Smoothness Discrepancies in Dynamics Models and How to Avoid Them

Anonymous Authors[1]

Abstract

A defining limitation of modern graph neural networks (GNNs) is the phenomenon of over smoothing, wherein node features of a graph become increasingly close to the node features of their neighbors. While recent architectures have proposed various inductive biases to reduce over smoothing, these biases are often designed independently of tasks where a small amount of smoothing is actually desirable. This paper studies over smoothing in GNNs on tasks where the ground truth smoothing factor is known. In doing so, we elucidate when inductive biases that aim to preserve smoothness are useful and when they are over constraining. In the process, we propose methods to relax smoothness preserving constraints and show their efficacy on solving heat diffusion on graphs and meshes. We also clarify the general smoothing behaviors of different GNN architectures by studying their ability to solve dynamical systems on mesh manifolds. We find that models with inductive biases that approximately preserve smoothness are competitive with models that have explicit geometric priors such as equivariance. We supplement our experiments with an approximation error lower bound that clarifies the generalization limits of smoothness preserving (unitary) functions. Our anonymized artifact is available here.

1. Introduction

Partial Differential Equation (PDE) solving is crucial for several scientific and engineering domains, including turbulent and fluid flow, micro-mechanics, and weather. Neural networks are useful solvers for PDEs, offering fast inference, discretization free solutions (Li et al., 2021), robustness to partial observability (Schlaginhaufen et al., 2021; Huang et al., 2024; Morel et al., 2025), and synergy with existing finite element methods (Gupta & Lermusiaux, 2023). For PDE solving on a given geometry, such as an airplane wing or mechanical part, the PDE solution is often discretized as a signal on a graph or mesh and solved using a graph neural network.

Unfortunately, a defining limitation of Graph Neural Networks (GNNs) is their tendency to exhibit over smoothing (Li et al., 2018), wherein adjacent node features become increasingly similar over successive iterations of message passing. This over smoothing phenomena has been proven to occur in a variety of settings (Cai & Wang, 2020; Bodnar et al., 2022; Keriven, 2022; Rusch et al., 2023; Balla, 2023; Kiani et al., 2024; Arroyo et al., 2025; Su & Wu, 2025; Mishayev et al., 2025) and prevents GNNs from being able to scale with the number of layers in a network.

Kiani et al. (2024) employ the Rayleigh quotient, the average distance between node features connected by an edge, as a measure for smoothness. They then prove that their proposed solution of unitary convolution prevents over smoothing, as it preserves the Rayleigh quotient. However, this poses a new problem: many dynamics problems commonly solved using GNNs require *some* amount of smoothing. For example, heat diffusion on graphs and meshes naturally smooth the input node features (see Tab. 1).

Our work directly addresses this problem, demonstrating that a novel *relaxed* unitary convolution network is best at capturing both the solution and smoothness of the heat equation on graphs. Furthermore, our work studies the problem of smoothness discrepancies in more generality: On a more complete set of dynamical systems on complex mesh datums, we conduct a systematic investigation of the smoothness tendencies of different mesh-GNN architectures and find that inductive biases for retaining smoothness can be even more useful for solving PDEs than geometric biases such as equivariance. We compliment our empirical results with theory that bridges results from over smoothing in GNNs to Mesh-GNNs. In particular, we generalize both the Rayleigh quotient smoothness metric and unitary convolution framework to the mesh setting. Our theoretical contributions also include an approximation error lower bound for smoothness preserving (unitary) functions. Our bound illustrates that unitary functions are particularly over con-

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

strained for dynamical systems where the solution's norm has a significant angular dependence.

In summary, our contributions are the following:

1. Numerical experiments that illustrate when unitary convolution is over constraining and when it is beneficial for learning and smoothness (Sec. 5, Sec. 6).
2. Architectural constraint relaxations for unitary convolution (Sec. 5, Sec. 6).
3. A generalization of unitary convolution and the Rayleigh quotient to mesh datums (Sec. 6).
4. A comparison of the smoothing behavior of different architectures on graphs and meshes (Sec. 6).
5. An approximation error lower bound for smoothness preserving (unitary) functions (Sec. 4).

2. Related Works

Over and under smoothing in GNNs. Our work quantifies the effect of neural networks on the Rayleigh quotient (Chung, 1997) of a graph, an approach similarly employed by Kiani et al. (2024). Moreover, Kiani et al. (2024) prove that unitary functions, and in particular the unitary convolution network, strictly preserve the Rayleigh quotient and therefore the smoothness of input graphs. Our work illustrates both theoretically and empirically how this property can be over constraining in GNNs. Similar approaches have studied the problem of over and under smoothing in PDE solutions using the Matérn kernel (Borovitskiy et al., 2021; Daniels et al., 2025) or fitting decay rate exponents (Kulick et al., 2025), but no previous works have used the Rayleigh quotient as we do for dynamics models.

Our work is perhaps most similar to Keriven (2022), who point out that some over smoothing can be useful for certain classification and regression tasks. Similarly, Li et al. (2018) point out that GCNs (Kipf & Welling, 2017) can be understood as a special case of Laplacian smoothing and is a key reason why GCNs work at all. In fact, Kipf & Welling (2017) compare their architecture to hashing and the Weisfeiler-Leman 1 test (Leman & Weisfeiler, 1968), indicating that even randomly initialized GCNs can be performant due to the way they smooth out information throughout the network. Only Marisca et al. (2025) studied these over smoothing behaviors in the context of spatio-temporal modeling, and does not consider dynamics modeling specifically. Our work builds off of these studies by further clarifying that unitary constraints can be over constraining for the task of learning dynamical systems where the solution's smoothness has a temporal dependence.

Dynamics modeling over graphs and meshes. Our work focuses on dynamics modeling where PDE solutions are discretized as signals on graphs and meshes as a case study for over smoothing. Many physical systems such as fluid flows (Constantin & Foiaş, 1988; Anandkumar et al., 2020), climate (Ghil & Simonnet, 2020), phase fields (Cahn & Hilliard, 1958; Li et al., 2024) heat diffusion (Park et al., 2023), and wave propagation (Park et al., 2023), can be expressed as PDEs. Deep learning based approaches are increasingly used to solve these PDEs on these domains where numerical solving is difficult (Wang et al., 2020; Cranmer et al., 2020; Anandkumar et al., 2020; Li et al., 2021; Mustafa et al., 2021; Cai et al., 2021; Maurizi et al., 2022; Park et al., 2023; Liu et al., 2024; Yu & Wang, 2024; Daniels & Rigollet, 2025). For PDE solving on meshes, these dynamics can be formulated extrinsically by embedding the manifold into euclidean space (Satorras et al., 2021; Pfaff et al., 2021), or intrinsically by defining evolution directly on coordinates of local tangent spaces (Cohen et al., 2019; de Haan et al., 2021; Mitchel et al., 2021; Basu et al., 2022; Park et al., 2023; Suk et al., 2024). While Park et al. (2023) study how these design choices can affect neural network convergence to PDE solutions under MSE loss, our work is distinct in that we directly assess how these design choices affect neural network *smoothing behavior*.

While the physical symmetries of many dynamical systems are well understood (Olver, 1993; Wang et al., 2021; Borovitskiy et al., 2021), there is comparatively less work done to understand the smoothness. The performance of deep dynamics models are typically measured either via quantitative error metrics against the ground truth or their preservation of underlying physical laws, such as spectral energy errors (Wang et al., 2021) or equivariance errors (Wang et al., 2021; 2022a;b). Our work is novel in our application of the Rayleigh quotient in quantifying the smoothing effect of trained GNN deep dynamics models. Furthermore, we are among the first to design architectures with inductive biases that encourage the model to exactly match the Rayleigh quotient of the labeled graphs. Other works have explored using the Rayleigh quotient as an auxiliary loss (Rowan et al., 2025) or positional encodings (Dong et al., 2024), and works such as Kiani et al. (2024) have developed a constrained model that preserves the Rayleigh quotient regardless of the true labels. In contrast, only our work and the work of Shao et al. (2024) tries to match the smoothness of labeled graphs via inductive biases in the network architecture, and only our work assesses how well the true smoothness of dynamical systems are recovered in evaluation.

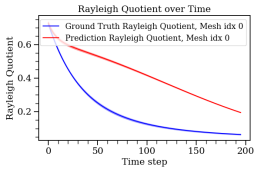
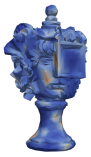
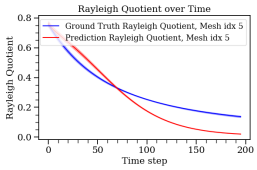
Smoothness Discrepancy	Predicted Heat $T = 190$	True Heat $T = 190$	Rayleigh quotient for Roll outs
Under Smooth			
Over Smooth			

Table 1. Two graph neural networks (Basu et al., 2022; Park et al., 2023) are tasked to auto-regressively predict the heat distribution at time t. **Top:** The Hermes (Park et al., 2023) prediction is rougher than the true solution. Accordingly, the Rayleigh quotient is lower for the ground truth for each step of the roll out. This indicates *under* smoothing. **Bottom:** The EMAN (Basu et al., 2022) predicted heat distribution starts to become too smooth over time. Accordingly, the Rayleigh quotient is lower for the prediction during the latter half of the roll out. This indicates *over* smoothing. A more complete comparison can be found in Sec. C.3, Tab. 4 and Tab. 5.

3. Background

3.1. Rayleigh quotient

To measure smoothness of a graph, we use the Rayleigh quotient as defined in Chung (1997). Consistent with Kiani et al. (2024), we use the following notation:

Definition 1 (Rayleigh quotient, (Chung, 1997), also Definition 5 in Kiani et al. (2024))**.** Given an undirected graph $\mathcal{G} = (V, E)$ on $|V| = n$ nodes with adjacency matrix $\mathbf{A} \in \{0,1\}^{n\times n}$, let $\mathbf{D} \in \mathbb{R}^{n\times n}$ be a diagonal matrix where the i-th entry $\mathbf{D}_{ii} = d_i$ and d_i is the degree of node i. Let $f : V \to \mathbb{C}^d$ be a function from nodes to features. Then the Rayleigh quotient $R_{\mathcal{G}}(\mathbf{X})$ is equal to

$$R_{\mathcal{G}}(\mathbf{X}) = \frac{1}{2}\frac{\sum_{(u,v)\in E} ||\frac{f(u)}{\sqrt{d_u}} - \frac{f(v)}{\sqrt{d_v}}||^2}{\sum_{w\in V} ||f(w)||^2} \tag{1}$$

or equivalently

$$R_{\mathcal{G}}(\mathbf{X}) = \frac{\mathrm{Tr}\left(\mathbf{X}^\dagger(\mathbf{I} - \tilde{\mathbf{A}})\mathbf{X}\right)}{||\mathbf{X}||_F^2} \tag{2}$$

where $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is the normalized adjacency matrix and $\mathbf{X} \in \mathbb{C}^{n\times d}$ is a matrix with the i-th row set to feature vector $f(i)$. We will often abbreviate the normalized graph Laplacian as $\mathbf{L} = (\mathbf{I} - \tilde{\mathbf{A}})$.

Intuitively, the Rayleigh quotient measures the average difference in node features for nodes connected by an edge. A graph with identical node features has a Rayleigh quotient of zero.

3.2. Unitary Convolution

Kiani et al. (2024) contribute two models that preserve the Rayleigh quotient using unitary operations, separable and lie unitary convolution (Eq. (3) and Eq. (4)):

$$f_{\mathrm{Uconv}}(\mathbf{X}; \mathbf{A}) = \exp(\mathbf{iAt})\mathbf{XU}, \qquad \mathbf{U}^\dagger\mathbf{U} = \mathbf{I} \tag{3}$$

$$f_{\mathrm{Uconv}}(\mathbf{X}; \mathbf{A}) = \exp(\mathbf{AXW}), \qquad \mathbf{W} = -\mathbf{W}^\dagger \tag{4}$$

where $\exp(\cdot)$ denotes the matrix exponential. We provide further background material on the matrix exponential and its relationship to unitary matrices in Sec. A.1. The authors show that networks constructed from unitary convolutions are architecturally constrained to preserve the Rayleigh quotient:

Proposition 1 (Invariance of Rayleigh quotient, Proposition 6 in Kiani et al. (2024))**.** *Given an undirected graph $\mathcal{G}$ on n nodes with normalized adjacency matrix $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, the Rayleigh quotient $R_{\mathcal{G}}(\mathbf{X}) = R_{\mathcal{G}}(f_{\mathrm{Uconv}}(\mathbf{X}))$ is invariant under normalized unitary or orthogonal graph convolution (see Eq. (3) and Eq. (4)).*

Crucially, separable unitary convolution is often relaxed by dropping the constraint that $\mathbf{U}^\dagger\mathbf{U} = \mathbf{I}$, meaning that Proposition 1 is not strictly satisfied. Additionally, Taylor series truncation errors in the matrix exponential can cause both separable and Lie unitary convolution to violate Proposition 1. (Kiani et al., 2024) also show that more conventional architectures such as GCNs (Kipf & Welling, 2017) are likely to exhibit over smoothing. We include the proposition in Sec. A.2 for completeness.

3.3. The Mesh Datum

A (triangular) mesh $\mathcal{M}$ consists of a set $(\mathcal{V}, \mathcal{E}, \mathcal{F})$, where $\mathcal{V}$ is a set of vertices, $\mathcal{E} = \{(i, j)\}$ is a set of ordered vertex indices i, j connected by an edge, and $\mathcal{F} = \{(i, j, k)\}$ is the

set of ordered vertex indices i,j,k connected by a triangular face. The mesh datum generalizes graphs by including high order connectivity information via the inclusion of faces. We assume that the mesh is a $2-$dimensional manifold embedded in $\mathbb{R}^3$, i.e. a manifold mesh. In Sec. A.3, we provide further detail on various equivariance constraints commonly employed for tasks on mesh datums.

4. Theory on Overly Constrained Unitary Functions

While unitary functions on graphs can be useful because they preserve the Rayleigh quotient, this section illustrates how unitary functions can be *overly* constrained. In particular, we derive an approximation error lower bound that clarifies the generalization limits of unitary functions. We start by establishing our unitary approximation learning framework.

4.1. Preliminaries

Let $Z = \mathbb{C}^n$ be a domain. Let $p : Z \to \mathbb{R}$ be the density on Z. Let $u : \mathbb{C}^n \to \mathbb{C}^n$ be a unitary function and let $f : \mathbb{C}^n \to \mathbb{C}^n$ be an arbitrary function. Denote the regression error by

$$\text{err}_{\text{reg}}(u) = \int_Z p(z)\|u(z) - f(z)\|_2^2 dz.$$

We assume no covariate shift during testing, i.e., $p(z)$ is always the underlying data distribution. Our result in Theorem 1 will use concepts from the approximation error of group invariant functions h. We review these concepts in detail in Sec. A.4 and provide informal definitions in the paragraph that follows.

A group invariant function h satisfies $h(z) = h(gz)$ for all $g \in G$, $z \in Z$. Let $Gz = \{gz : g \in G\}$ be the orbit of z. A *fundamental domain* F of a group G in Z is a set of orbit representatives. Z can be written as the union of conjugates, $Z = \cup_{g\in G} gF$, where the conjugate is defined $gF = \{gz : z \in F\}$. Denote the integrated density on an orbit by $p(Gz) = \int_{Gz} p(z)dz$. Finally, denote the average and variance of a function f on an orbit Gz by $\mathbb{E}_{Gz}[f]$ and $\mathbb{V}_{Gz}[f]$ respectively. The approximation error lower bound for an invariant function is given by the following proposition.

Proposition 2 (Theorem 4.8 in Wang et al. (2023))**.** *For a $G-$invariant function h, the regression error is bounded below by err $\geq \int_F p(Gz)\mathbb{V}_{Gz}[f]dz$.*

Proposition 2 was initially stated for real valued functions in Wang et al. (2023), but can be applied to complex valued functions without loss of generality. Furthermore, Wang et al. (2023) provide numerical evidence that Proposition 2 is a tight bound. This instills confidence that the bound we will prove in Theorem 1 is also tight, since it is a direct application of the proposition.

4.2. Unitary Approximation Error Lower Bound

This subsection states our main theoretical result, which demonstrates that unitary neural networks are particularly over constrained when the norm of the ground truth function has a high angular dependance. Recall the definition of $SU(n)$, the group of rotations in $\mathbb{C}^n$:

$$SU(n) = \{U \in \mathbb{C}^{n\times n} : \det(U) = 1\}.$$

We now have the necessary background to prove our approximation error lower bound in the case of an incorrect unitary constraint. A proof is in Sec. A.5.

Theorem 1. *Let F be a fundamental domain of $SU(n)$ in Z. In particular, $F = \{te : t \in \mathbb{R}_+\}$ where e is a standard basis vector of $\mathbb{C}^n$. The approximation error lower bound can be expressed as*

$$\int_Z p(z)\|u(z) - f(z)\|_2^2 dz \geq \int_F p(\|te\|)\mathbb{V}_{Gz}[\|f\|]dz.$$

Intuitively, the fundamental domain enumerates all concentric spheres S^{2n-1} embedded in $\mathbb{C}^n$. Unitary functions are complex valued rotations and reflections that preserve the norm of data points that live on each sphere. The error lower bound is given by the variance of the norm of f averaged over each concentric sphere wherein the norm of u is constant. Our result suggests that unitary functions can be particularly over constraining when the norm of f has a high angular dependence.

5. Unitary Convolution Constraint Relaxation

In this section, we prescribe a method for relaxed unitary convolution and show that it significantly outperforms normal graph convolution and marginally outperforms standard Lie Unitary Convolution on a synthetic heat diffusion task on graphs. We attribute the relative success of the method not only to the constraint relaxation but also the tendencies of GCNs at initialization. We provide evidence for this by studying the training curves of an ensemble of runs for each model.

5.1. Relaxed Unitary Convolution

Since Theorem 1 informs us that a unitary convolution network may be over constraining, this section proposes a constraint relaxation to the architecture. We note that Kiani et al. (2024) propose their own constraint relaxation by allowing $\mathbf{U}$ to be unconstrained in Eq. (3), and that our approach instead relaxes Eq. (4). This allows us to isolate the architectural component that alters the Rayleigh quotient. In contrast, the relaxation in Kiani et al. (2024) can be

achieved via two different mechanisms: early Taylor series truncation of the matrix exponential and letting $\mathbf{U}$ remain unconstrained in Eq. (3). The downside of this approach is that the relative contributions of each are difficult to measure and it is therefore harder to tune the extent of the relaxation. Our relaxation is simply Taylor series truncation of Eq. (4). Instead of approximating the the matrix exponential using enough Taylor series terms so that the truncation error is vanishingly small, we truncate at some $T = \mathbf{T}_{\max}$ where $\mathbf{T}_{\max}$ controls the extent of the relaxation. Our relaxed model is then defined

$$f_{\text{Relaxed}}(\mathbf{X}; \mathbf{A}, \mathbf{T}_{\max}) = \sum_{i=0}^{\mathbf{T}_{\max}} \frac{1}{i!} \mathbf{L}^i(\mathbf{X}) \tag{5}$$

where $\mathbf{L}(\mathbf{X}) = \mathbf{AXW}$, $\mathbf{W} = -\mathbf{W}^\dagger$. This approach does not preserve the Rayleigh quotient for small $\mathbf{T}_{\max}$. In the limit as $T \to \infty$ we recover standard Lie Unitary convolution in Eq. (4). Empirically, we find that $T = 10$ is sufficient to preserve the Rayleigh quotient, which is consistent with what is employed in Kiani et al. (2024). Accordingly, we will use $f_{\text{Relaxed}}(\mathbf{X}; \mathbf{A}, 10)$ interchangeably with $f_{\text{LieUniConv}}(\mathbf{X}; \mathbf{A})$.

5.2. Rayleigh Quotient Sensitivity

Motivated by the desire to find an appropriate $\mathbf{T}_{\max}$ that applies only a small perturbation to the Rayleigh quotients of input graphs, we conduct a sensitivity analysis of the Rayleigh quotient to different Taylor series truncations. For completeness, we also compare with standard GCNs and Separable Unitary networks. We study these tendencies at initialization for a heat diffusion dataset that we will use for the experiment in Sec. 5.3 as well. Our analysis echos a theme similar to Gruver et al. (2023) and Gao et al. (2025) that practitioners should be more thorough in evaluating when numerical approximations break strict theoretical guarantees.

Heat Diffusion Data Curation. We use `PyGSP` (Defferrard et al., 2017) to generate $10,000$ two-dimensional grids with 20 randomly placed heat sources and use `pyGSP` to model the true heat diffusion. Our setup mirrors the common task of modeling heat flow on a hot plate. Denote by $f : \mathbb{R}_+ \to \mathbb{R}^n$ a function that maps time t to the heat distribution of the graph. In particular, $t \mapsto e^{-\tau t \mathbf{L}} f(0)$ where τ is a diffusivity constant, $\mathbf{L}$ is the graph Laplacian, and $f(0)$ is the initial heat graph. The exponential term is sometimes referred to as the Green's function of the graph diffusion equation (Borovitskiy et al., 2021). The first three time steps are designated as training and the fourth as validation.

Experimental Setup. We use time step 3 to conduct the sensitivity analysis, and evaluate on the models f_{GCN}, $f_{\text{SepUniConv}}$, and $f_{\text{LieUniConv}}$. For each model f and truncation length $\mathbf{T}_{\max} \in \{1, \ldots, 10\}$, we compute the Rayleigh quotients $R_{\mathcal{G}}(\mathbf{X})$ and $R_{\mathcal{G}}(f(\mathbf{X}))$ for all graph mini batches $\mathbf{X}$. We denote the distribution of Rayleigh quotients before applying the model by $P_{\mathbf{X}}$ and after applying the model by $P_{f(\mathbf{X})}$. To quantify the deviation between these distributions, we compute the KL divergence $D_{\text{KL}}\left(P_{\mathbf{X}} \parallel P_{f(\mathbf{X})}\right)$, which measures the change in the distribution of Rayleigh quotients caused by the model at initialization.

Results. We see in Fig. 1 the effect of Taylor series truncation on the unitarity of the network. In particular, we observe that the KL divergence between the two distributions decreases exponentially with the number of terms. This is to be expected, we know from Taylor's theorem that a truncation at term t gives truncation error $\mathcal{O}\left(\frac{[\|\mathbf{AXW}\|_{\mathbf{O}}]^{t+1}\|\mathbf{X}\|_2}{(t+1)!}\right)$ where $\|\cdot\|_{\mathbf{O}}$ is the operator norm. Furthermore, works such as Ferrandi & Hochstenbach (2024) and Dong et al. (2024) show theoretically that small truncation errors will not compound into large deviations in the Rayleigh quotient. For details on the relevant propositions from Ferrandi & Hochstenbach (2024) and Dong et al. (2024), see Sec. A.6.

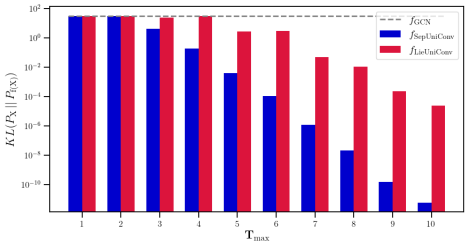


Figure 1. KL divergence between distribution of Rayleigh quotients before and after applying the model. Results are averaged over 10 runs.

In the supplementary material we include a video that shows the evolving Rayleigh quotient distribution as we increase $\mathbf{T}_{\max}$. Fig. 1 is also clickable and links to the same video in our anonymous artifact.

5.3. Heat Diffusion Performance and Smoothness

Having established tendencies of the different architectures at initialization, we now show how this manifests in different observed training dynamics and that the relaxed unitary convolution network in Eq. (5) provides the best performance.

Experimental Setup. We use the same dataset and objective as in Sec. 5.2. The network must predict the heat distribution on the graph at time t given the heat distribution at time $t-1$. We denote the true node feature labels (heat) as $\mathbf{Y}$. We compare the performance in terms of MSE loss and mean Rayleigh quotient for three models: f_{GCN}, f_{Relxaed},

and $f_{\text{LieUniConv}}$. We use $\mathbf{T}_{\max} = 3$ for the relaxed model. In order to understand how tendencies at initialization manifest in different training dynamics, we examine training curves over multiple different runs. Further details are given in Sec. B.

Results. We see in Tab. 2 and Fig. 2 that the relaxed model significantly outperforms the GCN and marginally outperforms the Lie Unitary model. Moreover, the relaxed model is best able to produce graphs whose smoothness matches that of the true labels. We note that while only 5 runs are shown for visual clarity, extensive ablations confirmed the behavior over a much larger set of runs. Our results are grounded theoretically by Proposition 7 in Kiani et al. (2024), which is provided in Sec. A.2 for convenience. The proposition demonstrates that GCNs with weights commonly seen at initialization are likely to exhibit smoothing. The nuance exhibited by our experiment is that this tendency at initialization makes GCNs perform poorly relative to other architectures even in a simulated task where one of the baselines is overly constrained. We also validate empirically that not only does the GCN smooth, but it *over smooths* on a consistent basis. In contrast, the relaxed convolution network often is initialized in a state of being *under smooth* and is able to learn how to compress the node features. We will see in Sec. 6 that these insights will hold for heat flow on more intricate mesh datums.

Model \ Metric	MSE (↓)	$\lvert\overline{R_{\mathcal{G}}(f(\mathbf{X}))} - \overline{R_{\mathcal{G}}(\mathbf{Y})}\rvert$ (↓)
GCN	1.080×10^{-2}	5.99×10^{-2}
UniConv	1.41×10^{-3}	8.86×10^{-2}
Relaxed UniConv **(Ours)**	$\mathbf{1.08 \times 10^{-3}}$	$\mathbf{2.07 \times 10^{-2}}$

Table 2. Best MSE and Rayleigh quotient error over an ensemble of models for a GCN, a Unitary Convolution Network, and our relaxed Unitary Convolution. Best performing model is given bold text.

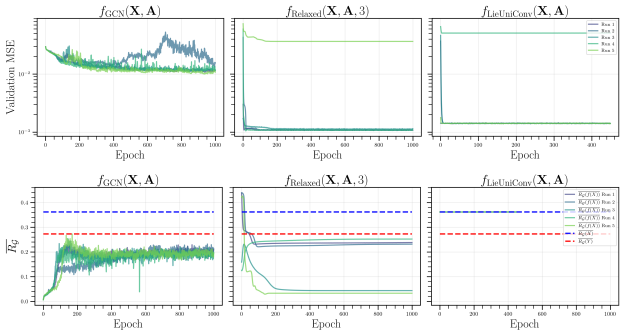


Figure 2. **Top:** Validation MSE for an ensemble of 5 runs for a GCN (left), a Lie Unitary Convolution network with 3 truncation terms (middle), and a Lie Unitary Convolution network with 10 truncation terms (right). The 3 truncation Lie Unitary network significantly outperforms the GCN and marginally outperforms the 10 truncation term Lie Unitary network. **Bottom:** The average Rayleigh quotient over all graphs for an ensemble of 5 runs for the same models. The GCN is under constrained and biased towards over smoothing at initialization. The 3 truncation term Lie Unitary network is able to roughly match the true smoothness of the labeled graphs. The 10 truncation term Lie Unitary network is over constrained and can not model the Rayleigh quotient of the labels because it is forced to preserve the Rayleigh quotient of the input graphs.

6. A Practitioners Guide to Smoothness

Having illustrated when inductive biases that preserve smoothness are helpful for learning on graphs, we now turn our attention towards architectures developed for more intricate dynamical systems and mesh datasets where convolution is often insufficient. Our experiments unearth the following conclusions relevant to practitioners: (**i**) Models with inductive biases that preserve smoothness are just as performant as models with inductive biases that preserve symmetry for dynamics modeling. (**ii**) Approximately unitary networks are especially useful for heat diffusion tasks. (**iii**) More inductive biases are required for more intricate cross mesh generalization.

We support these conclusions with experiments on a variety of real and simulated datasets for dynamics modeling on mesh manifolds, including PDE solving on the `PyVista` (Sullivan & Kaszynski, 2019) meshes from Park et al. (2023) and weather forecasting on the Earth mesh from WeatherBench2 (Rasp et al., 2024).

In order to keep our work appropriately scoped, we focus on which inductive biases are most useful for learning and smoothness in the main text. However, Sec. F contains an additional experiment that illustrates how smoothness discrepancies can compound with the number of layers in a network.

6.1. Models and Data

In this section, we detail the models and datasets used for our analysis, including a novel generalization of unitary convolution to the mesh setting.

Models. Our analysis starts with standard GNN models without any specific inductive biases for working on meshes, including a GCN (Kipf & Welling, 2017) and an MPNN (Gilmer et al., 2017). Additionally, we study symmetry preserving equivariant models, including Gauge and Euclidean equivariance (formally defined in Sec. A.3). Informally, Euclidean equivariant models are invariant to roto-translations of the mesh in Cartesian coordinates and Gauge Equivariant GNNs are invariant to a choice of reference angle for models that work in local coordinates of the mesh-manifold. We benchmark a state of the art Euclidean equivariant model (Satorras et al., 2021) as well as all existing flavors of Gauge

	Convolutional			Attentional		Message Passing		
Metric	GCN	GemCNN	UniMesh **(Ours)**	EMAN	Transformer	MPNN	EGNN	Hermes
Heat ($\alpha = 1$)								
INRMSE (↓)	–	–	$\mathbf{51.9 \pm 3.6}$	73.50 ± 3.8	92.5 ± 5.6	99.45 ± 4.8	344.25 ± 110.5	73.02 ± 4.7
ISMAPE (↓)	–	375.4 ± 0.53	$\mathbf{79.7 \pm 5.6}$	110.9 ± 13.3	213.9 ± 2.7	223.6 ± 1.5	319.33 ± 7.59	107.6 ± 7.4
IRE (↓)	–	52.21 ± 9.4	$\mathbf{9.1 \pm 7.4}$	14.2 ± 1.4	46.0 ± 3.7	76.06 ± 3.6	81.5 ± 8.77	39.76 ± 4.7
Wave ($c = 1$)								
INRMSE (↓)	–	–	$\mathbf{236.5 \pm 6.4}$	281.3 ± 15.5	864.9 ± 184.9	563.6 ± 7.75	2280.1 ± 559.9	458.5 ± 13.0
ISMAPE (↓)	–	318.8 ± 3.9	385.2 ± 1.2	$\mathbf{301.0 \pm 4.2}$	327.0 ± 4.4	318.0 ± 2.8	354.3 ± 11.0	316.4 ± 4.5
IRE (↓)	–	107.9 ± 3.158	93.5 ± 25.4	73.57 ± 6.5	$\mathbf{48.0 \pm 7.9}$	139.3 ± 10.1	157.2 ± 14.8	70.03 ± 6.1
Cahn-Hilliard								
INRMSE (↓)	–	$\mathbf{121.2 \pm 1.8}$	123.9 ± 2.6	137.5 ± 0.69	144.4 ± 0.8	147.4 ± 11.36	1001.04 ± 5.73	122.0 ± 7.8
ISMAPE (↓)	–	204.3 ± 2.4	167.3 ± 10.6	$\mathbf{143.7 \pm 2.5}$	191.7 ± 2.0	201.22 ± 32.79	336.5 ± 2.777	173.3 ± 4.3
IRE (↓)	–	$\mathbf{10.68 \pm 3.3}$	18.9 ± 10.4	48.57 ± 3.49	27.42 ± 2.87	23.98 ± 6.51	41.8 ± 1.997	14.38 ± 11.5

Table 3. INRMSE, ISMAPE, and IRE averaged over all roll outs on all test meshes for the heat, wave, and Cahn-Hilliard. equations. Errors and standard deviations are reported over all test meshes and all initializations. Cells with a dash (–) correspond to models which do not converge for a given metric. Models are grouped together by flavor: convolutional, attention, or message passing (Bronstein et al., 2021). UniMesh is competitive across all task and excels at solving the heat equation on unseen meshes.

Equivariant GNNs, including convolutional with GemCNN (de Haan et al., 2021), attentional with EMAN (Basu et al., 2022), and message passing with Hermes (Park et al., 2023). We note that EMAN is the only model that is both Gauge and Euclidean equivariant. We also consider and a state of the art mesh transformer (Janny et al., 2023). Finally, we adapt unitary convolution to the mesh setting. In Sec. A.7, we define the Rayleigh quotient for a mesh and prove that combining unitary convolution layers with a particular edge weighting scheme preserves this Rayleigh quotient. Accordingly, we construct a model by stacking unitary layers together with either a MLP or GCN readout layer to achieve approximate smoothness preserving behavior. We note that in order to increase the number of parameters in our model, our relaxation strategy differs from Sec. 5. We coin our model UniMesh . Further details of the mesh Rayleigh quotient are provided in the next section (Sec. 6.2) and in Sec. A.7, and further details of the architecture are provided in Sec. E.

Datasets. Our first task is to auto-regressively predict the solution to the heat, wave, and Cahn-Hilliard equations on test meshes given an initial condition (Eq. (7)-Eq. (8)). We use the same `PyVista` meshes generated in Park et al. (2023) as data. These meshes are highly intricate and stress test the models ability to handle nonlinear dynamics on complicated geometries. Before defining the PDEs to be solved, let us establish notation. Let α be the thermal diffusivity, c a constant, and $\tilde{\mathbf{L}}$ the symmetric cotangent Laplacian (Reuter et al., 2009). In particular, for a scalar function u, the symmetric cotangent Laplacian is defined

$$(\tilde{\mathbf{L}}(u))_i = \frac{1}{2A_i} \sum_{j \in \mathcal{N}(i)} (\cot \alpha_{ij} + \cot \beta_{ij}) (u_j - u_i) \quad (6)$$

where $\mathcal{N}(i)$ denotes the adjacent vertices of i, α_{ij} and β_{ij} are the angles opposite edge (i, j), and A_i is the vertex area of i, where we use the barycentric cell area. The heat and wave equations on the mesh are then given by

$$\frac{\partial u}{\partial t} = \alpha \tilde{\mathbf{L}} u, \text{(Heat)} \qquad \frac{\partial^2 u}{\partial t^2} = c^2 \tilde{\mathbf{L}} u, \text{(Wave)}. \quad (7)$$

We now define the Cahn-Hilliard equation (Cahn & Hilliard, 1958). Let c be the fluid concentration, M the diffusion coefficient, μ the chemical potential, L the Laplacian, f the double-free energy function, and λ a positive constant. The Cahn-Hilliard equation is often represented by the following two coupled second order equations:

$$\frac{\partial c}{\partial t} - ML(\mu) = 0, \quad \mu - \frac{\partial f}{\partial c} + \lambda L(c) = 0. \quad (8)$$

Our task is to approximate these PDEs by auto-regressively predicting the spatial distribution of the solution u or c at each time step t. We use auto-regressive roll outs on unseen meshes as validation over five different initializations. We note that the test meshes from Park et al. (2023) are much more complex for the heat and wave datasets than for Cahn-Hilliard. Training details are given in Sec. C.1.

Our second task consists of weather forecasting using WeatherBench 2 (WB2) (Rasp et al., 2024), a widely used benchmark for data-driven global weather forecasting based on historic data. Specifically, given a dataset $\mathcal{D} = \{X_i\}_{i=1}^N$ of historical weather data, the task of weather forecasting is to predict future weather conditions $X_T \in \mathbb{R}^{V \times H \times W}$ given initial conditions $\{X_i\}_{i=1}^K, X_i \in \mathbb{R}^{V \times H \times W}$, where T is the target lead time, K is the number of input time steps to the model, V is the number of atmospheric variables, and $H \times W$ is the spatial resolution of the data, which depends on how densely we grid the globe.

We train and evaluate our models on the ERA5 dataset from WB2, which is the curated version of the ERA5 reanalysis data provided by the European Centre for Medium-Range

Weather Forecasts (ECMWF) (Hersbach et al., 2020). We use the 1.5 (240×120) degree spatial resolution data with a 6 hours temporal resolution, consistent with the evaluation performed in WB2. Further details on mesh construction can be found in Sec. D.3. We evaluate on two variables, temperature at pressure level 850 (T850) and geopotential at pressure level 500 (Z500). Note that unlike many large scale weather forecasting models that predict multiple variables and levels simultaneously, we train one model for each variable and level pair. We take data from 2013-01-01 to 2019-12-31 UTC as training data. We use a smaller subset of the ERA5 data that is commonly used for other large scale data-based weather models due to compute constraints, but remain consistent to WB2 in evaluating on data from 2020-01-01 to 2020-12-31.

6.2. Evaluation

In this section, we establish our metrics and evaluation protocol: this includes a generalization of the Rayleigh quotient to mesh datums and a review for domain-specific weather forecasting metrics. First, we note that the Rayleigh quotient in Eq. (2) uses the Laplacian $\mathbf{L} = \mathbf{I} - \tilde{\mathbf{A}}$. While it is tempting to simply replace $\mathbf{L}$ with the symmetric cotangent Laplacian $\tilde{\mathbf{L}}$ in Eq. (6), this does not work. The cotangent weights in Eq. (6) may be negative, which in turn means that the Rayleigh quotient is no longer a valid measure of smoothness (Definition 1, Rusch et al., 2023). Instead, we use the *Robust Laplacian* (Sharp & Crane, 2020), which performs a minimal edge rewiring of the mesh so that the cotangent weights obey the *Delaunay criterion*. Concretely, this means that our Laplacian weights are both symmetric and the off-diagonals are nonnegative. The Laplacian remains symmetric with nonnegative off-diagonals even if the manifold assumption on our mesh does not hold (cf. Figure 6, Sharp & Crane, 2020). For the PDE task, our metrics include NRMSE, SMAPE, and Rayleigh quotient errors aggregated over all time-steps. Further details are provided in Sec. C.2. See also Definition 7, Sec. A.7 for a formal definition of the Rayleigh quotient for meshes. For WB2, we additionally report the root mean squared error (RMSE) and the anomaly correlation coefficient (ACC), both latitude weighted as recommended by the benchmark authors. RMSE measures forecast accuracy, while ACC is the Pearson correlation coefficient between forecast anomalies and ground-truth anomalies relative to a climatological baseline. The climatology is computed as a multi-year average over corresponding times of the year (approximately a 30-year mean). Precise definitions of the latitude weights, RMSE, ACC, and the climatology computation are provided in Sec. D. While our study is concerned with $1-$hop smoothness tendencies in accordance with the Rayleigh quotient formalism, in Sec. C.4 we conduct an additional experiment that compares with a more global metric for smoothness. We also validate our metrics with qualitative diagnostics in Sec. C.3.

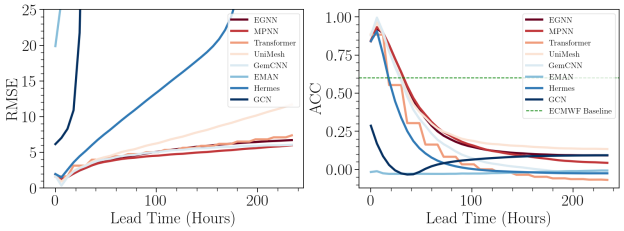


Figure 3. RMSE and ACC as a function of lead time for all models. UNIMESH has a competitive RMSE, especially at early lead time. UNIMESH also maintains viability for lead times of roughly 2 days according to the ECMWF baseline.

6.3. Results

Our main result is that our UNIMESH model is able to perform just as well or better than more expressive attention and message passing based models on most tasks without any equivariance constraints. For the PDEs task, this is indicated by the INRMSE, ISMAPE, and IRE for each of the models, which is given in Tab. 3. UNIMESH is especially performant on heat modeling, where it achieves the lowest error on all three metrics. We remind the reader that we validate these metrics with qualitative diagnostics in Sec. C.3. Another important conclusion is that nearly all models are able to perform comparably well on the Cahn-Hilliard dataset where the test mesh (toroid) is simple. The only models that perform poorly on this task are the GCN and EGNN models, which also struggle across all other tasks. This suggests that more inductive biases are necessary for more intricate cross mesh generalization such as for the heat and wave datasets. This is further supported by our results on WB2. As seen in Fig. 3, the equivariant and unitary models show no significant advantages in this setting, where there is no cross mesh generalization. We also note that, despite restricting our training set size due to compute limitations, our best performing models are reasonably close to the state of the art (Figure 1, Rasp et al., 2024) according to domain specific metrics RMSE and ACC.

7. Conclusion

Our work marks a step forward towards understanding smoothness discrepancies in GNNs for dynamics modeling on graphs and meshes. In particular, we elucidate the approximation limits of unitary functions and unitary convolution networks, and show how constraint relaxations can aid performance on various dynamics modeling tasks on graphs and meshes without the need for any explicit geometric biases. Future work may explore using approximately unitary networks for solving PDEs under partial observability by using them as backbones for generative models.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Anandkumar, A., Azizzadenesheli, K., Bhattacharya, K., Kovachki, N., Li, Z., Liu, B., and Stuart, A. Neural operator: Graph kernel network for partial differential equations. In *ICLR 2020 workshop on integration of deep neural models and differential equations*, 2020.

Anil, C., Lucas, J., and Grosse, R. Sorting out lipschitz function approximation. In *International conference on machine learning*, pp. 291–301. PMLR, 2019.

Arroyo, A., Gravina, A., Gutteridge, B., Barbero, F., Gallicchio, C., Dong, X., Bronstein, M. M., and Vandergheynst, P. On vanishing gradients, over-smoothing, and over-squashing in GNNs: Bridging recurrent and graph learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=N4cyRMuLyl.

Artin, M. *Algebra*. Birkhäuser, 1998.

Balla, J. Over-squashing in riemannian graph neural networks. In *The Second Learning on Graphs Conference*, 2023. URL https://openreview.net/forum?id=UUnYi0yLcM.

Basu, S., Gallego-Posada, J., Viganò, F., Rowbottom, J., and Cohen, T. Equivariant mesh attention networks. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=3IqqJh2Ycy. Expert Certification.

Bodnar, C., Di Giovanni, F., Chamberlain, B., Lio, P., and Bronstein, M. Neural sheaf diffusion: A topological perspective on heterophily and oversmoothing in gnns. *Advances in Neural Information Processing Systems*, 35: 18527–18541, 2022.

Borovitskiy, V., Azangulov, I., Terenin, A., Mostowsky, P., Deisenroth, M., and Durrande, N. Matérn gaussian processes on graphs. In *International Conference on Artificial Intelligence and Statistics*, pp. 2593–2601. PMLR, 2021.

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

Cahn, J. W. and Hilliard, J. E. Free energy of a nonuniform system. i. interfacial free energy. *The Journal of chemical physics*, 28(2):258–267, 1958.

Cai, C. and Wang, Y. A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*, 2020.

Cai, S., Wang, Z., Wang, S., Perdikaris, P., and Karniadakis, G. E. Physics-informed neural networks for heat transfer problems. *Journal of Heat Transfer*, 143(6):060801, 2021.

Chung, F. R. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

Cohen, T., Weiler, M., Kicanaoglu, B., and Welling, M. Gauge equivariant convolutional networks and the icosahedral cnn. In *International conference on Machine learning*, pp. 1321–1330. PMLR, 2019.

Constantin, P. and Foiaş, C. *Navier-stokes equations*. University of Chicago press, 1988.

Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., and Ho, S. Lagrangian neural networks. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.

Daniels, M. and Rigollet, P. Splat regression models. *arXiv preprint arXiv:2511.14042*, 2025.

Daniels, M., Hodgkinson, L., and Mahoney, M. Uncertainty-aware diagnostics for physics-informed machine learning. *arXiv preprint arXiv:2510.26121*, 2025.

de Haan, P., Weiler, M., Cohen, T., and Welling, M. Gauge equivariant mesh cnns: Anisotropic convolutions on geometric graphs. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Jnspzp-oIZE.

Defferrard, M., Martin, L., Pena, R., and Perraudin, N. Pygsp: Graph signal processing in python, 2017. URL https://github.com/epfl-lts2/pygsp/.

Dong, X., Zhang, X., and Wang, S. Rayleigh quotient graph neural networks for graph-level anomaly detection. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=4UIBysXjVq.

Esteves, C. Theoretical aspects of group equivariant neural networks. *arXiv preprint arXiv:2004.05154*, 2020.

Ferrandi, G. and Hochstenbach, M. E. A homogeneous rayleigh quotient with applications in gradient methods. *Journal of Computational and Applied Mathematics*, 437: 115440, 2024.

Gao, W., Xu, R., Deng, Y., and Liu, Y. Discretization-invariance? on the discretization mismatch errors in neural operators. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ghil, M. and Simonnet, E. Geophysical fluid dynamics, nonautonomous dynamical systems, and the climate sciences. In *Mathematical Approach to Climate Change and its Impacts: MAC2I*, pp. 3–81. Springer, 2020.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. Pmlr, 2017.

Gruver, N., Finzi, M. A., Goldblum, M., and Wilson, A. G. The lie derivative for measuring learned equivariance. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=JL7Va5Vy15J.

Gupta, A. and Lermusiaux, P. F. Generalized neural closure models with interpretability. *Scientific Reports*, 13(1): 10634, 2023.

Hall, B. C. Lie groups, lie algebras, and representations. In *Quantum Theory for Mathematicians*, pp. 333–366. Springer, 2013.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R. J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., and Thépaut, J.-N. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730): 1999–2049, 2020. doi: https://doi.org/10.1002/qj.3803. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3803.

Huang, J., Su, H., and Guibas, L. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018.

Huang, J., Yang, G., Wang, Z., and Park, J. J. Diffusionpde: Generative pde-solving under partial observation. *Advances in Neural Information Processing Systems*, 37: 130291–130323, 2024.

Janny, S., Bénéteau, A., Nadri, M., Digne, J., Thome, N., and Wolf, C. EAGLE: Large-scale learning of turbulent fluid dynamics with mesh transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=mfIX4QpsARJ.

Jarvis, M., Bernstein, G., and Jain, B. The skewness of the aperture mass statistic. *Monthly Notices of the Royal Astronomical Society*, 352(1):338–352, 2004.

Keriven, N. Not too little, not too much: a theoretical analysis of graph (over) smoothing. *Advances in Neural Information Processing Systems*, 35:2268–2281, 2022.

Kiani, B., Fesser, L., and Weber, M. Unitary convolutions for learning on graphs and groups. *Advances in Neural Information Processing Systems*, 37:136922–136961, 2024.

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

Kulick, C., Birnir, B., and Tang, S. Investigating zero-shot size transfer of graph neural differential equations for learning graph diffusion dynamics. In *Topology, Algebra, and Geometry in Data Science*, 2025. URL https://openreview.net/forum?id=qgbyLknKXy.

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., Merose, A., Hoyer, S., Holland, G., Vinyals, O., Stott, J., Pritzel, A., Mohamed, S., and Battaglia, P. Graphcast: Learning skillful medium-range global weather forecasting, 2023. URL https://arxiv.org/abs/2212.12794.

Leman, A. and Weisfeiler, B. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya*, 2(9):12–16, 1968.

Li, Q., Han, Z., and Wu, X.-M. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Li, W., Fang, R., Jiao, J., Vassilakis, G. N., and Zhu, J. Tutorials: Physics-informed machine learning methods of computing 1d phase-field models. *APL Machine Learning*, 2(3), 2024.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

Li, Z., Huang, D. Z., Liu, B., and Anandkumar, A. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023.

Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljacic, M., Hou, T. Y., and Tegmark, M. Kan: Kolmogorov–arnold networks. In *The Thirteenth International Conference on Learning Representations*, 2024.

Marisca, I., Bamberger, J., Alippi, C., and Bronstein, M. M. Over-squashing in spatiotemporal graph neural networks. *arXiv preprint arXiv:2506.15507*, 2025.

Maurizi, M., Gao, C., and Berto, F. Predicting stress, strain and deformation fields in materials and structures with graph neural networks. *Scientific reports*, 12(1):21834, 2022.

Mishayev, Y., Sverdlov, Y., Amir, T., and Dym, N. Short-range oversquashing. In *The Fourth Learning on Graphs Conference*, 2025.

Mitchel, T. W., Kim, V. G., and Kazhdan, M. Field convolutions for surface cnns. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10001–10011, 2021.

Morel, R., Ramunno, F. P., Shen, J., Bietti, A., Cho, K., Cranmer, M., Golkar, S., GUGNIN, O., Krawezik, G., Marwah, T., et al. Predicting partially observable dynamical systems via diffusion models with a multiscale inference scheme. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Mustafa, M., Wu, J., Jiang, C., Wang, R., et al. Physics-informed machine learning: case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A*, 379(2194):20200093, 2021.

NVIDIA. Nvidia h200 tensor core gpu datasheet, 2025. URL https://resources.nvidia.com/en-us-data-center-overview-mc/en-us-data-center-overview/hpc-datasheet-sc23-h200. Retrieved from NVIDIA website.

Olver, P. J. *Applications of Lie groups to differential equations*, volume 107. Springer Science & Business Media, 1993.

Pandya, S., Yang, Y., Van Alfen, N., Blazek, J., and Walters, R. Iaemu: Learning Galaxy Intrinsic Alignment Correlations. *The Open Journal of Astrophysics*, 8, dec 2 2025. doi: 10.33232/001c.151749.

Park, J. Y., Wong, L., and Walters, R. Modeling dynamics over meshes with gauge equivariant nonlinear message passing. *Advances in Neural Information Processing Systems*, 36:15277–15302, 2023.

Pfaff, T., Fortunato, M., Sanchez-Gonzalez, A., and Battaglia, P. Learning mesh-based simulation with graph

networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=roNqYL0_XP.

Rasp, S., Hoyer, S., Merose, A., Langmore, I., Battaglia, P., Russell, T., Sanchez-Gonzalez, A., Yang, V., Carver, R., Agrawal, S., et al. Weatherbench 2: A benchmark for the next generation of data-driven global weather models. *Journal of Advances in Modeling Earth Systems*, 16(6): e2023MS004019, 2024.

Reuter, M., Biasotti, S., Giorgi, D., Patanè, G., and Spagnuolo, M. Discrete laplace–beltrami operators for shape analysis and segmentation. *Computers & Graphics*, 33 (3):381–390, 2009.

Rowan, C., Doostan, A., Maute, K., and Evans, J. Solving engineering eigenvalue problems with neural networks using the rayleigh quotient. *International Journal for Numerical Methods in Engineering*, 126(24):e70209, 2025.

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

Satorras, V. G., Hoogeboom, E., and Welling, M. E (n) equivariant graph neural networks. In *International conference on machine learning*, pp. 9323–9332. PMLR, 2021.

Schlaginhaufen, A., Wenk, P., Krause, A., and Dörfler, F. Learning stable deep dynamics models for partially observed or delayed dynamical systems. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=u8HmtBBSVJS.

Schneider, P. Weak gravitational lensing. In *Gravitational lensing: strong, weak and micro*, pp. 269–451. Springer, 2006.

Shao, Z., Shi, D., Han, A., Guo, Y., Zhao, Q., and Gao, J. Unifying over-smoothing and over-squashing in graph neural networks: A physics informed approach and beyond, 2024. URL https://openreview.net/forum?id=swPf2hwKl8.

Sharp, N. and Crane, K. A laplacian for nonmanifold triangle meshes. In *Computer Graphics Forum*, volume 39, pp. 69–80. Wiley Online Library, 2020.

Su, J. and Wu, C. On the interplay between graph structure and learning algorithms in graph neural networks. In *Forty-second International Conference on Machine Learning*, 2025.

Suk, J., de Haan, P., Lippe, P., Brune, C., and Wolterink, J. M. Mesh neural networks for se (3)-equivariant hemodynamics estimation on the artery wall. *Computers in biology and medicine*, 173:108328, 2024.

Sullivan, C. and Kaszynski, A. Pyvista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (vtk). *Journal of Open Source Software*, 4(37):1450, 2019.

Tönshoff, J., Ritzert, M., Rosenbluth, E., and Grohe, M. Where did the gap go? reassessing the long-range graph benchmark. In *The Second Learning on Graphs Conference*, 2023.

Tönshoff, J., Ritzert, M., Rosenbluth, E., and Grohe, M. Where did the gap go? reassessing the long-range graph benchmark. *Transactions on Machine Learning Research*, 2024.

Tran, A., Mathews, A., Xie, L., and Ong, C. S. Factorized fourier neural operators. In *The Eleventh International Conference on Learning Representations*, 2023.

Trockman, A. and Kolter, J. Z. Orthogonalizing convolutional layers with the cayley transform. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Pbj8H_jEHYv.

Wang, D., Zhu, X., Park, J. Y., Jia, M., Su, G., Platt, R., and Walters, R. A general theory of correct, incorrect, and extrinsic equivariance. *Advances in Neural Information Processing Systems*, 36:40006–40029, 2023.

Wang, R., Kashinath, K., Mustafa, M., Albert, A., and Yu, R. Towards physics-informed deep learning for turbulent flow prediction. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pp. 1457–1466, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403198. URL https://doi.org/10.1145/3394486.3403198.

Wang, R., Walters, R., and Yu, R. Incorporating symmetry into deep dynamics models for improved generalization. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=wta_8Hx2KD.

Wang, R., Walters, R., and Yu, R. Approximately equivariant networks for imperfectly symmetric dynamics. In *International Conference on Machine Learning*, pp. 23078–23091. PMLR, 2022a.

Wang, R., Walters, R., and Yu, R. Data augmentation vs. equivariant networks: A theory of generalization on dynamics forecasting. *International Conference on Machine Learning (ICML) Principles of Distribution Shift Workshop*, 2022b.

Yu, R. and Wang, R. Learning dynamical systems from data: An introduction to physics-guided deep learning. *Proceedings of the National Academy of Sciences*, 121(27):e2311808121, 2024. doi: 10.1073/pnas.2311808121. URL https://www.pnas.org/doi/abs/10.1073/pnas.2311808121.

A. Deferred Theory

This section provides both theoretical background and deferred proofs from the main text.

A.1. Lie Algebras and the Exponential Map

In this section we review the formalism behind Lie algebras and the exponential map. A group is a mathematical structure that formalizes what it means for something to be *symmetric*. We say that a group is a matrix *Lie group*, if it is a differentiable manifold and a subgroup of the set of invertible $n \times n$ matrices. Lie groups are equipped with a *lie algebra*, which is the tangent space at the identity element. Our work encounters the orthogonal and unitary lie groups

$$O(n) = \{O \in \mathbb{R}^{n\times n} : OO^T = I\}, \qquad U(n) = \{U \in \mathbb{C}^{n\times n} : UU^\dagger = I\}$$

as well as the special unitary group

$$SU(n) = \{U \in \mathbb{C}^{n\times n} : \det(U) = 1\}.$$

The associated lie algebras for $O(n)$ and $U(n)$ are given by

$$\mathfrak{o}(n) = \{M \in \mathbb{R}^{n\times n} : M + M^T = 0\}, \qquad \mathfrak{u}(n) = \{M \in \mathbb{C}^{n\times n} : M + M^\dagger = 0\}.$$

The exponential map provides a mechanism of parameterizing lie groups with elements in the lie algebra. For matrix lie groups, the exponential map is simply the matrix exponential:

$$\exp(\mathbf{X}) = \sum_i^\infty \frac{1}{i!}\mathbf{X}^i.$$

Applying the exponential map to a linear operator is given by

$$\exp(\mathbf{L})(\mathbf{X}) = \sum_i^\infty \frac{1}{i!}\mathbf{L}^i(\mathbf{X}) = \mathbf{X} + \mathbf{L}(\mathbf{X}) + \frac{1}{2}\mathbf{L}\circ\mathbf{L}(\mathbf{X}) + \frac{1}{6}\mathbf{L}\circ\mathbf{L}\circ\mathbf{L}(\mathbf{X}) + \dots$$

In the case of Eq. (4), $\mathbf{L}$ is graph convolution, $\mathbf{L}(\mathbf{X}) = \mathbf{AXW}$. Further background on group theory and abstract algebra can be found in Artin (1998); Hall (2013); Esteves (2020).

A.2. Convolutional Over smoothing

This section provides a result from Kiani et al. (2024) which establishes that Graph Convolution Networks (Kipf & Welling, 2017) have a high probability to exhibit smoothing.

Proposition 3 (Proposition 7 in Kiani et al. (2024))**.** *Given a simple undirected graph $\mathcal{G}$ on n nodes with normalized adjacency matrix $\widetilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ and node degree bounded by D, let $\mathbf{X} \in \mathbb{R}^{n\times d}$ have rows drawn i.i.d. from the uniform distribution on the hypersphere in dimension d. Let $f_{conv}(\mathbf{X}) = \widetilde{\mathbf{A}}\mathbf{X}\mathbf{W}$ denote convolution with orthogonal feature transformation matrix $\mathbf{W} \in O(d)$. Then, the event below holds with probability $1 - \exp(-\Omega(\sqrt{n}))$:*

$$R_{\mathcal{G}}(\mathbf{X}) \geq 1 - O\left(\frac{1}{n^{1/4}}\right) \quad \textit{and} \quad R_{\mathcal{G}}(f_{conv}(\mathbf{X})) \leq 1 - \frac{\mathrm{Tr}(\widetilde{\mathbf{A}}^3)}{\mathrm{Tr}(\widetilde{\mathbf{A}}^2)} + O\left(\frac{1}{n^{1/4}}\right).$$

A.3. Gauge and Euclidean Equivariance

In this section, we introduce the necessary background and formal definitions for the equivariance constraints commonly applied to the mesh datum. While working with arbitrary meshes, many commonly used network architectures compute distances between node positions. One has the option of computing these distances in either global Cartesian coordinates or in local tangent spaces of the mesh. In both cases, we may exploit the symmetry of these coordinate systems by enforcing equivariance with respect to transformations from a certain symmetry group into the network architecture, which allows the network to automatically generalize across orbits.

We now give precise definitions of equivariance and invariance.

Definition 2. Let $f : \mathcal{X} \to \mathcal{Y}$ be a map between input and output vector spaces $\mathcal{X}$ and $\mathcal{Y}$. Let G be a group with representations $\rho^{\mathcal{X}}$ and $\rho^{\mathcal{Y}}$ which transform vectors in $\mathcal{X}$ and $\mathcal{Y}$ respectively. Representations map group elements to invertible linear transformations. The map $f : \mathcal{X} \to \mathcal{Y}$ is *equivariant* if

$$\rho^{\mathcal{Y}}(g)[f(x)] = f(\rho^{\mathcal{X}}(g)[x]) \text{ , for all } g \in G, x \in \mathcal{X} .$$

Invariance is a special case of equivariance in which $\rho^{\mathcal{Y}} = \mathrm{Id}^{\mathcal{Y}}$ for all $g \in G$. With an invariant operator, the output of f is unaffected by the transformations applied to the input.

Definition 3. A map $f : \mathcal{X} \to \mathcal{Y}$ is *invariant* if:

$$f(x) = f(\rho^{\mathcal{X}}(g)[x]) \text{ , for all } g \in G, x \in \mathcal{X}.$$

A.3.1. Euclidean Equivariance

For a mesh defined over a global coordinate system, a common choice of symmetry constraint is equivariance to the Euclidean group in n dimensions, $E(n)$. In this setting, the mesh is treated as a graph with positional encodings, and the equivariance constraint ensures generalization to different roto-translations of the mesh.

Definition 4. Let $t \in \mathbb{R}^n$ be a translation vector and $Q \in \mathbb{R}^{n \times n}$ an orthogonal matrix representing a rotation or reflection. A function f is equivariant to the Euclidean group $E(n)$ if for any $t \in \mathbb{R}^n$ and $Q \in \mathbb{R}^{n \times n}$ we have:

$$f(Qx + t) = Qf(x) + t.$$

A.3.2. Gauge Equivariance

We may also choose to embed coordinates locally, using coordinates that are intrinsic to the 2D mesh rather than the extrinsic 3D coordinates of the embedding space. This approach arises from the desire for a general convolution-like operator over arbitrary manifolds discretized as a mesh. To encode data over a mesh it is still necessary to make a choice of local coordinate frame at each vertex. In order to guarantee the equivalence of the features resulting from different choices of reference frames, the model should be invariant to change of coordinates frame at each vertex, i.e. gauge equivariant.

We specifically adapt the strategy described in de Haan et al. (2021) and define the local coordinate frame at each vertex in terms of a reference neighboring vertex. Denote v_a as the reference neighbor for gauge A, in which the neighbors have angles θ_A, and denote v_b as the reference neighbor for gauge B with angles θ_B. Comparing the two gauges, we see that they are related by a rotation of angle ϕ, so that $\theta_B = \theta_A - \phi$. This change of gauge is called a gauge transformation of angle $g := \phi$.

Definition 5 (Equations 3 and 4 in de Haan et al. (2021))**.** Let ρ_{in} and ρ_{out} be input and output types with dimensions C_{in} and C_{out}. Let $K_{\text{self}} \in \mathbb{R}^{\mathbb{C}_{\text{out}} \times \mathbb{C}_{\text{in}}}$ and $K_{\text{neigh}} : [0, 2\pi) \to \mathbb{R}^{C_{\text{out}} \times C_{in}}$ be two kernels. We say the kernels are *gauge equivariant* if they satisfy for any gauge transformation $g \in [0, 2\pi)$ and angle $\theta \in [0, 2\pi)$:

$$K_{\text{neigh}}(\theta - g) = \rho_{\text{out}}(-g) K_{\text{neigh}}(\theta) \rho_{\text{in}}(g), \qquad K_{\text{self}} = \rho_{\text{out}}(-g) K_{\text{self}}\, \rho_{\text{in}}(g).$$

Finally, as features at different nodes live in different tangent spaces and thus have different gauges, it is invalid to sum them directly. Let f_p and f_q be node features of a pair of neighboring nodes p and q. Before performing gauge equivariant convolution, we must parallel transport each f_q to T_pM along the mesh edge that connects the two vertices for them to be in the same gauge. For more details, we refer the reader to de Haan et al. (2021).

A.4. Unitary Learning Framework

This section provides rigorous definitions for the mathematical tools used in the main text and additionally clarifies necessary hypotheses.

We start with the fundamental domain. Assume X has dimension n. Let d be the dimension of a generic orbit of G in X. Let ν be the $(n - d)$ dimensional Hausdorff measure in X.

Definition 6 (Fundamental Domain, Definition 4.1 in Wang et al. (2023))**.** A closed subset F of X is called a fundamental domain of G in X if X is the union of conjugates of F, i.e., $X = \cup_{g \in G} gF$, and the intersection of any two conjugates has measure 0 under ν.

Next, we note that our proof of Theorem 1 satisfies the integrability assumption on the fundamental domain F and orbits Gz established in Wang et al. (2023):

Assumption 1 (Integrability Hypothesis, Sec. A in Wang et al. (2023))**.** The fundamental domain F and orbit Gx are differentiable manifolds and the union of all pairwise intersections $\cap_{g_1 \neq g)2}(g_1 F \cap g_2 F)$ has measure zero.

We now provide more formal definitions for $\mathbb{E}_{Gx}[f]$ and $\mathbb{V}_{Gx}[f]$ used in Proposition 2. Denote by $q(z) = \frac{p(z)}{p(Gx)}$ the density of the orbit Gx so that $\int_{Gx} q(z)dz = 1$. The mean and variance of a function f on Gx are given by

$$\mathbb{E}_{Gx}[f] = \int_{Gx} q(z)f(z)dz, \qquad \mathbb{V}_{Gx}[f] = \int_{Gx} q(z)\|\mathbb{E}_{Gx}[f] - f(z)\|_2^2 dz.$$

A.5. Proof of Main Theorem

We now provide proof of our main theoretical result in the main text. We repeat the theorem here for convenience.

Theorem 1. Let F be a fundamental domain of $SU(n)$ in Z. In particular, $F = \{te : t \in \mathbb{R}_+\}$ where e is a standard basis vector of $\mathbb{C}^n$. The approximation error lower bound can be expressed as

$$\int_Z p(z)\|u(z) - f(z)\|_2^2 dz \geq \int_F p(\|te\|)\mathbb{V}_{Gz}[\|f\|]dz.$$

Proof of Theorem 1. By the reverse triangle inequality,

$$\int_Z p(z)\|u(z) - f(z)\|_2^2 dz \geq \int_Z p(z)\left(\|u(z)\| - \|f(z)\|\right)^2 dz.$$

Notice that $\|u(z)\|$ is invariant under the action of $SU(n)$ on the sphere S^{2n-1} with radius $\|te\|$ and recall that $SU(n)$ acts transitively on the sphere. Thus, $F = \{te : t \in \mathbb{R}_+\}$ is a valid fundamental domain that indexes each orbit Gz, the spheres with radii $\|te\|$. Our theorem then follows from Proposition 2. □

A.6. Rayleigh Quotient Sensitivity

We include results from Ferrandi & Hochstenbach (2024) and Dong et al. (2024) that illustrate the sensitivity of the Rayleigh quotient to small perturbations of the input, such as Taylor series truncation errors. While the hypotheses are stronger than what we may actually see in practice, the following proposition provides an intuition for the Rayleigh quotient sensitivity.

Proposition 4 (Proposition 4 in Ferrandi & Hochstenbach (2024))**.** *Suppose* $\mathbf{u} = \mathbf{x} + \mathbf{e}$ *is an approximate eigenvector corresponding to a simple eigenvalue* $\lambda \neq 0$ *of a symmetric* A*, with* $\|\mathbf{x}\| = 1$*,* $\mathbf{e} \perp \mathbf{x}$*, and* $\varepsilon = \|\mathbf{e}\|$*. Then, up to* $\mathcal{O}(\varepsilon^4)$*-terms, for the sensitivity of the Rayleigh quotient (as a function of* $\mathbf{u}$*) it holds that*

$$\min_{\lambda_i \neq \lambda} \frac{|\lambda_i - \lambda|}{|\lambda_i|}\varepsilon^2 \lesssim \frac{|R_{\mathcal{G}}(\mathbf{u}) - \lambda|}{|\lambda|} \lesssim \max_{\lambda_i \neq \lambda} \frac{|\lambda_i - \lambda|}{|\lambda_i|}\varepsilon^2.$$

This indicates that the Rayleigh quotient sensitivity is quadratic in perturbations ε. For $\varepsilon < 1$, this means that the sensitivity of the Rayleigh quotient is even less than the truncation error. We also have the following results from Dong et al. (2024):

Proposition 5 (Theorem 1 in Dong et al. (2024))**.** *For any given graph* G*, if there exists a perturbation* Δ *on* $\mathbf{L}$*, the change of Rayleigh quotient can be bounded by* $\|\Delta\|_2$*.*

Proposition 6 (Theorem 2 in Dong et al. (2024))**.** *For any given graph* G*, if there exists a perturbation* δ *on* $\mathbf{x}$*, the change of Rayleigh quotient can be bounded by* $2\mathbf{x}^T\mathbf{L}\delta + o(\delta)$*. If* δ *is small enough, in which case* $o(\delta)$ *can be ignored, the change can be further bounded by* $2\mathbf{x}^T\mathbf{L}\delta$*.*

The results from Dong et al. (2024) state fewer hypotheses than Ferrandi & Hochstenbach (2024). Proposition 5 outlines a bound similar to Proposition 4 in that they are both related to the norm of the perturbing vector, and Proposition 6 states an alternative bound related to the graph's Laplacian.

A.7. Unitary Convolution on Meshes

In this section we generalize unitary convolution from graphs to meshes. In particular, we show that under modest assumptions on the mesh triangulation, unitary convolution with a weighted adjacency matrix still preserves the Rayleigh quotient for meshes (Definition 7).

Let $\odot$ represent the Hadamard product which performs element-wise matrix multiplication, i.e., for $C = A \odot B$ we have $C_{ij} = A_{ij}B_{ij}$. Recall that $\mathcal{N}(i)$ denotes the adjacent vertices of i, α_{ij} and β_{ij} are the angles opposite edge (i, j), and A_i is the vertex area of i, where we use the barycentric cell area. Let $\mathbb{W}$ be the cotangent weights given by

$$\mathbb{W}_{ij} = \begin{cases} \frac{1}{2}\left(\cot \alpha_{ij} + \cot \beta_{ij}\right), & j \in \mathcal{N}(i) \\ -\sum_{k \in \mathcal{N}(i)} \mathbb{W}_{ik}, & i = j \\ 0, & \text{Otherwise.} \end{cases} \tag{9}$$

As noted in Sec. 6, the Rayleigh quotient for meshes is well defined for the *Robust Laplacian* from Sharp & Crane (2020).

Definition 7 (Mesh Rayleigh Quotient)**.** Let $\mathcal{M} = (V, E, F)$ be a mesh on $|V| = n$ nodes. Denote by $\mathbf{A}, \mathbb{W}, \mathbf{L} \in \{0, 1\}^{n \times n}$ the adjacency matrix, cotangent weights, and Robust Laplacian for an edge rewired mesh $\mathcal{M}'$. Let $f : V \to \mathbb{C}^d$ be a function from nodes to features. Then the Rayleigh quotient $R_{\mathcal{M}}(\mathbf{X})$ is equal to

$$R_{\mathcal{M}}(\mathbf{X}) = \frac{1}{2} \frac{\sum_{(u,v)\in E} \mathbb{W}_{uv} || \frac{f(u)}{\sqrt{d_u}} - \frac{f(v)}{\sqrt{d_v}} ||^2}{\sum_{w \in V} ||f(w)||^2} = \frac{\mathrm{Tr}\left(\mathbf{X}^\dagger \mathbf{L} \mathbf{X}\right)}{||\mathbf{X}||_F^2}.$$

In this setting, the edges of a mesh $\mathcal{M}$ are rewired so that the cotangent weights are symmetric and the off-diagonals are nonnegative. Formally, the weights must obey the Delaunay criterion given by Definition 8:

Definition 8 (Delaunay Criterion)**.** For all faces connected by an edge (i, j) with opposite angles α_{ij} and β_{ij}, $\alpha_{ij} + \beta_{ij} \leq \pi$. This is true if and only if $\cot \alpha_{ij} + \cot \beta_{ij} \geq 0$.

To generalize unitary convolution from graphs to meshes, we modify the functions in Eq. (3) and Eq. (4) by incorporating the cotangent weights into the normalized adjacency matrix $\tilde{\mathbf{A}}$. We assume that these weights obey the Delaunay Criterion given by Definition 8, which is stated again in Assumption 2 for completeness. We note that in practice there are existing triangulation strategies that a practitioner can use to ensure that mesh edges satisfy these criterion (Sharp & Crane, 2020). We also assume the mesh is manifold, which similarly can be achieved by various triangulation strategies (Huang et al., 2018; Sharp & Crane, 2020). Incorporating the cotangent weights into the function allows us to ensure that we preserve the Rayleigh quotient for meshes in Definition 7.

Assumption 2 (Mesh Weights Obey the Delaunay Criterion)**.** For a mesh $\mathcal{M}$, the mesh is manifold and all angles obey the Delaunay Criterion given by Definition 8.

With this assumption, we now defined our mesh analogue for unitary convolution and prove that it preserves the mesh Rayleigh quotient given by Definition 7. We use the cotangent weights in Eq. (9) to define separable and lie mesh-unitary convolution. Let $\mathbf{D}$ be the degree matrix defined by $\mathbf{D}_{ii} = \sum_{i \neq j} \mathbb{W}_{ij}$. Our mesh-unitary convolution operator is defined

$$\text{(Separable)} \quad f_{\text{MeshUniConv}}(\mathbf{X}; \mathbf{A}, \mathbb{W}) = \exp(i\mathbf{D}^{-1/2}\left(\mathbb{W} \odot \mathbf{A}\right)\mathbf{D}^{-1/2})\mathbf{X}\mathbf{U}, \quad \mathbf{U}\mathbf{U}^\dagger = \mathbf{I} \tag{10}$$

$$\text{(Lie)} \quad f_{\text{MeshUniConv}}(\mathbf{X}; \mathbf{A}, \mathbb{W}) = \exp(\mathbf{D}^{-1/2}\left(\mathbb{W} \odot \mathbf{A}\right)\mathbf{D}^{-1/2}\mathbf{X}\mathbf{W}), \quad \mathbf{W} + \mathbf{W}^\dagger = \mathbf{0}. \tag{11}$$

We now show that Eq. (10) and Eq. (11) preserve the Rayleigh quotient on meshes.

Corollary 1 (Corollary to Proposition 1)**.** *Given a manifold mesh $\mathcal{M}$ on n nodes with normalized adjacency matrix $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}(\mathbb{W} \odot \mathbf{A})\mathbf{D}^{-1/2}$. We assume the weights $\mathbb{W}$ obey Assumption 2 (i.e., the mesh is already Delaunay without rewiring). The mesh Rayleigh quotient (Definition 7) is invariant under normalized unitary or orthogonal graph convolution (see Eq. (10) and Eq. (11)). Equivalently, $R_{\mathcal{M}}(\mathbf{X}) = R_{\mathcal{M}}(f_{\text{MeshUniConv}}(\mathbf{X}))$.*

Our proof follows the same structure as the proof of Proposition 1 in Kiani et al. (2024) with modifications to account for the weighted adjacency matrix. Namely, we invoke Assumption 2 which ensures that $f_{\text{MeshUniConv}}$ is norm preserving and therefore the the strategy in Kiani et al. (2024) still holds.

Proof. We first prove invariance for Eq. (10). By the circulant property of the trace,

$$\mathrm{Tr}\left(\left(\exp(i\tilde{\mathbf{A}})\mathbf{X}\mathbf{U}\right)^{\dagger}\left(\mathbf{I}-\tilde{\mathbf{A}}\right)\left(\exp(i\tilde{\mathbf{A}})\mathbf{X}\mathbf{U}\right)\right)=\mathrm{Tr}\left(\mathbf{X}^{\dagger}\exp(-i\tilde{\mathbf{A}})\left(\mathbf{I}-\tilde{\mathbf{A}}\right)\exp(i\tilde{\mathbf{A}})\mathbf{X}\right).$$

Because $\exp(-i\tilde{\mathbf{A}})$, $\exp(i\tilde{\mathbf{A}})$, and $(\mathbf{I}-\tilde{\mathbf{A}})$ share an eigenbasis, they commute, so

$$\mathrm{Tr}\left(\left(\exp(i\tilde{\mathbf{A}})\mathbf{X}\mathbf{U}\right)^{\dagger}\left(\mathbf{I}-\tilde{\mathbf{A}}\right)\left(\exp(i\tilde{\mathbf{A}})\mathbf{X}\mathbf{U}\right)\right)=\mathrm{Tr}\left(\mathbf{X}^{\dagger}\left(\mathbf{I}-\tilde{\mathbf{A}}\right)\mathbf{X}\right).$$

For the denominator, we need to show that $\|\exp(i\tilde{\mathbf{A}})\mathbf{X}\mathbf{U}\|_F^2 = \|\mathbf{X}\|_F^2$. By Assumption 2 we have that $\mathbb{W}$ is symmetric. Because $\mathbf{A}$ is also symmetric, we have that $i\tilde{\mathbf{A}}$ is skew-symmetric and therefore $\exp(i\tilde{\mathbf{A}}) \in SU(n)$. Thus, $\|\exp(i\tilde{\mathbf{A}})\mathbf{X}\mathbf{U}\|_F^2 = \|\mathbf{X}\|_F^2$ and finally $R_{\mathcal{M}}(\mathbf{X}) = R_{\mathcal{M}}(f_{\mathrm{MeshUniConv}}(\mathbf{X}))$.

We now show that Eq. (11) also preserves the Rayleigh quotient. First, we need to show that $\|\exp(\mathbf{A}\mathbf{X}\mathbf{W})\|_F^2 = \|\mathbf{X}\|_F^2$. To do this, we note that Eq. (11) can equivalently be viewed us a function that acts on a vector in $\mathbb{C}^{nd}$. By properties of the Kroneckor tensor product,

$$f_{\mathrm{MeshUniConv}}(\mathbf{X};\mathbf{A})=\exp(\mathbf{A}\mathbf{X}\mathbf{W}) \iff \mathrm{vec}\left(f_{\mathrm{MeshUniConv}}(\mathbf{X};\mathbf{A})\right)=\exp\left(\mathbf{A}\otimes\mathbf{W}^{\mathbf{T}}\right)\mathrm{vec}(\mathbf{X}).$$

Since

$$\left(\mathbf{A}\otimes\mathbf{W}^{\mathbf{T}}\right)+\left(\mathbf{A}\otimes\mathbf{W}^{\mathbf{T}}\right)^{\dagger}=\mathbf{A}\otimes\left(\mathbf{W}+\mathbf{W}^{\dagger}\right)^{T}=0,$$

we have that $\left(\mathbf{A}\otimes\mathbf{W}^{\mathbf{T}}\right)$ is in the lie algebra of the unitary group and therefore preserves the norm of $\mathrm{vec}(\mathbf{X})$. This holds for any symmetric edge weighting $\tilde{\mathbf{A}} = \mathbb{W}\odot\mathbf{A}$, which is guaranteed by Assumption 2. Thus, $\|\exp(\mathbf{A}\mathbf{X}\mathbf{W})\|_F^2 = \|\mathbf{X}\|_F^2$. Next, note that $\exp\left(\tilde{\mathbf{A}}\otimes\mathbf{W}^{\mathbf{T}}\right)$ commutes with $(\tilde{\mathbf{A}}\otimes\mathbf{I})$. Thus,

$$\begin{aligned}
&\mathrm{Tr}\left(f_{\mathrm{MeshUniConv}}(\mathbf{X};\tilde{\mathbf{A}})^{\dagger}(\mathbf{I}-\tilde{\mathbf{A}})f_{\mathrm{MeshUniConv}}(\mathbf{X};\tilde{\mathbf{A}})\right)\\
&\quad=\mathrm{vec}(\mathbf{X})^{\dagger}\exp(\tilde{\mathbf{A}}\otimes\mathbf{W}^{\mathbf{T}})^{\dagger}\left[(\mathbf{I}-\tilde{\mathbf{A}})\otimes\mathbf{I}\right]\exp(\tilde{\mathbf{A}}\otimes\mathbf{W}^{\mathbf{T}})\mathrm{vec}(\mathbf{X})\\
&\quad=\mathrm{vec}(\mathbf{X})^{\dagger}\left[(\mathbf{I}-\tilde{\mathbf{A}})\otimes\mathbf{I}\right]\mathrm{vec}(\mathbf{X}).
\end{aligned}$$

Multipliying the above by $\|\mathbf{X}\|_F^{-2}$ recovers $R_{\mathcal{M}}(\mathbf{X})$. We conclude that $R_{\mathcal{M}}(\mathbf{X}) = R_{\mathcal{M}}(f(\mathbf{X}))$. □

Remark 2. Corollary 1 was applied to convolution with the symmetric cotangent weights in Eq. (9), but the proof extends without loss of generality to any set of symmetric weights.

B. Simulated Heat Diffusion Dataset

This section details dataset generation specifications for our experiment in Sec. 5. We generate grid-graphs with an average of 10 nodes and a standard deviation of 2 nodes. On the grid we randomly set 20 nodes to be heat sources. They are given a heat value of 1 and all other nodes start at 0. Using `PyGSP`, We simulate heat flow on $10,000$ graphs for training, and the task is to predict the next time step given the previous one. The simulation proceeds until time $T = 10$ in increments of $\Delta T = 0.5$ time steps. A sample graph data point is given in Fig. 4.

C. `PyVista` Mesh Training Details, Evaluation Details, and Further Experiments

This section provides extra experimental details and results for our dynamical systems modeling on `PyVista` meshes.

C.1. `PyVista` Mesh Training Details

We extend the publicly available code base from Park et al. (2023) to train our baselines for the `PyVista` and WeatherBench2 datasets: https://github.com/jypark0/hermes/. For GemCNN, EMAN, and Hermes, we use the already available pretrained model checkpoints. For GCN, UniMesh , Mesh Transformer, MPNN, and EGNN we train our own models. We performed ablations over learning rate and latent space sizes. Following Park et al. (2023) we keep models

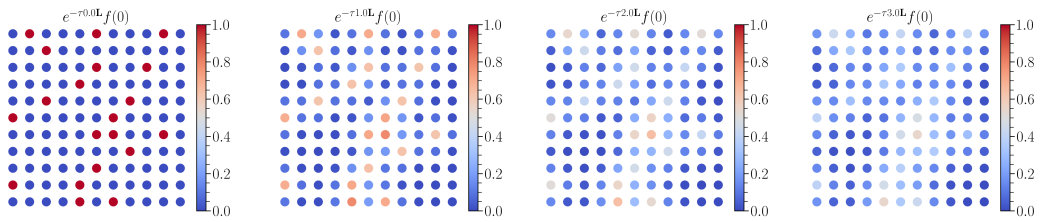


Figure 4. Sample heat diffusion process on a grid discretized as a graph. Node neighbors are the nodes that sit adjacent in the grid.

within a $\sim 40,000 - 50,000$ parameter budget. We note that this budget is relatively small, and that models that diverge in our experiments could potentially perform better under a more forgiving budget. All runs were performed on a single H200 GPU (NVIDIA, 2025). We use the previous 5 time steps as input node feature vectors and backpropagate through 3 steps of auto-regressive inference.

Hyper parameters are given in our artifact. Defaults are taken from Park et al. (2023) if provided and otherwise optimized via grid search. Considered hyper parameters include learning rate, optimizer, training epochs, latent size, and skip connections. We also consider z-scoring of normed edge lengths for EGNN, different decoder heads for UNIMESH , and the number of clusters for the mesh transformer.

C.2. `PyVista` Evaluation Details

In order to aggregate discrepancies between the smoothness of the labeled graph with the one produced by the model over all time steps, we introduce a new metric. Define the Integrated Rayleigh Error (IRE) by $\int_0^\infty |R_{\mathcal{M}}(\mathbf{Y_t}) - R_{\mathcal{M}}(f(\mathbf{X_t}))|dt$. In practice we approximate this by summing over the time steps where we are able to perform inference:

$$\text{IRE}(f) = \sum_t |R_{\mathcal{M}}(\mathbf{Y_t}) - R_{\mathcal{M}}(f(\mathbf{X_t}))|dt.$$

Following Janny et al. (2023) and Pandya et al. (2025), we also consider the scale invariant metrics NRMSE and SMAPE integrated over the entire roll out:

$$\begin{aligned}
\text{INRMSE}(f) &= \sum_t \sqrt{\frac{\frac{1}{n}\sum_{i=1}^n \left(f(\mathbf{X_t})_i - (\mathbf{Y_t})_i\right)^2}{\frac{1}{n}\sum_{i=1}^n (\mathbf{Y_t})_i^2}} \\
\text{ISMAPE}(f) &= \sum_t \frac{1}{n}\sum_{i=1}^n \frac{2|(\mathbf{Y_t})_i - f(\mathbf{X_t})_i|}{|(\mathbf{Y_t})_i| + |f(\mathbf{X_t})_i| + \varepsilon}
\end{aligned}$$

$\varepsilon = 10^{-8}$ is a stability constant. SMAPE is generally more robust than NMRSE in that it is less sensitive to outliers, but it is also more sensitive to small values. The scale invariant property of these metrics is crucial especially for heat diffusivity because solutions tend to decrease proportionally to e^{-t}. Thus, we need to consider deviations across several orders of magnitude in order to see how accurately we are modeling the decay.

C.3. `PyVista` Mesh Qualitative Diagnostics

In this section, we validate the superior performance of UNIMESH on solving the heat equation with qualitative diagnostics. In Tab. 4 and Tab. 5 we show that UNIMESH is the best at capturing the true smoothness of an unseen mesh during each step of the rollout.

C.4. Beyond $1-$hop Smoothness

Since the Rayleigh quotient is a $1-$hop metric, this section performs additional comparisons with a more global smoothness metric and finds that our $1-$hop smoothness tendencies also hold more generally for the gauge equivariant models we study. In particular, we define smoothness according to the $2-$point correlation function. Let $\delta : \mathbb{R}^3 \to \mathbb{R}$ be a function that maps a point $\mathbf{x}$ on a mesh to the scalar solution $u(\mathbf{x})$ (or approximation thereof) to the PDE at that point. The smoothness is then

Time	Truth	UNIMESH **(Ours)**	EMAN	Hermes
10				
50				
100				
150				
190				

Table 4. Qualitative comparison of model performance for the heat equation on the armadillo mesh. Our UNIMESH model remains faithful to the ground truth during each step of the rollout, whereas the EMAN model over smooths and the Hermes model under smooths.

defined by the $2-$point correlation function ξ given in Eq. (12):

$$\xi(r;\delta) = \mathbb{E}\left[\delta(\mathbf{x})\delta(\mathbf{x}+r)\right]. \tag{12}$$

Intuitively, if node features are similar at a distance of r apart, the correlation will be high. This allows us to study smoothness beyond $1-$hop neighbors by considering larger r. Fig. 5 shows an example correlation function for Hermes at a given time step. We note that this characterization of smoothness is common in the weak gravitational lensing literature for point-cloud datasets (Schneider, 2006) and are easily computed with the `TreeCorr` library (Jarvis et al., 2004).

Let δ_{ij} be the scalar field for the ground truth on a mesh $\mathcal{M}_i$ at time step j and $\widehat{\delta_{ij}}$ be the approximation thereof. We define our smoothness error by

$$\text{err}_{\text{smooth}}(\widehat{\delta_{ij}}) = \frac{1}{r_{\text{bins}}}\frac{1}{\mathbf{T}_{\max}}\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{\mathbf{T}_{\max}}\sum_{k=1}^{r_{\text{bins}}}|\xi(r_k;\delta_{ij}) - \xi(r_k;\widehat{\delta_{ij}})|.$$

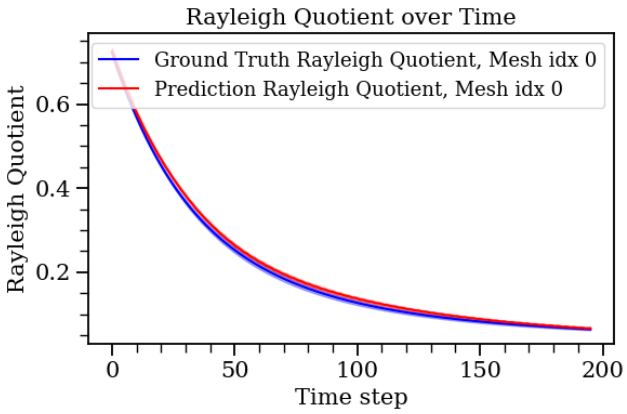
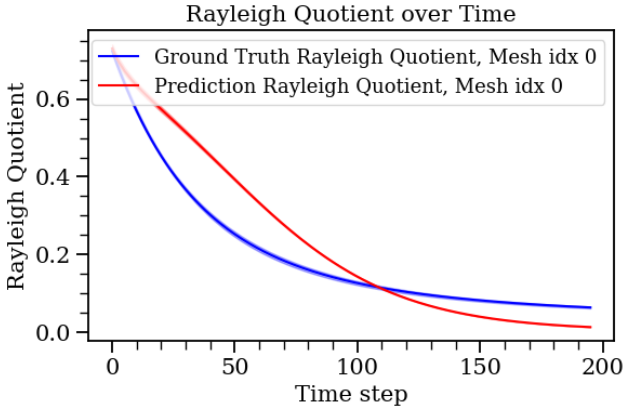
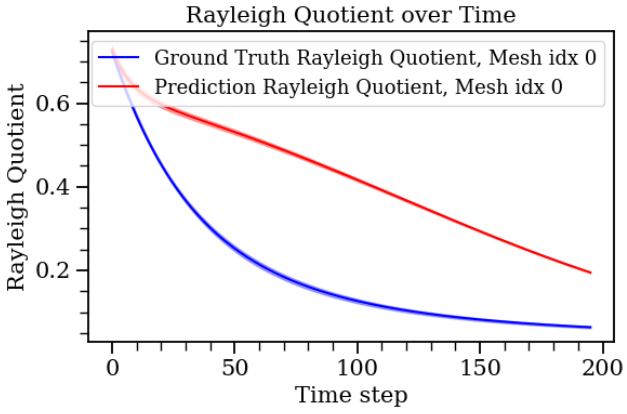
Model	Rayleigh quotient Rollout
UniMesh **(Ours)**	
EMAN	
Hermes	

Table 5. The Rayleigh quotient for each timestep on an unseen mesh for Hermes, EMAN, and UniMesh models. The UniMesh is the best at capturing the ground truth smoothness.

We note that the correlation function is related to the Fourier space power spectrum $P(k)$ by

$$\xi(r) = \frac{1}{2\pi^2} \int k^2 P(k) \frac{\sin(kr)}{kr} dk. \tag{13}$$

Thus, Eq. (13) informs us that our metric for smoothness as a function of r is related to traditional energy spectrum errors (e.g., Wang et al., 2021). We leave a more systematic comparison between the measures as an opportunity for future work.

As seen in Tab. 6, the more expressive attention and message passing based models are much better at capturing the underlying smoothness. The CNN model diverges for the heat and wave datasets, but performs reasonably well on Cahn-Hilliard. This is mirrored by our results in Tab. 3 in the main text for the Gauge Equivariant models.

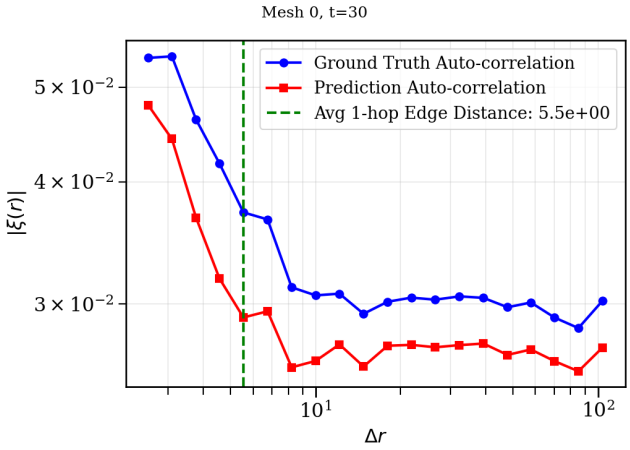


Figure 5. Smoothness for a Hermes model as measured by the $2-$point correlation function. The plot indicates under smoothing in each radial bin.

Heat ($\alpha = 1$)	
Model	$\text{err}_{\text{smooth}}$ ($\downarrow$)
GemCNN (de Haan et al., 2021)	–
EMAN (Basu et al., 2022)	$\mathbf{4.04 \times 10^{-3}}$
Hermes (Park et al., 2023)	9.71×10^{-3}
Wave ($c = 1$)	
Model	$\text{err}_{\text{smooth}}$ ($\downarrow$)
GemCNN (de Haan et al., 2021)	–
EMAN (Basu et al., 2022)	$\mathbf{1.78 \times 10^{-3}}$
Hermes (Park et al., 2023)	1.38×10^{-2}
Cahn–Hilliard	
Model	$\text{err}_{\text{smooth}}$ ($\downarrow$)
GemCNN (de Haan et al., 2021)	1.89×10^{-1}
EMAN (Basu et al., 2022)	4.59×10^{-1}
Hermes (Park et al., 2023)	$\mathbf{9.61 \times 10^{-3}}$

Table 6. $\text{err}_{\text{smooth}}$ for Gauge Equivariant models on the `PyVista` Mesh datasets. Dashes (–) indicate non-convergence. Best performing model is indicated with **bold text**.

D. WeatherBench2 Further Details

We lay out the relevant training, evaluation, and dataset details for WB2.

D.1. Training Details

We follow the same training and hyper parameter optimization strategy as in Sec. C.1. The only difference is that we use the 3 previous time steps as input instead of 5.

D.2. Evaluation Details

Here we give precise definitions of the evaluation and metrics omitted in the main text. We begin by establishing some notation common to the subsections, and consistent with the notation used in (Rasp et al., 2024).

Let f denote the forecast, o the ground-truth observation, and c the climatology. Let $t \in \{1, \dots, T\}$ denote the verification time, $l \in \{1, \dots, L\}$ the lead time, $i \in \{1, \dots, I\}$ the latitude index, and $j \in \{1, \dots, J\}$ the longitude index. Forecasts are indexed as $f_{t,l,i,j}$, while observations and climatology are indexed by absolute time as $o_{t,i,j}$ and $c_{t,i,j}$.

D.2.1. Latitude weighting

In an equiangular latitude-longitude grid, grid cells at the poles have a much smaller area compared to grid cells at the equator. Weighting all cells equally in the computation of RMSE and ACC would result in an inordinate bias towards the polar regions. As a result both metrics are latitude-weighted with weights computed as follows:

$$w(i) = \frac{\sin\theta_i^u - \sin\theta_i^l}{\frac{1}{I}\sum_i^I (\sin\theta_t^u - \sin\theta_i^l)},$$

where θ_i^u and θ_i^l indicate upper and lower latitude bounds, respectively.

D.2.2. Climatology

The climatology c is a function of the day of year and time of day, it is computed by taking the mean of ERA5 data from 1990 to 2019 (inclusive) for each grid point. A sliding window of 61 days is used around each day of year and time of day combination with weights linearly decaying to zero from the center. For notational consistency, we also define the lead-time–indexed climatology $c_{t,l,i,j} := c_{t+l,i,j}$, corresponding to the climatology at the forecast valid time.

D.2.3. Root mean squared error (RMSE)

Following the WB2 convention, our work measures error in terms of RMSE. For each variable and level pair, the RMSE at lead time l is defined as:

$$RMSE_l = \sqrt{\frac{1}{TIJ}\sum_t^T\sum_i^I\sum_j^J w(i)(f_{t,l,i,j} - o_{t,i,j})^2}.$$

This choice is important for temperature forecasting, as we are invariant to choice of unit (e.g., temperature in terms of Kelvin and Celsius will have the same RMSE). Moreover, the change in scale over time is less dramatic as it was for the `PyVista` meshes, where we considered NRMSE.

D.2.4. Anomaly correlation coefficient (ACC)

The ACC is computed as the Pearson correlation coefficient of the anomalies with respect to the climatology c. Denote the differences between forecast and climatology and between observation and climatology by

$$f'_{t,l,i,j} = f_{t,l,i,j} - c_{t,l,i,j}; \quad o'_{t,i,j} = o_{t,i,j} - c_{t,i,j}.$$

The ACC at lead time l is then defined as

$$ACC_l = \frac{1}{T}\sum_t^T \frac{\sum_i^I\sum_j^J w(i) f'_{t,l,i,j}\, o'_{t,i,j}}{\sqrt{\sum_i^I\sum_j^J w(i) {f'_{t,l,i,j}}^2 \sum_i^I\sum_j^J w(i) {o'_{t,i,j}}^2}}.$$

ACC ranges from 1, indicating perfect correlation, to -1, indicating perfect anti-correlation. The ECMWF states that when the ACC value falls below 0.6, it is considered that the positioning of synoptic scale features ceases to have value for forecasting purposes.

D.3. Earth Mesh Discretization

We construct a spherical mesh of the Earth by directly projecting the latitude–longitude grid points onto the unit sphere, and define mesh connectivity according to the original grid neighborhood structure. In order to obtain triangular faces, we

further subdivide each cell into two triangles. The resulting mesh has 29040 nodes, 57600 faces, and 86640 edges. We note that this mesh construction is simpler than those used in other graph-based models, such as GraphCast (Lam et al., 2023), which employs a subdivided icosahedron as the underlying mesh. However, our approach has the advantage that it operates directly on the native latitude–longitude grid and therefore does not require interpolation or regridding of the ERA5 data. Although more elaborate mesh constructions is likely to improve performance in real weather forecasting applications, our focus is on methodological experimentation rather than optimized weather prediction, and we therefore leave mesh optimization as an oppurtunity for future work.

E. UniMesh Details

This section contains architectural details for UniMesh . Our model uses the mesh unitary convolution layers (Eq. (11)) outlined in Sec. A.7 as an encoder. We note that these layers are constrained to preserve the dimension of input node features. Therefore, in order to increase the number of parameters in our model, we perform zero padding on the input node features. Zero padding preserves the Rayleigh quotient as the normed difference between node feature remains unchanged. We also used orthogonal weight matrices instead of unitary ones for a more direct comparison of the number of learnable parameters. We note that the theory in Sec. A.7 still applies, and that Kiani et al. (2024) find no significant differences empirically between unitary and orthogonal networks. The norm preserving GroupSort nonlinearity from Kiani et al. (2024) is used between unitary layers (see also Anil et al. (2019) and Trockman & Kolter (2021)). To get a final scalar valued prediction, we used either a MLP or GCN decoder. This decoder serves to break unitary. Through ablation, we found that using a MLP readout with sinusoidal activation functions was a key ingredient for strong performance with UniMesh on the PDE datasets. This supports previous work on how to train GNNs for long range tasks (Tönshoff et al., 2023; 2024). However, the GCN decoder exhibited the best performance on WeatherBench2.

F. Smoothness Discrepancies Compound With Layers

In this section, we show that we can explain a model's inability to scale with the number of layers by analyzing its smoothness. We illustrate the success of the Factorized Fourier Neural Operator by its ability to perform (Fourier space) edge rewiring as a means of combating over smoothing.

F.1. Models and Data

We compare Fourier neural operators developed for graphs and meshes. Fourier neural operators are a class of neural network used for PDE solving where the learnable parameters live in Fourier space in order to avoid solutions that are dependent on specific discretizations of the mesh. We compare Factorized Fourier Neural Operators (FFNOs) (Tran et al., 2023) with Geometric Fourier Neural Operators (FFNOs) (Li et al., 2023). Our choice of models is motivated by the comparisons in Tran et al. (2023) that demonstrate that the process of factorization, wherein Fourier transforms act independently on each dimension of the input, leads to improved scalability with the number of layers.

We use the structured mesh datasets from (Li et al., 2023; Tran et al., 2023). In contrast to the `PyVista` meshes in Sec. 6 which were used to stress test the expressiveness of the models, the irregular geometries in this setting are meant to be indicative of real world scenarios. Further dataset, training, and evaluation details are given in Sec. G.

F.2. Results

As seen in Fig. 6, the Geo-FNO model diverges at the same time it starts to over smooth. In fact, once the number of layers hit 16 for the Airfoil dataset and 20 for the Plasticity dataset, the Geo-FFNO Rayleigh quotients were within machine precision of zero. One interpretation of the result is that the factorization in FFNO rewires the mesh edge connectivity (in frequency space) and thereby prevents nodes from sending messages to their typical neighbors.

G. FNO Dataset, Training, and Evaluation Details

In this section we provide further experimental details for Sec. F.

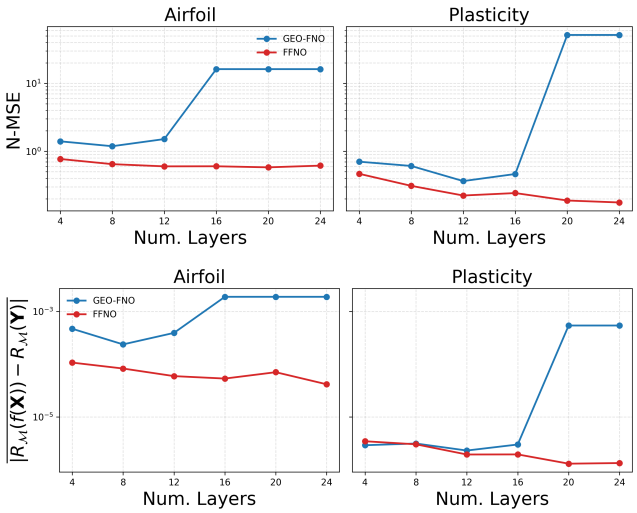


Figure 6. **Top:** NMSE as a function of the number of layers in the surrogate predictions. The FFNO model is better able to scale with the number of layers. **Bottom:** The average absolute error between the Rayleigh quotient of the ground truth and the surrogate predictions as a function of the number of layers in the surrogate. The FFNO model is better able to scale with the number of layers as it does not exhibit over smoothing.

G.1. FNO Datasets

The two structure mesh datasets from (Li et al., 2023; Tran et al., 2023) are the airfoil dataset and the plasticity dataset. We recap each briefly below.

G.1.1. Airfoil Dataset

The airfoil dataset considers the transonic flow over an airfoil. Let ρ^f be the fluid density, $\mathbf{v}$ the velocity vector, p the pressure, and E the total energy. The governing equation is the Euler equation:

$$\frac{\partial \rho^f}{\partial t} + \nabla \cdot \left(\rho^f \mathbf{v}\right) = 0, \qquad \frac{\partial \rho^f \mathbf{v}}{\partial t} + \nabla \cdot \left(\rho^f \mathbf{v} \otimes \mathbf{v} + p\mathbb{I}\right) = 0, \qquad \frac{\partial E}{\partial t} + \nabla \cdot \left((E+p)\mathbf{v}\right) = 0.$$

Concretely, the task is to predict a scalar field, the velocity mach number, for each input coordinate pair (x, y).

G.1.2. Plasticity Dataset

The plasticity dataset considers the plastic forging problem where a block of material $\Omega = [0, L] \times [0, H]$ is impacted by a frictionless, rigid die at $t = 0$. Let λ be the plastic multiplier constrained by $\lambda \geq 0$, $f(\sigma) \leq 0$, $\lambda \cdot f(\sigma) = 0$, let C be the isotropic stiffness tensor with Young's modulus $E = 200$ GPa and Poisson's ratio 0.3, and let σ_Y be the yield strength. The yield strength is set to $\sigma_Y = 70$ MPa with the mass density $\rho^s = 7850\ kg \cdot m^{-3}$. The dynamical system is given by

$$\sigma = C : (\varepsilon - \varepsilon_p), \qquad \dot{\varepsilon_p} = \lambda \nabla_\sigma f(\sigma), \qquad f(\sigma) = \sqrt{\frac{3}{2}} \left| \sigma - \frac{1}{3} tr(\sigma) \cdot I \right|_F - \sigma_Y.$$

Concretely, the task is to predict the displacement vector field for all time steps given a boundary condition.

G.2. FNO Training Details

We follow the same procedure as in Tran et al. (2023), using their publicly available code base `https://github.com/alasdairtran/fourierflow`. Hyper parameters are tuned for each layer number and model to achieve best performance under N-MSE loss. All runs were performed on a single H200 GPU (NVIDIA, 2025).

G.3. FNO Evaluation Details

Following Tran et al. (2023), we use the normalized mean squared error

$$\text{N-MSE} = \frac{1}{B}\sum_{i=1}^{B}\frac{\|f(\mathbf{X}_i) - \mathbf{Y}\|_2}{\|\mathbf{Y}\|_2}$$

where B is the batch size. We observed that the errors reported in Tran et al. (2023) were scaled up by a factor of 100, and we do the same for consistency. Instead of using the default mesh discretizations, we use Delaunay triangulations so that we can compute the mesh Rayleigh quotient in Definition 7. For the plasticity dataset, we average the Rayleigh quotient error over all data points and over all time steps.