PINOT: PARTITION-INVARIANT NEURAL OPERATOR TRANSFORMER FOR FLEXIBLE LATENT TOKENS

Anonymous authors
Paper under double-blind review

ABSTRACT

To reduce computational cost of global processing, several recent neural operators compress physical observations into a smaller set of latent tokens. Using fewer tokens lowers computation, but excessive compression can discard local spatial information, creating a trade-off between predictive accuracy and efficiency. It is therefore desirable to flexibly adapt the partitioning of physical observations into latent tokens according to the problem. This flexibility is particularly useful for complex geometries and nonuniform discretizations, where geometry, observation density, and local spatial variation can differ substantially across regions, making a fixed latent configuration inefficient. However, existing latent-token-based neural operators typically rely on prescribed latent configurations and do not explicitly ensure consistency under changes in latent tokenization. We introduce **PINOT**, a **Partition-Invariant Neural Operator Transformer** for flexible latent partitioning. PINOT uses patchwise scale-normalized polynomial coefficients for patch-scale-consistent latent tokens while preserving higher-order local information. Measure-weighted attention further makes global processing consistent with changes in patch density by accounting for the physical measure represented by each token. We theoretically establish partition and discretization consistency together with local expressivity of PINOT. Experiments show that PINOT generalizes to unseen patch resolutions and preserves substantially more spatial information with far fewer latent tokens. Based on these properties, PINOT demonstrates strong predictive performance against representative neural operator competitors, achieving the lowest error on all benchmarks with nonuniform discretizations. The code is available at `https://anonymous.4open.science/r/PINOT`.

1 INTRODUCTION

Data-driven surrogate models offer an efficient alternative to numerical solvers for partial differential equations (PDEs) by learning mappings from input conditions to solution fields. Neural operators (Kovachki et al., 2023) formulate this task as learning mappings between function spaces, aiming to support consistent prediction across different source discretizations (see Figure 1(a)). However, neural operators involving global spatial interactions can become computationally expensive as the number of spatial observations grows, limiting their applicability to large-scale real-world problems (Bonnet et al., 2022). To mitigate this cost, several recent neural operators (Wang & Wang, 2024; Wen et al., 2026) adopt a latent-token-based encoder–processor–decoder structure. The encoder compresses physical observations into a smaller number of latent tokens to reduce the cost of subsequent global interaction between the latent tokens in the processor, and the output field is then reconstructed from the processed latent tokens.

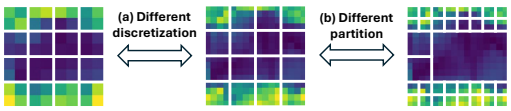

Figure 1: **Source discretization vs. patch partition.** The same underlying field can be represented by (a) different source discretizations, illustrated by different observation resolutions, and/or (b) different patch partitions, illustrated by white boundaries grouping the same observations into latent patches.

This compression introduces an inherent trade-off between computational efficiency and predictive accuracy: fewer tokens reduce computational cost, whereas excessive compression can disregard

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

local spatial information. This trade-off is particularly important for practical problems typically involving complex geometries and nonuniform observations, where observation density and local spatial variation can differ substantially across regions and problem instances (Liu et al., 2025a). Consequently, some regions may require more latent tokens to preserve local information, whereas others can be represented with fewer tokens without substantial information loss. Adapting the latent configuration—the number and spatial distribution of latent tokens—can therefore provide a more favorable balance between predictive accuracy and computational efficiency.

Such flexibility can be facilitated by consistency across latent configurations. More precisely, for the same underlying physical problem, predictions obtained from different latent configurations converge to an identical limiting output field under sufficient refinement. For brevity, we refer to this asymptotic property as consistency across latent configurations hereafter. Without such consistency, changes in latent configuration can introduce configuration-dependent variations in the model output, potentially limiting generalization to unseen latent configurations.

However, existing latent-token-based neural operators either offer limited flexibility in their latent configurations or do not explicitly ensure consistency under changes in latent configuration. For example, Transolver (Wu et al., 2024) and LNO (Wang & Wang, 2024) fix the number of latent tokens by design, limiting their flexibility at inference time. GINO (Li et al., 2023b) constructs its latent tokens on a regular grid, which restricts the spatial flexibility of the latent configuration. GAOT (Wen et al., 2026) can operate with different spatial latent configurations, but does not explicitly enforce consistency across such configurations. We refer the reader to Appendix B for a more detailed discussion.

To address this problem, we introduce **PINOT**, a Partition-Invariant Neural Operator Transformer. PINOT partitions the physical domain into spatial patches and represents the observations within each patch as a latent token. **PINOT** is designed to remain consistent as the patch partitioning changes (see Figure 1(b)). Additionally, it aims to preserve local spatial information when observations within each patch are compressed into a single latent token, enabling a favorable balance between predictive accuracy and computational efficiency. More specifically, PINOT employs a scale-normalized coefficient encoder and a continuous local expansion decoder, which represent each patch using coefficients of a shared polynomial basis while retaining higher-order local information. Scaling of the coefficients normalizes out the leading-order dependence on patch scale to maintain consistency across different patch partitions. PINOT further employs measure-weighted attention, which accounts for the physical measure of each patch (length, area, or volume depending on the spatial dimension) to prevent densely partitioned regions from exerting disproportionate influence and maintain consistent global processing across different patch partitions.

Theoretically, we prove that PINOT's predictions converge to the same limiting field under any admissible joint refinement, in which the patch partitions become finer and source discretizations become denser at a sufficient rate to keep each shrinking patch sufficiently well resolved. We further prove that, in the refinement limit, the scale-normalized coefficients retain local derivatives up to the prescribed polynomial degree and are strictly more informative than the analyzed class of smooth, scale-independent local aggregations based on the graph neural operator (GNO) (Li et al., 2020). Experimentally, we demonstrate generalization to unseen patch partitions and source discretizations, as well as substantially richer spatial-information preservation with far fewer latent tokens. Finally, across six PDE benchmarks, **PINOT** achieves the lowest prediction error on five, including all three benchmarks with nonuniform discretizations, where adaptive partitioning is employed.

Our contributions are threefold:

- **Flexible latent representation.** We introduce a scale-normalized coefficient encoder, a measure-weighted transformer, and a continuous local expansion decoder to support flexible latent configurations while preserving rich local spatial information.
- **Consistency and expressivity guarantees.** We establish consistency across patch partitions and source discretizations while showing the local expressivity of the scale-normalized coefficient representation by proving that it retains higher-order local information under refinement.
- **Partition generalization and adaptive partitioning.** We experimentally demonstrate generalization to unseen partitions and strong predictive performance, particularly on benchmarks with nonuniform discretizations through adaptive partitioning.

2 METHODOLOGY

2.1 PROBLEM DEFINITION

We now formalize PDE operator learning with variable source and target discretizations and flexible latent patch partitions, together with consistency under changes in such partitions and discretizations. Let $\Omega \subset \mathbb{R}^d$ denote a physical domain. Then we consider a family of parameterized PDEs with solution operator $\mathcal{G}^\dagger$:

$$\mathcal{G}^\dagger : \mathcal{V}(\Omega) \to \mathcal{U}(\Omega), \qquad u = \mathcal{G}^\dagger(v), \tag{1}$$

where $v : \Omega \to \mathbb{R}^{c_{\text{in}}}$ and $u : \Omega \to \mathbb{R}^{c_{\text{out}}}$ denote the input and solution fields, respectively.

As shown in Figure 2, the input field v is observed on source points that define the source discretization $\mathbb{X}_{\text{src}} = \{\boldsymbol{x}_{\text{src}}^j\}_{j=1}^{N_{\text{src}}} \subset \Omega$, and the solution is queried on target points that define the target discretization $\mathbb{X}_{\text{tar}} = \{\boldsymbol{x}_{\text{tar}}^j\}_{j=1}^{N_{\text{tar}}} \subset \Omega$. Their cardinalities and spatial distributions may vary.

Additionally, the physical domain Ω is partitioned into patches $\mathcal{P} = \{P_i\}_{i=1}^M$, with the source points in each patch P_i later represented by a single latent token. Each patch P_i is characterized by its center $\boldsymbol{r}_i$ and scale h_i, which may vary independently of the source $\mathbb{X}_{\text{src}}$ and target discretizations $\mathbb{X}_{\text{tar}}$. For simplicity, we assume symmetric and non-overlapping patches. However, extensions to nonsymmetric and clipped boundary patches are discussed in Remark 4, while overlapping patches are considered in Section C.8.

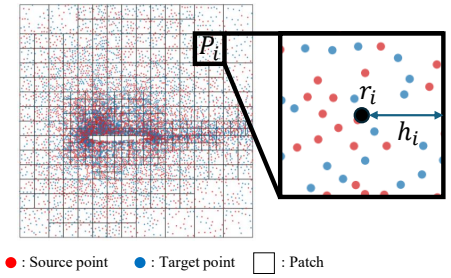

Figure 2: **Problem setup.** The input field is observed at the source points, while the output field is queried at the target points. The domain is partitioned into patches, where each patch P_i is characterized by its center $\boldsymbol{r}_i$ and scale h_i.

We aim to develop a neural operator $\mathcal{G}_{\boldsymbol{\theta}}$ that approximates the solution operator $\mathcal{G}^\dagger$ under flexible choices of the source discretization $\mathbb{X}_{\text{src}}$, the target discretization $\mathbb{X}_{\text{tar}}$, and the patch partition $\mathcal{P}$:

$$\mathcal{G}_{\boldsymbol{\theta}}\left(\mathbb{X}_{\text{src}}, v|_{\mathbb{X}_{\text{src}}}, \mathbb{X}_{\text{tar}}; \mathcal{P}\right) \approx \mathcal{G}^\dagger(v)\big|_{\mathbb{X}_{\text{tar}}}. \tag{2}$$

Here, $f|_{\mathbb{X}}$ denotes the observation or evaluation of a field f on discretization $\mathbb{X}$.

To facilitate flexibility across varying source discretizations $\mathbb{X}_{\text{src}}$ and patch partitions $\mathcal{P}$, we design the neural operator $\mathcal{G}_{\boldsymbol{\theta}}$ such that its prediction converges to a limiting field $\widehat{u}_{\boldsymbol{\theta}}(v)$ under admissible joint refinement. Specifically, as the patch partition $\mathcal{P}$ is refined and the source discretization $\mathbb{X}_{\text{src}}$ becomes denser at a rate sufficient to keep each patch P_i well resolved by source points, the limit is independent of the particular refinement sequence:

$$\mathcal{G}_{\boldsymbol{\theta}}\left(\mathbb{X}_{\text{src}}, v|_{\mathbb{X}_{\text{src}}}, \mathbb{X}_{\text{tar}}; \mathcal{P}\right) \longrightarrow \widehat{u}_{\boldsymbol{\theta}}(v)|_{\mathbb{X}_{\text{tar}}}. \tag{3}$$

2.2 MODEL ARCHITECTURE

To achieve the operator-approximation objective in Eq. (2) while satisfying the consistency requirement in Eq. (3), we propose PINOT, which comprises a *scale-normalized coefficient encoder*, a *measure-weighted transformer*, and a *continuous local expansion decoder*. As illustrated in Figure 3, the three components form a conventional encoder–processor–decoder architecture (Wang & Wang, 2024; Wen et al., 2026).

2.2.1 SCALE-NORMALIZED COEFFICIENT ENCODER

The scale-normalized coefficient encoder is designed to convert an arbitrary[1] set of local observations into a patch-scale-consistent representation while retaining higher-order local information. Specifically, as illustrated in Figure 3(a), each source point observation, consisting of its position $\boldsymbol{x} \in \mathbb{X}_{\text{src}}$ and input field value $v(\boldsymbol{x})$, is first lifted pointwise by a multi-layer perceptron (MLP) to a higher-dimensional feature $L(\boldsymbol{x})$.

[1]Here, an arbitrary set of local observations refers to one for which the local least-squares problem is well-posed, *i.e.*, the corresponding design matrix has full column rank.

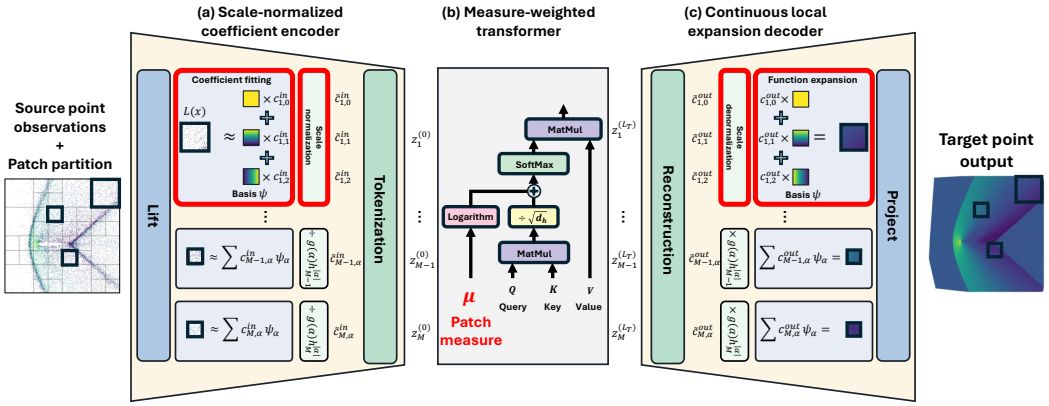

Figure 3: **Overview of PINOT**. (a) The *scale-normalized coefficient encoder* employs polynomial coefficient fitting and scale normalization to encode the source point observations into latent tokens while supporting partition consistency and preserving local spatial variation. Next, (b) the *measure-weighted transformer* incorporates patch measure into conventional attention to support partition-consistent global processing of the latent tokens. Finally, (c) the *continuous local expansion decoder* restores the original coefficient scale and reconstructs continuous local functions in the same polynomial basis as in (a), enabling evaluation at arbitrary target coordinates.

Then, we approximate the lifted features $L(\boldsymbol{x})$ as a patchwise linear combination of the polynomial basis functions $\{\psi_\alpha\}$ with coefficients $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$:

$$L(\boldsymbol{x}) \approx \sum_{|\alpha|\le K} \boldsymbol{c}^{\mathrm{in}}_{i,\alpha}\psi_\alpha(N_i(\boldsymbol{x}))\,, \qquad \boldsymbol{x}\in P_i\cap\mathbb{X}_{\mathrm{src}}. \tag{4}$$

Here, $N_i(\boldsymbol{x}) = \frac{\boldsymbol{x}-\boldsymbol{r}_i}{h_i}$ maps each patch P_i to a common reference domain, allowing patches with different centers $\boldsymbol{r}_i$ and scales h_i to share the same set of polynomial basis functions $\{\psi_\alpha\}$. Also, $\alpha=(\alpha_1,\dots,\alpha_d)\in\mathbb{N}_0^d$ is a multi-index specifying the polynomial degree along each coordinate, and $|\alpha|=\sum_{k=1}^d\alpha_k$ denotes the total degree. The polynomial basis functions $\{\psi_\alpha\}$ are chosen to satisfy two conditions: (1) they span all polynomials of total degree at most K, and (2) each basis function is even or odd according to its total degree $|\alpha|$. Examples of such polynomial basis functions $\{\psi_\alpha\}$ include Legendre polynomials and Chebyshev polynomials of the first and second kinds. For a detailed description of the polynomial basis functions $\{\psi_\alpha\}$, refer to Appendix C.1. The coefficients $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$ are obtained by least-squares fitting to minimize the discrepancy between the polynomial expansion in Eq. (4) and the lifted features $L(\boldsymbol{x})$ at the source points.

In this way, the lifted features $L(\boldsymbol{x})$ within each patch P_i can be represented by a fixed number of basis coefficients $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$, independent of the number of source points, thereby accommodating varying source discretizations $\mathbb{X}_{\mathrm{src}}$. More importantly, coefficients $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$ associated with higher-order polynomial basis functions can retain information about local spatial variations, such as slopes, curvatures, and higher-order structure, reducing the information loss caused by compressing the lifted features $L(\boldsymbol{x})$ within each patch P_i into a compact latent representation.

To compensate for the patch-scale dependence of the coefficients $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$, we scale the coefficients $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$:

$$\widetilde{\boldsymbol{c}}^{\mathrm{in}}_{i,\alpha} = s_{i,\alpha}\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}, \qquad s_{i,\alpha}=\frac{1}{g(|\alpha|)h_i^{|\alpha|}}. \tag{5}$$

Here, the $h_i^{|\alpha|}$ term in the scaling factor $s_{i,\alpha}$ is motivated by a Taylor expansion of the lifted field $L(\boldsymbol{x})$ around the patch center $\boldsymbol{r}_i$, which shows that the magnitude of the coefficient $\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$ scales as $h_i^{|\alpha|}$ to leading order, as described in Appendix C.2. Dividing by this factor therefore removes the leading-order dependence on the patch scale h_i. As a practical implementation detail, the scaling factor $s_{i,\alpha}$ further includes a learnable factor $g(|\alpha|)$, which is shared across all patches in $\mathcal{P}$, allowing the magnitude of $\widetilde{\boldsymbol{c}}^{\mathrm{in}}_{i,\alpha}$ to be adjusted to a numerically favorable range for training.

Finally, the normalized coefficients $\widetilde{\boldsymbol{c}}^{\mathrm{in}}_{i,\alpha}$, together with their basis indices α, are embedded patchwise by an MLP to produce a fixed-dimensional patch token $\boldsymbol{z}_i^{(0)}$, which is then passed to the measure-weighted transformer for global processing, as described in the following section.

2.2.2 MEASURE-WEIGHTED TRANSFORMER

The measure-weighted transformer in Figure 3(b) leverages the measure-weighted attention to maintain consistency under changes in partition density. Since standard attention assigns the same prior weight to every token, subdividing a region into more patches can artificially increase its influence even when the represented field remains unchanged. To account for such dependence on the partition density, the measure-weighted attention modifies the conventional attention head by incorporating the physical measure $\mu_i = |P_i|$ of each patch P_i into the attention logits:

$$\boldsymbol{o}_i^{(\ell)} = \sum_{j=1}^{M} A_{i,j}^{(\ell)} \boldsymbol{v}_j^{(\ell)}, \qquad A_{i,j}^{(\ell)} = \mathrm{softmax}_j \left(\log \mu_j + s\left(\boldsymbol{q}_i^{(\ell)}, \boldsymbol{k}_j^{(\ell)} \right) \right). \tag{6}$$

Here, for $\ell = 0, \cdots, L_T - 1$, $\boldsymbol{q}_i^{(\ell)}$, $\boldsymbol{k}_i^{(\ell)}$, and $\boldsymbol{v}_i^{(\ell)}$ denote the query, key, and value vectors obtained from the patch token $\boldsymbol{z}_i^{(\ell)}$ through learned linear projections, respectively, where L_T denotes the total number of transformer layers; d_h is the feature dimension of each attention head; $s(\boldsymbol{q}, \boldsymbol{k}) = \boldsymbol{q}^\top \boldsymbol{k} / \sqrt{d_h}$ is the scaled dot-product score; $A_{i,j}^{(\ell)}$ is the attention weight; and $\boldsymbol{o}_i^{(\ell)}$ is the resulting attention output. The physical measure μ_j corresponds to the length, area, or volume of each patch P_j in one, two, or three dimensions, respectively.

Addition of the $\log \mu_j$ term to the scaled dot-product score $s\left(\boldsymbol{q}_i^{(\ell)}, \boldsymbol{k}_j^{(\ell)} \right)$ in Eq. (6) allows each region in the physical domain Ω to contribute according to its physical measure μ_j rather than the number of patches used to partition the region. A detailed analysis of this consistency with respect to partition density is provided in Appendix C.4.

Through L_T measure-weighted transformer layers equipped with the measure-weighted attention, the input patch tokens $\{\boldsymbol{z}_i^{(0)}\}_{i=1}^M$ are transformed into $\{\boldsymbol{z}_i^{(L_T)}\}_{i=1}^M$, which are then passed to the continuous local expansion decoder described in the following section.

2.2.3 CONTINUOUS LOCAL EXPANSION DECODER

The continuous local expansion decoder is designed to enable evaluation at arbitrary target points. Specifically, as depicted in Figure 3(c), the processed token $\boldsymbol{z}_i^{(L_T)}$ from the measure-weighted transformer is passed through a token-wise prediction MLP, whose output is interpreted as the normalized expansion coefficients $\{\widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}}\}_{|\alpha| \le K}$. These coefficients $\widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}}$ are then denormalized by inverting the scaling factor $s_{i,\alpha}$ used by the scale-normalized coefficient encoder in Eq. (5). The resulting coefficients $\boldsymbol{c}_{i,\alpha}^{\mathrm{out}}$ together with the polynomial basis functions $\{\psi_\alpha\}$ used by the scale-normalized coefficient encoder define a continuous local expansion over P_i:

$$\sum_{|\alpha| \le K} \boldsymbol{c}_{i,\alpha}^{\mathrm{out}} \psi_\alpha \left(N_i(\boldsymbol{x}) \right), \qquad \boldsymbol{x} \in P_i. \tag{7}$$

The continuous local expansion is evaluated at the target points and subsequently mapped to the final output through a pointwise MLP projection. Since the expansion defines a continuous function over each patch P_i, it can be evaluated at any target coordinate $\boldsymbol{x}_{\mathrm{tar}} \in P_i$, allowing flexible target discretizations $\mathbb{X}_{\mathrm{tar}}$ with varying numbers and arrangements of target points.

2.3 THEORETICAL ANALYSES

We now theoretically analyze two key properties of PINOT: (1) consistency with respect to source discretization and patch partition, and (2) local expressivity of the scale-normalized coefficients.

Theorem 1 (**Source discretization and patch partition invariance**). *For a fixed PINOT model $\mathcal{G}_{\boldsymbol{\theta}}$ and a fixed input field v, there exists a limiting output field $\widehat{u}_{\boldsymbol{\theta}}(v)$, independent of the particular admissible joint refinement sequence, such that, for every target point set $\mathbb{X}_{\mathrm{tar}}$,*

$$\mathcal{G}_{\boldsymbol{\theta}} \left(\mathbb{X}_{\mathrm{src}}, v|_{\mathbb{X}_{\mathrm{src}}}, \mathbb{X}_{\mathrm{tar}}; \mathcal{P} \right) \longrightarrow \widehat{u}_{\boldsymbol{\theta}}(v)|_{\mathbb{X}_{\mathrm{tar}}}. \tag{8}$$

Theorem 2 (**Local expressivity**). *Under admissible refinement, the scale-normalized coefficients $\{\widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{in}}\}_{|\alpha| \le K}$ collectively retain all local derivative information of the lifted field up to order K.*

In Theorem 1, we use the terms **source discretization invariance** and **patch partition invariance** to denote *asymptotic consistency* under admissible joint refinement, where the patch partition is refined while the source discretization within each patch remains sufficiently well resolved relative to the patch scale. Under such refinement, representations and model predictions obtained along different admissible refinement sequences converge to the same continuum limits. Precise technical conditions are provided in Appendix C.1. Theorem 1 shows that PINOT converges to a common continuum prediction under admissible refinements of the source discretization and patch partition.

Theorem 2 shows that the scale-normalized coefficients retain higher-order local information, including changes in slope, curvature, and higher-order spatial variations up to order K. For $K \geq 1$, Corollary 13 in Appendix C.6 shows that the limiting scale-normalized coefficient representation is strictly more informative than that induced by the analyzed class of smooth, scale-independent local GNO aggregations (Li et al., 2020), whose limit depends only on the lifted field at the patch center.

Proofs and supporting theoretical analyses of Theorems 1 and 2 as well as Corollary 13 are provided in Appendix C, with empirical validation presented in Sections 3.2.1 and 3.2.2.

3 Experimental Evaluation

3.1 Experimental Setup

Datasets. We focus on PDE benchmarks with nonuniform discretizations, where point density varies substantially across the physical domain, while also including uniformly discretized benchmarks to assess generality across discretization structures. The benchmarks with nonuniform discretizations include 2D airfoils (NACA0012 and RAE2822) (Wen et al., 2026) and 3D DeepJEB (Hong et al., 2024), while the benchmarks with uniform discretizations include 1D Burgers (Li et al., 2021), 2D Navier–Stokes (NS-Gauss) (Herde et al., 2024), and 2D Compressible Euler (CE-RP) (Herde et al., 2024). We use NACA0012 as the default dataset for our analyses. For the quantitative evaluation of patch-partition and source-discretization invariance, we instead use 1D Burgers to systematically vary patch and source point counts over a wide range. The implementation details are provided in Appendix D.1.

Competitors. We select competing methods to cover FNO (Li et al., 2021) and transformer (Vaswani et al., 2017)-based processor architectures, as well as different latent-representation strategies, as summarized in Table 1. Here, "explicit physical-space anchoring" indicates whether each latent token is explicitly associated with a prescribed local region in the physical domain, while "latent-configuration flexibility" indicates whether the model can operate with different numbers or spatial arrangements of latent tokens while keeping the learned parameters fixed regardless of prediction accuracy degradation. For example, the latent representations of Transolver (Wu et al., 2024) and LNO (Wang & Wang, 2024) are not explicitly associated with prescribed local regions in the physical domain. Since OFormer (Li et al., 2023a) operates directly on the source discretization without constructing a separate latent token set, the two criteria concerning latent configurations are not applicable. The implementation details are provided in Appendix D.2.

Table 1: **Summary of competing methods.** Here ✓ and × indicate whether each property is supported, while △ indicates limited flexibility.

Method	Processor	Uses latent tokens	Explicit physical-space anchoring	Latent-configuration flexibility
GINO (Li et al., 2023b)	FNO	✓	✓	△ (regular grid)
OFormer (Li et al., 2023a)	Transformer	×	–	–
GAOT (Wen et al., 2026)	Transformer	✓	✓	✓
Transolver (Wu et al., 2024)	Transformer	✓	×	×
LNO (Wang & Wang, 2024)	Transformer	✓	×	×

PINOT configuration. Unless otherwise specified, we use Legendre polynomials as the local basis for PINOT. For patch construction, we partition the domain into nonadaptive equal-sized patches for

datasets with uniform discretizations. For datasets with nonuniform discretizations, we use point-density-adaptive partitioning (Carrier et al., 1988). Starting from a uniform square partition in 2D or cubic partition in 3D, each patch exceeding a predefined point-count threshold is recursively subdivided by halving it along each spatial dimension, producing four child patches in 2D and eight in 3D. Empty patches are removed. This yields finer patches in densely sampled regions while avoiding unnecessary tokens in sparsely sampled or empty regions. The maximum total polynomial degree K, initial partition, and point-count threshold for each dataset are provided in Appendix D.3.

3.2 Results and Analyses

We organize our experiments around two properties that enable adaptive, data-dependent patch partitioning in PINOT: generalization across patch partitions and the local expressivity of the coefficient representation. We then assess the practical benefit of the adaptive partitioning compared to nonadaptive partitioning. Finally, we evaluate whether these advantages translate into improved predictive performance across diverse PDE benchmarks.

3.2.1 Generalization Across Patch Partitions

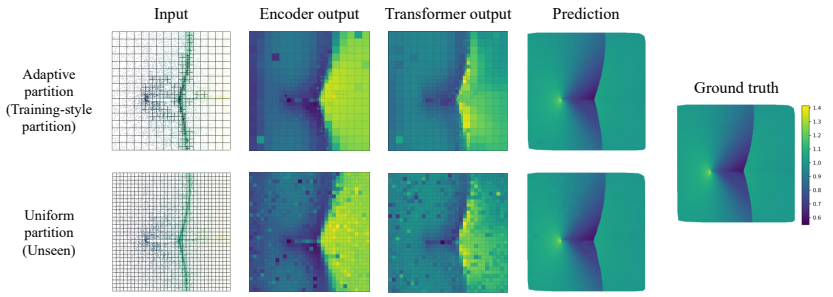

Figure 4: **Qualitative illustration of patch partition consistency.** PINOT is trained on NACA0012 using adaptive partitions and evaluated on the same test sample using either a training-style adaptive partition (top) or an unseen uniform partition (bottom). Despite the change in the partition, the representations of the scale-normalized coefficient encoder and the measure-weighted transformer exhibit similar spatial structures, and the final predictions are also consistent.

We first qualitatively examine how PINOT behaves when the patch partition is changed at inference. Figure 4 presents a representative example in which a model trained with adaptive partitions is evaluated using either an adaptive partition generated in the same manner as during training or an unseen uniform partition. Despite the substantial change in patch partition, the representations of the scale-normalized coefficient encoder and the measure-weighted transformer preserve similar spatial structures, and the final predictions are also consistent. This provides an intuitive illustration that the representation learned by PINOT is not tied to a particular patch partition.

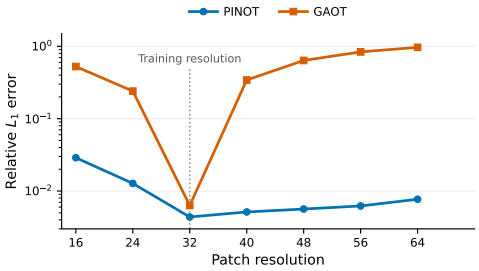

Figure 5: **Quantitative evaluation of patch-partition consistency.** PINOT and GAOT are trained on 1D Burgers using $M = 32$ patch tokens and evaluated at different patch resolutions without retraining. PINOT maintains a low relative L_1 error across unseen patch resolutions, whereas GAOT is highly sensitive to changes in the partition. The dashed line marks the training patch resolution.

We next quantitatively evaluate consistency under changes in the patch partition. GAOT is selected as a competitor because it is the only baseline in Table 1 with latent-configuration flexibility, which allows changes in the number and spatial arrangement of patch tokens (*i.e.*, latent tokens). We train both models with a total of $M = 32$ patch tokens and evaluate them at different patch resolutions without retraining. Here, M denotes the number of patch tokens provided to the transformer attention layers.

As shown in Figure 5, PINOT maintains a low relative L_1 error across a broad range of unseen patch resolutions, whereas GAOT degrades substantially as the patch resolution moves away from the training configuration. This indicates that PINOT is considerably less sensitive to changes in the latent patch resolution. These results provide quantitative evidence that PINOT can generalize to unseen latent tokenizations at inference without retraining. As discussed in Section 2.2, this generalizability depends strongly on the basis used for local approximation, and we provide additional experiments evaluating generalization across patch partitions using different bases in Appendix E.2. Beyond patch-partition changes, we also evaluate generalization to unseen source discretizations. The results are available in Figure 6 of Appendix E.1.

3.2.2 Expressivity of the Coefficient Representation

Table 2: **Input reconstruction for latent representation evaluation.** Input data sampled from the NACA0012 solution fields are encoded into latent tokens and reconstructed by the corresponding decoders. PINOT achieves the lowest reconstruction error (relative L_1 error) while using the fewest latent tokens, indicating better preservation of the local spatial information in the input field.

Model	Encoder	# Latent tokens	Reconstruction error (%)
GINO	GNO	4,096	15.0
GAOT	MAGNO	4,096	13.8
PINOT	Scale-normalized coefficient encoder	501.9	**2.92**

To assess the ability of PINOT's scale-normalized coefficient encoder to preserve local spatial information under substantial compression, we design an input reconstruction experiment. Specifically, input data sampled from the NACA0012 solution fields are passed through the encoder and decoder of each model without the processor. The encoder and decoder are then trained to reconstruct the original input. A lower reconstruction error indicates less information loss during compression by the encoder. We exclude the processor to directly measure the information preservation capability of the latent representation.

As competitors, we select GINO and GAOT, which satisfy the "explicit physical-space anchoring" criterion in Table 1, because they encode local spatial information from specific physical regions into latent tokens, unlike the other competitors. This enables a direct comparison of local spatial information preservation. The GNO and MAGNO encoders used by the selected competitors utilize a fixed set of $4{,}096$ uniformly spaced latent tokens, whereas our scale-normalized coefficient encoder uses a variable number of tokens with an average of 501.9, supported by adaptive partitioning.

As shown in Table 2, our scale-normalized coefficient encoder achieves a substantially lower relative L_1 reconstruction error of 2.92%, compared to the competitors, while using approximately one-eighth as many latent tokens. These results show that our scale-normalized coefficient representation preserves substantially more spatial information with a more compact latent representation, providing empirical support for our theoretical results in Theorem 2 and Corollary 13.

3.2.3 Adaptive Partitioning for Nonuniform Discretizations

To evaluate the benefit of adaptive partitioning facilitated by the generalization across patch partitions and the expressive coefficient representation demonstrated in Sections 3.2.1 and 3.2.2, we compare nonadaptive uniform partitions with point-density-adaptive partitions while keeping the PINOT architecture and all other training settings fixed. As shown in Table 3, adaptive partitioning achieves the lowest relative L_1 error while using fewer latent tokens than the finer uniform partitions. This improved efficiency is enabled by allocating finer latent resolution to densely sampled regions while avoiding unnec-

Table 3: **Effect of adaptive partitioning**. Adaptive and nonadaptive partitioning are compared on NACA0012. Adaptive partitioning achieves the lowest relative L_1 error while using fewer latent tokens than the finer uniform partitions.

Patch partition	# Latent tokens	Relative L_1 error (%)
Nonadaptive 16×16	256	6.91
Adaptive	501.9	**5.72**
Nonadaptive 23×23	529	6.41
Nonadaptive 32×32	1,024	6.18

essary patches in sparse or empty regions. Additional computational-efficiency results, including theoretical complexity, peak GPU memory, and inference time, are provided in Appendix F.1.

3.2.4 Overall Predictive Accuracy

Table 4: **Predictive performance comparison.** We compare the predictive performance of PINOT with competitors on PDE benchmarks with both uniform and nonuniform observation sampling densities. PINOT utilizes adaptive partitioning for the nonuniform benchmarks. Relative L_1 errors are reported as percentages (mean $\pm$ standard deviation over multiple runs); a lower relative L_1 error indicates better predictive performance. The best result is shown in bold and the second-best is underlined.

Model	Nonuniform			Uniform		
	2D NACA0012	2D RAE2822	3D DeepJEB	1D Burgers	2D NS-Gauss	2D CE-RP
GINO	9.68 ± 0.08	8.63 ± 0.41	31.8 ± 0.49	1.20 ± 0.35	14.4 ± 0.06	20.2 ± 1.59
OFormer	8.78 ± 2.18	13.5 ± 7.16	-	1.48 ± 0.23	27.4 ± 4.94	26.8 ± 1.08
GAOT	11.9 ± 0.17	12.7 ± 0.26	30.9 ± 0.90	$\underline{0.57 \pm 0.08}$	$\underline{5.08 \pm 0.14}$	$\mathbf{8.40 \pm 0.28}$
Transolver	$\underline{6.44 \pm 0.11}$	$\underline{5.53 \pm 0.16}$	$\underline{23.5 \pm 0.48}$	2.51 ± 0.43	41.78 ± 6.04	31.9 ± 3.95
LNO	7.61 ± 1.05	6.92 ± 1.06	27.3 ± 0.83	1.50 ± 0.28	11.7 ± 1.54	16.5 ± 0.59
PINOT	$\mathbf{5.72 \pm 0.08}$	$\mathbf{5.35 \pm 0.04}$	$\mathbf{19.4 \pm 0.28}$	$\mathbf{0.56 \pm 0.11}$	$\mathbf{3.98 \pm 0.53}$	$\underline{9.58 \pm 0.67}$

Finally, we evaluate whether the preceding advantages—the generalization across patch partitions, the expressive coefficient representation, and adaptive partitioning, demonstrated in Sections 3.2.1, 3.2.2, and 3.2.3—translate into improved predictive performance across diverse PDE benchmarks.

Table 4 compares the predictive performance of PINOT with the competitors summarized in Table 1 across benchmarks with uniform and nonuniform observation sampling densities, using the relative L_1 error. Among the competing methods, GAOT is the only model with latent-configuration flexibility; however, as shown in Figure 5, its predictive accuracy degrades substantially when the latent configuration differs from that used during training. Therefore, across datasets with nonuniform discretizations, we apply adaptive partitioning only to PINOT, while evaluating the competing methods with their prescribed latent configurations.

PINOT achieves the lowest error across all benchmarks with nonuniform discretizations. This strong predictive performance reflects the combined benefits of adaptive partitioning, which allocates finer latent resolution to densely sampled regions, and the expressive local coefficient representation. GINO and GAOT, whose encoders preserve less local spatial information than PINOT in Table 2, exhibit substantially higher errors on these datasets. PINOT also remains highly competitive on benchmarks with uniform discretizations, achieving the lowest error except for CE-RP, indicating that the benefit of its local coefficient representation can vary across PDEs. Overall, the results demonstrate that the flexibility and expressivity of latent representations of PINOT translate into strong predictive performance across diverse discretization structures and spatial dimensions.

4 Conclusion

We presented PINOT, which enables flexible latent partitioning while maintaining consistency across source discretizations and patch configurations. Theoretically, we established asymptotic consistency across patch partitions and source discretizations while showing that the scale-normalized coefficient representation preserves higher-order local information. Experiments demonstrated generalization to unseen partitions and discretizations, richer spatial-information preservation with fewer latent tokens, and strong predictive performance, particularly on benchmarks with nonuniform discretizations using adaptive partitioning. Taken together, these results showed that the flexibility and expressiveness of PINOT's latent representation enable a favorable trade-off between predictive accuracy and latent token count. PINOT's performance may depend on the choice of patch partition, suggesting a need for a more systematic study of alternative schemes. Although our theoretical analysis accommodates different patch shapes and overlapping patches, these settings remain to be explored empirically.

AI USE STATEMENT

We used generative AI tools to assist with writing refinement, literature searches, mathematical review, and code adaptation and refinement. The authors developed the manuscript's main narrative and logical structure and wrote the initial drafts. Generative AI tools assisted in improving clarity, grammar, and readability. For related work, generative AI tools helped identify relevant publications and review the accuracy of our descriptions of existing methods. The authors read and analyzed the relevant papers and drafted the characterization and comparison of baseline methods.

The authors developed the proposed method and formulated the mathematical claims, theorems, and initial arguments. Generative AI tools provided feedback on mathematical correctness, logical consistency, and the scope and wording of these claims. This feedback informed revisions to the reasoning and presentation, which the authors reviewed before incorporating them into the manuscript.

For the proposed method, the authors implemented the core algorithmic logic, and generative AI tools assisted with code refinement. The authors repeatedly reviewed the resulting changes for correctness and consistency with the intended method. For baseline implementations, generative AI coding assistants helped adapt official implementations to the training and evaluation pipelines used for our benchmarks. The authors repeatedly reviewed these adaptations and their hyperparameter settings against the original papers and implementations, checking fidelity to the baseline methods while applying a unified experimental protocol.

Across these activities, revisions made with generative AI assistance underwent iterative cycles of feedback, author review, and further revision. The authors cross-checked the relevant literature, mathematical arguments, and implementation details throughout this process and made the final decisions about which changes to incorporate. We take full responsibility for the final manuscript, mathematical claims, code, and reported results.

REPRODUCIBILITY STATEMENT

The architecture of PINOT is described in Section 2.2, with implementation details and dataset-specific patch configurations in Appendix D.3 and the batched coefficient encoder in Appendix F.2. The assumptions and proofs supporting our theoretical results are provided in Appendix C. Dataset sources, preprocessing, train–test splits, and evaluation metrics are documented in Appendix D.1, while baseline configurations and the shared training protocol are detailed in Appendix D.2. Computational complexity, hardware configuration, and empirical memory and inference-time measurements are reported in Appendix F.1; the efficiency experiments were conducted on an NVIDIA RTX A6000 GPU. The source code for PINOT is available through the link provided in the Abstract to support reproduction of our results.

REFERENCES

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators, 2025. URL https://arxiv.org/abs/2402.12365.

Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 23463–23478. Curran Associates, Inc., 2022. doi: 10.52202/068431-1705. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/94ab7b23a345f93333eac8748a66c763-Paper-Datasets_and_Benchmarks.pdf.

Shuhao Cao. Choose a transformer: Fourier or galerkin, 2021. URL https://arxiv.org/abs/2105.14995.

J. Carrier, L. Greengard, and V. Rokhlin. A fast adaptive multipole algorithm for particle simulations. *SIAM Journal on Scientific and Statistical Computing*, 9(4):669–686, 1988. doi: 10.1137/0909044. URL https://doi.org/10.1137/0909044.

Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pp. 481–490, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939738. URL https://doi.org/10.1145/2939672.2939738.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for pdes, 2024. URL https://arxiv.org/abs/2405.19101.

Seongjun Hong, Yongmin Kwon, Dongju Shin, Jangseop Park, and Namwoo Kang. Deepjeb: 3d deep learning-based synthetic jet engine bracket dataset, 2024. URL https://arxiv.org/abs/2406.09047.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023. URL http://jmlr.org/papers/v24/21-1524.html.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning, 2023a. URL https://arxiv.org/abs/2205.13671.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations, 2020. URL https://arxiv.org/abs/2003.03485.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2021. URL https://arxiv.org/abs/2010.08895.

Zongyi Li, Nikola Borislavov Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Prakash Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Anima Anandkumar. Geometry-informed neural operator for large-scale 3d pdes, 2023b. URL https://arxiv.org/abs/2309.00583.

Menghan Liu, Jianhuan Cen, Ziyang Zhou, Haolong Fan, Hongji Li, Ping Wei, Guohang Peng, Changye He, Yuzhe Qin, Yutong Lu, and Qingsong Zou. Cfdoneval: A comprehensive evaluation of operator-learning neural network models for computational fluid dynamics. In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 5752–5760. International Joint Conferences on Artificial Intelligence Organization, 8 2025a. doi: 10.24963/ijcai.2025/640. URL https://doi.org/10.24963/ijcai.2025/640. Main Track.

Qibang Liu, Weiheng Zhong, Hadi Meidani, Diab Abueidda, Seid Koric, and Philippe Geubelle. Geometry-informed neural operator transformer, 2025b. URL https://arxiv.org/abs/2504.19452.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, March 2021. ISSN 2522-5839. doi: 10.1038/s42256-021-00302-5. URL http://dx.doi.org/10.1038/s42256-021-00302-5.

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating eulerian fluid simulation with convolutional networks, 2022. URL https://arxiv.org/abs/1607.03597.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,

2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse pde problems, 2024. URL https://arxiv.org/abs/2406.03923.

Shizheng Wen, Arsh Kumbhat, Levi Lingsch, Sepehr Mousavi, Yizhou Zhao, Praveen Chandrashekar, and Siddhartha Mishra. Geometry aware operator transformer as an efficient and accurate neural surrogate for pdes on arbitrary domains, 2026. URL https://arxiv.org/abs/2505.18781.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries, 2024. URL https://arxiv.org/abs/2402.02366.

Yinhao Zhu, Nicholas Zabaras, Phaedon-Stelios Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, October 2019. ISSN 0021-9991. doi: 10.1016/j.jcp.2019.05.024. URL http://dx.doi.org/10.1016/j.jcp.2019.05.024.

APPENDIX CONTENTS

A Notation **14**

B Related Work **15**

B.1 Neural PDE Solvers 15

B.2 Latent Representations in Neural Operators 15

B.3 Relation to PINOT 16

C Theoretical Analysis **16**

C.1 Setting and Assumptions 16

C.2 Scale-Normalized Local Coefficient Representation 19

C.3 Continuum Token Field 21

C.4 Partition Consistency of Measure-Weighted Attention 22

C.5 Continuous Decoding and Target Discretization 23

C.6 Comparison with Smooth Scale-Independent Local GNO Aggregations 25

C.7 Proofs of Theorems 1 and 2 27

C.8 Extension to Overlapping Patches 27

D Experimental Details **29**

D.1 Dataset Configuration 29

D.2 Baseline Implementations 33

D.3 Implementation Details of PINOT 35

E Additional Experimental Results **36**

E.1 Source-discretization invariance 36

E.2 Effect of basis choice on patch-partition consistency 37

F Computational Efficiency and Implementation **37**

F.1 Computational Complexity and Empirical Efficiency 37

F.2 Efficient Batched Implementation of the Coefficient Encoder 39

A NOTATION

Table 5 summarizes the principal notation used in the main text, grouped by problem setting and model component. Subscripts src and tar distinguish source observations from target queries. For patchwise representations, i indexes patches, α indexes polynomial basis functions, and ℓ indexes transformer layers. Superscripts in and out distinguish encoder and decoder coefficients, while a tilde denotes scale normalization.

Table 5: Principal notation used in the main text.

Symbol	Description	Symbol	Description
Problem and discretization			
$\Omega \subset \mathbb{R}^d$	Physical domain in d dimensions.	$\mathcal{V}(\Omega), \mathcal{U}(\Omega)$	Input and solution function spaces.
v, u	Input and solution fields, $u = \mathcal{G}^\dagger(v)$.	$c_{\mathrm{in}}, c_{\mathrm{out}}$	Input and output channel counts.
$\mathcal{G}^\dagger, \mathcal{G}_{\boldsymbol{\theta}}$	Ground-truth and learned solution operators.	$\widehat{u}_{\boldsymbol{\theta}}(v)$	Limiting output under admissible joint refinement.
$\mathbb{X}_{\mathrm{src}}, \mathbb{X}_{\mathrm{tar}}$	Source observation and target query sets.	$N_{\mathrm{src}}, N_{\mathrm{tar}}$	Source and target point counts.
$\boldsymbol{x}^j_{\mathrm{src}}, \boldsymbol{x}^j_{\mathrm{tar}}$	Source and target coordinates.	$f\|_{\mathbb{X}}$	Field f restricted to point set $\mathbb{X}$.
Patch geometry			
$\mathcal{P} = \{P_i\}_{i=1}^M$	Partition into M patches P_i, each defining one token.	$\boldsymbol{r}_i, h_i$	Center and spatial scale of patch P_i.
$\mu_i = \lvert P_i \rvert$	Physical measure of patch P_i.	$N_i(\boldsymbol{x})$	Reference-coordinate map $(\boldsymbol{x} - \boldsymbol{r}_i)/h_i$.
Scale-normalized coefficient encoder			
$L(\boldsymbol{x})$	Pointwise feature lifted from $\boldsymbol{x}$ and $v(\boldsymbol{x})$.	ψ_α	Local polynomial basis function.
$\alpha, \lvert\alpha\rvert$	Polynomial multi-index and total degree.	K	Maximum polynomial total degree.
$\boldsymbol{c}^{\mathrm{in}}_{i,\alpha}$	Input coefficient for mode α in patch P_i.	$\widetilde{\boldsymbol{c}}^{\mathrm{in}}_{i,\alpha}$	Scale-normalized input coefficient.
$s_{i,\alpha}$	Normalization factor $1/(g(\lvert\alpha\rvert)h_i^{\lvert\alpha\rvert})$.	$g(\lvert\alpha\rvert)$	Learned positive degree-dependent factor.
Measure-weighted transformer			
$\boldsymbol{z}_i^{(\ell)}$	Patch token at layer ℓ; $\boldsymbol{z}_i^{(0)}$ is the encoder output.	L_T, d_h	Transformer depth and per-head feature dimension.
$\boldsymbol{q}_i^{(\ell)}, \boldsymbol{k}_i^{(\ell)}, \boldsymbol{v}_i^{(\ell)}$	Query, key, and value vectors.	$s(\boldsymbol{q}, \boldsymbol{k})$	Attention score $\boldsymbol{q}^\top \boldsymbol{k}/\sqrt{d_h}$.
$A_{i,j}^{(\ell)}$	Measure-weighted attention weight from j to i.	$\boldsymbol{o}_i^{(\ell)}$	Attention-head output for token i.
Continuous local expansion decoder			
$\widetilde{\boldsymbol{c}}^{\mathrm{out}}_{i,\alpha}$	Predicted scale-normalized output coefficient.	$\boldsymbol{c}^{\mathrm{out}}_{i,\alpha}$	Denormalized coefficient for local reconstruction.

B Related Work

B.1 Neural PDE Solvers

Many early data-driven PDE surrogates learned finite-dimensional mappings defined on fixed input–output discretizations (Guo et al., 2016; Tompson et al., 2022; Zhu et al., 2019). Their architectures were therefore often tied to the discretizations used during training. Neural operators instead aim to approximate mappings between function spaces, making evaluation on varying discretizations possible in principle (Lu et al., 2021; Kovachki et al., 2023). Among them, transformer-based (Cao, 2021) and Fourier-based (Li et al., 2021) approaches have emerged as two prominent families.

Transformer-based neural operators use self-attention to capture long-range interactions in PDE solution operators (Cao, 2021; Li et al., 2023a; Wu et al., 2024). Because self-attention has quadratic computational complexity in the number of tokens, applying it directly to all physical points is especially costly. Consequently, encoder–transformer–decoder architectures that operate on a smaller set of latent tokens have been extensively studied (Wang & Wang, 2024; Alkin et al., 2025; Wen et al., 2026). The encoder maps the input field to a compact set of latent tokens, the transformer captures interactions among these tokens, and the decoder reconstructs the solution at target coordinates.

Latent representations are also used in Fourier-based neural operators. For example, GINO (Li et al., 2023b) maps observations on an irregular physical discretization to a regular latent grid, processes them with an FNO, and reconstructs the solution at target coordinates. In this setting, the latent mapping not only reduces the number of points processed by the global operator but also converts the irregular physical discretization into a regular grid, enabling efficient FFT-based Fourier operations.

Across both families, predictive accuracy and computational cost therefore depend strongly on how the latent representation is constructed.

B.2 Latent Representations in Neural Operators

Existing latent-token-based neural operators construct discretization-flexible latent representations in two broad ways. Latent tokens may either correspond explicitly to prescribed local regions in the physical domain or be defined in a learned latent space without such a correspondence. These representations generally accommodate varying source discretizations, but their behavior under changes in latent tokenization depends strongly on how the tokens are constructed.

First, when latent tokens retain an explicit correspondence to prescribed local regions in the physical domain, a common approach is to construct the encoder using smooth, scale-independent local GNO (Li et al., 2020) or related message-passing operations, as adopted in several neural operator architectures (Li et al., 2023b; Wen et al., 2026). These methods aggregate source point observations in nearby local regions onto a smaller set of spatially located latent tokens, naturally supporting nonuniform point clouds and complex geometries. However, local spatial details such as slope, curvature, and higher-order spatial variations are encoded only implicitly in the aggregated token features. These features depend on both the input field values and the local region over which they are aggregated. Even when the input field remains unchanged, changing the local region can therefore change the latent token features. Since the effects of the input field and the local region are not explicitly separated, reducing sensitivity to the local region may also reduce sensitivity to local spatial details. This creates a potential trade-off between consistency across different latent configurations and the spatial detail retained in each token.

Second, latent tokens can be defined in a learned latent space without explicit correspondence to particular physical locations or regions. Some methods use learned linear mappings to produce input-dependent assignment weights between physical points and latent tokens (Wu et al., 2024), while others use cross-attention to aggregate physical-point features into a prescribed set of latent queries (Wang & Wang, 2024; Alkin et al., 2025). Such representations can adapt their aggregation patterns to each input without explicitly anchoring latent tokens to the physical domain. However, the number of latent tokens is typically fixed as a model hyperparameter and is not designed to vary at inference time. Moreover, because the tokens are not tied to specific physical regions, there is no natural spatial correspondence between tokens before and after a change in latent tokenization. For example, when the number of latent tokens changes, there is no spatial criterion for determining

which information should be preserved, split, or merged across tokens. This makes it difficult to define a representation that remains consistent across different latent tokenizations.

B.3 Relation to PINOT

Overall, the aforementioned two approaches face different limitations under changes in latent configuration. Physical-domain representations provide a natural way to aggregate nonuniform observations, but maintaining consistency across different aggregation regions can come at the expense of the spatial variation retained in the representation. Learned latent representations avoid explicit dependence on physical regions, but typically use a fixed number of tokens and lack a natural correspondence across different latent tokenizations.

Existing approaches therefore do not readily provide a latent representation whose tokenization can change while consistently preserving the information needed to represent complex spatial structure. This leaves open the challenge of constructing a representation that remains consistent as both the source discretization and latent tokenization change, without sacrificing the expressive capacity required to represent complex fields.

This motivates us to develop PINOT, a neural operator transformer whose latent representation is constructed from scale-normalized local expansion coefficients. Unlike conventional physical-domain aggregation, scale-normalized coefficients retain higher-order local variation under refinement, thereby providing the expressive capacity required to represent complex fields; unlike learned latent tokens, scale-normalized coefficients describe the field within specific physical regions through a common normalized expansion, giving them a consistent interpretation across various patch partitions. PINOT is thus designed to maintain a consistent and expressive latent representation as both the source discretization and patch partition vary.

C Theoretical Analysis

This appendix provides the assumptions and proofs underlying the consistency and local expressivity of PINOT. We analyze the encoder, processor, and decoder to establish convergence to a common limiting output field under admissible joint refinement of source discretizations and patch partitions (Theorem 1). We further characterize the local derivative information retained by the scale-normalized coefficients (Theorem 2) and show that, for $K \geq 1$, their limiting representation is strictly more informative than that of the analyzed class of smooth, scale-independent local GNO aggregations. For clarity, we present the coefficient analysis for a scalar-valued lifted field L; the vector-valued result follows by applying the same argument to each channel.

C.1 Setting and Assumptions

Geometry and partition refinement. Let $\Omega \subset \mathbb{R}^d$ be a bounded measurable domain with compact closure and $0 < |\Omega| < \infty$. Let $\widehat{P} \subset \mathbb{R}^d$ be a bounded centrally symmetric reference domain of positive measure containing the origin:

$$\widehat{P} = -\widehat{P}, \qquad 0 \in \widehat{P}. \tag{9}$$

Each patch in $\mathcal{P} = \{P_i\}_{i=1}^M$ is represented as

$$P_i = \boldsymbol{r}_i + h_i \widehat{P}, \qquad 0 < h_i \leq 1, \qquad \mu_i = |P_i|. \tag{10}$$

For the main argument, the patches form a measurable partition of Ω up to measure-zero boundaries:

$$\left| \Omega \triangle \bigcup_{i=1}^{M} P_i \right| = 0, \qquad \Omega \subseteq \bigcup_{i=1}^{M} \overline{P_i}, \qquad |P_i \cap P_j| = 0 \quad \text{for } i \neq j. \tag{11}$$

Consequently, $\sum_i \mu_i = |\Omega|$. The patches further satisfy

$$\operatorname{diam}(P_i) \leq C_{\widehat{P}} h_i. \tag{12}$$

We denote the maximum patch scale by

$$H_{\mathcal{P}} := \max_i h_i. \tag{13}$$

A partition refinement is admissible if

$$H_{\mathcal{P}} \longrightarrow 0. \tag{14}$$

For every $\boldsymbol{x} \in \Omega$, choose a patch index

$$\iota_{\mathcal{P}}(\boldsymbol{x}) \in \left\{ i : \boldsymbol{x} \in \overline{P_i} \right\}. \tag{15}$$

At a point belonging to more than one patch boundary, the choice is arbitrary but deterministic. This convention makes the patchwise decoder uniquely defined at every $\boldsymbol{x} \in \Omega$.

The displayed affine and centrally symmetric setting is used for the main proof. For a general physical boundary, the same construction extends to a clipped patch $P_i^{\Omega} \subset \Omega$ equipped with a representative point $\boldsymbol{r}_i \in \overline{P_i^{\Omega}}$, a scale h_i, and normalized support

$$\widehat{P}_i^{\Omega} := \frac{P_i^{\Omega} - \boldsymbol{r}_i}{h_i} \subseteq \widehat{P}. \tag{16}$$

In this extension, $\mu_i := |P_i^{\Omega}|$ is the measure of the actual physical patch, and the clipped patches are assumed to form the physical partition, so $\sum_i \mu_i = |\Omega|$. The partition-quadrature and decoder arguments below are unchanged. The consistency proof for the scale-normalized coefficient representation remains valid, but a nonsymmetric limiting sampling measure generally degrades the normalized remainder from $O(h_i^2)$ to $O(h_i)$, as detailed in Remark 4.

Polynomial basis. Let

$$\Lambda_K := \left\{ \alpha \in \mathbb{N}_0^d : |\alpha| \le K \right\}, \qquad \Lambda_m^{\circ} := \left\{ \alpha \in \mathbb{N}_0^d : |\alpha| = m \right\}. \tag{17}$$

We assume that the degree-ordered polynomial basis

$$\{\psi_\alpha\}_{\alpha \in \Lambda_K} \tag{18}$$

satisfies

$$\operatorname{span}\{\psi_\alpha : |\alpha| \le m\} = \mathbb{P}_m, \qquad m = 0, \dots, K, \tag{19}$$

where $\mathbb{P}_m$ denotes the polynomials of total degree at most m. We further assume the parity condition

$$\psi_\alpha(-\boldsymbol{\xi}) = (-1)^{|\alpha|} \psi_\alpha(\boldsymbol{\xi}). \tag{20}$$

These conditions hold, for example, for tensor-product Legendre and Chebyshev polynomial bases on $[-1, 1]^d$, restricted to the total-degree index set Λ_K.

Although the basis may be chosen orthogonal under a reference measure $d\rho$, the analysis below does not require the source point sampling measure to coincide with $d\rho$. This is important because the encoder employs ordinary, rather than weighted, least squares.

Lifted field and regularity. The pointwise lifting operation defines

$$L(\boldsymbol{x}) := \mathrm{Lift}_{\boldsymbol{\phi}}(\boldsymbol{x}, v(\boldsymbol{x})) \in \mathbb{R}^{d_{\mathrm{lift}}}. \tag{21}$$

We assume that L belongs to C^{K+2} on an open neighborhood of $\overline{\Omega}$. This is a conditional assumption on both the input field and the lifting network. In particular, it requires sufficient regularity of v and a sufficiently smooth lifting map on the compact set encountered in the analysis; a ReLU lifting network does not, in general, satisfy this global C^{K+2} assumption. All network parameters, including the learned coefficient-balancing function g, are fixed throughout refinement. The learned pointwise maps are assumed to be continuous and Lipschitz on the bounded feature sets encountered during the forward pass. Layer normalization, when used, includes a fixed positive numerical constant, and no token-index-dependent positional embedding is introduced. Coordinate-dependent positional encodings that vary continuously with the patch center, including the patch-center encoding used in our implementation, are allowed; the restriction applies only to embeddings tied to discrete token indices.

Ordinary least-squares sampling. For notational brevity, let $\mathbb{X} := \mathbb{X}_{\mathrm{src}}$. For each patch, define

$$\mathbb{X}^i_{\mathrm{src}} := P_i \cap \mathbb{X}_{\mathrm{src}} = \{\boldsymbol{x}_{i,j}\}_{j=1}^{n_i}, \qquad \boldsymbol{\xi}_{i,j} := N_i(\boldsymbol{x}_{i,j}) = \frac{\boldsymbol{x}_{i,j} - \boldsymbol{r}_i}{h_i}. \tag{22}$$

The least-squares fitting described in the main text is explicitly

$$\left\{\boldsymbol{c}_{i,\alpha}^{\mathrm{in},\mathbb{X}}\right\}_{\alpha\in\Lambda_K} := \underset{\{\boldsymbol{c}_\alpha\in\mathbb{R}^{d_{\mathrm{lift}}}\}_{\alpha\in\Lambda_K}}{\arg\min} \frac{1}{n_i}\sum_{j=1}^{n_i}\left\| L(\boldsymbol{x}_{i,j}) - \sum_{\alpha\in\Lambda_K}\boldsymbol{c}_\alpha\psi_\alpha(\boldsymbol{\xi}_{i,j})\right\|_2^2. \tag{23}$$

Thus, the appendix superscript $\mathbb{X}$ records the source discretization used to obtain the input coefficients. For a scalar lifted channel, we collect the fitted coefficients into

$$\boldsymbol{c}_i^{\mathrm{in},\mathbb{X}} := \left[c_{i,\alpha}^{\mathrm{in},\mathbb{X}}\right]_{\alpha\in\Lambda_K}. \tag{24}$$

The unweighted empirical measure induced by the source points is

$$\sigma_{i,\mathbb{X}} := \frac{1}{n_i}\sum_{j=1}^{n_i}\delta_{\boldsymbol{\xi}_{i,j}}. \tag{25}$$

Equation (23) is therefore the empirical $L^2(\sigma_{i,\mathbb{X}})$ projection used by the encoder; no quadrature or density-correction weights are introduced.

For the analysis, associate each patch with a centrally symmetric probability measure σ_i on $\widehat{P}$:

$$\sigma_i(E) = \sigma_i(-E) \qquad \text{for every measurable } E\subset\widehat{P}. \tag{26}$$

The measure σ_i may depend on the patch and need not be an orthogonality measure of the polynomial basis.

Let

$$\boldsymbol{\psi}(\boldsymbol{\xi}) := [\psi_\alpha(\boldsymbol{\xi})]_{\alpha\in\Lambda_K}. \tag{27}$$

For the pulled-back field

$$L_i(\boldsymbol{\xi}) := L(\boldsymbol{r}_i + h_i\boldsymbol{\xi}), \tag{28}$$

define

$$\boldsymbol{G}_i := \int_{\widehat{P}}\boldsymbol{\psi}(\boldsymbol{\xi})\boldsymbol{\psi}(\boldsymbol{\xi})^\top\, d\sigma_i(\boldsymbol{\xi}). \tag{29}$$

$$\boldsymbol{G}_{i,\mathbb{X}} := \int_{\widehat{P}}\boldsymbol{\psi}(\boldsymbol{\xi})\boldsymbol{\psi}(\boldsymbol{\xi})^\top\, d\sigma_{i,\mathbb{X}}(\boldsymbol{\xi}). \tag{30}$$

$$\boldsymbol{b}_i := \int_{\widehat{P}}\boldsymbol{\psi}(\boldsymbol{\xi})L_i(\boldsymbol{\xi})\, d\sigma_i(\boldsymbol{\xi}). \tag{31}$$

$$\boldsymbol{b}_{i,\mathbb{X}} := \int_{\widehat{P}}\boldsymbol{\psi}(\boldsymbol{\xi})L_i(\boldsymbol{\xi})\, d\sigma_{i,\mathbb{X}}(\boldsymbol{\xi}). \tag{32}$$

We assume uniform least-squares stability:

$$\lambda_{\min}(\boldsymbol{G}_i) \geq \gamma, \qquad \lambda_{\min}(\boldsymbol{G}_{i,\mathbb{X}}) \geq \gamma > 0. \tag{33}$$

This condition implies that the empirical design matrix has full column rank, so the ordinary least-squares minimizer is unique.

Define the empirical moment error

$$\eta_{i,\mathbb{X}} := \|\boldsymbol{G}_{i,\mathbb{X}} - \boldsymbol{G}_i\| + \|\boldsymbol{b}_{i,\mathbb{X}} - \boldsymbol{b}_i\|, \tag{34}$$

and its scale-normalized form

$$\varepsilon_{i,\mathbb{X}} := h_i^{-K}\eta_{i,\mathbb{X}}. \tag{35}$$

An admissible joint source-discretization and partition refinement satisfies

$$H_{\mathcal{P}} \longrightarrow 0, \qquad \max_i \varepsilon_{i,\mathbb{X}} \longrightarrow 0. \tag{36}$$

For example, suppose that the source points within each patch are sampled i.i.d., so that the empirical moment error satisfies $\eta_{i,\mathbb{X}} = O_p(n_i^{-1/2})$, where n_i is the number of source points in P_i. Then the condition in Eq. (36) suggests the heuristic sampling regime

$$\max_i h_i^{-K} n_i^{-1/2} \longrightarrow 0, \tag{37}$$

or equivalently, $n_i \gg h_i^{-2K}$ uniformly over the patches. Thus, as the patches shrink, the number of source points per patch should increase sufficiently rapidly. This heuristic sampling regime becomes more demanding at higher polynomial degrees.

C.2 Scale-Normalized Local Coefficient Representation

We first identify the relation between the fitted polynomial coefficients and the local derivatives of the lifted field.

For each degree m, define the vector of degree-m basis functions and the vector of normalized monomials as

$$\boldsymbol{\psi}_m(\boldsymbol{\xi}) := [\psi_\alpha(\boldsymbol{\xi})]_{\alpha\in\Lambda_m^\circ}, \qquad \boldsymbol{m}_m(\boldsymbol{\xi}) := \left[\frac{\boldsymbol{\xi}^\beta}{\beta!}\right]_{\beta\in\Lambda_m^\circ}, \tag{38}$$

where $\Lambda_m^\circ := \{\alpha : |\alpha| = m\}$ denotes the set of multi-indices of total degree exactly m. The normalization by $\beta!$ matches the standard multivariate Taylor expansion, so that the coefficients in the normalized monomial basis are directly associated with the corresponding derivatives.

Because the polynomial basis is degree ordered, there exists an invertible matrix $\boldsymbol{T}_m$ such that

$$\boldsymbol{\psi}_m(\boldsymbol{\xi}) = \boldsymbol{T}_m\boldsymbol{m}_m(\boldsymbol{\xi}) + \boldsymbol{p}_{m-1}(\boldsymbol{\xi}), \tag{39}$$

where every component of $\boldsymbol{p}_{m-1}$ has degree at most $m-1$. The matrix $\boldsymbol{T}_m$ is invertible because the degree-m leading terms of the basis functions span the space of homogeneous polynomials of degree m. Moreover, under the parity condition $\psi_\alpha(-\boldsymbol{\xi}) = (-1)^{|\alpha|}\psi_\alpha(\boldsymbol{\xi})$, the lower-order term $\boldsymbol{p}_{m-1}$ contains only degrees having the same parity as m, namely $m-2, m-4, \ldots$.

For a scalar field L, define

$$\boldsymbol{J}_m L(\boldsymbol{r}) := \left[D^\beta L(\boldsymbol{r})\right]_{\beta\in\Lambda_m^\circ}, \qquad \boldsymbol{J}^K L(\boldsymbol{r}) := \left[(\boldsymbol{J}_0 L(\boldsymbol{r}))^\top, \ldots, (\boldsymbol{J}_K L(\boldsymbol{r}))^\top\right]^\top. \tag{40}$$

Define

$$\boldsymbol{B}_m := \boldsymbol{T}_m^{-\top}, \qquad \boldsymbol{B}_{\le K} := \mathrm{blkdiag}\left(\boldsymbol{B}_0, \ldots, \boldsymbol{B}_K\right). \tag{41}$$

The role of $\boldsymbol{B}_m$ follows from the degree-m Taylor expansion. For a patch centered at $\boldsymbol{r}$ with scale h, write $\boldsymbol{x} = \boldsymbol{r} + h\boldsymbol{\xi}$. The degree-$m$ Taylor term is

$$h^m \left(\boldsymbol{J}_m L(\boldsymbol{r})\right)^\top \boldsymbol{m}_m(\boldsymbol{\xi}). \tag{42}$$

If the same degree-m component is represented in the basis $\boldsymbol{\psi}_m$ with coefficient vector $\boldsymbol{c}_m$, its degree-m homogeneous term is

$$\boldsymbol{c}_m^\top \boldsymbol{T}_m \boldsymbol{m}_m(\boldsymbol{\xi}) = \left(\boldsymbol{T}_m^\top \boldsymbol{c}_m\right)^\top \boldsymbol{m}_m(\boldsymbol{\xi}). \tag{43}$$

The lower-order term $\boldsymbol{p}_{m-1}$ does not contribute to this degree-m component. Matching the coefficients of $\boldsymbol{m}_m(\boldsymbol{\xi})$ therefore gives

$$\boldsymbol{T}_m^\top \boldsymbol{c}_m = h^m \boldsymbol{J}_m L(\boldsymbol{r}), \qquad \boldsymbol{c}_m = h^m \boldsymbol{B}_m \boldsymbol{J}_m L(\boldsymbol{r}). \tag{44}$$

Thus, $\boldsymbol{B}_m = \boldsymbol{T}_m^{-\top}$ is the fixed change-of-coordinates matrix from the degree-m derivatives to the corresponding degree-m polynomial-basis coefficients. In particular, Eq. (44) shows that these coefficients carry the deterministic scale factor h^m. Since each $\boldsymbol{T}_m$ is invertible, both $\boldsymbol{B}_m$ and $\boldsymbol{B}_{\le K}$ are invertible and depend only on the selected polynomial basis.

To express the encoder normalization compactly, define

$$\begin{aligned} \boldsymbol{D}_{h_i} &:= \mathrm{diag}\left(h_i^{|\alpha|}\right)_{\alpha\in\Lambda_K}, \qquad \boldsymbol{D}_g := \mathrm{diag}\left(g(|\alpha|)\right)_{\alpha\in\Lambda_K}, \\ \boldsymbol{S}_i &:= \mathrm{diag}\left(s_{i,\alpha}\right)_{\alpha\in\Lambda_K} = \boldsymbol{D}_g^{-1}\boldsymbol{D}_{h_i}^{-1}. \end{aligned} \tag{45}$$

Here, $\boldsymbol{D}_{h_i}$ collects the deterministic scale factors associated with the polynomial degrees, while $\boldsymbol{D}_g$ collects the learned coefficient-balancing factors. Accordingly, $\boldsymbol{D}_{h_i}^{-1}$ removes the factor $h_i^{|\alpha|}$ carried by each coefficient, and $\boldsymbol{D}_g^{-1}$ applies the learned rescaling used by the encoder.

We assume $g(|\alpha|) > 0$ for every $\alpha \in \Lambda_K$, as required to invert the encoder normalization in the decoder. Since Λ_K is finite and g is fixed after training, both $\boldsymbol{D}_g$ and $\boldsymbol{S}_i$ are invertible. Combining the basis transformation with the learned normalization, define

$$\boldsymbol{A}_{\boldsymbol{\theta},K} := \boldsymbol{D}_g^{-1}\boldsymbol{B}_{\le K}. \tag{46}$$

The matrix $\boldsymbol{A}_{\boldsymbol{\theta},K}$ is invertible and depends only on the selected polynomial basis and the learned normalization g, but not on the patch center $\boldsymbol{r}_i$, patch scale h_i, or source discretization.

Lemma 3 (Continuous scale-normalized coefficient limit)**.** *Let $\boldsymbol{c}_i^{\mathrm{in}}$ denote the coefficient vector of the continuous least-squares projection*

$$\boldsymbol{c}_i^{\mathrm{in}} := \arg\min_{\boldsymbol{c}} \int_{\widehat{P}} \left| L_i(\boldsymbol{\xi}) - \boldsymbol{c}^\top \boldsymbol{\psi}(\boldsymbol{\xi}) \right|^2 d\sigma_i(\boldsymbol{\xi}). \tag{47}$$

Then

$$\boldsymbol{D}_{h_i}^{-1} \boldsymbol{c}_i^{\mathrm{in}} = \boldsymbol{B}_{\le K} \boldsymbol{J}^K L(\boldsymbol{r}_i) + \boldsymbol{\delta}_i, \qquad \|\boldsymbol{\delta}_i\| \le C h_i^2. \tag{48}$$

The constant C is uniform over admissible patches and for lifted fields L in bounded subsets of C^{K+2}.

Proof. Let $\mathcal{Q}_{i,K}$ denote the $L^2(\sigma_i)$ projection onto $\mathbb{P}_K$. A Taylor expansion on the reference domain gives

$$L_i(\boldsymbol{\xi}) = \sum_{m=0}^{K+1} h_i^m \left(\boldsymbol{J}_m L(\boldsymbol{r}_i) \right)^\top \boldsymbol{m}_m(\boldsymbol{\xi}) + R_i(\boldsymbol{\xi}), \tag{49}$$

where

$$\|R_i\|_{L^\infty(\widehat{P})} \le C h_i^{K+2}. \tag{50}$$

The projection $\mathcal{Q}_{i,K}$ exactly reproduces every polynomial in $\mathbb{P}_K$. Moreover, central symmetry of σ_i implies that $\mathcal{Q}_{i,K}$ preserves parity. Consequently, the degree-m coefficient block receives no contribution from Taylor terms of degree less than m, and the degree-$(m+1)$ term has the opposite parity and also contributes nothing to that block. All remaining Taylor terms that can contribute to the degree-m block have degree at least $m+2$, and hence have raw magnitude $O(h_i^{m+2})$. The projected remainder contributes $O(h_i^{K+2})$, which is also $O(h_i^{m+2})$ because $m \le K$ and $0 < h_i \le 1$.

Using Eq. (44), the degree-m block therefore satisfies

$$\boldsymbol{c}_{i,m}^{\mathrm{in}} = h_i^m \boldsymbol{B}_m \boldsymbol{J}_m L(\boldsymbol{r}_i) + \boldsymbol{\delta}_{i,m}, \qquad \|\boldsymbol{\delta}_{i,m}\| \le C h_i^{m+2}. \tag{51}$$

Dividing each block by h_i^m and stacking the degrees proves Eq. (48).

Uniformity follows from the Gram-matrix lower bound in Eq. (33), which uniformly bounds the continuous projection operator.

□

Remark 4 (Nonsymmetric and boundary patches). Central symmetry is used only to preserve parity when the unresolved Taylor terms are projected onto $\mathbb{P}_K$. Without this symmetry, the degree-$(K+1)$ term may project onto the degree-K coefficient block with raw magnitude $O(h_i^{K+1})$. After degree normalization, this produces the first-order bound

$$\boldsymbol{D}_{h_i}^{-1} \boldsymbol{c}_i^{\mathrm{in}} = \boldsymbol{B}_{\le K} \boldsymbol{J}^K L(\boldsymbol{r}_i) + O(h_i). \tag{52}$$

The lower-degree blocks are no worse, and the same conclusion applies to a clipped patch by integrating over its normalized support $\widehat{P}_i^\Omega$, provided the corresponding Gram matrices remain uniformly stable. Thus, central symmetry improves the rate from first to second order but is not required for consistency itself.

Proposition 5 (Consistency of ordinary local least squares)**.** *Let $\boldsymbol{c}_i^{\mathrm{in},\mathbb{X}}$ be the scalar coefficient vector obtained from Eq. (23). Define the normalized coefficient vector by the same scaling used in the main method:*

$$\widetilde{\boldsymbol{c}}_i^{\mathrm{in},\mathbb{X}} := \boldsymbol{S}_i \boldsymbol{c}_i^{\mathrm{in},\mathbb{X}}. \tag{53}$$

Then

$$\widetilde{\boldsymbol{c}}_i^{\mathrm{in},\mathbb{X}} = \boldsymbol{A}_{\boldsymbol{\theta},K} \boldsymbol{J}^K L(\boldsymbol{r}_i) + \boldsymbol{R}_{i,\mathbb{X}}, \qquad \|\boldsymbol{R}_{i,\mathbb{X}}\| \le C \left(h_i^2 + \varepsilon_{i,\mathbb{X}} \right). \tag{54}$$

Proof. The continuous and empirical normal equations are

$$\boldsymbol{G}_i \boldsymbol{c}_i^{\mathrm{in}} = \boldsymbol{b}_i, \qquad \boldsymbol{G}_{i,\mathbb{X}} \boldsymbol{c}_i^{\mathrm{in},\mathbb{X}} = \boldsymbol{b}_{i,\mathbb{X}}. \tag{55}$$

Hence

$$\begin{aligned} \boldsymbol{c}_i^{\mathrm{in},\mathbb{X}} - \boldsymbol{c}_i^{\mathrm{in}} = {} & \boldsymbol{G}_{i,\mathbb{X}}^{-1} \left(\boldsymbol{b}_{i,\mathbb{X}} - \boldsymbol{b}_i \right) \\ & + \boldsymbol{G}_{i,\mathbb{X}}^{-1} \left(\boldsymbol{G}_i - \boldsymbol{G}_{i,\mathbb{X}} \right) \boldsymbol{c}_i^{\mathrm{in}}. \end{aligned} \tag{56}$$

Uniform stability and boundedness of $\boldsymbol{c}_i^{\mathrm{in}}$ yield

$$\left\|\boldsymbol{c}_i^{\mathrm{in},\mathbb{X}} - \boldsymbol{c}_i^{\mathrm{in}}\right\| \le C\eta_{i,\mathbb{X}}. \tag{57}$$

Since $0 < h_i \le 1$,

$$\left\|\boldsymbol{D}_{h_i}^{-1}\right\| = h_i^{-K}. \tag{58}$$

Therefore,

$$\left\|\boldsymbol{S}_i\left(\boldsymbol{c}_i^{\mathrm{in},\mathbb{X}} - \boldsymbol{c}_i^{\mathrm{in}}\right)\right\| \le Ch_i^{-K}\eta_{i,\mathbb{X}} = C\varepsilon_{i,\mathbb{X}}. \tag{59}$$

Combining this estimate with Lemma 3 and $\boldsymbol{A}_{\boldsymbol{\theta},K} = \boldsymbol{D}_g^{-1}\boldsymbol{B}_{\le K}$ proves the result.

□

For the vector-valued lifted field $L : \Omega \to \mathbb{R}^{d_{\mathrm{lift}}}$, define

$$\begin{aligned}\boldsymbol{C}_i^{\mathrm{in},\mathbb{X}} &:= \left[\boldsymbol{c}_{i,\alpha}^{\mathrm{in},\mathbb{X}}\right]_{\alpha\in\Lambda_K} \in \mathbb{R}^{d_{\mathrm{lift}}\times|\Lambda_K|},\\ \boldsymbol{J}^K L(\boldsymbol{r}_i) &:= \left[D^\alpha L(\boldsymbol{r}_i)\right]_{\alpha\in\Lambda_K} \in \mathbb{R}^{d_{\mathrm{lift}}\times|\Lambda_K|}.\end{aligned} \tag{60}$$

Applying Proposition 5 componentwise gives

$$\widetilde{\boldsymbol{C}}_i^{\mathrm{in},\mathbb{X}} := \boldsymbol{C}_i^{\mathrm{in},\mathbb{X}}\boldsymbol{S}_i = \boldsymbol{J}^K L(\boldsymbol{r}_i)\boldsymbol{A}_{\boldsymbol{\theta},K}^\top + \boldsymbol{R}_{i,\mathbb{X}}, \tag{61}$$

with

$$\left\|\boldsymbol{R}_{i,\mathbb{X}}\right\|_{\mathrm{F}} \le C\left(h_i^2 + \varepsilon_{i,\mathbb{X}}\right). \tag{62}$$

In particular, two patches centered at the same point $\boldsymbol{r}$, with scales h and $\bar{h}$, satisfy

$$\left\|\widetilde{\boldsymbol{C}}_{\boldsymbol{r},h}^{\mathrm{in},\mathbb{X}} - \widetilde{\boldsymbol{C}}_{\boldsymbol{r},\bar{h}}^{\mathrm{in},\bar{\mathbb{X}}}\right\|_{\mathrm{F}} \le C\left(h^2 + \bar{h}^2 + \varepsilon_{\boldsymbol{r},h,\mathbb{X}} + \varepsilon_{\boldsymbol{r},\bar{h},\bar{\mathbb{X}}}\right). \tag{63}$$

Thus, the scale-normalized coefficient representation approaches the same transformed K-jet independently of the patch scale and the particular admissible source discretization.

For a nonsymmetric or clipped patch, the same perturbation argument combined with Remark 4 replaces every h_i^2 term in this subsection by h_i. In particular, the vector-valued remainder becomes

$$\left\|\boldsymbol{R}_{i,\mathbb{X}}\right\|_{\mathrm{F}} \le C\left(h_i + \varepsilon_{i,\mathbb{X}}\right). \tag{64}$$

C.3 Continuum Token Field

Let $\mathscr{E}_{\boldsymbol{\theta}_{\mathrm{E}}}$ denote the coefficient-to-token mapping implemented by the final embedding stage of the *scale-normalized coefficient encoder*. The finite-patch token is

$$\boldsymbol{z}_i^{(0)} := \mathscr{E}_{\boldsymbol{\theta}_{\mathrm{E}}}\left(\widetilde{\boldsymbol{C}}_i^{\mathrm{in},\mathbb{X}}\right). \tag{65}$$

Define the limiting token field by

$$\boldsymbol{z}_*^{(0)}(\boldsymbol{r}) := \mathscr{E}_{\boldsymbol{\theta}_{\mathrm{E}}}\left(\boldsymbol{J}^K L(\boldsymbol{r})\boldsymbol{A}_{\boldsymbol{\theta},K}^\top\right). \tag{66}$$

The Lipschitz continuity of $\mathscr{E}_{\boldsymbol{\theta}_{\mathrm{E}}}$ and Eq. (61) imply

$$\left\|\boldsymbol{z}_i^{(0)} - \boldsymbol{z}_*^{(0)}(\boldsymbol{r}_i)\right\| \le C\left(h_i^2 + \varepsilon_{i,\mathbb{X}}\right). \tag{67}$$

Therefore, the encoder tokens consistently sample the same continuum token field under admissible refinement.

For a nonsymmetric or clipped boundary patch, Eq. (67) remains valid with h_i^2 replaced by h_i.

C.4 Partition Consistency of Measure-Weighted Attention

We first record the quadrature property of the patch centers and measures.

Lemma 6 (Partition quadrature)**.** *For every continuous function $\varphi : \overline{\Omega} \to \mathbb{R}^p$,*

$$\sum_{i=1}^{M} \mu_i \varphi(\boldsymbol{r}_i) \longrightarrow \int_{\Omega} \varphi(\boldsymbol{x})\, d\boldsymbol{x} \qquad \text{as } H_{\mathcal{P}} \to 0. \tag{68}$$

Proof. Let ω_φ be the modulus of continuity of φ. Since $\|\boldsymbol{x} - \boldsymbol{r}_i\| \le C_{\widehat{P}} H_{\mathcal{P}}$ for $\boldsymbol{x} \in P_i$,

$$\left\| \sum_i \mu_i \varphi(\boldsymbol{r}_i) - \int_{\Omega} \varphi(\boldsymbol{x})\, d\boldsymbol{x} \right\| \le |\Omega|\, \omega_\varphi \left(C_{\widehat{P}} H_{\mathcal{P}} \right), \tag{69}$$

which converges to zero.

□

Consider one attention head acting on a continuous feature field

$$\boldsymbol{z} : \overline{\Omega} \to \mathbb{R}^{d_{\mathrm{emb}}}. \tag{70}$$

Define

$$\boldsymbol{q}(\boldsymbol{x}) = \boldsymbol{W}_Q \boldsymbol{z}(\boldsymbol{x}), \qquad \boldsymbol{k}(\boldsymbol{x}) = \boldsymbol{W}_K \boldsymbol{z}(\boldsymbol{x}), \qquad \boldsymbol{v}(\boldsymbol{x}) = \boldsymbol{W}_V \boldsymbol{z}(\boldsymbol{x}), \tag{71}$$

and

$$s_{\boldsymbol{z}}(\boldsymbol{x}, \boldsymbol{y}) := \frac{\boldsymbol{q}(\boldsymbol{x})^\top \boldsymbol{k}(\boldsymbol{y})}{\sqrt{d_h}}. \tag{72}$$

Here, d_h denotes the feature dimension of each attention head.

The corresponding continuous attention operator is

$$\mathcal{A}[\boldsymbol{z}](\boldsymbol{x}) := \frac{\displaystyle\int_{\Omega} \exp(s_{\boldsymbol{z}}(\boldsymbol{x}, \boldsymbol{y}))\, \boldsymbol{v}(\boldsymbol{y})\, d\boldsymbol{y}}{\displaystyle\int_{\Omega} \exp(s_{\boldsymbol{z}}(\boldsymbol{x}, \boldsymbol{y}))\; d\boldsymbol{y}}. \tag{73}$$

The discrete weights in Eq. (6) are equivalently

$$A_{i,j} = \frac{\mu_j \exp\!\left(\boldsymbol{q}_i^\top \boldsymbol{k}_j / \sqrt{d_h}\right)}{\displaystyle\sum_{p=1}^{M} \mu_p \exp\!\left(\boldsymbol{q}_i^\top \boldsymbol{k}_p / \sqrt{d_h}\right)}. \tag{74}$$

Proposition 7 (Consistency of measure-weighted attention)**.** *Suppose that $\boldsymbol{z}$ is continuous and bounded and that*

$$\max_i \|\boldsymbol{z}_i - \boldsymbol{z}(\boldsymbol{r}_i)\| \longrightarrow 0. \tag{75}$$

Then

$$\max_i \left\| \sum_{j=1}^{M} A_{i,j} \boldsymbol{v}_j - \mathcal{A}[\boldsymbol{z}](\boldsymbol{r}_i) \right\| \longrightarrow 0 \tag{76}$$

under admissible partition refinement.

Proof. Continuity on the compact domain implies that the exact attention scores are uniformly bounded. Moreover, Eq. (75) implies that the discrete features are uniformly bounded for all sufficiently refined partitions. Hence there exists a constant $C_s > 0$ such that

$$|s_{\boldsymbol{z}}(\boldsymbol{x}, \boldsymbol{y})| \le C_s, \qquad \left| \frac{\boldsymbol{q}_i^\top \boldsymbol{k}_j}{\sqrt{d_h}} \right| \le C_s. \tag{77}$$

Consequently, both the continuous and actual discrete denominators are uniformly bounded away from zero:

$$\int_{\Omega} \exp(s_{\boldsymbol{z}}(\boldsymbol{x}, \boldsymbol{y}))\, d\boldsymbol{y} \geq e^{-C_s}|\Omega|. \tag{78}$$

$$\sum_j \mu_j \exp\left(\frac{\boldsymbol{q}_i^\top \boldsymbol{k}_j}{\sqrt{d_h}}\right) \geq e^{-C_s} \sum_j \mu_j = e^{-C_s}|\Omega|. \tag{79}$$

For exact samples $\boldsymbol{z}(\boldsymbol{r}_i)$, the numerator and denominator of Eq. (74) converge uniformly to those of Eq. (73) by Lemma 6. Uniformity follows from the joint continuity of the corresponding integrands on $\overline{\Omega} \times \overline{\Omega}$.

Replacing $\boldsymbol{z}(\boldsymbol{r}_i)$ by $\boldsymbol{z}_i$ gives a vanishing perturbation because the linear projections, score function, exponential, and normalized ratio are Lipschitz on the bounded feature set. Combining the quadrature and feature errors proves the result.

□

Applying Proposition 7 to each head and each layer gives the following result.

Corollary 8 (Consistency of the transformer processor)**.** *For any fixed number L_{T} of transformer layers, there exists a continuous feature field*

$$\boldsymbol{z}_*^{(L_{\mathrm{T}})} : \overline{\Omega} \to \mathbb{R}^{d_{\mathrm{emb}}} \tag{80}$$

such that

$$\max_i \left\| \boldsymbol{z}_i^{(L_{\mathrm{T}})} - \boldsymbol{z}_*^{(L_{\mathrm{T}})}(\boldsymbol{r}_i) \right\| \longrightarrow 0. \tag{81}$$

The limiting field is independent of the particular admissible partition and source-discretization sequences.

Proof. Equation (67) establishes the claim at layer zero. Proposition 7 propagates convergence through each attention operation. Residual connections, normalization, and feed-forward networks preserve convergence by Lipschitz continuity. Induction over the finite number of transformer layers proves the claim.

□

C.5 Continuous Decoding and Target Discretization

For each processed token, the coefficient MLP produces the normalized output coefficients described in Section 2.2.3:

$$\widetilde{\boldsymbol{C}}_i^{\mathrm{out}} := \mathrm{reshape}_{\Lambda_K}\left(\mathrm{MLP}_{\mathrm{coeff}}\left(\boldsymbol{z}_i^{(L_{\mathrm{T}})}\right)\right) = \left[\widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}}\right]_{\alpha \in \Lambda_K}. \tag{82}$$

The decoder exactly inverts the normalization in Eq. (5):

$$\boldsymbol{C}_i^{\mathrm{out}} := \widetilde{\boldsymbol{C}}_i^{\mathrm{out}} \boldsymbol{S}_i^{-1}, \qquad \boldsymbol{c}_{i,\alpha}^{\mathrm{out}} = s_{i,\alpha}^{-1} \widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}} = g(|\alpha|) h_i^{|\alpha|} \widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}}. \tag{83}$$

Define the limiting normalized output-coefficient field by

$$\widetilde{\boldsymbol{C}}_*^{\mathrm{out}}(\boldsymbol{r}) := \mathrm{reshape}_{\Lambda_K}\left(\mathrm{MLP}_{\mathrm{coeff}}\left(\boldsymbol{z}_*^{(L_{\mathrm{T}})}(\boldsymbol{r})\right)\right). \tag{84}$$

Corollary 8 and continuity of the coefficient MLP imply

$$\delta_{\mathcal{P},\mathbb{X}}^{\mathrm{out}} := \max_i \left\| \widetilde{\boldsymbol{C}}_i^{\mathrm{out}} - \widetilde{\boldsymbol{C}}_*^{\mathrm{out}}(\boldsymbol{r}_i) \right\|_{\mathrm{F}} \longrightarrow 0. \tag{85}$$

For every patch P_i, define the reconstructed local feature by the same expansion used in the main decoder:

$$\begin{aligned} \widehat{w}_i^{\mathrm{loc}}(\boldsymbol{x}) &:= \sum_{\alpha \in \Lambda_K} \boldsymbol{c}_{i,\alpha}^{\mathrm{out}} \psi_\alpha\left(N_i(\boldsymbol{x})\right) \\ &= \sum_{\alpha \in \Lambda_K} g(|\alpha|) h_i^{|\alpha|} \widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}} \psi_\alpha\left(N_i(\boldsymbol{x})\right), \qquad \boldsymbol{x} \in \overline{P_i}. \end{aligned} \tag{86}$$

The pointwise output projection gives

$$\widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}) := \mathrm{Proj}_{\boldsymbol{\varphi}}\left(\widehat{w}_i^{\mathrm{loc}}(\boldsymbol{x})\right). \tag{87}$$

The deterministic boundary assignment introduced in Section C.1 then defines the pre-discretization model output field

$$\widehat{u}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)(\boldsymbol{x}) := \widehat{u}^{\mathrm{loc}}_{\iota_{\mathcal{P}}(\boldsymbol{x})}(\boldsymbol{x}), \qquad \boldsymbol{x}\in\Omega. \tag{88}$$

The discrete model output is exactly the restriction of this field to the requested target coordinates:

$$\mathcal{G}_{\boldsymbol{\theta}}\left(\mathbb{X}_{\mathrm{src}}, v|_{\mathbb{X}_{\mathrm{src}}}, \mathbb{X}_{\mathrm{tar}}; \mathcal{P}\right) = \widehat{u}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)|_{\mathbb{X}_{\mathrm{tar}}}. \tag{89}$$

Thus, the prediction at a fixed coordinate is independent of what other target points are queried.

Proposition 9 (Continuum limit of the decoder)**.** *Under admissible refinement, there exists a continuous limiting output field*

$$\widehat{u}_{\boldsymbol{\theta}}(v) : \overline{\Omega} \to \mathbb{R}^{c_{\mathrm{out}}} \tag{90}$$

such that the local predictions satisfy the stronger uniform convergence

$$\max_i \sup_{\boldsymbol{x}\in\overline{P_i}\cap\overline{\Omega}} \left\|\widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x})\right\| \longrightarrow 0. \tag{91}$$

Consequently,

$$\sup_{\boldsymbol{x}\in\Omega} \|\widehat{u}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x})\| \longrightarrow 0. \tag{92}$$

The limiting output is independent of the particular admissible refinement sequence.

Proof. Let $\mathbf{0}$ denote the zero multi-index and let ω_* be a modulus of continuity of $\widetilde{\boldsymbol{C}}_*^{\mathrm{out}}$ on $\overline{\Omega}$. For $\boldsymbol{x}\in\overline{P_i}\cap\overline{\Omega}$,

$$\|\boldsymbol{r}_i - \boldsymbol{x}\| \le C_{\widehat{P}} H_{\mathcal{P}}. \tag{93}$$

Because $\psi_{\mathbf{0}}$ is constant, the degree-zero feature satisfies

$$\begin{aligned}&\left\|g(0)\widetilde{\boldsymbol{c}}^{\mathrm{out}}_{i,\mathbf{0}}\psi_{\mathbf{0}} - g(0)\widetilde{\boldsymbol{c}}^{\mathrm{out}}_{*,\mathbf{0}}(\boldsymbol{x})\psi_{\mathbf{0}}\right\| \\ &\qquad \le C\left(\delta^{\mathrm{out}}_{\mathcal{P},\mathbb{X}} + \omega_*\left(C_{\widehat{P}}H_{\mathcal{P}}\right)\right).\end{aligned} \tag{94}$$

For every $|\alpha|\ge 1$, boundedness of the normalized coefficients and of ψ_α on $\widehat{P}$ gives

$$\left\|g(|\alpha|)h_i^{|\alpha|}\widetilde{\boldsymbol{c}}^{\mathrm{out}}_{i,\alpha}\psi_\alpha\left(N_i(\boldsymbol{x})\right)\right\| \le C H_{\mathcal{P}}^{|\alpha|}. \tag{95}$$

Since Λ_K is finite and $H_{\mathcal{P}}\le 1$, the sum of all positive-degree terms is bounded by $CH_{\mathcal{P}}$. Therefore, with

$$\widehat{w}_{\boldsymbol{\theta}}(v)(\boldsymbol{x}) := g(0)\widetilde{\boldsymbol{c}}^{\mathrm{out}}_{*,\mathbf{0}}(\boldsymbol{x})\psi_{\mathbf{0}}, \tag{96}$$

we obtain

$$\begin{aligned}&\max_i \sup_{\boldsymbol{x}\in\overline{P_i}\cap\overline{\Omega}} \left\|\widehat{w}_i^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{w}_{\boldsymbol{\theta}}(v)(\boldsymbol{x})\right\| \\ &\qquad \le C\left(\delta^{\mathrm{out}}_{\mathcal{P},\mathbb{X}} + \omega_*\left(C_{\widehat{P}}H_{\mathcal{P}}\right) + H_{\mathcal{P}}\right) \longrightarrow 0.\end{aligned} \tag{97}$$

The Lipschitz continuity of $\mathrm{Proj}_{\boldsymbol{\varphi}}$ proves Eq. (91), with

$$\widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x}) = \mathrm{Proj}_{\boldsymbol{\varphi}}\left(g(0)\widetilde{\boldsymbol{c}}^{\mathrm{out}}_{*,\mathbf{0}}(\boldsymbol{x})\psi_{\mathbf{0}}\right). \tag{98}$$

Equation (92) then follows from the deterministic assignment in Eq. (88).

□

Corollary 10 (Vanishing patch-boundary jumps)**.** *For two adjacent patches, define the jump between their local predictions at a shared boundary point by*

$$\mathcal{J}_{ij}(\boldsymbol{x}) := \left\|\widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{u}_j^{\mathrm{loc}}(\boldsymbol{x})\right\|, \qquad \boldsymbol{x}\in\overline{P_i}\cap\overline{P_j}\cap\overline{\Omega}. \tag{99}$$

Then

$$\max_{i\ne j} \sup_{\boldsymbol{x}\in\overline{P_i}\cap\overline{P_j}\cap\overline{\Omega}} \mathcal{J}_{ij}(\boldsymbol{x}) \longrightarrow 0 \qquad \textit{under Eq. (36)}, \tag{100}$$

where pairs with an empty intersection are omitted. Hence the possible discontinuity introduced by the deterministic boundary assignment vanishes under admissible joint source-discretization and partition refinement.

Proof. For every shared boundary point $\boldsymbol{x}$,

$$\begin{aligned}\mathcal{J}_{ij}(\boldsymbol{x}) \le &\left\|\widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x})\right\| \\ &+ \left\|\widehat{u}_j^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x})\right\|.\end{aligned} \tag{101}$$

The right-hand side converges uniformly to zero by Eq. (91). More explicitly, the proof of Proposition 9 gives the bound

$$\mathcal{J}_{ij}(\boldsymbol{x}) \le C\left(\delta_{\mathcal{P},\mathbb{X}}^{\mathrm{out}} + \omega_*\left(C_{\widehat{P}} H_{\mathcal{P}}\right) + H_{\mathcal{P}}\right). \tag{102}$$

□

Remark 11 (Role of the higher-order decoder terms). Equation (98) contains only the degree-zero output coefficient because it describes the value of the reconstructed field in the limit $H_{\mathcal{P}} \to 0$. This does not imply that the limiting prediction depends only on zeroth-order input information. The field $\widetilde{\boldsymbol{c}}_{*,\mathbf{0}}^{\mathrm{out}}(\boldsymbol{x})$ is produced from $\boldsymbol{z}_*^{(L_{\mathrm{T}})}(\boldsymbol{x})$, whose initial token field in Eq. (66) is constructed from the complete transformed input K-jet. Therefore, the surviving degree-zero output coefficient may already encode higher-order local input information as well as the global interactions computed by the transformer.

The positive-degree output coefficients have a separate role in finite-patch reconstruction. The degree-m part is

$$\widehat{w}_{i,m}^{\mathrm{loc}}(\boldsymbol{x}) := \sum_{\alpha \in \Lambda_m^\circ} g(|\alpha|) h_i^m \widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}} \psi_\alpha\left(N_i(\boldsymbol{x})\right). \tag{103}$$

At the level of field values, $\widehat{w}_{i,m}^{\mathrm{loc}}(\boldsymbol{x}) = O(h_i^m)$, which explains why every block with $m \ge 1$ vanishes from the pointwise continuum limit. This does not mean that the higher-order coefficients are discarded or irrelevant at finite resolution. For any multi-index β with $|\beta| = m$,

$$D_{\boldsymbol{x}}^\beta \widehat{w}_{i,m}^{\mathrm{loc}}(\boldsymbol{x}) = \sum_{\alpha \in \Lambda_m^\circ} g(|\alpha|) \widetilde{\boldsymbol{c}}_{i,\alpha}^{\mathrm{out}} D_{\boldsymbol{\xi}}^\beta \psi_\alpha\left(N_i(\boldsymbol{x})\right). \tag{104}$$

Thus, the degree-m block provides an $O(1)$ contribution at its natural m-th derivative scale. The positive-degree coefficients encode the within-patch variation and finite-scale corrections of the reconstructed field, while their amplitudes shrink at the order required for a common continuous limit across refined patches. If $\mathrm{Proj}_{\boldsymbol{\varphi}}$ is sufficiently smooth, these local variations propagate to the output through the chain rule; the continuum result above requires only Lipschitz continuity and therefore asserts convergence of field values rather than convergence of their derivatives.

C.6 Comparison with Smooth Scale-Independent Local GNO Aggregations

We compare the scale-normalized input coefficient representation with a smooth local GNO whose message is independent of the patch scale. No separate post-aggregation readout is introduced. Fix an interior point $\boldsymbol{r} \in \Omega^\circ$. For sufficiently small $h > 0$ such that $\boldsymbol{r} + h\widehat{P} \subset \Omega$, let

$$P_h(\boldsymbol{r}) := \boldsymbol{r} + h\widehat{P}. \tag{105}$$

Let

$$\kappa_{\boldsymbol{\theta}} : \overline{\Omega} \times \overline{\Omega} \times \mathbb{R}^{d_{\mathrm{lift}}} \times \mathbb{R}^{d_{\mathrm{lift}}} \to \mathbb{R}^{d_{\mathrm{GNO}}} \tag{106}$$

be a smooth vector-valued message function that is fixed after training and independent of h. We define the local GNO representation by

$$\mathcal{R}_h^{\mathrm{GNO}}[L](\boldsymbol{r}) := \frac{1}{|P_h(\boldsymbol{r})|} \int_{P_h(\boldsymbol{r})} \kappa_{\boldsymbol{\theta}}\left(\boldsymbol{r}, \boldsymbol{x}, L(\boldsymbol{r}), L(\boldsymbol{x})\right) d\boldsymbol{x}. \tag{107}$$

This form includes coordinate-dependent, feature-dependent, nonlinear, and multi-channel messages. A pointwise residual term such as $\boldsymbol{W}L(\boldsymbol{r})$ can also be included directly by adding the constant message $\boldsymbol{W}L(\boldsymbol{r})$ inside the integrand. The normalization by $|P_h(\boldsymbol{r})|$ gives a nontrivial local limit under patch refinement; an unnormalized shrinking-patch integral would instead vanish with the patch volume.

Proposition 12 (Patch-refinement limit of a local GNO)**.** *Suppose that $L \in C^2$ locally around $\boldsymbol{r}$ and that $\kappa_{\boldsymbol{\theta}}$ is C^2 on the bounded set encountered in the aggregation. Then*

$$\mathcal{R}_h^{\mathrm{GNO}}[L](\boldsymbol{r}) = \kappa_{\boldsymbol{\theta}}\left(\boldsymbol{r}, \boldsymbol{r}, L(\boldsymbol{r}), L(\boldsymbol{r})\right) + O(h^2). \tag{108}$$

Consequently, the partition-consistent local GNO limit depends on the input field only through its zeroth-order value at $\boldsymbol{r}$. If central symmetry of the reference patch is dropped, the same conclusion holds with an $O(h)$ remainder.

Proof. The change of variables $\boldsymbol{x} = \boldsymbol{r} + h\boldsymbol{\xi}$ gives

$$\mathcal{R}_h^{\mathrm{GNO}}[L](\boldsymbol{r}) = \frac{1}{|\widehat{P}|}\int_{\widehat{P}} F_{\boldsymbol{r}}(\boldsymbol{r} + h\boldsymbol{\xi})\, d\boldsymbol{\xi}, \tag{109}$$

where

$$F_{\boldsymbol{r}}(\boldsymbol{x}) := \kappa_{\boldsymbol{\theta}}\left(\boldsymbol{r}, \boldsymbol{x}, L(\boldsymbol{r}), L(\boldsymbol{x})\right). \tag{110}$$

Taylor expansion yields

$$F_{\boldsymbol{r}}(\boldsymbol{r} + h\boldsymbol{\xi}) = F_{\boldsymbol{r}}(\boldsymbol{r}) + hD_{\boldsymbol{x}}F_{\boldsymbol{r}}(\boldsymbol{r})\boldsymbol{\xi} + O(h^2). \tag{111}$$

Central symmetry gives

$$\int_{\widehat{P}} \boldsymbol{\xi}\, d\boldsymbol{\xi} = 0. \tag{112}$$

Therefore,

$$\mathcal{R}_h^{\mathrm{GNO}}[L](\boldsymbol{r}) = F_{\boldsymbol{r}}(\boldsymbol{r}) + O(h^2) = \kappa_{\boldsymbol{\theta}}\left(\boldsymbol{r}, \boldsymbol{r}, L(\boldsymbol{r}), L(\boldsymbol{r})\right) + O(h^2). \tag{113}$$

Without central symmetry, the first-order moment need not vanish, which gives the stated $O(h)$ remainder.

□

Define the limiting PINOT coefficient representation for the vector-valued lifted field by

$$\mathcal{R}_*^{\mathrm{PINOT}}[L](\boldsymbol{r}) := \boldsymbol{J}^K L(\boldsymbol{r}) \boldsymbol{A}_{\boldsymbol{\theta},K}^{\top}. \tag{114}$$

Equation (61) gives

$$\widetilde{\boldsymbol{C}}_{\boldsymbol{r},h}^{\mathrm{in},\mathbb{X}} \longrightarrow \mathcal{R}_*^{\mathrm{PINOT}}[L](\boldsymbol{r}) \tag{115}$$

under admissible source sampling and patch refinement centered at $\boldsymbol{r}$. Define

$$\Phi_{\boldsymbol{\theta},\boldsymbol{r}}(\boldsymbol{a}) := \kappa_{\boldsymbol{\theta}}\left(\boldsymbol{r}, \boldsymbol{r}, \boldsymbol{a}, \boldsymbol{a}\right). \tag{116}$$

Proposition 12 gives the limiting GNO representation

$$\mathcal{R}_*^{\mathrm{GNO}}[L](\boldsymbol{r}) := \Phi_{\boldsymbol{\theta},\boldsymbol{r}}\left(L(\boldsymbol{r})\right). \tag{117}$$

Corollary 13 (Strict informativeness relative to local GNO)**.** *Fix an interior point $\boldsymbol{r} \in \Omega^{\circ}$. For every $K \geq 1$, over the class of all C^{K+2} lifted fields L, the limiting PINOT coefficient representation in Eq. (114) is strictly more informative than the limiting local GNO representation in Eq. (117).*

Proof. Since $\boldsymbol{A}_{\boldsymbol{\theta},K}$ is invertible, $\mathcal{R}_*^{\mathrm{PINOT}}[L](\boldsymbol{r})$ uniquely determines the complete local jet $\boldsymbol{J}^K L(\boldsymbol{r})$, including its zeroth-order column $L(\boldsymbol{r})$. Hence the limiting GNO representation can be recovered from the PINOT representation:

$$\mathcal{R}_*^{\mathrm{GNO}}[L](\boldsymbol{r}) = \Phi_{\boldsymbol{\theta},\boldsymbol{r}}\left(\pi_{\mathbf{0}}\left(\mathcal{R}_*^{\mathrm{PINOT}}[L](\boldsymbol{r})\boldsymbol{A}_{\boldsymbol{\theta},K}^{-\top}\right)\right), \tag{118}$$

where $\pi_{\mathbf{0}}$ extracts the zeroth-order column of the vector-valued jet.

The converse does not hold. Choose two smooth fields satisfying

$$L_1(\boldsymbol{r}) = L_2(\boldsymbol{r}), \qquad \boldsymbol{J}^K L_1(\boldsymbol{r}) \neq \boldsymbol{J}^K L_2(\boldsymbol{r}). \tag{119}$$

For example, for $K \geq 1$, one may locally take a scalar pair

$$L_1(\boldsymbol{x}) = 0, \qquad L_2(\boldsymbol{x}) = x_1 - r_1. \tag{120}$$

Then

$$\mathcal{R}_*^{\mathrm{GNO}}[L_1](\boldsymbol{r}) = \mathcal{R}_*^{\mathrm{GNO}}[L_2](\boldsymbol{r}), \qquad \mathcal{R}_*^{\mathrm{PINOT}}[L_1](\boldsymbol{r}) \neq \mathcal{R}_*^{\mathrm{PINOT}}[L_2](\boldsymbol{r}). \tag{121}$$

Thus, the PINOT coefficient representation distinguishes local fields that are indistinguishable in the partition-consistent GNO limit.

□

Remark 14 (Scope of the GNO comparison). The comparison permits an arbitrary fixed output dimension and a general smooth vector-valued GNO message that depends nonlinearly on the query coordinate, source coordinate, query feature, and source feature. It is made directly at the GNO aggregation output, without assuming an additional readout layer. Any subsequent continuous pointwise map cannot recover derivative information that is absent from the limiting aggregation, although no such map is needed for the result above.

The conclusion concerns local GNO messages that are independent of the shrinking patch scale. A scale-aware GNO may explicitly preserve higher-order information through messages containing factors such as

$$h^{-|\alpha|}\psi_\alpha\left(\frac{\boldsymbol{x}-\boldsymbol{r}}{h}\right). \tag{122}$$

However, doing so introduces the same reference-coordinate transformation and degree-dependent scaling as the proposed encoder. The result therefore concerns the information retained beyond a standard scale-independent GNO, rather than every possible scale-aware GNO architecture.

C.7 Proofs of Theorems 1 and 2

Proof of Theorem 1. Proposition 5 and Eq. (61) establish that the scale-normalized input coefficients converge to a common continuum representation under admissible source sampling and patch refinement. Equation (67) transfers this convergence through the shared coefficient-to-token map.

The partition quadrature in Lemma 6 and the attention result in Proposition 7 yield the transformer limit in Corollary 8. Proposition 9 extends this convergence to the reconstructed output field.

Finally, combining the exact target restriction in Eq. (89) with the uniform decoder convergence in Eq. (92) gives, for every target point set,

$$\begin{aligned}&\left\|\mathcal{G}_{\boldsymbol{\theta}}\left(\mathbb{X}_{\mathrm{src}},\, v|_{\mathbb{X}_{\mathrm{src}}},\mathbb{X}_{\mathrm{tar}};\mathcal{P}\right)-\widehat{u}_{\boldsymbol{\theta}}(v)|_{\mathbb{X}_{\mathrm{tar}}}\right\|_{\ell^\infty(\mathbb{X}_{\mathrm{tar}})}\\ &\qquad\le\sup_{\boldsymbol{x}\in\Omega}\|\widehat{u}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)(\boldsymbol{x})-\widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x})\|\longrightarrow 0.\end{aligned} \tag{123}$$

This proves Eq. (8). Because Eq. (89) restricts a field that is already defined on Ω, the target discretization can be selected independently. Corollary 10 additionally shows that the deterministic boundary convention does not create a nonvanishing discontinuity in the refinement limit.

□

Proof of Theorem 2. Equations (61) and (62) give

$$\max_i\left\|\widetilde{\boldsymbol{C}}_i^{\mathrm{in},\mathbb{X}}-\boldsymbol{J}^K L(\boldsymbol{r}_i)\boldsymbol{A}_{\boldsymbol{\theta},K}^\top\right\|_{\mathrm{F}}\le C\left(H_{\mathcal{P}}^2+\max_i\varepsilon_{i,\mathbb{X}}\right)\longrightarrow 0 \tag{124}$$

under admissible refinement. Since $\boldsymbol{A}_{\boldsymbol{\theta},K}$ is invertible, the limiting normalized coefficient representation uniquely determines $\boldsymbol{J}^K L(\boldsymbol{r}_i)$, which contains all derivatives of the lifted field through order K. For nonsymmetric or clipped patches, the same conclusion follows from Eq. (64), with $H_{\mathcal{P}}$ replacing $H_{\mathcal{P}}^2$. □

C.8 Extension to Overlapping Patches

The preceding analysis uses non-overlapping patches to keep the notation and quadrature argument simple. The construction extends to an overlapping patch cover by introducing a partition of unity that assigns each physical location a total weight of one. Let

$$\mathcal{P}=\{P_i\}_{i=1}^M \tag{125}$$

cover Ω, and assume that there exists a refinement-independent constant M_{ov} such that

$$\sup_{\boldsymbol{x}\in\Omega}\operatorname{card}\{i:\boldsymbol{x}\in P_i\}\le M_{\mathrm{ov}}<\infty. \tag{126}$$

Choose nonnegative continuous weights $\{\chi_i\}_{i=1}^M$ satisfying

$$\operatorname{supp}(\chi_i)\subseteq\overline{P_i},\qquad\sum_{i=1}^M\chi_i(\boldsymbol{x})=1\quad\text{for every }\boldsymbol{x}\in\overline{\Omega}. \tag{127}$$

For example, let nonnegative tapering windows ω_i be supported on the corresponding patches and satisfy $\sum_i \omega_i(\boldsymbol{x}) > 0$. Then define

$$\chi_i(\boldsymbol{x}) := \frac{\omega_i(\boldsymbol{x})}{\sum_{p=1}^{M} \omega_p(\boldsymbol{x})}. \tag{128}$$

Local encoding. For each patch, the encoder continues to use

$$\mathbb{X}_{\mathrm{src}}^{i} = P_i \cap \mathbb{X}_{\mathrm{src}} \tag{129}$$

and the ordinary local least-squares fit in Eq. (23). A source point in an overlap is intentionally used in every local fit whose patch contains it. No partition-of-unity factor is needed inside the local least-squares objective, because that objective constructs a local representation rather than a global quadrature rule. The consistency result for the scale-normalized coefficient representation therefore remains unchanged, provided the stability and empirical-measure assumptions hold for every overlapping patch.

Measure-weighted attention. Using the raw patch volumes $|P_i|$ would count an overlapped physical region multiple times. Instead, associate token i with the effective measure

$$\mu_i^{\chi} := \int_{\Omega} \chi_i(\boldsymbol{x})\, d\boldsymbol{x}. \tag{130}$$

Patches with $\mu_i^{\chi} = 0$ are discarded; every active patch is assumed to satisfy $\mu_i^{\chi} > 0$. The attention weights are then computed as

$$A_{i,j}^{(\ell),\chi} = \mathrm{softmax}_j \left(\log \mu_j^{\chi} + \frac{\left(\boldsymbol{q}_i^{(\ell)}\right)^{\top} \boldsymbol{k}_j^{(\ell)}}{\sqrt{d_h}} \right). \tag{131}$$

Indeed, for every continuous φ,

$$\begin{aligned} &\left\| \sum_{i=1}^{M} \mu_i^{\chi} \varphi(\boldsymbol{r}_i) - \int_{\Omega} \varphi(\boldsymbol{x})\, d\boldsymbol{x} \right\| \\ &\quad = \left\| \int_{\Omega} \sum_{i=1}^{M} \chi_i(\boldsymbol{x}) \left(\varphi(\boldsymbol{r}_i) - \varphi(\boldsymbol{x})\right)\, d\boldsymbol{x} \right\| \\ &\quad \le |\Omega|\, \omega_{\varphi}\left(C_{\widehat{P}} H_{\mathcal{P}}\right) \longrightarrow 0. \end{aligned} \tag{132}$$

Thus, $\{(\boldsymbol{r}_i, \mu_i^{\chi})\}$ discretizes the physical measure without double-counting the overlaps, and the proof of measure-weighted attention consistency applies with μ_i replaced by μ_i^{χ}.

Overlapping decoder. At a target coordinate $\boldsymbol{x}$, evaluate the local decoder for every patch with $\chi_i(\boldsymbol{x}) > 0$ and combine the predictions using the same partition of unity:

$$\widehat{u}^{\mathrm{ov}}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)(\boldsymbol{x}) := \sum_{i:\, \chi_i(\boldsymbol{x})>0} \chi_i(\boldsymbol{x}) \widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}). \tag{133}$$

This replaces the deterministic single-patch assignment by a uniquely defined blended prediction. If the tapering weights and local predictions are continuous, then $\widehat{u}^{\mathrm{ov}}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)$ is continuous even at finite patch scale. It is again defined before selecting target points, so

$$\mathcal{G}_{\boldsymbol{\theta}}\left(\mathbb{X}_{\mathrm{src}},\, v|_{\mathbb{X}_{\mathrm{src}}},\, \mathbb{X}_{\mathrm{tar}};\, \mathcal{P}\right) = \widehat{u}^{\mathrm{ov}}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)\big|_{\mathbb{X}_{\mathrm{tar}}}. \tag{134}$$

Proposition 15 (Consistency under overlapping patch refinement)**.** *Assume the bounded-overlap and partition-of-unity conditions above, together with the admissible refinement in Eq. (36) and the same sampling and regularity conditions. Then the local encoder converges to the same continuum token field, and attention with μ_i^{χ} converges to the same continuous operator. The blended decoder satisfies*

$$\sup_{\boldsymbol{x}\in\Omega} \left\| \widehat{u}^{\mathrm{ov}}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x}) \right\| \longrightarrow 0. \tag{135}$$

Table 6: Dataset and task specifications used in our experiments. N denotes the number of spatial points per sample.

Discretization	Dataset	Learning task	N	Data source
Nonuniform	NACA0012	$(Ma, \alpha, \Omega) \mapsto u_\Omega$	8000	GAOT (Wen et al., 2026)
	RAE2822	$(Ma, \alpha, \Omega) \mapsto u_\Omega$	8000	GAOT (Wen et al., 2026)
	DeepJEB	$\Omega \mapsto \sigma_{\mathrm{VM}}$	18k–55k	DeepJEB (Hong et al., 2024)
Uniform	1D Burgers	$u_0(\cdot) \mapsto u(\cdot, 1)$	1024–4096	FNO (Li et al., 2021)
	NS-Gauss	$\mathbf{v}(\cdot, t) \mapsto \mathbf{v}(\cdot, t + 0.2)$	128^2	Poseidon (Herde et al., 2024)
	CE-RP	$\mathbf{q}(\cdot, t) \mapsto \mathbf{q}(\cdot, t + 0.2)$	128^2	Poseidon (Herde et al., 2024)

Proof. The local encoder argument is patchwise and is unaffected by overlap. Equation (132) replaces Lemma 6, so the attention and finite-depth transformer proofs proceed unchanged with the effective measures μ_i^χ. Finally, the partition-of-unity property gives

$$\begin{aligned} &\left\| \widehat{u}^{\mathrm{ov}}_{\boldsymbol{\theta};\mathcal{P},\mathbb{X}}(v)(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x}) \right\| \\ &\quad \le \sum_{i:\, \chi_i(\boldsymbol{x})>0} \chi_i(\boldsymbol{x}) \left\| \widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x}) \right\| \\ &\quad \le \max_{i:\, \chi_i(\boldsymbol{x})>0} \left\| \widehat{u}_i^{\mathrm{loc}}(\boldsymbol{x}) - \widehat{u}_{\boldsymbol{\theta}}(v)(\boldsymbol{x}) \right\|. \end{aligned} \tag{136}$$

Taking the supremum over $\boldsymbol{x}$ and applying Eq. (91) proves the result.

□

Remark 16 (Hard overlap weights). The algebraic choice

$$\chi_i(\boldsymbol{x}) = \frac{\mathbf{1}_{P_i}(\boldsymbol{x})}{\sum_p \mathbf{1}_{P_p}(\boldsymbol{x})} \tag{137}$$

equally averages all patches containing $\boldsymbol{x}$ and still avoids measure double-counting when used in Eq. (130). However, these weights are generally discontinuous where the overlap multiplicity changes. Continuous tapering windows are therefore preferable when finite-resolution continuity is desired; with hard weights, continuity is recovered only asymptotically from the local uniform convergence in Eq. (91).

D Experimental Details

D.1 Dataset Configuration

We evaluate the performance of PINOT and competing methods on six PDE benchmarks spanning uniform and nonuniform discretizations. All experiments use the officially released data files from the cited sources. We construct our own train–test splits rather than using the splits from the original studies, and apply the dataset-specific postprocessing described below. The PDE benchmark suite includes three benchmarks with irregular and nonuniform discretizations, NACA0012, RAE2822, and DeepJEB; one irregular but uniformly sampled benchmark, 1D Burgers; and two time-dependent benchmarks on regular Cartesian grids, NS-Gauss and CE-RP. Table 6 provides an overview of the datasets and learning tasks considered in our experiments, including their sampling characteristics, spatial resolutions, and data sources. Unless otherwise stated, dataset splits are constructed deterministically by taking contiguous index ranges without shuffling.

Following GAOT (Wen et al., 2026), we use the sample-wise relative L_1 error as the evaluation metric. We report its mean over the test samples, rather than the median, to measure average predictive performance over the test distribution and retain the contribution of high-error cases. All competing methods are evaluated using the same aggregation protocol.

NACA0012 and RAE2822. The NACA0012 and RAE2822 datasets are steady-state compressible-flow benchmarks introduced in GAOT (Wen et al., 2026). We use the officially released GAOT data and construct our own input representation and train–test split as described below.

The flow is governed by the steady compressible Euler equations,

$$\nabla \cdot \mathbf{F}(\mathbf{U}) = 0, \tag{138}$$

where

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho\mathbf{v} \\ E \end{bmatrix}, \qquad \mathbf{F}(\mathbf{U}) = \begin{bmatrix} \rho\mathbf{v} \\ \rho\mathbf{v} \otimes \mathbf{v} + pI \\ (E+p)\mathbf{v} \end{bmatrix}, \tag{139}$$

with

$$E = \frac{1}{2}\rho\|\mathbf{v}\|^2 + \frac{p}{\gamma - 1}, \qquad \gamma = 1.4. \tag{140}$$

The datasets cover free-stream conditions

$$0.5 \le Ma \le 1.4, \qquad 0.5^\circ \le \alpha_{\mathrm{AoA}} \le 5.0^\circ, \tag{141}$$

where Ma denotes the Mach number and α_{AoA} is the angle of attack.

To introduce geometric variation, GAOT perturbs the reference NACA0012 and RAE2822 airfoil geometries using the class-function/shape-function transformation (CST). Letting $\xi = x/c$ denote the chord-normalized coordinate, the upper and lower surfaces are represented as

$$\eta^U(\xi) = C(\xi)S^U(\xi), \qquad \eta^L(\xi) = C(\xi)S^L(\xi), \tag{142}$$

where the class function is

$$C(\xi) = \sqrt{\xi}(1-\xi), \tag{143}$$

and the shape functions are parameterized by Bernstein polynomials of degree $n = 7$. Random perturbations of the corresponding CST coefficients generate different airfoil geometries.

The released flow solutions were generated using the SU2 EULER solver with the Jameson–Schmidt–Turkel flux scheme on O-type unstructured meshes generated by Delaundo. A Mach-based anisotropic adaptive-mesh refinement procedure is used to resolve shocks, resulting in strongly nonuniform spatial discretizations. GAOT subsamples $8{,}000$ points from each final mesh.

In our benchmark implementation, each point is represented by

$$[x, y, Ma, \alpha_{\mathrm{AoA}}, \mathrm{SDF}_\Omega(x, y)], \tag{144}$$

where SDF_Ω denotes the signed distance to the airfoil geometry. The corresponding task can be viewed abstractly as

$$\mathcal{G} : (Ma, \alpha, \Omega) \mapsto u_\Omega, \tag{145}$$

where $u_\Omega = \rho_\Omega$ denotes the steady-state fluid density field around the airfoil, with one scalar density value at each sampled spatial point.

For both NACA0012 and RAE2822, we use the first $8{,}192$ samples for training and the following $1{,}024$ samples for testing. Each sample contains $8{,}000$ nonuniformly distributed spatial points. Models are trained with a batch size of 64 for 500 epochs.

DeepJEB. DeepJEB (Hong et al., 2024) is a dataset of $2{,}138$ synthetic three-dimensional jet-engine bracket geometries together with finite-element structural-analysis results. The underlying static structural problem can be written in the generic linear-elasticity form

$$-\nabla \cdot \boldsymbol{\sigma}(\mathbf{d}) = 0 \qquad \text{in } \Omega, \tag{146}$$

where $\mathbf{d}$ denotes the displacement field and

$$\boldsymbol{\varepsilon}(\mathbf{d}) = \frac{1}{2}\left(\nabla\mathbf{d} + \nabla\mathbf{d}^\top\right), \qquad \boldsymbol{\sigma} = \mathbb{C} : \boldsymbol{\varepsilon}, \tag{147}$$

subject to the corresponding loading and support boundary conditions. DeepJEB provides displacement and stress fields for multiple loading scenarios.

We use the processed Jet Engine Bracket data released with GINOT (Liu et al., 2025b). GINOT considers the von Mises stress field under vertical loading. To reduce the size of the original quadratic finite-element meshes, the original 10-node tetrahedra are converted into 4-node linear tetrahedra by

removing edge-midpoint nodes. This results in a variable number of volume-mesh nodes, ranging approximately from $18{,}085$ to $55{,}009$ per geometry.

The original GINOT formulation separately encodes a CAD-derived boundary point cloud and evaluates the solution at volume-mesh query points. In our benchmark, however, we use a simplified representation and retain only the volume-mesh node coordinates and nodal von Mises stress. Thus, the current learning task is

$$\{(x_i, y_i, z_i)\}_{i=1}^{N} \mapsto \{\sigma_{\mathrm{VM},i}\}_{i=1}^{N}, \tag{148}$$

where the number of points, N, varies across geometries. Tetrahedral connectivity and the separate CAD boundary point cloud are not provided to the baseline models in this benchmark.

We use the first $1{,}024$ samples for training and the following 256 samples for testing, with a batch size of 16 and $1{,}000$ training epochs. We do not report OFormer on DeepJEB because its official repository does not provide a three-dimensional implementation. Moreover, OFormer performs attention over all source points, leading to prohibitive memory consumption for complex three-dimensional geometries such as DeepJEB.

1D Burgers. We construct a 1D Burgers benchmark with uniform but irregular discretizations from the dataset originally introduced for the Fourier Neural Operator (FNO) (Li et al., 2021). The underlying PDE is the one-dimensional viscous Burgers equation

$$\partial_t u(x,t) + \partial_x \left(\frac{u(x,t)^2}{2} \right) = \nu \, \partial_{xx} u(x,t), \qquad x \in (0,1), \quad t \in (0,1], \tag{149}$$

with periodic boundary conditions and

$$u(x,0) = u_0(x), \qquad \nu = 0.1. \tag{150}$$

The initial condition is sampled from the Gaussian random field

$$u_0 \sim \mathcal{N}\left(0,\, 625(-\Delta + 25I)^{-2}\right). \tag{151}$$

The original FNO data are generated on a uniform grid containing 8192 spatial points, and the learning objective is the solution operator

$$\mathcal{G}^\dagger : u_0(\cdot) \mapsto u(\cdot, 1). \tag{152}$$

In our experiments, we convert this originally regular-grid dataset into an irregular point-cloud benchmark. For each sample, we first draw $N_{\mathrm{src}} \sim \mathrm{Unif}\{1{,}024, \ldots, 4{,}096\}$ and then uniformly sample N_{src} indices without replacement from the original $8{,}192$-point grid and extract both the initial condition and the final-time solution at the same coordinates. The sampled indices are generated once with a fixed random seed and used consistently across all methods.

Thus, the pointwise model input is

$$\big[x,\ u_0(x)\big], \tag{153}$$

and the prediction target is $u(x, 1)$. We use the first $1{,}024$ samples for training and the following 256 samples for testing, with a batch size of 16 and $1{,}000$ training epochs. Importantly, the irregular but uniform spatial discretization is introduced by our sampling procedure; the original FNO dataset itself is generated on a regular grid.

NS-Gauss. NS-Gauss is taken from the Poseidon benchmark suite (Herde et al., 2024) and is governed by the two-dimensional incompressible Navier–Stokes equations

$$\partial_t \mathbf{v} + (\mathbf{v} \cdot \nabla)\mathbf{v} + \nabla p = \nu \Delta \mathbf{v}, \qquad \nabla \cdot \mathbf{v} = 0, \tag{154}$$

on the periodic domain $\Omega = [0,1]^2$. The released trajectories are generated using Poseidon's spectral-viscosity discretization. The numerical dissipation acts only on sufficiently high Fourier modes, with $N = 128$, $m_N = \sqrt{N}$, and $\epsilon_N = 0.05/N \approx 3.9 \times 10^{-4}$. Thus, this value should be interpreted as the effective high-mode spectral viscosity, rather than a spatially uniform physical viscosity multiplying all modes.

The initial condition is specified through the vorticity

$$\omega_0(x,y) = \sum_{i=1}^{100} \frac{\alpha_i}{\sigma_i} \exp\left(-\frac{(x-x_i)^2 + (y-y_i)^2}{2\sigma_i^2} \right), \tag{155}$$

where

$$\alpha_i \sim \mathcal{U}[-1,1], \qquad \sigma_i \sim \mathcal{U}[0.01,0.1], \qquad x_i, y_i \sim \mathcal{U}[0,1]. \tag{156}$$

The corresponding divergence-free initial velocity $\mathbf{v}_0 = (v_x, v_y)$ is recovered from the vorticity.

The original trajectories contain 21 uniformly spaced snapshots over $t \in [0,1]$, each represented on a 128×128 Cartesian grid.

In our benchmark, we use the officially released trajectories but define our own trajectory split and temporal-pair construction. We retain all $128^2 = 16{,}384$ spatial points and use 1024 trajectories for training and 256 for testing. Rather than training on the full trajectory, we use a step-wise prediction task with stride four:

$$t_0 \to t_4, \quad t_4 \to t_8, \quad t_8 \to t_{12}, \quad t_{12} \to t_{16}, \quad t_{16} \to t_{20}, \tag{157}$$

corresponding to a physical time interval

$$\Delta t = 0.2. \tag{158}$$

Each trajectory therefore generates five input–output pairs. The actual pointwise input to the models is

$$[v_x(x,t),\, v_y(x,t),\, x,\, y,\, t], \tag{159}$$

and the target is

$$[v_x(x,t+0.2),\, v_y(x,t+0.2)]. \tag{160}$$

Consequently, the 1,024 training trajectories yield 5,120 training pairs. We train with a batch size of 64 for 1,000 epochs.

CE-RP. The compressible Euler Riemann-problem dataset (CE-RP) is also taken from Poseidon (Herde et al., 2024). The governing equations are the time-dependent compressible Euler equations

$$\partial_t \mathbf{U} + \nabla \cdot \mathbf{F}(\mathbf{U}) = 0, \tag{161}$$

where $\mathbf{U}$ and $\mathbf{F}$ are defined in Eq. (139).

CE-RP is a stochastic two-dimensional four-quadrant Riemann problem. The periodic domain $\Omega = [0,1]^2$ is partitioned into four equal quadrants

$$D_{ij} = \left\{ (x,y) \in \mathbb{T}^2 : \frac{i-1}{2} \le x < \frac{i}{2},\ \frac{j-1}{2} \le y < \frac{j}{2} \right\}, \qquad i,j \in \{1,2\}. \tag{162}$$

Within each quadrant, the initial primitive variables are spatially constant:

$$(\rho_0, v_{x,0}, v_{y,0}, p_0)\big|_{D_{ij}} = (\rho_{ij}, v_{x,ij}, v_{y,ij}, p_{ij}), \tag{163}$$

where

$$\rho_{ij}, p_{ij} \sim \mathcal{U}[0.1,1], \qquad v_{x,ij}, v_{y,ij} \sim \mathcal{U}[-1,1]. \tag{164}$$

The released CE-RP trajectories contain 21 uniformly spaced snapshots over $t \in [0,1]$ on a 128×128 Cartesian grid.

In our benchmark, we use the officially released trajectories but define our own trajectory split and temporal-pair construction. We retain all 16,384 spatial points and use 1,024 trajectories for training and 256 for testing. With a temporal stride of four snapshots, each trajectory is converted into five prediction pairs with $\Delta t = 0.2$. Defining the primitive state as

$$\mathbf{q} = [\rho, v_x, v_y, p], \tag{165}$$

our benchmark learns the step-wise solution operator

$$\mathbf{q}(\cdot, t) \mapsto \mathbf{q}(\cdot, t+0.2). \tag{166}$$

The pointwise model input is

$$[\rho, v_x, v_y, p, x, y, t], \tag{167}$$

and the prediction target is the four-dimensional primitive state $\mathbf{q}(\cdot, t+0.2)$. Consequently, the 1,024 training trajectories yield 5,120 training pairs. Models are trained with a batch size of 64 for 1,000 epochs.

For NS-Gauss and CE-RP, we perform five autoregressive prediction steps with $\Delta t = 0.2$ from the initial state and report the relative L_1 error at the final time $t = 1$.

Common preprocessing. We apply a common preprocessing procedure for all benchmark models. Spatial coordinates are independently scaled along each axis to $[-1, 1]$ using statistics computed from the training set:

$$\widetilde{x}_k = 2\frac{x_k - x_k^{\min}}{x_k^{\max} - x_k^{\min}} - 1. \tag{168}$$

Solution and conditioning channels are standardized channel-wise using the training-set mean and standard deviation,

$$\widetilde{u}_c = \frac{u_c - \mu_c^u}{\sigma_c^u}, \qquad \widetilde{c}_j = \frac{c_j - \mu_j^c}{\sigma_j^c}, \tag{169}$$

and time is normalized analogously for time-dependent datasets,

$$\widetilde{t} = \frac{t - \mu_t}{\sigma_t}. \tag{170}$$

All normalization statistics are computed exclusively from the training set and subsequently fixed when processing the test set.

D.2 Baseline Implementations

Table 7: Processor configurations used for the baseline models and PINOT. d denotes the spatial dimension and N_{src} the number of input points. For GINO, processor depth refers to the number of FNO layers, whereas for attention-based models it denotes the number of transformer blocks.

Method	Hidden dim.	Processor depth	# processor tokens	Aggregation radius
GINO	64	4	32^d 8^3	$0.066, \quad d = 1, 2$ $0.266, \quad d = 3$
OFormer	256	5	N_{src}	–
GAOT†	256	5	32^d 8^3	$\{0.022, 0.033, 0.044\}$, $d = 1, 2$ $\{0.088, 0.132, 0.176\}$, $d = 3$
Transolver	128	8	64	–
LNO	256	8	256	–
PINOT (Ours)	256	8	32^d / adaptive	–

† GAOT first constructs a 64^d latent grid and groups neighboring latent elements using a patch size of 2 along each spatial dimension, resulting in 32^d tokens processed by the transformer for $d = 1, 2$. For the 3D benchmark, we use a 16^3 latent grid, resulting in 8^3 tokens.

We implement all competing methods based on their official implementations and published architectural configurations, while adapting the model capacity when necessary to accommodate the dimensionality and complexity of our benchmark problems. The optimization protocol is standardized across all methods as described below.

Table 7 summarizes the processor configurations used in our experiments. Here, the processor hidden dimension and depth refer to the feature width and the number of processing layers, respectively. For methods that operate on an intermediate discretization, the number of processor tokens denotes the number of elements processed by the main FNO (Li et al., 2021) or attention-based processor. All aggregation radii are applied after the coordinate normalization described above.

For configuration sources, GINO follows the CarCFD architecture with dimension-wise extension; OFormer follows its steady-state configuration with capacity scaled to Airfoil; GAOT uses the official multiscale setting; Transolver and LNO use the official Airfoil and NS2D settings, respectively.

GINO. We implement GINO (Li et al., 2023b) following its official CarCFD configuration, using a GNO architecture to map between the input point cloud and a uniform latent grid, followed by an FNO processor. We retain the official processor width of 64 and four FNO layers. For 1D and 2D

Table 8: Dataset-specific training configurations. For time-dependent datasets (NS-Gauss and CE-RP), the train/test counts denote trajectories; each trajectory is further converted into five temporal input–output pairs with a time interval of $\Delta t = 0.2$.

Dataset	Train	Test	Batch	Epochs
1D Burgers	1024	256	16	1000
NACA0012	8192	1024	64	500
RAE2822	8192	1024	64	500
DeepJEB	1024	256	16	1000
NS-Gauss	1024	256	64	1000
CE-RP	1024	256	64	1000

benchmarks, we use the official latent grid resolution of 32^d with a neighborhood aggregation radius of 0.066. For the 3D benchmark, we instead use an 8^3 latent grid with a neighborhood aggregation radius of 0.266. For the input reconstruction experiment in Section 3.2.2 only, we use a 64×64 latent grid with a neighborhood aggregation radius of 0.033 and set the latent embedding dimension to 256 to match that of GAOT's MAGNO encoder. This gives both encoders $4{,}096$ latent tokens with the same embedding dimension.

OFormer. For OFormer (Li et al., 2023a), we use its steady-state irregular-grid formulation. Accordingly, the encoded input features are directly processed to predict the output field without using a latent temporal propagator. Since the original steady-state configurations were designed for relatively small static boundary-value problems, we increase the processor capacity based on the official Airfoil configuration, using a hidden dimension of 256 and five processor blocks. OFormer processes the physical input points directly, and thus its processor token count is N_{src}.

GAOT. We follow the official multiscale GAOT (Wen et al., 2026) configuration, including its three neighborhood aggregation radii $\{0.022, 0.033, 0.044\}$. For the 1D and 2D benchmarks, GAOT first constructs a 64^d latent grid and groups neighboring latent elements with a patch size of 2 along each dimension. Consequently, the transformer operates on 32^d processor tokens. For the 3D benchmark, we adapt the setting to use 8^3 processor tokens with three neighborhood aggregation radii $\{0.088, 0.132, 0.176\}$ to control the computational cost. The transformer processor uses a hidden dimension of 256 and five layers. The $4{,}096$ latent tokens used in the input reconstruction experiment in Section 3.2.2 correspond to the 64×64 MAGNO output grid before the ViT-style patchification used by its transformer processor.

Transolver. We follow the official Transolver (Wu et al., 2024) configuration used for the Airfoil benchmark, with a hidden dimension of 128, eight transformer blocks, and 64 learned physical slices. We retain these settings across our benchmarks without increasing the processor capacity.

LNO. For LNO (Wang & Wang, 2024), we use the official NS2D configuration with a processor hidden dimension of 256, eight processing layers, and 256 latent tokens. The same processor configuration is used across datasets, while the pointwise input and output projections are adjusted according to the dimensionality of each benchmark.

Optimization. To ensure a consistent optimization budget across architectures, we train all competing methods and PINOT using the same optimization protocol. All models are optimized with AdamW (Loshchilov & Hutter, 2019) using an initial learning rate of 5×10^{-4} and a weight decay of 10^{-5}. The learning rate is decayed to 10^{-5} using cosine annealing over the full training schedule, without warmup. We minimize the mean-squared error in the normalized target space, evaluated only over valid, non-padded output points. No gradient clipping is applied. Within each dataset, all methods are trained using the same training budget and data split. No model selection is performed using the test set. We report the checkpoint from the final training epoch for every method. For the overall benchmark results in Table 4 of Section 3.2.4, we report the mean and standard deviation of the test errors over three independent training runs with different random seeds. The dataset-specific training settings are summarized in Table 8.

D.3 Implementation Details of PINOT

We summarize the implementation corresponding to the three architectural components in Figure 3. Detailed implementation of the batched local coefficient fitting in the encoder is provided separately in Appendix F.2. Unless otherwise specified, the network architecture is shared across all datasets, while the maximum total polynomial degree K and patch partition are dataset-dependent.

Table 9: Network architecture of PINOT corresponding to Figure 3. K denotes the dataset-dependent maximum total polynomial degree, and $|\Lambda_K| = \binom{d+K}{d}$ is the resulting number of local basis functions in spatial dimension d.

Component	Layer	Configuration
(a) Scale-normalized coefficient encoder	Lift	Pointwise MLP: $d_{\text{in}} \to 64 \to 64 \to 64$
	Time embedding	64-dim. sinusoidal embedding $\to 256 \to 64$ (time-dependent datasets only)
	Coefficient fitting	Local Legendre projection using $\{\psi_\alpha\}_{\lvert\alpha\rvert\le K}$ (Appendix F.2)
	Scale normalization	Degree-scaling MLP: $1 \to 16 \to 16 \to 1$
	Degree embedding	Shared MLP: $(64 + d) \to 64 \to 64$
	Tokenization	MLP: $64\lvert\Lambda_K\rvert \to H \to H \to 256$
(b) Measure-weighted transformer	Positional encoding	256-dim. sinusoidal encoding of patch centers
	Self-attention	8 heads, $d_{\text{model}} = 256$, $d_{\text{head}} = 32$
	Feed-forward	SwiGLU, hidden dimension 1024
	Transformer stack	8 pre-norm blocks with RMSNorm
(c) Continuous local expansion decoder	Coefficient prediction	MLP: $256 \to H \to H \to 64\lvert\Lambda_K\rvert$
	Local reconstruction	Legendre expansion: $\lvert\Lambda_K\rvert \times 64 \to 64$ pointwise features
	Project	Pointwise MLP: $64 \to 64 \to 64 \to d_{\text{out}}$

Network architecture. Table 9 summarizes the layer configurations used in the scale-normalized coefficient encoder, measure-weighted transformer, and continuous local expansion decoder. The pointwise feature width is fixed to 64, while each valid patch is represented by a single 256-dimensional transformer token. Let $\Lambda_K = \{\alpha \in \mathbb{N}_0^d : |\alpha| \le K\}$ denote the index set for polynomials of total degree at most K. The local expansion therefore contains $|\Lambda_K| = \binom{d+K}{d}$ basis functions, and we set $H = \max(64|\Lambda_K|, 256)$.

We use SiLU activations between consecutive linear layers in the Lift, coefficient-degree projection, tokenization, coefficient-prediction, and Project MLPs. For time-dependent datasets, the time embedding is added to the lifted point features followed by LayerNorm. After tokenization, an absolute sinusoidal encoding of the patch center is added to each token, followed by LayerNorm. For simplicity, the patch-center dependence of the token embedding is suppressed in the theoretical notation. Adding a shared continuous positional encoding with parameters fixed during refinement preserves consistency under admissible refinement.

Each transformer block consists of RMSNorm pre-normalization, multi-head self-attention, and a SwiGLU feed-forward network with residual connections. A dropout rate of 0.1 is applied to the feed-forward output. For nonuniform patch partitions, we augment the attention logits with the logarithm of the key-patch measure,

$$\frac{q_i^\top k_j}{\sqrt{d_h}} + \log \mu_j. \tag{171}$$

Table 10: Dataset-specific implementation settings. For adaptive partitions, "Base partition" denotes the initial grid before recursive refinement. "Max pts." is the refinement threshold, i.e., a leaf is subdivided when its point count exceeds this value.

Dataset	Maximum total polynomial degree K	Base partition	Partition type	Max pts.
NACA0012	2	16×16	Adaptive	40
RAE2822	2	16×16	Adaptive	40
DeepJEB	2	$8 \times 8 \times 8$	Adaptive	125
1D Burgers	3	32	Fixed	–
CE-RP	3	32×32	Fixed	–
NS-Gauss	3	32×32	Fixed	–

Otherwise, standard scaled dot-product self-attention is used. The transformer output is mapped to a residual correction of the normalized local coefficients. We omit this coefficient residual connection in the input reconstruction experiment reported in Table 2 of Section 3.2.2, where the normalized output coefficients are predicted directly from the encoder tokens without a processor. The resulting coefficients are subsequently denormalized and evaluated using the local Legendre expansion before the final pointwise projection.

Dataset-specific patch configurations. The patch partition and maximum total polynomial degree K used for each dataset are summarized in Table 10. We use fixed uniform-grid partitions for uniformly discretized datasets and point-density-adaptive partitions for nonuniformly discretized datasets. Fixed partitions use the base partitions without refinement. Adaptive partitions start from the specified base partitions and recursively subdivide any cell whose point count exceeds the specified threshold. Each subdivision uniformly splits every spatial axis, corresponding to quadtree refinement in 2D and octree refinement in 3D. Empty children are discarded, and refinement is terminated at a maximum depth of 8. Each final nonempty leaf corresponds to one transformer token; consequently, the number of tokens varies across samples for adaptive partitions. The listed base partition therefore does not denote the final token count.

E Additional Experimental Results

E.1 Source-discretization invariance

We evaluate whether PINOT maintains consistent predictions when the source discretization X_{src} at inference differs from that used during training. For this additional experiment, the model is separately trained on the 1D Burgers dataset using a uniformly spaced source discretization with $N_{\text{src}} = |X_{\text{src}}| = 1{,}024$, and is then evaluated, without retraining, on both coarser and finer unseen source discretizations. For each test input field v, we vary only X_{src}, which changes the observed samples $v|_{X_{\text{src}}}$, while keeping the underlying field v and the patch partition $\mathcal{P}$ fixed. This isolates the effect of changing the source discretization from other variations in the input.

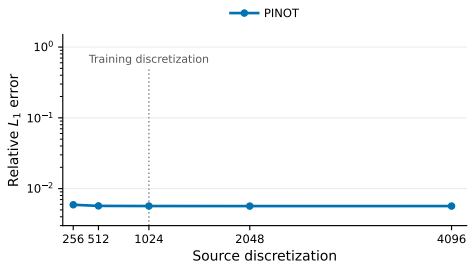

Figure 6: **Source-discretization invariance.** PINOT is trained at $N_{\text{src}} = 1{,}024$ on 1D Burgers and evaluated without retraining at $N_{\text{src}} \in [256, 4{,}096]$. The dashed line marks the training discretization.

Figure 6 reports the relative L_1 error as a function of N_{src}, with the training discretization indicated by the vertical dashed line. The prediction error remains nearly unchanged across all tested discretizations, from $N_{\text{src}} = 256$ to $4{,}096$, despite the model being trained only at $N_{\text{src}} = 1{,}024$. In particular, PINOT retains comparable accuracy when the source point count is decreased by a factor of four or increased by a factor of four. These results provide empirical evidence that a single trained PINOT model can generalize across different source discretizations without retraining.

E.2 Effect of basis choice on patch-partition consistency

We examine how the choice of local basis in PINOT affects the patch-partition invariance. As shown in Figure 7, polynomial bases, including Legendre and Chebyshev polynomials of the first and second kinds, maintain stable relative L_1 errors across changes in patch resolution. In contrast, the Fourier basis exhibits a substantial increase in relative L_1 error as the patch resolution deviates from that used during training. This is consistent with our analysis: the leading patch-scale dependence of polynomial coefficients is removed by the coefficient normalization, whereas the corresponding dependence of Fourier coefficients is not. Thus, patch-partition invariance depends not only on coefficient normalization, but also on the use of basis functions with the required patch-scale behavior.

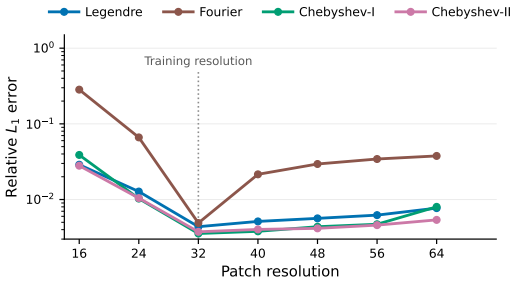

Figure 7: **Effect of basis choice on patch-partition consistency.** PINOT models with different bases are trained on 1D Burgers with $M = 32$ patch tokens and evaluated at different patch resolutions without retraining. Polynomial bases remain stable under partition changes, whereas the Fourier basis exhibits substantially increased relative L_1 error as the patch resolution deviates from the training partition. The dashed line marks the training patch resolution.

F Computational Efficiency and Implementation

F.1 Computational Complexity and Empirical Efficiency

We compare the computational complexity, peak GPU memory, and wall-clock inference time of PINOT with LNO, GAOT, Transolver, OFormer, and GINO. All empirical measurements are performed on NACA0012 using the same model configurations and evaluation setting as in the main experiment (Table 4). We report the dominant representation-mixing cost rather than expanding every pointwise linear layer. Constant channel expansion ratios and pointwise projection or feed-forward costs are omitted uniformly, but dependence on the embedding dimension d_{emb} is retained in the mixing operations. Under this convention, softmax attention over M tokens costs $\mathcal{O}(M^2 d_{\mathrm{emb}})$, Galerkin or factorized attention costs $\mathcal{O}(M d_{\mathrm{emb}}^2)$, and cross-domain assignment between N_{src} points and M latent elements costs $\mathcal{O}(N_{\mathrm{src}} M d_{\mathrm{emb}})$. The empirical memory and latency measurements include all operations omitted from these asymptotic summaries.

Since this experiment uses the same number of source and target points, we write N_{src} for their common size. We use M consistently for the number of elements processed by the global operator: PINOT patches, LNO latent samples, GAOT patch tokens entering the transformer, Transolver slices, and GINO latent grid points. OFormer instead applies Galerkin-type factorized attention directly to the N_{src} physical points, giving $\mathcal{O}(L_{\mathrm{proc}} N_{\mathrm{src}} d_{\mathrm{emb}}^2)$. Here, L_{proc} denotes the number of processor layers.

For GAOT,

$$E_{\mathrm{MAGNO}} = \sum_{s=1}^{S} \left(E_s^{\mathrm{enc}} + E_s^{\mathrm{dec}} \right) \tag{172}$$

counts the encoder and decoder edges over all S MAGNO scales. Thus, the scale count affects the graph cost, while ViT-style patchification changes M and hence the quadratic transformer cost. For GINO, E_{GINO} denotes the total number of edges in the encoder and decoder graphs, and we use M to denote the number of latent FNO grid points rather than introducing a separate grid-size symbol. With the Fourier truncation fixed, we report the $O(M \log M d_{\mathrm{emb}})$ cost of FFT and inverse FFT operations and omit Fourier-mode channel mixing from this complexity summary. For PINOT, as detailed in Sec. F.2, with the spatial dimension d and maximum total polynomial degree K fixed and the padding overhead bounded, its local and token-interface operations reduce to $\mathcal{O}(N_{\mathrm{src}} d_{\mathrm{emb}})$, followed by measure-weighted softmax attention over M patch tokens.

Table 11: Computational complexity and empirical inference efficiency on NACA0012. The computational complexity reports dominant representation-mixing costs; pointwise projection and feed-forward terms are suppressed uniformly, while dependence on d_{emb} is retained. Peak GPU memory and inference time are measured with batch size $B = 1$ under the same setting as the main experiment. The best result is shown in bold and the second-best is underlined.

Method	Computational complexity	Peak memory (MB)	Time (ms)
GINO	$\mathcal{O}(E_{\text{GINO}} d_{\text{emb}} + L_{\text{proc}} M \log M \, d_{\text{emb}})$	166.9	**10.9**
OFormer	$\mathcal{O}\big(L_{\text{proc}} N_{\text{src}} d_{\text{emb}}^2\big)$	267.5	13.8
GAOT	$\mathcal{O}\big(E_{\text{MAGNO}} d_{\text{emb}} + L_{\text{proc}} M^2 d_{\text{emb}}\big)$	<u>39.9</u>	23.1
Transolver	$\mathcal{O}\big(L_{\text{proc}}(N_{\text{src}} M d_{\text{emb}} + M^2 d_{\text{emb}})\big)$	62.5	14.4
LNO	$\mathcal{O}\big(N_{\text{src}} M d_{\text{emb}} + L_{\text{proc}} M d_{\text{emb}}^2\big)$	70.5	12.9
PINOT (Ours)	$\mathcal{O}\big(N_{\text{src}} d_{\text{emb}} + L_{\text{proc}} M^2 d_{\text{emb}}\big)$	**37.2**	<u>11.1</u>

Measurement protocol. Peak GPU memory and inference time are measured over 100 distinct NACA0012 test samples with a batch size of one ($B = 1$) on an RTX A6000 GPU. Each sample is evaluated once, and inference time is reported as the mean across 100 samples. The model is executed in evaluation mode without gradient computation. CUDA execution is explicitly synchronized when measuring inference latency. Data loading, preprocessing, and host-to-device transfer are excluded. Peak GPU memory is measured using `torch.cuda.max_memory_allocated()` and reported as the mean peak additional allocation observed across the 100 forward passes.

Results and analyses. Table 11 compares the theoretical complexity of representation mixing with the empirical inference efficiency of the evaluated methods. PINOT achieves the lowest peak GPU memory usage and the second-fastest inference time, requiring 37.2 MB and 11.1 ms per sample, respectively. In terms of memory, PINOT slightly improves over GAOT, while requiring substantially less memory than Transolver, LNO, OFormer, and GINO. The inference time of PINOT is close to that of GINO (10.9 ms), which uses an FNO-based processor. Among all evaluated methods with transformer-based processors, PINOT achieves the fastest inference time.

The relatively large memory footprints of GINO and OFormer arise from different aspects of their representations. For GINO, NACA0012 provides a highly nonuniform point-cloud discretization, and the neighborhood graph used by its geometric integral operator is constructed directly from the input geometry during inference rather than supplied as a precomputed structure. Constructing and retaining the resulting neighborhood relations introduces substantial additional memory overhead on this nonuniform discretization. In contrast, OFormer does not first compress the input into a smaller latent space, but instead performs its attention-based processing directly over the N_{src} source points. Its activation memory therefore remains directly tied to the size of the source discretization, resulting in a considerably larger memory footprint than methods operating on compact latent representations.

These results highlight a practical advantage of PINOT's compact patch-level representation. PINOT first compresses the N_{src} input observations into M local patch tokens and performs global interaction in this reduced representation. Consequently, when $M \ll N_{\text{src}}$, the dominant transformer processing cost depends on the number of patches rather than directly on the input discretization size. On the NACA0012 benchmark with nonuniform discretizations, this design translates into favorable practical memory usage and inference latency while retaining the predictive performance reported in the main experiments.

Another practical advantage of PINOT is that it constructs physically grounded latent representations while retaining dense batching across heterogeneous source discretizations. In contrast, graph-based latent mappings, as used in GINO and GAOT, rely on sample-dependent neighborhood structures, making them difficult to fully tensorize when the input discretization varies across samples. Although our efficiency measurements use $B = 1$, the implementation may provide additional benefits in batched training or inference.

F.2 Efficient Batched Implementation of the Coefficient Encoder

A naïve implementation of the coefficient encoder would process each sample and patch independently, resulting in a large number of small matrix operations. Although these local projection problems are mutually independent, executing them sequentially leads to poor GPU utilization and substantial overhead from kernel launches and Python loops. We therefore implement the coefficient encoder in a fully batched manner, jointly vectorizing the projection over both samples and patches.

Let

$$L_{\mathrm{src}} \in \mathbb{R}^{B\times N_{\mathrm{src}}\times d_{\mathrm{lift}}} \tag{173}$$

denote the sampled lifted features $L(x)$ for a mini-batch of size B, where N_{src} is the maximum source point count in the batch and d_{lift} is the dimension of $L(x)$. Samples with fewer source points are padded and masked. Given the assignment of each source point to a patch $P_i \in \mathcal{P}$, we rearrange these features into

$$L_{\mathrm{pack}} \in \mathbb{R}^{B\times M\times n_{\max}\times d_{\mathrm{lift}}}, \tag{174}$$

where M is the maximum number of patches among samples in the mini-batch and $n_{\max}$ is the maximum number of source points contained in any patch across all samples in the mini-batch. For each patch, we write

$$n_i = |X_i^{\mathrm{src}}| = |P_i \cap X_{\mathrm{src}}|. \tag{175}$$

Here and below, sample indices on patch-local quantities are suppressed when unambiguous. Samples containing fewer than M patches and patches containing fewer than $n_{\max}$ source points are zero-padded, and corresponding binary masks exclude padded entries from subsequent computations. For the non-overlapping, label-based patch assignments used in our experiments, the packing indices are constructed through vectorized tensor indexing, eliminating the need for explicit loops over samples or patches.

In our experiments, the prescribed maximum total degree K determines the full degree-ordered basis $\{\psi_\alpha\}_{\alpha\in\Lambda_K}$. When a patch contains too few source points to use all $|\Lambda_K|$ coefficients, we activate only the first $\min(n_i, |\Lambda_K|)$ modes in this fixed degree order. Let $a_{b,i,\alpha} \in \{0,1\}$ denote the resulting active-mode mask, where $a_{b,i,\alpha} = 1$ if mode α is active for the corresponding patch and 0 otherwise. This prevents a patch from activating more basis modes than the number of available source observations. We refer to this restriction on active basis modes as rank limiting. This is used only as a finite-resolution numerical safeguard; the analysis in Appendix C assumes admissible refinements for which the full prescribed basis is supported and the local design matrices remain uniformly well conditioned.

We form the basis matrix

$$\Phi \in \mathbb{R}^{B\times M\times n_{\max}\times|\Lambda_K|}, \tag{176}$$

where $\Lambda_K = \{\alpha \in \mathbb{N}_0^d : |\alpha| \le K\}$ indexes the polynomial basis functions of total degree at most K, with $|\Lambda_K| = \binom{d+K}{d}$. For each valid point and active mode,

$$\Phi_{b,i,j,\alpha} = \psi_\alpha(N_i(x_{i,j})), \tag{177}$$

while entries corresponding to padded points or inactive modes are set to zero. Thus, inactive modes appear as zero columns in Φ, while all patches retain the same fixed basis dimension $|\Lambda_K|$ for batched computation. To represent these inactive coordinates within the fixed-size linear system, we add an additional unit diagonal through $\mathrm{diag}(1-a)$. The local coefficient estimation is therefore

$$G = \Phi^\top\Phi + \lambda\widetilde{n}I + \mathrm{diag}(1-a), \qquad Q = \Phi^\top L_{\mathrm{pack}}, \qquad C^{\mathrm{in}} = \mathrm{solve}(G, Q). \tag{178}$$

Here, the transpose is taken over the point and basis-mode dimensions for each patch. For active modes, $\mathrm{diag}(1-a)$ contributes zero, whereas each inactive mode receives an additional unit diagonal entry and no data-dependent Gram contribution. The corresponding inactive coefficients are set to zero after solving. Here, $C^{\mathrm{in}}_{b,i,\alpha,:}$ stores the fitted input coefficient, and $\widetilde{n}_i = \max(n_i, 1)$ is broadcast over the corresponding Gram matrix. Positive regularization $\lambda > 0$ makes each system positive definite in exact arithmetic, including for empty padded patches. We use this ridge term as a numerical safeguard for coefficient fitting at finite resolution, and this term is not required for a unique least-squares solution when the local design matrix has full column rank.

We do not explicitly compute G^{-1}. Instead, all local systems are solved simultaneously using batched Cholesky factorization followed by batched Cholesky solves. The principal tensor shapes are

$$\begin{aligned} \Phi : [B, M, n_{\max}, |\Lambda_K|], \qquad L_{\mathrm{pack}} : [B, M, n_{\max}, d_{\mathrm{lift}}], \\ G : [B, M, |\Lambda_K|, |\Lambda_K|], \qquad Q : [B, M, |\Lambda_K|, d_{\mathrm{lift}}], \qquad C^{\mathrm{in}} : [B, M, |\Lambda_K|, d_{\mathrm{lift}}]. \end{aligned} \tag{179}$$

Thus, both the sample and patch axes are treated as batch dimensions. The products $\Phi^\top \Phi$ and $\Phi^\top L_{\mathrm{pack}}$ are evaluated using batched tensor contractions, and the resulting systems for all (b, i) pairs are solved in parallel.

Batched tokenization. Before tokenizing coefficients, each fitted coefficient $c^{\mathrm{in}}_{i,\alpha}$ is normalized according to Eq. (5), producing $\tilde{c}^{\mathrm{in}}_{i,\alpha} = s_{i,\alpha} c^{\mathrm{in}}_{i,\alpha}$.

Subsequently, the corresponding basis-degree features are concatenated with each per-mode coefficient representation and projected back to dimension d_{lift} using a shared MLP. The resulting coefficient tensor is then reshaped as

$$C^{\mathrm{in}}_{\mathrm{feat}} \in \mathbb{R}^{B \times M \times (|\Lambda_K| d_{\mathrm{lift}})}. \tag{180}$$

A shared tokenization MLP is applied along the final dimension. Patch-center absolute positional embeddings are then added, followed by layer normalization, to obtain the initial transformer tokens

$$Z^{(0)} = [z_i^{(0)}]_{i=1}^{M} \in \mathbb{R}^{B \times M \times d_{\mathrm{emb}}}. \tag{181}$$

Since the leading $[B, M]$ dimensions are treated as batch dimensions by the linear layers, the projection is effectively executed as a large matrix operation over all samples and patches. The resulting tokens are then passed to the transformer, with invalid patches excluded using an attention mask.

After L_T measure-weighted transformer layers, the processed tokens are denoted by $Z^{(L_T)} = [z_i^{(L_T)}]_{i=1}^{M}$. A shared coefficient prediction MLP then predicts a residual in the normalized-coefficient space. The residual is added to the corresponding coefficient representation, after which the coefficient normalization $s_{i,\alpha}$ is inverted to recover the batched output coefficients $C^{\mathrm{out}} = [c^{\mathrm{out}}_{i,\alpha}]_{i,\alpha}$ before function expansion.

For the implementation and cost analysis below, we consider function expansion at the source locations, for which the corresponding pointwise representation is obtained by the batched tensor contraction

$$W_{\mathrm{rec}} = \Phi C^{\mathrm{out}}. \tag{182}$$

The local basis projection and shared tokenization are mathematically equivalent to applying the same operations independently to each valid patch. The measure-weighted transformer subsequently models interactions among the resulting patch tokens, while the independent local operations are carried out as large batched tensor computations that are well suited to parallel GPU execution.

Geometry caching. For datasets in which the same spatial geometry and patch partition are repeated across samples, geometry-dependent quantities do not need to be recomputed for every mini-batch. After verifying that the source coordinates, patch assignments, and patch ranges are identical, we cache the packing layout and masks, normalized local coordinates $\xi_{i,j} = N_i(x_{i,j})$, patch centers r_i, scales h_i, measures μ_i, the masked basis matrices Φ, the Cholesky factor of the regularized Gram system, and the active-mode mask associated with this per-patch truncation of the degree-ordered basis. The Gram matrix itself is not retained after factorization.

For subsequent samples sharing the same verified geometry, these cached quantities are reused, while the input-dependent right-hand side

$$Q = \Phi^\top L_{\mathrm{pack}} \tag{183}$$

and the corresponding coefficient solve are recomputed. This eliminates repeated packing setup, coordinate normalization, basis evaluation, Gram-matrix construction, and Cholesky factorization for identical geometries.

Grouped batching. Mixing substantially different patch configurations within the same mini-batch can increase $n_{\max}$, the maximum patch occupancy in the mini-batch, and the number of padded entries. During training, samples with the same partition configuration are therefore assigned to the same batching group. In our implementation, this grouping is determined by the partition configuration associated with each sample. Importantly, sharing the same batching group does not by itself guarantee that geometry caching is valid. Caching is enabled only after separately verifying that the source coordinates, patch assignments, and patch ranges are identical within the group. Grouped batching can therefore reduce padding while increasing the opportunity for safe reuse of geometry-dependent quantities.

Computational cost and memory trade-off. For the following cost analysis, assume that each sample contains the same number N_{src} of valid source points. Because the patch assignments are non-overlapping in our experiments, it follows that $\sum_{i=1}^{M} n_i = N_{\mathrm{src}}$. Padding each of the M patches to $n_{\max}$ points allocates $Mn_{\max}$ local point slots per sample. The resulting padding overhead is $\rho_{\mathrm{pad}} = Mn_{\max}/N_{\mathrm{src}} \geq 1$, or equivalently $Mn_{\max} = \rho_{\mathrm{pad}} N_{\mathrm{src}}$.

Before geometry caching, the basis-projection and reconstruction cost is

$$\mathcal{O}\big(B\rho_{\mathrm{pad}} N_{\mathrm{src}} |\Lambda_K|^2 + B\rho_{\mathrm{pad}} N_{\mathrm{src}} |\Lambda_K| d_{\mathrm{lift}} + BM|\Lambda_K|^3 + BM|\Lambda_K|^2 d_{\mathrm{lift}}\big)\,, \tag{184}$$

where the four terms of Eq. (184) correspond to Gram-matrix construction, right-hand-side formation and basis reconstruction, Cholesky factorization, and Cholesky solves. These expressions count the projection and reconstruction operations; packing-layout construction, including sorting, is excluded. Basis evaluation and masking do not increase the displayed order for fixed spatial dimension and polynomial degree. When geometry is cached, the basis matrices and Cholesky factors are reused, avoiding Gram-matrix construction and factorization and leaving

$$\mathcal{O}\big(B\rho_{\mathrm{pad}} N_{\mathrm{src}} |\Lambda_K| d_{\mathrm{lift}} + BM|\Lambda_K|^2 d_{\mathrm{lift}}\big)\,. \tag{185}$$

For fixed basis size, feature widths, and MLP depths, the shared coefficient mappings, including the per-mode degree-feature projection, add $\mathcal{O}(BM)$ operations. The L_T-layer measure-weighted transformer adds

$$\mathcal{O}\big(BL_T(M^2 d_{\mathrm{emb}} + M d_{\mathrm{emb}}^2)\big)\,, \tag{186}$$

since measure weighting does not change the attention-matrix dimensions. Here, the spatial dimension d, maximum total polynomial degree K, feature widths, and network depths are fixed; hence $|\Lambda_K| = \binom{d+K}{d}$ is fixed. Assuming bounded ρ_{pad}, all local terms scale linearly with N_{src} or M.

For a per-sample comparison with cached geometry, after consistently omitting pointwise projections and feed-forward networks as in Table 11, the retained cost is

$$\mathcal{O}\big(\rho_{\mathrm{pad}} N_{\mathrm{src}} |\Lambda_K| d_{\mathrm{lift}} + M|\Lambda_K|^2 d_{\mathrm{lift}} + L_T M^2 d_{\mathrm{emb}}\big)\,. \tag{187}$$

With fixed basis size, bounded padding overhead ρ_{pad}, $M = \mathcal{O}(N_{\mathrm{src}})$, and $d_{\mathrm{lift}} = \mathcal{O}(d_{\mathrm{emb}})$, this reduces to the compact cost

$$\mathcal{O}(N_{\mathrm{src}} d_{\mathrm{emb}} + L_T M^2 d_{\mathrm{emb}}). \tag{188}$$

Batching changes hardware utilization, but not the arithmetic order of these local linear-algebra operations. Excluding model parameters, optimizer states, and intermediate activations of the coefficient MLPs and transformer, the packed projection and reconstruction tensors occupy

$$\mathcal{O}\big(B\rho_{\mathrm{pad}} N_{\mathrm{src}} (|\Lambda_K| + d_{\mathrm{lift}}) + BM(|\Lambda_K|^2 + |\Lambda_K| d_{\mathrm{lift}})\big) \tag{189}$$

memory. Thus, larger variation in patch sizes increases ρ_{pad} and peak memory, while batched tensor contractions improve GPU utilization.