\top$ to both sides of* (18)*: the spectral coefficient at frequency λ_i obeys the scalar second-order recurrence*

$$\widetilde{\mathbf{Z}}_{i,:}^{[k]} \;=\; (m_k\lambda_i + c_k + \beta_k)\,\widetilde{\mathbf{Z}}_{i,:}^{[k-1]} \;-\; \beta_k\,\widetilde{\mathbf{Z}}_{i,:}^{[k-2]}, \tag{28}$$

with the convention $\widetilde{\mathbf{Z}}^{[-1]} = \widetilde{\mathbf{Z}}^{[0]}$.

Proof. $\mathbf{U}^\top T_k\mathbf{Z}^{[k-1]} = \mathbf{U}^\top(m_k\widehat{\mathbf{W}} + c_k\mathbf{I})\mathbf{Z}^{[k-1]} = (m_k\mathbf{\Lambda} + c_k\mathbf{I})\widetilde{\mathbf{Z}}^{[k-1]}$, which has ith row $(m_k\lambda_i + c_k)\widetilde{\mathbf{Z}}_{i,:}^{[k-1]}$. The momentum term becomes $\beta_k(\widetilde{\mathbf{Z}}_{i,:}^{[k-1]} - \widetilde{\mathbf{Z}}_{i,:}^{[k-2]})$. Summing gives (28). □

Theorem 17 (Closed form for the energy under uniform routing, $\beta \equiv 0$)**.** *Suppose at every step the routing is node-uniform and $\beta_k \equiv 0$. Then for every $K \geq 0$,*

$$\mathcal{E}(\mathbf{Z}^{[K]}) \;=\; \sum_{i=1}^{N}(1-\lambda_i)\Big(\prod_{k=1}^{K}(m_k\lambda_i + c_k)\Big)^2 \|\widetilde{\mathbf{Z}}^{(0)}_{i,:}\|_2^2. \tag{29}$$

The fraction $\mathcal{E}(\mathbf{Z}^{[K]})/\mathcal{E}(\mathbf{Z}^{(0)})$ is a weighted average of $\{(\prod_k(m_k\lambda_i+c_k))^2\}_{i:\,\lambda_i<1}$ with non-negative weights $(1-\lambda_i)\|\widetilde{\mathbf{Z}}^{(0)}_{i,:}\|_2^2$ summing to $\mathcal{E}(\mathbf{Z}^{(0)})$.

Proof. By Lemma 16 with $\beta_k = 0$, the per-mode recurrence is $\widetilde{\mathbf{Z}}^{[k]}_{i,:} = (m_k\lambda_i + c_k)\widetilde{\mathbf{Z}}^{[k-1]}_{i,:}$, so $\widetilde{\mathbf{Z}}^{[K]}_{i,:} = \prod_{k=1}^{K}(m_k\lambda_i + c_k)\,\widetilde{\mathbf{Z}}^{(0)}_{i,:}$. Plugging into (17) gives (29). The weighted-average claim is direct. □

Corollary 18 (Smoothing causes exponential collapse)**.** *If $\boldsymbol{\alpha}_k = (1,0,0)$ (equivalently $(m_k, c_k) = (1,0)$) at every step and $\beta_k \equiv 0$, then*

$$\mathcal{E}(\mathbf{Z}^{[K]}) \;=\; \sum_{i=1}^{N}(1-\lambda_i)\,\lambda_i^{2K}\,\|\widetilde{\mathbf{Z}}^{(0)}_{i,:}\|_2^2.$$

Provided $\widehat{\mathbf{W}}$ has spectral gap $1-|\lambda_2| > 0$ (where λ_2 is the largest non-trivial eigenvalue in absolute value, possibly 1 in the kernel-of-Dirichlet sense),

$$\mathcal{E}(\mathbf{Z}^{[K]}) \;\leq\; |\lambda_2|^{2K}\,\mathcal{E}(\mathbf{Z}^{(0)}),$$

which decays at the explicit exponential rate $|\lambda_2|^{2K} \to 0$.

Proof. With $(m_k, c_k) = (1,0)$, $m_k\lambda_i + c_k = \lambda_i$, so $\prod_k(m_k\lambda_i + c_k) = \lambda_i^K$. The energy is $\sum_i(1-\lambda_i)\lambda_i^{2K}\|\widetilde{\mathbf{Z}}^{(0)}_{i,:}\|^2$. The contribution from $\lambda_i = 1$ vanishes *(*factor $(1-\lambda_i) = 0$*)*; contributions from $\lambda_i < 1$ are bounded by $|\lambda_2|^{2K}(1-\lambda_i)\|\widetilde{\mathbf{Z}}^{(0)}_{i,:}\|^2$, which sums to $|\lambda_2|^{2K}\mathcal{E}(\mathbf{Z}^{(0)})$. □

Corollary 19 (Identity preserves energy exactly)**.** *If $\boldsymbol{\alpha}_k = (0,0,1)$ (equivalently $(m_k, c_k) = (0,1)$) at every step and $\beta_k \equiv 0$, then $\mathbf{Z}^{[K]} = \mathbf{Z}^{(0)}$ for every K, and $\mathcal{E}(\mathbf{Z}^{[K]}) = \mathcal{E}(\mathbf{Z}^{(0)})$.*

Proof. $T_k = \mathbf{I}$, and $\beta_k = 0$ kills the momentum term, so the recurrence reduces to $\mathbf{Z}^{[k]} = \mathbf{Z}^{[k-1]}$. □

Theorem 20 (No-collapse via residual or identity)**.** *Suppose at every step the routing is node-uniform with $\boldsymbol{\alpha}_k \in \Delta_3$ and $\beta_k = 0$. If there exists $\eta > 0$ such that $\alpha_k^{id} \geq \eta$ for every k, then for every K,*

$$\mathcal{E}(\mathbf{Z}^{[K]}) \;\geq\; \eta^{2K}\,\mathcal{E}(\mathbf{Z}^{(0)}). \tag{30}$$

If instead $\alpha_k^{hi} \geq \eta$ for every k and $\sigma(\widehat{\mathbf{W}}) \subseteq [-1, 1-2\eta]$ (so that the high-pass primitive has spectral coefficient $1-\lambda \geq 2\eta$), then (30) *again holds.*

Proof. With node-uniform routing and $\beta_k = 0$, the per-mode recurrence (28) simplifies to $\widetilde{\mathbf{Z}}^{[k]}_{i,:} = (m_k\lambda_i + c_k)\widetilde{\mathbf{Z}}^{[k-1]}_{i,:}$ and the per-mode contraction at frequency λ_i is the absolute value $|m_k\lambda_i + c_k|$. We bound this from below in each case.

Identity-dominant case. $\alpha_k^{\mathrm{id}} \geq \eta$ implies that the convex combination $T_k\mathbf{Z} = \alpha_k^{\mathrm{sm}}\widehat{\mathbf{W}}\mathbf{Z} + \alpha_k^{\mathrm{hi}}(\mathbf{I} - \widehat{\mathbf{W}})\mathbf{Z} + \alpha_k^{\mathrm{id}}\mathbf{Z}$ has eigenvalue at λ_i given by $\alpha_k^{\mathrm{sm}}\lambda_i + \alpha_k^{\mathrm{hi}}(1-\lambda_i) + \alpha_k^{\mathrm{id}}$. Among the three terms, the third equals $\alpha_k^{\mathrm{id}} \geq \eta$. Each of the first two has absolute value at most $|\lambda_i|$ and $|1-\lambda_i|$ respectively, both bounded by ≤ 2. Triangle inequality gives $|m_k\lambda_i + c_k| \geq \alpha_k^{\mathrm{id}} - \alpha_k^{\mathrm{sm}}|\lambda_i| - \alpha_k^{\mathrm{hi}}|1-\lambda_i|$. For $|\lambda_i| < 1$, taking η small enough so that $\eta(1 + |\lambda_i| + |1-\lambda_i|) \leq \eta \cdot 4$, we get $|m_k\lambda_i + c_k| \geq \eta$ for η in a sufficiently small window. We conclude $|m_k\lambda_i + c_k|^{2K} \geq \eta^{2K}$, and inserting into (29) yields (30).

Residual-dominant case. If $\alpha_k^{\mathrm{hi}} \geq \eta$ and $\sigma(\widehat{\mathbf{W}}) \subseteq [-1, 1-2\eta]$, then $1-\lambda_i \geq 2\eta$ for every $\lambda_i \in \sigma(\widehat{\mathbf{W}})$. Hence $|\alpha_k^{\mathrm{hi}}(1-\lambda_i)| \geq \eta \cdot 2\eta = 2\eta^2$ contributes to the lower bound, and a similar triangle inequality gives $|m_k\lambda_i + c_k| \geq \eta$ in that range. The conclusion is the same. □

Remark 21 (Compatibility with momentum). For $\beta_k \neq 0$, the per-mode recurrence has companion matrix $\mathbf{C}_{k,i} = \left[\begin{smallmatrix} m_k\lambda_i + c_k + \beta_k & -\beta_k \\ 1 & 0 \end{smallmatrix}\right]$ with eigenvalues $r_\pm = \frac{(m_k\lambda_i + c_k + \beta_k) \pm \sqrt{(m_k\lambda_i + c_k + \beta_k)^2 - 4\beta_k}}{2}$ and product $r_+ r_- = \beta_k$. Hence the geometric mean of the two roots' absolute values equals $\sqrt{|\beta_k|}$, and at each step the spectral content at frequency λ_i contracts by at most $\sqrt{|\beta_k|}$ in geometric mean. For any $|\beta_k| \geq \eta > 0$, this provides an additional independent mechanism preventing collapse: $\mathcal{E}(\mathbf{Z}^{[K]}) \geq \eta^K \mathcal{E}(\mathbf{Z}^{(0)})$ in the appropriate basis.

F.5 Sensitivity bounds and over-squashing

We bound the Jacobian $\partial \mathbf{Z}_v^{[K]} / \partial \mathbf{Z}_u^{(0)}$ that controls how much a perturbation of node u's initial state propagates to node v at depth K. Following Topping et al. [2022], Di Giovanni et al. [2023], Black et al. [2023], this Jacobian is the standard quantity for diagnosing over-squashing.

Theorem 22 (Jacobian under uniform routing)**.** *Suppose the routing is node-uniform with $(m_k, c_k) \in \mathcal{T}$ and $\beta_k \in \mathbb{R}$. Let P_K be the polynomial realized by the schedule per Theorem 5. Then for every $u, v \in \mathcal{V}$,*

$$\frac{\partial \mathbf{Z}_{v,c}^{[K]}}{\partial \mathbf{Z}_{u,c'}^{(0)}} \;=\; \big[P_K(\widehat{\mathbf{W}})\big]_{vu}\, \delta_{cc'}, \qquad c, c' \in \{1, \ldots, C\}, \tag{31}$$

so the operator norm of the per-node Jacobian block is $\big|[P_K(\widehat{\mathbf{W}})]_{vu}\big|$. In particular, if $u \neq v$ and there is no walk of length $\leq K$ between u and v in $\mathcal{G}$, then the diagonal of $\widehat{\mathbf{W}}^j$ at (v, u) is zero for $j \leq K$ and the only contribution to $[P_K(\widehat{\mathbf{W}})]_{vu}$ comes from terms in P_K of degree $\geq K + 1$, which are zero by construction; hence the Jacobian vanishes.

Proof. Theorem 5 gives $\mathbf{Z}^{[K]} = P_K(\widehat{\mathbf{W}})\mathbf{Z}^{(0)}$, a linear map applied separately to each of the C channels of $\mathbf{Z}^{(0)}$. The partial derivative of the (v, c) entry of the output with respect to the (u, c') entry of the input is the (v, u) entry of $P_K(\widehat{\mathbf{W}})$ when $c = c'$ and zero otherwise, giving (31). Walk-counting: $[\widehat{\mathbf{W}}^j]_{vu} \neq 0$ requires a walk of length j from u to v. □

Theorem 23 (Effective-resistance bound)**.** *Let $A = P_K(\widehat{\mathbf{W}})$ be the linear operator under uniform routing. Then for every $u, v \in \mathcal{V}$,*

$$\sum_{k=0}^{K} \big|[\widehat{\mathbf{W}}^k]_{vu}\big| \;\leq\; \frac{C_K}{R(u, v)} \tag{32}$$

for an explicit constant C_K depending only on K and the spectral profile of $\widehat{\mathbf{W}}$.

The bound (32) is a verbatim restatement of Black et al. [2023]'s Theorem 3.6, which we cite without re-proof: large effective resistance forces the cumulative walk weight $\sum_k |[\widehat{\mathbf{W}}^k]_{vu}|$ to be small. Combined with (31), it gives a precise constant on the over-squashing decay of CASTOR under *uniform* routing: the Jacobian is bounded above by a polynomial-weighted sum of $|[\widehat{\mathbf{W}}^k]_{vu}|$, hence by $C_K / R(u, v)$. The next theorem establishes that per-node routing breaks this resistance dependence for the subset of nodes that route to identity early.

Theorem 24 (Per-node over-squashing relief)**.** *Fix any $K \geq 1$ and any non-empty subset $S \subseteq \mathcal{V}$. There exists a per-node routing schedule $\{\mathbf{a}_k(\cdot)\}_{k=1}^K$ with $\mathbf{a}_k(v) = (0, 0, 1)$ (identity primitive) and $\beta_k = 0$ for every $v \in S$ and every $k > K_*$, where $K_* < K$ is a fixed cutoff, such that the operator A produced by the* CASTOR *block satisfies*

$$\frac{\partial \mathbf{Z}_{v,c}^{[K]}}{\partial \mathbf{Z}_{u,c'}^{(0)}} \;=\; \big[P_{K_*}^{(v)}(\widehat{\mathbf{W}})\big]_{vu}\, \delta_{cc'} \qquad \forall v \in S,\ \forall u \in \mathcal{V}, \tag{33}$$

where $P_{K_*}^{(v)}$ is a polynomial of degree $\leq K_*$ depending only on $\mathbf{a}_1(v), \ldots, \mathbf{a}_{K_*}(v)$. In particular, the right-hand side of (33) *is independent of K for every $K \geq K_*$, and the resulting sensitivity is bounded by $C_{K_*} / R(u, v)$ rather than by $C_K / R(u, v)$.*

Proof. For nodes $v \in S$, the routing past step K_* is $\mathbf{a}_k(v) = \mathbf{e}_3$, so the per-step routed update at v is the identity: $\mathbf{Z}_v^{[k]} = \mathbf{Z}_v^{[k-1]}$ for $k > K_*$ (the momentum term vanishes since $\beta_k = 0$). Hence $\mathbf{Z}_v^{[K]} = \mathbf{Z}_v^{[K_*]}$ for every $K \geq K_*$, and the Jacobian at v is the same as the Jacobian of the depth-K_* block. The polynomial $P_{K_*}^{(v)}$ is determined by the routing-and-momentum schedule restricted to $\{1, \ldots, K_*\}$. □

Remark 25. Theorem 24 formalizes the per-node empirical pattern of Section 2.1: classes whose discriminative information is shallow (small K_*) should route to identity once that information is gathered, decoupling their sensitivity from depth K. A uniform spectral filter cannot do this — the polynomial P_K applies to every node identically, so its sensitivity at every node is bounded by $C_K / R(u, v)$ with the same K.

F.6 Stability under graph perturbation and the second-order region

We close with two stability results: a first-order perturbation bound under graph changes, and an explicit non-expansive region in the second-order recurrence parameters.

Theorem 26 (First-order graph-perturbation bound)**.** *Let $\widehat{\mathbf{W}}$ and $\widehat{\mathbf{W}}'$ be two symmetric normalized graph shift operators with $\|\widehat{\mathbf{W}} - \widehat{\mathbf{W}}'\|_{\mathrm{op}} \leq \varepsilon$. Suppose a uniform-routing* CASTOR *schedule with $(m_k, c_k) \in \mathcal{T}$ and $\beta_k \in \mathbb{R}$, $|\beta_k| \leq B$. Let $\mathbf{Z}^{[K]}$ and $\mathbf{Z}'^{[K]}$ be the outputs from the same initial state $\mathbf{Z}^{(0)}$ under $\widehat{\mathbf{W}}$ and $\widehat{\mathbf{W}}'$ respectively. Define*

$$L_K \;:=\; \sum_{k=1}^{K} \big(|m_k| + 2|\beta_k| + |c_k|\big) \cdot \prod_{j=k+1}^{K} \big(|m_j| + |c_j| + |\beta_j| + |\beta_{j-1}|\big).$$

Then $\big\|\mathbf{Z}^{[K]} - \mathbf{Z}'^{[K]}\big\|_F \leq L_K \cdot \varepsilon \cdot \|\mathbf{Z}^{(0)}\|_F + O(\varepsilon^2)$.

Proof. Define the difference $\mathbf{\Delta}^{[k]} := \mathbf{Z}^{[k]} - \mathbf{Z}'^{[k]}$ and $\delta T_k := T_k - T_k' = m_k(\widehat{\mathbf{W}} - \widehat{\mathbf{W}}')$ (since the routing parameters are the same). Subtracting the two recurrences:

$$\mathbf{\Delta}^{[k]} = T_k \mathbf{\Delta}^{[k-1]} + \delta T_k \, \mathbf{Z}'^{[k-1]} + \beta_k \big(\mathbf{\Delta}^{[k-1]} - \mathbf{\Delta}^{[k-2]}\big).$$

Iterating from $\mathbf{\Delta}^{[0]} = 0$ and using $\|T_k\|_{\mathrm{op}} \leq |m_k| + |c_k|$ from Lemma 3 and $\|\delta T_k\|_{\mathrm{op}} \leq |m_k|\varepsilon$, taking norms via the triangle inequality, we get $\|\mathbf{\Delta}^{[K]}\|_F \leq L_K \varepsilon \|\mathbf{Z}^{(0)}\|_F + O(\varepsilon^2)$. The cross-product terms in L_K collect the contributions of δT_k at step k propagated through subsequent operators, with β-amplification accumulating across the recurrence. □

The bound is linear in ε. The constant L_K depends only on the routing-and-momentum schedule, which is bounded automatically because $(m_k, c_k) \in \mathcal{T}$ implies $|m_k| + |c_k| \leq 2$ and the softmax in the Router enforces $\boldsymbol{\alpha}_k \in \Delta_3$.

Theorem 27 (Non-expansive region of the second-order recurrence)**.** *Consider the per-mode recurrence*

$$x_k \;=\; (\xi + \beta)\, x_{k-1} \;-\; \beta\, x_{k-2}, \qquad k \geq 2, \quad \xi, \beta \in \mathbb{R}, \tag{34}$$

with arbitrary initial conditions $x_0, x_1 \in \mathbb{R}$. The recurrence is non-expansive — i.e., $\sup_k |x_k| \leq \max(|x_0|, |x_1|)$ — if and only if $(\xi, \beta) \in \mathcal{S}$, where

$$\mathcal{S} \;:=\; \big\{(\xi, \beta) \in \mathbb{R}^2 \;:\; |\beta| \leq 1,\; |\xi + \beta| \leq 1 + \beta\big\}. \tag{35}$$

Inside $\mathcal{S}$, the regimes are:

1. ***First-order.*** *$\beta = 0$: $\mathcal{S} \cap \{\beta = 0\} = \{|\xi| \leq 1\}$, the first-order contraction interval.*
2. ***Acceleration.*** *$0 < \beta < 1$: ξ may lie in $[-1 - 2\beta, 1]$, extending below -1. This is the heavy-ball regime [Polyak, 1964], in which the iteration converges to the dominant eigenmode at rate $\sqrt{\beta}$.*
3. ***Damping.*** *$-1 < \beta < 0$: ξ must lie in $[-1 + 2|\beta|, 1]$, with the lower endpoint moving* inward*. Trajectories near $\xi = -1$ are dampened back into the strict interior of $\mathcal{S}$.*

Proof. The characteristic polynomial of (34) is $r^2 - (\xi + \beta)r + \beta = 0$ with roots $r_\pm = \frac{(\xi+\beta)\pm\sqrt{(\xi+\beta)^2-4\beta}}{2}$. By Vieta, $r_+r_- = \beta$ and $r_+ + r_- = \xi + \beta$. The recurrence is non-expansive iff both roots have $|r_\pm| \le 1$. For real distinct roots, this is equivalent to the Schur–Cohn conditions *(*for a degree-2 monic with constant term β and linear term $-(\xi + \beta)$*)*:

$$|r_+r_-| \le 1 \iff |\beta| \le 1, \qquad |r_+ + r_-| \le 1 + r_+r_- \iff |\xi + \beta| \le 1 + \beta.$$

For complex-conjugate roots, $|r_\pm| = \sqrt{r_+\bar{r}_-} = \sqrt{|r_+r_-|} = \sqrt{|\beta|}$, so $|r_\pm| \le 1 \iff |\beta| \le 1$. The cases match. The case-wise interpretations follow by substituting $\beta = 0$, $\beta > 0$, and $\beta < 0$ into the boundary $|\xi + \beta| = 1 + \beta$. □

Remark 28 (Why CASTOR learns negative β on dense heterophilic graphs). Theorem 27 explains the empirical pattern of Section 2.2. On dense heterophilic graphs whose normalized adjacency has eigenvalues approaching -1, the per-mode effective frequency $\xi = m\lambda + c$ is close to -1. With $\beta = 0$, the marginal stability $|\xi| \le 1$ is just barely met, and discrete iterates oscillate on the boundary. Setting $\beta < 0$ pulls the lower endpoint of the stability region inward to $-1 + 2|\beta|$, dampening the oscillation. Conversely, sparse heterophilic graphs whose smoothing dynamics converge slowly along the dominant eigenvector benefit from $\beta > 0$: the iteration becomes equivalent to the heavy-ball method with rate $\sqrt{\beta}$, faster than the first-order rate when $\beta \in (0, 1)$.

F.7 Special-case recovery and structural exclusions

Proposition 29 (Exact recovery of prior architectures)**.** *The following architectures are exact instances of* CASTOR *for specific routing-and-momentum schedules:*

1. **SGC** *[Wu et al., 2019].* $\boldsymbol{\alpha}_k = (1, 0, 0)$ *(equivalently* $(m_k, c_k) = (1, 0)$*) and* $\beta_k = 0$ *for all* k*. Output:* $\mathbf{Z}^{[K]} = \widehat{\mathbf{W}}^K\mathbf{Z}^{(0)}$.

2. **APPNP** *[Gasteiger et al., 2018].* $\boldsymbol{\alpha}_k = (1 - \alpha, 0, \alpha)$ *(equivalently* $(m_k, c_k) = (1 - \alpha, \alpha)$*) for* $\alpha \in (0, 1)$ *and* $\beta_k = 0$*, with* $\mathbf{Z}^{(0)}$ *acting as the personalization vector. Output:* $\mathbf{Z}^{[K]} = (1-\alpha)^K\widehat{\mathbf{W}}^K\mathbf{Z}^{(0)} + \sum_{k=0}^{K-1}\alpha(1-\alpha)^k\widehat{\mathbf{W}}^k\mathbf{Z}^{(0)}$*, the truncated personalized PageRank propagation.*

3. **GPRGNN** *[Chien et al., 2020]. With* $\beta_k = 0$*, the simplex products* $\prod_k(m_k\widehat{\mathbf{W}} + c_k\mathbf{I})$ *realize every degree-*K *polynomial in* $\widehat{\mathbf{W}}$ *that factors over* $\mathcal{T}$*, including a strict subset of the polynomials reachable by* GPRGNN*. Specifically, the sub-class with non-negative coefficients summing to* 1 *is reachable.*

4. **BERNNET** *[He et al., 2021]. Bernstein polynomials with non-negative coefficients summing to* 1 *on the rescaled spectrum* $(\widehat{\mathbf{W}} + \mathbf{I})/2 \in [0, 1]$ *are products of degree-*1 *factors of the form* $1 - x$ *and* x*, hence elements of* $\mathcal{P}_K^{\text{simp}}$ *via the change of basis* $(m, c) = (1, 0)$ *and* $(m, c) = (-1, 1)$ *respectively.*

5. ***Rescaled-spectrum*** **CHEBNET** *[Defferrard et al., 2016]. Setting* $(m_k, c_k) = (1, 0)$ *on* $\widehat{\mathbf{W}}/2$ *and* $\beta_k = 1$ *for* $k \ge 2$ *yields a three-term recurrence in the rescaled spectrum, matching the algebraic family (though not the exact coefficients) of Chebyshev-type recurrences (cf. Remark 6).*

Proof. Direct substitution into Eqs. (18) and (23) for each item, using the change of variables (20) between $\boldsymbol{\alpha} \in \Delta_3$ and $(m, c) \in \mathcal{T}$. □

Remark 30 (Architectures *not* contained). A small group of architectures is structurally outside CASTOR's reachable set:

- **ACM-GCN** [Luan et al., 2022]. ACM per-layer applies three *distinct* learned weight matrices $\mathbf{W}_{\text{low}}, \mathbf{W}_{\text{high}}, \mathbf{W}_{\text{mlp}}$ with non-linearities to each branch and mixes them with per-node attention; the per-branch transformations break the spectral interpretation across layers. CASTOR holds zero per-channel weights inside the propagation block and applies no non-linearities mid-recurrence. The two architectures share the abstract idea of a per-node low/high/identity mixture but realize it in incompatible function classes.

- **MIXHOP** [Abu-El-Haija et al., 2019]. MIXHOP concatenates $[\widehat{\mathbf{W}}^0\mathbf{X}; \ldots; \widehat{\mathbf{W}}^k\mathbf{X}]$ across hops, lifting the state dimension and applying per-hop weight matrices; CASTOR keeps the state dimension fixed at C.
- **JK-NET** [Xu et al., 2018]. JK-NET aggregates intermediate states $\{\mathbf{Z}^{[k]}\}_{k\geq 1}$ at the end via concatenation, max-pool, or LSTM. CASTOR uses only $\mathbf{Z}^{[K]}$ as final output.
- **FA-GCN** [Bo et al., 2021]. FA-GCN learns per-edge gating coefficients in $[-1, 1]$, yielding a graph-dependent shift operator that is not a polynomial in $\widehat{\mathbf{W}}$. CASTOR keeps $\widehat{\mathbf{W}}$ fixed and varies only $\boldsymbol{\alpha}$ and β.

The exclusions are deliberate design choices: CASTOR is the spectrally-honest, weight-free, fixed-shift counterpart to a family that combines per-node mixing with non-linear branches or learnable shifts. Theorem 12 shows that this counterpart loses no expressivity in the per-node sense.

F.8 The CASTOR block as a discretized port-Hamiltonian system

Remark 31 (ODE bridge). Let $t_k = k\Delta t$ and treat $\mathbf{Z}^{[k]}$ as samples of a continuous trajectory $\mathbf{Z}(t_k)$. Substituting $\mathbf{Z}^{[k-1]} - \mathbf{Z}^{[k-2]} \approx \Delta t\, \dot{\mathbf{Z}}(t_{k-1})$ and $\mathbf{Z}^{[k]} - 2\mathbf{Z}^{[k-1]} + \mathbf{Z}^{[k-2]} \approx \Delta t^2 \ddot{\mathbf{Z}}(t_{k-1})$ into (18), and writing $T_k \approx \mathbf{I} + \Delta t^2\, \mathbf{F}(t_{k-1})$ for some operator $\mathbf{F}$, gives in the limit $\Delta t \to 0$ with $\beta_k \sim 1 - \gamma\Delta t$, $\gamma \geq 0$:

$$\ddot{\mathbf{Z}}(t) \;+\; \gamma\, \dot{\mathbf{Z}}(t) \;=\; \mathbf{F}\, \mathbf{Z}(t), \tag{36}$$

the equation of motion of a unit-mass particle in $\mathbb{R}^{N\times C}$ subject to a position-dependent force $\mathbf{F}\,\mathbf{Z}$ and viscous damping $\gamma\,\dot{\mathbf{Z}}$. The Hamiltonian is $\mathcal{H}(\mathbf{Z}, \dot{\mathbf{Z}}) = \frac{1}{2}\|\dot{\mathbf{Z}}\|_F^2 - \frac{1}{2}\langle \mathbf{Z}, \mathbf{F}\,\mathbf{Z}\rangle_F$. The three regimes of β_k in Section 4.2 correspond to:

1. **Underdamped** *($\beta_k > 0$).* Oscillatory trajectories with accelerated convergence in the smoothing eigenmode [Rusch et al., 2022].
2. **Critically damped or overdamped** *($\beta_k = 0$).* Monotone exponential decay to the equilibrium of $\mathbf{F}$, the standard gradient flow on the Dirichlet energy [Di Giovanni et al., 2023].
3. **Negative damping** *($\beta_k < 0$).* Adds an extra contraction beyond the conservative dynamics, useful when a discrete iteration would otherwise overshoot at the chosen Δt.

This places CASTOR in the broader port-Hamiltonian-on-graphs family [Heilig et al., 2025], with the additional feature — absent in those architectures — that the routed step T_k varies with the state and across the trajectory and the damping coefficient β_k varies in sign per step.

F.9 Summary

The framework establishes:

- **Realization geometry.** CASTOR with $\beta \equiv 0$ and uniform routing realizes the simplex-product polynomial family $\mathcal{P}_K^{\mathrm{simp}}$ *(*Thm. 4*)*; turning on β enlarges this to $\mathcal{P}_K^{\mathrm{rec}} \supsetneq \mathcal{P}_K^{\mathrm{simp}}$ via three-term recurrences *(*Thm. 5*)*; per-node routing leaves USF_K for NCSF_K *(*Thm. 7*)*, with at least N extra degrees of freedom *(*Cor. 8*)*.
- **Approximation.** CASTOR interpolates any prescribed values on a discrete spectrum within the simplex-product family *(*Thm. 11*)*, and achieves per-node interpolation under mild input-separation hypotheses *(*Thm. 12*)*.
- **Energetics.** The Dirichlet energy admits a closed-form per-mode evolution *(*Thm. 17*)*; uniform smoothing causes exponential collapse at rate $|\lambda_2|^{2K}$ *(*Cor. 18*)*, while identity preserves it exactly *(*Cor. 19*)*; a positive lower bound on α^{id} *(*or on α^{hi} paired with appropriate spectrum*)* prevents collapse *(*Thm. 20*)*, and non-zero $|\beta|$ provides an independent mechanism *(*Rem. 21*)*.
- **Sensitivity.** The uniform-routing Jacobian is $|[P_K(\widehat{\mathbf{W}})]_{vu}|$ and obeys the effective-resistance bound of Black et al. [2023] *(*Thms. 22, 23*)*. Per-node routing decouples the sensitivity from depth K for the subset of nodes that route to identity early *(*Thm. 24*)*.

- **Stability.** A first-order ε-bound under graph perturbation *(*Thm. 26*)*; the second-order recurrence is non-expansive on the explicitly characterized region $\mathcal{S} = \{(\xi, \beta) : |\beta| \leq 1, |\xi + \beta| \leq 1 + \beta\}$ *(*Thm. 27, Rem. 28*)*.
- **Recovery and exclusion.** SGC, APPNP, GPRGNN, BERNNET, and rescaled-spectrum CHEBNET are exact instances *(*Prop. 29*)*; ACM-GCN, MIXHOP, JK-NET, FA-GCN are not, for structural reasons that are precisely identified *(*Rem. 30*)*.
- **Continuum interpretation.** The block is a finite-difference discretization of a damped second-order ODE, placing CASTOR in the port-Hamiltonian-on-graphs family with sign-flexible per-step damping *(*Rem. 31*)*.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state four contributions, each of which is supported in the body of the paper: (1) two empirical axes along which the right graph filter varies on heterophilic benchmarks — *across nodes within a hop* and *across hops within a trajectory* — together with the structural observation that no fixed-coefficient polynomial in $\widehat{\mathbf{W}}$ can adapt to either (Section 2); (2) the CASTOR architecture, in which every step is a per-node convex combination of three elementary primitives (smoothing, residual, identity) chosen by a shared router and iterated as a second-order recurrence with a learnable, sign-flexible per-step momentum coefficient (Sections 4.1–4.2, Eqs. (1)–(6)); (3) a strict expressiveness statement showing that the resulting filter family contains every fixed-coefficient polynomial filter of the same depth and exits the polynomial family altogether once per-node routing is enabled, formalized in Theorems 4, 5, 7 and Proposition 29 of Appendix F; and (4) empirical results in Table 1 showing that a single instantiation of CASTOR attains the best average rank (2.07, more than 3.5 ranks ahead of the second-best architecture) on fourteen standard node-classification benchmarks spanning the full homophily-heterophily spectrum (Section 5).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.

- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The closing paragraph of Section 6 acknowledges three concrete limitations of the present work: (i) the experiments are confined to transductive node classification on undirected graphs — behavior under inductive settings, link prediction, graph-level prediction, or large distribution shift is not directly addressed; (ii) although the propagation block's per-step memory is constant in the router depth K (Appendix B), latency grows linearly with K, which may matter at the largest scales; and (iii) the spectral interpretation and the structural results of Appendix F are tied to the symmetric normalized propagation $\widehat{\mathbf{W}}$, so extensions to attention-weighted, signed, or otherwise learnable propagation operators are left to future work.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All formal results are stated, assumed, and proved in Appendix F, organized around the node-conditional spectral filter (NCSF) family introduced in Appendix F.1. The realization results characterize the reachable filter set (Lemma in Appendix F.2; Theorems 4, 5, 7, with Corollary 8); spectral universality is treated in Appendix F.3 (Theorems 11, 12); the Dirichlet-energy analysis is in Appendix F.4 (Theorem 17, Corollaries 18, 19, Theorem 20); the sensitivity / over-squashing analysis is in Appendix F.5 (Theorem 22 et seq.); and the stability analysis is in Appendix F.6, with prior-architecture embeddings as exact special cases in Appendix F.7 (Proposition 29). Each theorem statement names its assumptions in the hypothesis (e.g., uniform routing, $\beta \equiv 0$, separating inputs,

trajectory non-degeneracy), the proofs are self-contained in the same appendix, and every result invoked in the main text (Theorem 20, Theorem 7, Proposition 29, Corollary 18) is cross-referenced.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 4 specifies the full CASTOR architecture: the routing kernel (Eqs. (1)–(4)), the smoothing-then-preserve initialisation of the per-step bias (Eq. (5)), the inertial second-order recurrence with sign-flexible per-step momentum (Eq. (6)), and the anchor encoder and classifier head (Eqs. (8)–(10)). Section 5 ("Configuration and setup" and "Training recipe") documents the implementation framework (PyTorch Geometric), the hardware (NVIDIA A100 80 GB GPU, CUDA 12.8), the common two-layer backbone with fair ~ 64 effective hidden dimension on the Platonov datasets, the per-(architecture, dataset) hyperparameter sweep, the evaluation protocol — the BernNet-style $10 \times 48/32/20$ splits of Zhu et al. [2020] for the nine small datasets of Pei et al. [2020] (Cora, CiteSeer, PubMed, Texas, Wisconsin, Cornell, Actor, Squirrel, Chameleon) and the official $10 \times 50/25/25$ splits of Platonov et al. [2023] for the five Platonov mid-scale datasets (Roman-empire, Amazon-ratings, Minesweeper, Tolokers, Questions) — and the reporting protocol (mean $\pm$ 95% non-parametric bootstrap CI over $1{,}000$ resamples across the ten splits, as in Platonov et al. [2023]). Appendix C expands the protocol with the per-architecture fair-capacity audit (Appendix C.2), the explicit hyperparameter grids (Appendix C.3), the AdamW training loop with $1{,}000$-epoch budget and 200-epoch validation patience (Appendix C.4), and the bootstrap reporting protocol (Appendix C.5). All datasets are public benchmarks loaded through PyTorch Geometric and are properly cited.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: All fourteen datasets used in the main experiments — Cora, CiteSeer, PubMed (homophilic); Texas, Wisconsin, Cornell, Actor, Squirrel, Chameleon (small heterophilic, from Pei et al. [2020]); Roman-empire, Amazon-ratings, Minesweeper, Tolokers, Questions (Platonov mid-scale, from Platonov et al. [2023]) — as well as the de-duplicated Squirrel-filtered and Chameleon-filtered benchmarks used in Appendix D, are standard benchmarks publicly available through the PyTorch Geometric library and the official Platonov repository, which are properly cited. The reference implementation of CASTOR together with scripts that reproduce every reported number (the per-(dataset, architecture) training pipeline, the hyperparameter grids of Appendix C.3, the bootstrap-CI reporting protocol of Appendix C.5, and the per-dataset selected configurations) will be made available alongside the camera-ready paper, accompanied by the exact commands and the conda environment specification needed to rerun the experiments end-to-end.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

Justification: Section 5 ("Configuration and setup" and "Training recipe") documents the full experimental protocol used for every reported number, and Appendix C (Datasets and splits, Common backbone and fair-capacity audit, Hyperparameter grids, Training and early stopping, Reporting protocol) gives every parameter required to reproduce it. We fix a common two-layer backbone with fair ~ 64 effective hidden dimension on the Platonov datasets (per-stream width divided by the aggregation count for multi-power, multi-head, or JK-cat architectures, audited in Appendix C.2), train every architecture with AdamW (Appendix C.4; baselines whose reference implementation prescribes a different optimizer keep the original choice), use cross-entropy for multi-class targets and binary cross-entropy on the binary Platonov datasets (Minesweeper, Tolokers, Questions, where ROC-AUC is the evaluation metric), and select the configuration that maximizes the per-dataset validation metric (accuracy on multi-class targets, ROC-AUC on the binary Platonov datasets) via 200-epoch validation-patience early stopping over a 1,000-epoch budget. The data-splitting protocol is the BernNet-style $10\times48/32/20$ splits of Zhu et al. [2020] applied to the nine small datasets of Pei et al. [2020] (Cora, CiteSeer, PubMed, Texas, Wisconsin, Cornell, Actor, Squirrel, Chameleon) and the official $10\times50/25/25$ splits of Platonov et al. [2023] for the five Platonov mid-scale datasets (Roman-empire, Amazon-ratings, Minesweeper, Tolokers, Questions). We sweep learning rate $\in \{10^{-3}, 5{\cdot}10^{-3}, 10^{-2}, 5{\cdot}10^{-2}\}$, weight decay $\in \{0, 5{\cdot}10^{-4}\}$, pre-input dropout $\in \{0.0, 0.3, 0.5, 0.7\}$, and pre-classifier dropout $\in \{0.0, 0.3, 0.5, 0.7, 0.8\}$ for every architecture, with CASTOR-specific sweeps over propagation depth $K \in \{2, 3, 5, 8, 10, 12\}$, hop-bias initialization, momentum parameterization, and momentum initialization (Appendix C.3). Test scores are aggregated as mean $\pm$ 95% non-parametric bootstrap CI over 1,000 resamples across the ten splits (Appendix C.5).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Every classification metric (accuracy on multi-class targets, ROC-AUC on the binary Platonov datasets) reported in Table 1 is presented as the empirical mean of the ten per-split test scores, with the half-width of a non-parametric 95% bootstrap confidence interval over 1,000 bootstrap resamples in subscript — the same protocol used by Platonov et al. [2023] on the Platonov mid-scale benchmarks. The factor of variability captured by the interval is the random/fixed train/validation/test split together with the associated parameter initialisation per run, as described in the "Training recipe" paragraph of Section 5 and detailed in Appendix C.5. The bootstrap interval is non-parametric and therefore makes no Normality assumption on the per-split scores. The same reporting protocol is applied on the de-duplicated benchmarks of Appendix D, where we additionally use the standard $\sqrt{\mathrm{ci}_1^2 + \mathrm{ci}_2^2}$ approximation to assess pairwise gaps (Appendix D.1).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 5 ("Configuration and setup") states that all experiments were conducted on NVIDIA A100 GPUs (80 GB VRAM) [NVIDIA Corporation, 2020] with CUDA 12.8, that all models are implemented in PyTorch Geometric [Fey and Lenssen, 2019], and that profiling for memory and runtime measurements relied on NVIDIA Nsight Compute CLI [NVIDIA Corporation, 2025]. The propagation block's cost is itemized in Appendix B: K sparse matvecs and K Router passes per forward pass, total cost $\mathcal{O}(K(|\mathcal{E}|C + NCH_\rho))$, propagation memory $\mathcal{O}(NC + NH_\rho)$ that is independent of K, and propagation parameter count $\mathcal{O}(CH_\rho + K)$ — which together with the disclosed hardware budget bounds the resources required to reproduce every reported number (Table 2). Each (architecture, dataset, split) configuration is trained for up to 1,000 epochs with 200-epoch validation patience (Appendix C.4). Preliminary experiments and unsuccessful configurations during architecture development used the same hardware and required modestly more total compute than the final reported runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics. The work uses public benchmark datasets with proper attribution, presents transparent and statistically sound evaluations across 10 splits per (dataset, model) pair, and concerns a foundational architectural question about graph neural networks — whether the propagation operator can be made adaptive to node and graph structure — without raising ethical concerns related to privacy, bias, or potential misuse. We have preserved anonymity in the submission and have reviewed the Code of Ethics in preparing this checklist.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper does not include an explicit broader-impacts section. CASTOR is foundational machine learning research focused on the architectural question of how the propagation operator of a graph neural network is parameterized, and the empirical evaluation is conducted on standard public node-classification benchmarks. While not discussed in the paper, the most plausible positive impact is that a single model that adapts to the homophily-heterophily spectrum without per-dataset architectural switching reduces the wall-clock and energy cost of model selection on real graphs (such as fraud, recommendation, or biological-interaction networks), and we are not aware of a direct path from this work to specific negative applications beyond those that already apply to graph neural networks in general.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper releases neither pre-trained language models, image generators, nor scraped datasets. The artifact released is a node-classification GNN architecture trained on standard, already-public graph benchmarks; it poses no particular risk for misuse and therefore does not require model-access controls, usage gating, or content safety filters.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [No]

    Justification: The paper credits the creators of every asset it uses by citing the original publications — for the datasets (Cora, CiteSeer, PubMed, Texas, Wisconsin, Cornell, Actor, Squirrel, Chameleon from Pei et al. [2020]; Roman-empire, Amazon-ratings, Minesweeper, Tolokers, Questions, Squirrel-filtered, and Chameleon-filtered from Platonov et al. [2023]), the baseline architectures (GCN, GAT, SGC, GCN-II, GPRGNN, BernNet, MixHop, H2GCN, GCN-JK, GAT-JK, M2MGNN, CO-GNN, GAMMA, FaGcn), and the implementation tools (PyTorch Geometric, CUDA, NVIDIA Nsight Compute CLI). However, the paper does not explicitly enumerate the specific license (e.g., MIT, Apache-2.0, CC-BY) under which each of these assets is distributed. All of them are standard, freely available academic resources that we use within their stated terms of use, and we will state explicit license information for every reused asset in the released code repository alongside the camera-ready version.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The new asset introduced by this paper is the CASTOR architecture itself, which is documented in the paper as a self-contained set of equations: the routing kernel of Section 4.1 (Eqs. (1)–(5)), the inertial second-order propagation block of Section 4.2 (Eq. (6)), and the full pipeline of Section 4.3 (Eqs. (8)–(10)). Together with the experimental protocol of Section 5, this is sufficient to reproduce the model from scratch. The accompanying reference implementation will be released alongside the camera-ready paper with a README that documents the model, the conda environment, the per-(dataset, model) hyperparameter grid, and the scripts used to produce every reported number.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or any research with human subjects. All experiments are computational, conducted on standard publicly available graph benchmarks (Cora, CiteSeer, PubMed, Texas, Wisconsin, Cornell, Actor, Squirrel, Chameleon, Roman-empire, Amazon-ratings, Minesweeper, Tolokers, Questions, and the de-duplicated Squirrel-filtered / Chameleon-filtered variants); no human participants were recruited, instructed, or compensated as part of this research.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The research does not involve human subjects. The paper presents an algorithmic contribution evaluated exclusively on public node-classification benchmarks, with no human participants, no risks to disclose to subjects, and no associated requirement for institutional review board (or equivalent) approval.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

Justification: Large language models are not a component of the core methodology of this research. CASTOR is a graph neural network whose propagation operator is composed adaptively at every step from three elementary spectral primitives via a small router and a learnable second-order recurrence; no language model is used at training time, at inference time, or in producing any of the reported empirical results.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.