LOCAL BASIS NEURAL FIELDS: SCALABLE SCIENTIFIC SIGNAL FITTING WITH EXPLICIT ORDER OF ACCURACY FOR VALUES AND DERIVATIVES

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural field representations are increasingly used to represent, visualize, and analyze large-scale scientific signals, such as astronomical simulations and turbulent flows. These applications demand scalable representations that are accurate not only in reconstructed values, but also under differentiation. However, purely MLP-based networks scale poorly to large signals, while hybrid methods that pair MLPs with grid-based features (e.g., hash grids) introduce significant artifacts under differentiation. To bridge this gap, we introduce Local Basis Neural Fields (LBNFs), a framework that represents a signal using a set of overlapping local polynomial functions placed adaptively across the domain, paired with a lightweight MLP that decodes these local features into signal values. The polynomial degree and the placement density together give LBNFs explicit, independent control over modeling complexity and its allocation around complex signal structures. Because LBNFs build on classical partition-of-unity (PoU) approximation theory, they inherit its convergence guarantees: for smooth targets, this yields $O(h^{p+1})$ convergence in value and $O(h^{p+1-\nu})$ convergence for derivatives of order $\nu \leq p$, where h is the spacing between polynomials and p their degree. Empirically, LBNFs realize these hp-approximation rates on analytic benchmarks while achieving significantly lower absolute error than classical approximants. On large-scale scientific data, LBNFs scale comparably to state-of-the-art neural field methods (e.g., Instant-NGP and NeuRBF), matching or improving their reconstruction error while reducing first- and second-order gradient errors by an order of magnitude. Code found here: `https://anonymous.4open.science/r/LBNFs-anon`

1 INTRODUCTION

Scientific domains ranging from cosmology to fluid dynamics routinely generate massive, high-dimensional signals, such as simulated density fields and turbulent velocity fields, whose scale makes traditional discrete grid representations costly to store, visualize, and analyze. Neural fields, or implicit neural representations (INRs), have emerged as a compelling continuous alternative for representing such signals (Sun et al., 2024; Zorek et al., 2026; Wurster et al., 2023; Lu et al., 2021; Weiss et al., 2022). Rather than storing values on a rigid grid, a neural field approximates a signal $f : \mathbb{R}^d \to \mathbb{R}^c$ mapping a d-dimensional input coordinate to a c-valued vector with a neural network $\hat{f}_\Theta$ that can be queried at arbitrary spatial coordinates, often using far fewer parameters Θ than raw data samples. Standard neural fields are fit to observed values and evaluated primarily in terms of reconstruction accuracy and compression. In many scientific applications, however, signal values alone are insufficient; quantities derived from spatiotemporal derivatives are equally central to analysis. For example, fluid dynamicists identify vortical structures via vorticity and the Q-criterion, cosmologists analyze gradients and Laplacians of density and gravitational potential fields, and volume rendering relies on gradient-based normals for shading and isosurface extraction. Since neural fields are supervised only on signal values, derivative accuracy is never directly optimized but must instead emerge as a byproduct of the representation itself. A neural field for scientific data must therefore scale efficiently to large signals while still yielding accurate derivatives from value supervision alone.

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.

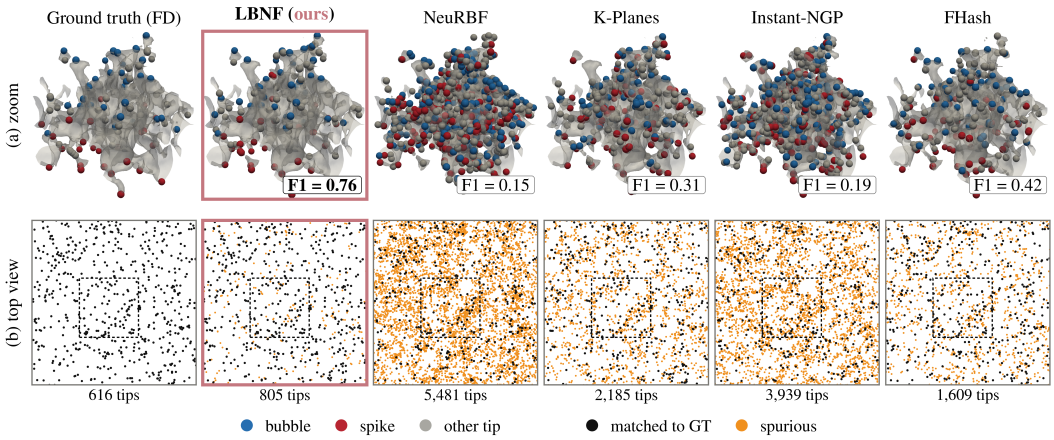

Figure 1: **LBNFs enable accurate gradients for scientific analysis.** We locate bubble and spike tips in a **Rayleigh–Taylor** mixing layer, a canonical turbulent-mixing problem, to characterize mixing-region growth. Tips are points on the $\rho = 0.5$ isosurface where horizontal density gradient vanishes (App. D.4.1). (a) Zoomed window with detected tips: bubbles (light fluid rising), spikes (heavy fluid falling), and other extrema. (b) All interior tips viewed from above: black if within 2 voxels of a reference tip and orange if spurious. **F1** (↑) uses the finite-difference reference. **LBNF** recovers the reference structure with substantially fewer spurious detections than baselines.

However, existing neural fields fail to simultaneously achieve scalability and high-order derivative accuracy. Purely MLP-based representations offer infinite differentiability, but because their parameters have global support, their capacity and training costs scale poorly with signal size (Martel et al., 2021). Hybrid methods address scaling by pairing a lightweight MLP with spatially organized feature encodings, such as dense or hashed grids (Müller et al., 2022; Sun et al., 2026) or planar factorizations (Fridovich-Keil et al., 2023). While computationally efficient, these features rely on low-order interpolation: first derivatives are discontinuous across cell boundaries, second derivatives vanish or are ill-defined within cells, and spatial hash collisions introduce non-local high-frequency noise under differentiation. Unstructured RBF-based neural fields such as NeuRBF (Chen et al., 2023) relax rigid grid constraints, but use signal-dependent initialization to allocate their bases. More broadly, existing RBF-based neural encodings do not establish an explicit relationship between representation complexity and approximation order for values and derivatives. Compounding these practical limits, the theoretical error behavior of scalable neural fields remains largely uncharacterized, as existing convergence analyses focus almost exclusively on MLP-only designs (Roddenberry et al., 2023; Yüce et al., 2022; DeVore et al., 2020; Opschoor & Schwab, 2024). Consequently, there is a clear need for a neural field framework that scalably fits large signals while providing accurate derivatives with a predictable, controllable order of accuracy.

To resolve this fundamental tradeoff, we introduce *Local Basis Neural Fields* (LBNFs), a framework that bridges classical polynomial approximation theory with scalable neural representations. LBNFs decompose signal representation into two stages. First, they map input coordinates to local polynomial feature vectors anchored at spatial locations and aggregated via Gaussian partition-of-unity (PoU) weights. Second, a lightweight MLP decodes these aggregated features into signal values. This formulation provides explicit control over local polynomial approximation degree p and spatial resolution h (governed by anchor count N), the two parameters that govern LBNFs' provable orders of accuracy. Crucially, LBNF does not constrain anchors to a rigid grid. Instead, anchor locations, polynomial basis coefficients, PoU shape parameters, and decoder weights are optimized jointly end-to-end, enabling the representation to adaptively concentrate capacity around complex, high-frequency signal features.

On large-scale scientific datasets, LBNFs improve reconstruction accuracy over the evaluated baselines, such as Instant-NGP, K-Planes, and NeuRBF. In addition, LBNFs match the derivative accuracy of smooth MLPs, where grid-based methods are an order of magnitude worse, while retaining the computational cost of grid-based methods. These improvements support gradient-based

scientific analysis, from surface-normal estimation to identifying features of a mixing front. In the **Rayleigh-Taylor** example, LBNFs substantially reduce spurious bubble and spike detections (Fig. 1). Finally, we connect LBNFs to classical hp-approximation theory, establishing explicit approximation orders for values and derivatives. We show empirically that LBNFs exhibit convergence consistent with these orders on analytic functions.

2 Related Work

Neural fields range from globally parameterized smooth MLPs such as SIREN (Sitzmann et al., 2020) and WIRE (Saragadam et al., 2023), to spatially decomposed models such as MINER (Saragadam et al., 2022), to scalable hybrid encodings. The latter include grid-based methods such as Instant-NGP (Müller et al., 2022), F-Hash (Sun et al., 2026), and K-Planes (Fridovich-Keil et al., 2023), and RBF methods (Chen et al., 2023; Kairanda et al., 2026). As discussed in Section 1, these hybrids trade global capacity for spatially localized features at the cost of derivative accuracy. The most closely related work to LBNFs is NeuRBF (Chen et al., 2023), which places anisotropic RBFs throughout a signal's domain via weighted k-means on signal-derived importance weights (e.g., gradient magnitude) and holds their positions and shapes fixed during training. ∂^∞-Grid (Kairanda et al., 2026) instead recovers smooth derivatives by Gaussian RBF interpolation of features on a fixed grid. These works do not generally connect tunable representation parameters to explicit refinement rates for values and derivatives, as we do with LBNFs. We optimize basis locations end-to-end and treat approximation order and spatial resolution as explicit refinement controls linked to classical approximation rates. Such rates provide a principled way to reason about parameter allocation with respect to accuracy–scale tradeoffs. LBNFs inherit these approximation properties by building on classical partition-of-unity and hp-approximation methods (Melenk & Babuška, 1996; Babuška & Suri, 1994), in which local polynomial degree and spatial resolution jointly determine convergence rates. PoUnets (Lee et al., 2021) learn partition functions with local polynomials whose coefficients are obtained through a globally coupled least-squares problem, which can become increasingly expensive and poorly conditioned as the number of partitions grows. LBNFs instead learn local polynomial features and a nonlinear neural decoder jointly by gradient descent, avoiding this separate global solve while adding nonlinear representational capacity for large, complex scientific signals.

3 Methods

Let $\Omega \subset \mathbb{R}^d$ denote a bounded coordinate domain and let $f : \Omega \to \mathbb{R}^c$ denote a target signal. Our goal is to fit an accurate coordinate-based neural approximation $\hat{f}_\Theta$ of f over Ω using a compact set of parameters Θ. We consider the standard signal fitting setting, in which only samples of the signal itself, $\{(\mathbf{x}_i, f(\mathbf{x}_i))\}_{i=1}^{N_s}$, are observed, with no supervision on derived quantities such as gradients. In this setting, the representation is fit end-to-end by minimizing the empirical loss

$$\mathcal{L}(\Theta) = \frac{1}{N_s} \sum_{i=1}^{N_s} \ell\Big(\hat{f}_\Theta(\mathbf{x}_i), f(\mathbf{x}_i)\Big)\,, \tag{1}$$

where $\ell(\cdot,\cdot)$ is a per-sample reconstruction loss, such as the squared ℓ_2 error.

We propose *Local Basis Neural Fields* (LBNFs), a scalable and controllable neural parameterization of $\hat{f}_\Theta$. Intuitively, an LBNF encodes an input coordinate $\mathbf{x}$ with features computed from a set of locally defined polynomials, and then decodes those features into an output value with a small MLP. Concretely, an LBNF defines features throughout Ω by a set of N "anchor" points $\mathcal{A} = \{(\mathbf{a}_i, \mathbf{B}_i)\}_{i=1}^{N}$, where $\mathbf{a}_i \in \Omega$ is the position of the ith anchor and $\mathbf{B}_i$ holds the coefficients of its P polynomials. Evaluated at an input coordinate $\mathbf{x}$, an anchor's polynomials yield P local features. The LBNF aggregates features from anchors near $\mathbf{x}$ and passes them to a small MLP decoder D_θ with parameters θ, which maps them to an output signal value. Formally,

$$\hat{f}_\Theta(\mathbf{x}) = D_\theta(E_{\mathcal{A},\phi}(\mathbf{x}))\,, \qquad \Theta = \{\mathcal{A}, \theta, \phi\}, \tag{2}$$

where $E_{\mathcal{A},\phi} : \Omega \to \mathbb{R}^F$ is an encoding that combines the polynomial features of nearby anchors using a neighborhood-weighting function parameterized by ϕ.

In the following sections, we describe LBNF design details (Sec. 3.1), an adaptive anchor placement strategy (Sec. 3.2), and approximation properties (Sec. 3.3).

3.1 LBNF DESIGN

The main design choices concerning LBNFs are the local polynomials stored in each anchor point, and the aggregation strategy to combine features from local anchors. We first assume that anchors are placed on a uniform grid in this section, and then relax this assumption to allow anchor positions to also be optimized in Sec. 3.2.

3.1.1 LOCAL POLYNOMIAL FEATURES

Consider anchor i located at $\mathbf{a}_i$ storing P polynomials, each of which has degree p. Each polynomial consists of all possible individual and joint product terms of coordinate dimensions, yielding $\binom{p+d}{d}$ coefficients. More formally, let $\boldsymbol{\delta} = \mathbf{x} - \mathbf{a}_i$ denote the relative location of coordinate $\mathbf{x}$ with respect to the anchor. Define the multi-index $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d) \in \mathbb{N}_0^d$, $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_d$, and the monomial $\boldsymbol{\delta}^{\boldsymbol{\alpha}} = \delta_1^{\alpha_1}\delta_2^{\alpha_2}\cdots\delta_d^{\alpha_d}$. Then, the jth polynomial of this anchor has the form:

$$q_{i,j}(\boldsymbol{\delta}) = \sum_{|\boldsymbol{\alpha}| \le p} b_{i,j,\boldsymbol{\alpha}}\, \boldsymbol{\delta}^{\boldsymbol{\alpha}}, \tag{3}$$

where the sum runs over all $\binom{p+d}{d}$ multi-indices satisfying $|\boldsymbol{\alpha}| \le p$. Evaluating all P polynomials at $\mathbf{x}$ yields a P-dimensional local feature vector, which we denote $\mathbf{q}_i(\mathbf{x}) \in \mathbb{R}^P$, contributed by anchor i. We subsequently aggregate features from neighboring anchors and decode them into one signal value, as described in the following section.

3.1.2 ANCHOR WEIGHTING, FEATURE AGGREGATION, AND VALUE PREDICTION

Anchor weighting. Given a query coordinate $\mathbf{x}$, we aim to aggregate the P features from all local anchor points. We first find the K nearest neighbor anchors in Ω to $\mathbf{x}$, which we denote $\mathcal{N}_K(\mathbf{x})$, with $K \ll N$ to reduce computational cost. We then weight the contribution of all anchors in $\mathcal{N}_K(\mathbf{x})$, with a nonzero weighting function parameterized by ϕ that monotonically decreases with distance to $\mathbf{x}$. In our experiments we use a Gaussian kernel:

$$\psi_i(\mathbf{x}; \phi_i) = \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{a}_i)^\top \boldsymbol{\Sigma}_i(\phi_i)^{-1}(\mathbf{x} - \mathbf{a}_i)\right], \qquad \boldsymbol{\Sigma}_i(\phi_i) \succ 0. \tag{4}$$

We consider both the full covariance case where $\boldsymbol{\Sigma}_i(\phi_i) = \mathbf{L}_i\mathbf{L}_i^\top$ and the isotropic case where $\boldsymbol{\Sigma}_i(\phi_i) = \sigma_i^2\mathbf{I}$, with ϕ_i denoting the parameters that determine $\boldsymbol{\Sigma}_i$.

$$w_i^{(K)}(\mathbf{x}) = \frac{\psi_i(\mathbf{x})}{\sum_{j \in \mathcal{N}_K(\mathbf{x})} \psi_j(\mathbf{x})}, \qquad i \in \mathcal{N}_K(\mathbf{x}), \tag{5}$$

so that $\sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i^{(K)}(\mathbf{x}) = 1$ and $w_i^{(K)}(\mathbf{x}) \ge 0$, forming a partition of unity at $\mathbf{x}$.

Aggregated local encoding. We combine the evaluated local feature vectors by *weighted summation* or *concatenation*. Summation imposes a pooled partition-of-unity representation with fixed feature dimension P, whereas concatenation preserves anchor-specific local features at the cost of a decoder input dimension that grows linearly with K. Under equivalent parameter settings, we find that summation maintains higher-order accuracy on smoother signals, while concatenation performs better on more complex signals, such as the scientific data. The decoder maps the encoding $E_{\mathcal{A},\phi}(\mathbf{x})$ to a smooth MLP decoder D_θ defined in equation 2. Unless otherwise stated, we use a small fully connected MLP with smooth $\tanh$ activations and a linear output layer. The encoder construction exposes three primary approximation controls: the local polynomial order p, the anchor resolution h (or equivalently anchor quantity N), and the aggregation locality K. The feature width F provides an additional learned capacity parameter without changing the local polynomial order. In the special case $F = 1$, $K = N$, and D_θ equal to the identity, equation 2 reduces to a classical local-polynomial partition-of-unity approximant.

3.2 ADAPTIVE ANCHOR PLACEMENT

LBNFs can also treat the anchor locations as trainable parameters and optimize them jointly with their local coefficients and the decoder. The goal of anchor relocation is to concentrate representation

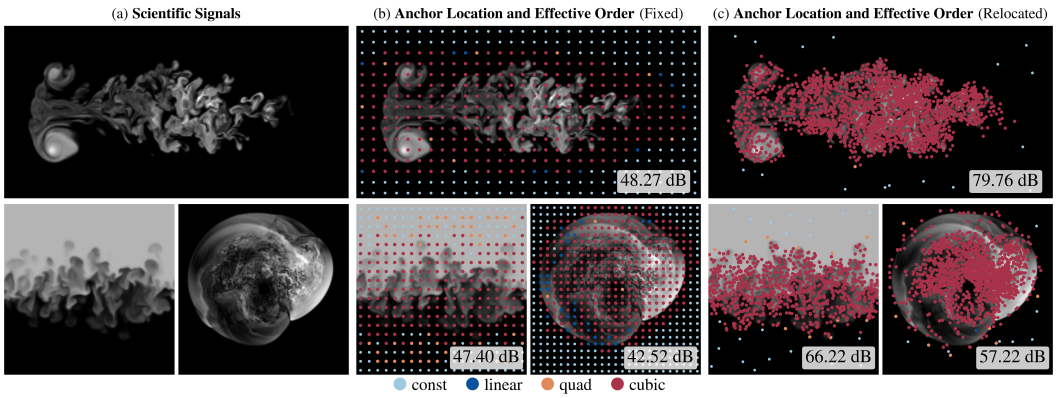

Figure 2: **Trained Anchor Locations** colored by effective order (D.3). **(a)** Target signals and anchor positions after training **(b)** fixed and **(c)** relocated. Relocation concentrates high-order anchors at signal structure and raises PSNR (insets) by 15–32 dB at equal parameter count.

capacity near complex signal boundaries, where approximation error is typically largest. We empirically find that this continuous relocation allows the anchor geometry to subtly refine around signal structure without changing the number of basis functions. In addition, to permit larger changes in spatial allocation while preserving the model size, we incorporate periodic split-and-merge updates. Given randomly subsampled training points $\{(\mathbf{x}_s, \mathbf{y}_s)\}$, we assign each anchor a residual score:

$$e_i = \sum_s \mathbb{I}[i \in \mathcal{N}_K(\mathbf{x}_s)]\, w_i^{(K)}(\mathbf{x}_s) \left\| \hat{f}_\Theta(\mathbf{x}_s) - \mathbf{y}_s \right\|_2^2, \tag{6}$$

representing the weighted reconstruction error associated with anchor i. At a relocation step, we split a subset of anchors with largest e_i. For a selected anchor $\mathbf{a}_i$, we initialize two children with positions:

$$\mathbf{a}_i^{\pm} = \mathbf{a}_i \pm \epsilon\, \mathbf{u}_i, \qquad \|\mathbf{u}_i\|_2 = 1, \tag{7}$$

with the parent's local coefficients copied to both children as a warm start. In our implementation, $\mathbf{u}_i$ is sampled randomly and we scale ϵ with the mean anchor spacing. To keep the total parameter count fixed, we pair each split with a merge among low-error anchors. Defining the weight mass

$$m_i = \sum_s \mathbb{I}[i \in \mathcal{N}_K(\mathbf{x}_s)]\, w_i^{(K)}(\mathbf{x}_s),$$

we replace pair (i, j) with the mass-weighted anchor

$$\mathbf{a}_{ij} = \frac{m_i \mathbf{a}_i + m_j \mathbf{a}_j}{m_i + m_j}, \tag{8}$$

with its local coefficients merged analogously. We select merge pairs from a low-error candidate pool and preferentially merge spatially nearby anchors. This method reallocates capacity from low-error to high-error regions. The resulting anchors form a corset around non-trivial signal boundaries, concentration capacity near structurally complex regions and improving accuracy (Figure 2).

3.3 Approximation Properties

The explicit polynomial structure of LBNFs allows their approximation properties to be related directly to classical partition-of-unity (PoU) approximation. We separate the approximation capacity of the representation from optimization: first, we show that the full LBNF hypothesis class can inherit the error of a polynomial PoU approximant up to a decoder approximation term; we then combine this result with the classical smooth-function PoU rate. For clarity, we state the results for scalar-valued targets; the extension to multiple output channels is componentwise.

For a finite anchor set $\mathcal{A} = \{\mathbf{a}_i\}_{i=1}^N \subset \Omega$, we characterize its spatial resolution by the fill distance

$$h = \sup_{\mathbf{x} \in \Omega} \min_{1 \le i \le N} \|\mathbf{x} - \mathbf{a}_i\|_2, \tag{9}$$

and the separation distance

$$q = \frac{1}{2} \min_{i \neq j} \|\mathbf{a}_i - \mathbf{a}_j\|_2. \tag{10}$$

We call a family of anchor sets quasi-uniform if its mesh ratio $\rho = h/q$ remains bounded as $h \to 0$. For a regular M^d anchor grid, $N = M^d$ and $h \asymp M^{-1}$.

Lemma 1 (LBNF approximation inheritance)**.** *Let $S_h f$ be a scalar polynomial PoU approximant realizable by one LBNF feature channel, and define*

$$\mathbf{z}_h(\mathbf{x}) = \big(S_h f(\mathbf{x}), 0, \ldots, 0\big) \in \mathbb{R}^F.$$

Let

$$\varepsilon_D(h) = \inf_{\theta} \sup_{\mathbf{x} \in \Omega} |D_\theta(\mathbf{z}_h(\mathbf{x})) - S_h f(\mathbf{x})|$$

denote the best decoder approximation error on the corresponding feature range. Then the LBNF hypothesis class satisfies

$$\inf_{\Theta} \|f - \hat{f}_\Theta\|_{L^\infty(\Omega)} \leq \|f - S_h f\|_{L^\infty(\Omega)} + \varepsilon_D(h). \tag{11}$$

Proof sketch. On the bounded feature range, the $\tanh$ decoder can approximate the linear readout $D_\theta(\mathbf{z}) = z_1$ arbitrarily closely by operating in its linear regime, so the infimum defining $\varepsilon_D(h)$ is zero and the LBNF hypothesis class can reproduce $S_h f$ to arbitrary accuracy; see Appendix B.2.

Theorem 1 (Approximation-order inheritance for smooth targets)**.** *Let $\Omega \subset \mathbb{R}^d$ be bounded convex and let $f \in C^{p+1}(\overline{\Omega})$. Let $\{\mathcal{A}_h\}_{h>0}$ be a quasi-uniform family of anchor sets with fill distance h. Consider the untruncated LBNF formulation ($K = N$) with degree-p local polynomial features and normalized isotropic Gaussian weights with bandwidth*

$$\sigma_h = \kappa h, \qquad \kappa > 0.$$

Then there exists a sequence of LBNF parameter sets $\{\Theta_h\}$ such that

$$\|f - \hat{f}_{\Theta_h}\|_{L^\infty(\Omega)} \leq C h^{p+1}. \tag{12}$$

Proof sketch. Using degree-p Taylor polynomials at the anchors, quasi-uniformity and $\sigma_h \asymp h$ imply that the normalized Gaussian PoU has a $(p+1)$ moment of order h^{p+1}, so the Taylor remainder yields $\|f - S_h f\|_{L^\infty(\Omega)} \leq C h^{p+1}$. Lemma 1 then transfers this rate to the LBNF hypothesis class; see Appendix B.3 for the complete proof.

Corollary 1 (Derivative-order inheritance)**.** *Under the hypotheses of Theorem 1, for derivative order $0 \leq \nu \leq p$, there exists a sequence of LBNF parameter sets $\{\Theta_h\}$ such that*

$$\|f - \hat{f}_{\Theta_h}\|_{W^{\nu,\infty}(\Omega)} \leq C h^{p+1-\nu}. \tag{13}$$

For example, when $\nu = 1$,

$$\|\nabla f - \nabla \hat{f}_{\Theta_h}\|_{L^\infty(\Omega)} \leq C h^p.$$

Proof sketch. Applying Leibniz's rule, derivatives of the Gaussian weights scale as $h^{-|\gamma|}$ while the corresponding Taylor remainders scale as $h^{p+1-\nu+|\gamma|}$, yielding an overall $\mathcal{O}(h^{p+1-\nu})$ rate. The near-linear $\tanh$ decoder can approximate the corresponding linear readout and its derivatives through order ν arbitrarily closely, so the LBNF hypothesis class inherits the same derivative order. The complete proof and an additional corollary on approximation order inheritance under truncation (Corollary 2) are provided in Appendix B.4.

4 Experiments

First, we show how LBNFs can be used to fit large scientific datasets, and how to leverage their natural differentiability to perform scientifically relevant gradient-based tasks. We compare LBNF performance to both smooth MLP and grid-based methods on four volumetric scientific datasets, **Rayleigh–Taylor instability** (Ohana et al., 2024), **Argon Bubble**, **Supernova**, and **Asteroid impact** (Gisler et al., 2018) (details in App. D.1), and evaluate a series of post-fitting analyses on them. Second, we test whether trained LBNFs exhibit the convergence behavior predicted in Sec. 3.3 by systematically varying approximation order and anchor density on analytic benchmarks. On smooth signals, we find that LBNFs converge at least as fast as the theoretical rates predict in both values and gradients. Finally, we study how neighborhood truncation and the neural decoder itself affect the trade-off between accuracy and computational scaling (Sec. 4.2).

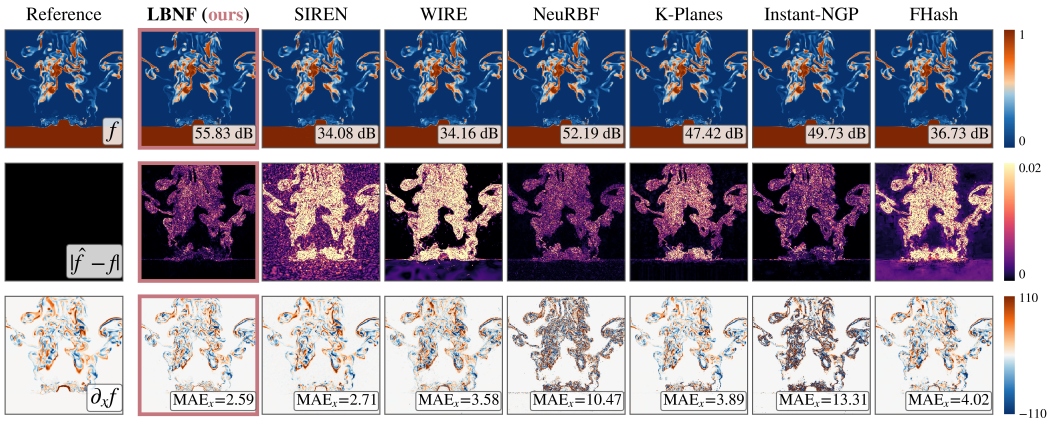

Figure 3: **Asteroid impact signal fitting.** Rows show reconstructed value, absolute error, and first derivative $\partial_x f$ for a central x-slice. Insets report value **PSNR** and **mean absolute error** of $\partial_x f$, evaluated over the interior of the volume. LBNFs outperform grid-based methods on both derivative and value accuracy.

4.1 Scientific Signal Fitting

In this signal fitting task, all models are trained only on sampled signal values using a squared ℓ_2 reconstruction loss, with no higher-order supervision. At inference, derivatives are computed via autograd. Reference derivatives from the ground truth are computed with a high-order finite-difference stencil. Model, optimization, and dataset settings are provided in App. D.2. For the grid-based baselines, we report both their default implementations and custom PyTorch smooth variants, denoted "(B-spline)," which use $\tanh$ decoders and replace trilinear interpolation with cubic B-spline interpolation, yielding a C^2 continuous representation. Slices and their reconstruction and first-derivative errors are shown in Fig. 3 (Rayleigh–Taylor, argon bubble, and supernova in App. D.2), with aggregate value, first-order, and second-order errors reported in Table 1 (supernova and asteroid impact in App. Table 6). Additional results and discussion of adaptive anchor placement can be found in App. D.3. Across the datasets, LBNFs combine the derivative accuracy of smooth MLP-based representations such as SIREN with computational costs comparable to scalable grid-based methods, while also matching or exceeding the latter in value reconstruction accuracy.

4.1.1 Gradient-Based Analysis of Scientific Data

Derivative accuracy matters primarily through the scientific quantities computed from the fitted field, rather than through derivative error alone. We therefore evaluate three representative gradient-based analyses: interface feature detection, isosurface analysis, and surface-normal estimation.

Rayleigh–Taylor bubble and spike detection. Bubble and spike tips are standard diagnostics of a Rayleigh–Taylor mixing layer and occur on the $\rho = 0.5$ interface where $\partial_x \rho = \partial_y \rho = 0$. We extract each model's interface, locate candidate tips from its gradient, and distinguish tips from saddles using the winding number of $(\partial_x \rho, \partial_y \rho)$ (App. D.4.1). Although every model reconstructs the interface to within 0.08 voxels, the other models detect 2.6–$8.9\times$ as many tips as the ground truth, largely due to spurious gradient structure (Fig. 1).

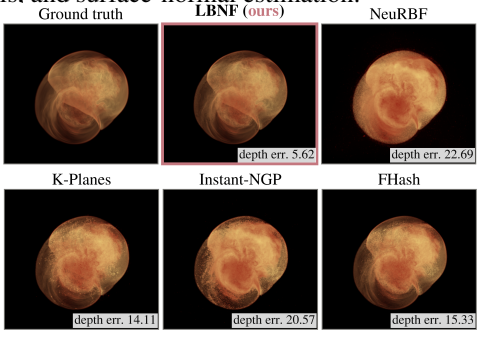

Figure 4: **Gradient-Weighted Rendering of Supernova** (angular momentum) rendered with opacity $\propto |\nabla f|$ and cutaway through center. LBNF achieves the lowest visible-surface depth error.

Gradient-Weighted Rendering. On the supernova volume, we evaluate how derivative error propagates to rendering. We render the volume with opacity proportional to $|\nabla f|$, so that spurious gradients between level sets appear as surfaces absent from the

Table 1: **Scientific signal fitting** accuracy. E_ν^∞ denotes the maximum absolute error over all components. †The representation is not twice differentiable under its default configuration. Peak memory at derivative inference includes all autograd passes required for reported derivative orders, methods with second-order results include additional double-backward cost.

		Value		First order		Second order		Peak memory (GB)		
	Method	PSNR ↑	E_0^∞ ↓	NRMSE_1 ↓	E_1^∞ ↓	NRMSE_2 ↓	E_2^∞ ↓	Train	Value inf.	Deriv. inf.
Rayleigh–Taylor	**LBNF (ours)**	**66.22**	**8.0e-3**	**0.02**	58.0	**0.06**	2.52e4	0.32	0.25	1.69
	SIREN	49.40	6.06e-2	0.02	**48.6**	0.06	**2.24e4**	3.00	0.80	0.58
	WIRE	57.59	1.45e-2	0.02	53.3	0.11	3.92e4	4.47	2.75	0.84
	NeuRBF†	58.68	2.72e-2	0.30	1.69e4	—	—	0.37	0.33	0.15
	K-Planes†	54.49	2.99e-2	0.08	388	—	—	0.26	0.14	0.07
	K-Planes (B-spline)	55.59	1.58e-2	0.03	85.7	0.09	2.83e4	0.56	0.24	0.18
	Instant-NGP†	52.03	3.86e-2	0.14	232	—	—	0.22	0.14	0.07
	Instant-NGP (B-spline)	52.00	2.78e-2	0.04	131	0.18	5.84e4	5.12	5.11	0.41
	FHash†	45.42	8.09e-2	0.08	940	—	—	0.34	0.14	0.09
	FHash (B-spline)	48.57	3.75e-2	0.02	183	0.81	2.05e6	0.82	0.43	0.18
Argon bubble	**LBNF (ours)**	**78.33**	**1.91e-2**	**0.01**	105	0.10	**1.49e5**	0.32	0.25	1.69
	SIREN	50.05	1.40e-1	0.01	**99.8**	**0.07**	1.53e5	3.00	0.80	0.58
	WIRE	53.82	4.07e-2	0.03	161	0.25	2.50e5	4.90	3.86	0.80
	NeuRBF†	75.66	6.53e-2	1.74	1.89e4	—	—	0.38	0.33	0.15
	K-Planes†	58.61	1.11e-1	0.05	1.27e3	—	—	0.25	0.14	0.07
	K-Planes (B-spline)	59.19	7.10e-2	0.03	472	0.13	9.48e5	0.56	0.24	0.18
	Instant-NGP†	63.34	9.11e-2	0.09	265	—	—	0.22	0.14	0.07
	Instant-NGP (B-spline)	69.72	8.96e-2	0.05	195	0.37	2.08e5	5.12	5.11	0.41
	FHash†	42.70	2.21e-1	0.05	1.21e3	—	—	0.28	0.14	0.08
	FHash (B-spline)	44.02	1.81e-1	0.06	2.72e3	3.18	9.38e7	0.79	0.31	0.18

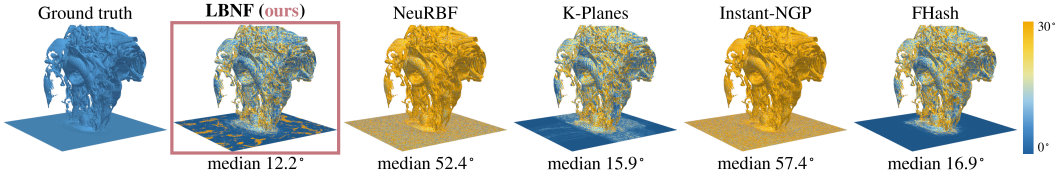

Figure 5: **Surface normals of Asteroid impact** water surface ($v_{02} = 0.5$), colored by the angle between each model's gradient and the finite-difference ground truth (color capped at 30° for visualization). Median angular error is reported over the entire plume.

ground truth (App. D.4.2). LBNF places the visible surfaces closest to the ground truth, with a mean depth error of 5.6 voxels versus 14.1–22.7 for the baselines (Fig. 4).

Surface normals. We next evaluate gradient direction through surface-normal estimation on the asteroid-impact volume. We extract the reference water interface ($v_{02} = 0.5$), evaluate each model's autograd gradient at the resulting surface vertices, and measure the angular error between its normalized gradient and the finite-difference reference normal. Over the water plume, LBNF achieves a median normal error of 12.2°, compared with 15.9–16.9° for K-Planes and FHash and more than 50° for NeuRBF and Instant-NGP (Fig. 5).

4.2 Controlled Analytical Benchmarks

We use 2D analytic targets to isolate the approximation controls of LBNFs, with function values and derivatives available in closed form. Models are trained only on sampled values, and analytic derivatives are used solely for evaluation. Unless otherwise stated, we use fixed uniform anchors, isotropic Gaussian weights, summation aggregation, and a smooth $\tanh$ decoder. The complete training details are given in Appendix C.1. We consider several functions, each with unique attributes: **Sinusoid**, a smooth periodic target; **Ackley**, with oscillatory structure and a localized cusp; **Holder Table**, with repeated derivative discontinuities; **RidgeMax**, with a curved derivative discontinuity; and **Multi-scale Motifs**, with sharp structure across multiple spatial scales (App. C.1).

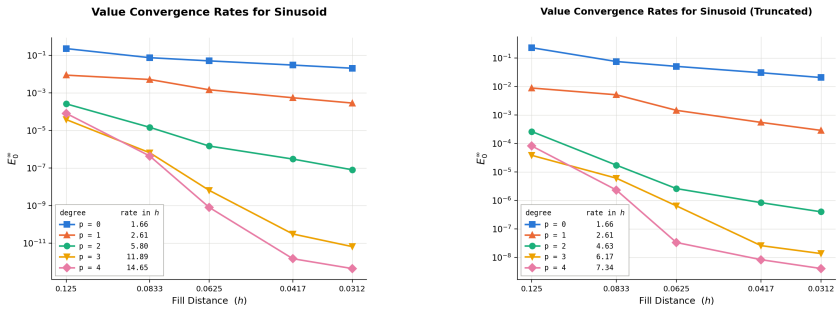

Figure 6: Max value error vs. fill distance h on **Sinusoid**, $p \in \{0, \ldots, 4\}$. Left: untruncated ($K = N$). Right: truncated ($K = 32$). Fitted rates in h (legend) meet or exceed the theoretical $\mathcal{O}(h^{p+1})$ in both settings.

Approximation order and spatial resolution. Theorem 1 and Corollary 1 predict value and gradient rates of $\mathcal{O}(h^{p+1})$ and $\mathcal{O}(h^p)$, respectively. We refine a uniform anchor grid and vary $p \in \{0, \ldots, 4\}$ on the smooth **Sinusoid** target. Across all orders, trained LBNFs converge at least as fast as the predicted value rates, indicating that the theoretical approximation order survives end-to-end training (Fig. 6). Gradient convergence rates and implementation details are in Appendix C.2. We additionally run ablations on the affect order has on representing smooth and non-smooth signals in Appendix C.3.

Neighborhood truncation. We next vary the number of neighboring anchors K to characterize the accuracy–scale tradeoff introduced by local aggregation. Increasing K generally improves reconstruction accuracy while peak evaluation memory grows nearly linearly; moderate truncation retains most of the accuracy of the full neighborhood at substantially lower memory (Table 2). Additional targets, derivative orders, and truncation sweeps are provided in Appendix C.4.

Table 2: **Neighborhood truncation** on **Holder Table** ($p = 2$, $N = 1024$). E_0^∞: max value error. Larger K improves accuracy at higher memory cost.

K	Mem. (GB)↓	PSNR↑	$E_0^\infty \downarrow$	$L_2^\nabla \downarrow$
16	0.21	66.75	2.19e-2	0.149
32	0.36	67.94	1.54e-2	0.135
64	0.65	72.63	6.84e-3	0.134
128	1.24	73.34	6.80e-3	0.128
256	2.41	74.74	6.31e-3	0.123
1024	9.41	75.71	5.66e-3	0.124

Role of the neural decoder. Finally, we compare LBNFs with a decoder-free polynomial PoU using the same local basis construction. In under-resolved and truncated regimes, the classical PoU develops substantial value and derivative artifacts, while the nonlinear decoder recovers a more accurate representation (Figure 12). The effect is strongest on non-smooth and multiscale targets, where the decoder provides additional representation capacity beyond the local polynomial approximation, matched parameters results at additional polynomial order are reported in Appendix C.5

5 Discussion and Conclusion

We introduce Local Basis Neural Fields, a neural representation that combines classical local polynomial approximation with scalable learned feature representations. LBNFs expose approximation order and spatial resolution as explicit controls, inherit classical convergence rates for values and derivatives under refinement, and retain the flexibility to adapt basis placement to complex signals. Across large-scale scientific datasets, LBNFs combine reconstruction accuracy and computational scaling comparable to grid-based neural fields with substantially improved derivative fidelity, enabling reliable downstream gradient-based analysis. Our theoretical results characterize the approximation capacity of the representation under smoothness and quasi-uniformity assumptions; they do not address optimization dynamics or arbitrary learned anchor configurations. Extending these guarantees to adaptive, non-uniform placement and non-smooth signals is left for future work. More broadly, LBNFs show that classical approximation structure can provide neural fields with controllable accuracy–scale tradeoffs that can help guide the user in how to best allocate representation capacity.

REFERENCES

I. Babuška and M. Suri. The p and h-p versions of the finite element method, basic principles and properties. *SIAM Review*, 36(4):578–632, 1994.

Benjamin Charlier, Jean Feydy, Joan Alexis Glaunès, François-David Collin, and Ghislain Durif. Kernel operations on the GPU, with autodiff, without memory overflows. *Journal of Machine Learning Research*, 22(74):1–6, 2021. URL http://jmlr.org/papers/v22/20-275.html.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022.

Zhang Chen, Zhong Li, Liangchen Song, Lele Chen, Jingyi Yu, Junsong Yuan, and Yi Xu. Neurbf: A neural fields representation with adaptive radial basis functions, 2023. URL https://arxiv.org/abs/2309.15426.

Aditya Chetan, Guandao Yang, Zichen Wang, Steve Marschner, and Bharath Hariharan. Accurate differential operators for hybrid neural fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 530–539, 2025.

Wojciech M. Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Świrszcz, and Razvan Pascanu. Sobolev training for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Tim De Ryck, Samuel Lanthaler, and Siddhartha Mishra. On the approximation of functions by tanh neural networks. *Neural Networks*, 143:732–750, 2021.

Ronald DeVore, Boris Hanin, and Guergana Petrova. Neural network approximation, 2020. URL https://arxiv.org/abs/2012.14501.

Ronald A. DeVore. Nonlinear approximation. *Acta Numerica*, 7:51–150, 1998.

Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance, 2023. URL https://arxiv.org/abs/2301.10241.

Galen R. Gisler, Tamra Heberling, Catherine S. Plesko, and Robert P. Weaver. Three-dimensional simulations of oblique asteroid impacts into water. *Journal of Space Safety Engineering*, 2018. doi: 10.1016/j.jsse.2018.06.001.

Ingo Gühring, Gitta Kutyniok, and Philipp Petersen. Error bounds for approximations with deep ReLU neural networks in $W^{s,p}$ norms. *Analysis and Applications*, 18(05):803–859, 2020.

Weizhang Huang and Robert D. Russell. *Adaptive Moving Mesh Methods*, volume 174 of *Applied Mathematical Sciences*. Springer, 2011.

Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3d scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Navami Kairanda, Shanthika Naik, Marc Habermann, Avinash Sharma, Christian Theobalt, and Vladislav Golyanik. Dinf-grid: A neural differential equation solver with differentiable feature grids, 2026. URL https://arxiv.org/abs/2601.10715.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023.

Peter Lancaster and Kestutis Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

Kookjin Lee, Nathaniel A. Trask, Ravi G. Patel, Mamikon A. Gulian, and Eric C. Cyr. Partition of unity networks: deep hp-approximation, 2021. URL https://arxiv.org/abs/2101.11256.

David Levin. The approximation power of moving least-squares. *Mathematics of Computation*, 67 (224):1517–1531, 1998.

Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, 8 (3):29–37, 1988. doi: 10.1109/38.511.

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H. Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Yuzhe Lu, Kairong Jiang, Joshua A. Levine, and Matthew Berger. Compressive neural representations of volumetric scalar fields, 2021. URL https://arxiv.org/abs/2104.04523.

Julien N.P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: Adaptive coordinate networks for neural representation. *ACM Trans. Graph. (SIGGRAPH)*, 2021.

J. M. Melenk and I. Babuška. The partition of unity finite element method: Basic theory and applications. *Computer Methods in Applied Mechanics and Engineering*, 139(1–4):289–314, 1996.

Ben Moseley, Andrew Markham, and Tarje Nissen-Meyer. Finite basis physics-informed neural networks (FBPINNs): A scalable domain decomposition approach for solving differential equations. *Advances in Computational Mathematics*, 49(4):62, 2023.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):1–15, 2022. ISSN 1557-7368. doi: 10.1145/3528223.3530127. URL http://dx.doi.org/10.1145/3528223.3530127.

Ruben Ohana, Michael McCabe, Lucas Meyer, Rudy Morel, Fruzsina J. Agocs, et al. The well: a large-scale collection of diverse physics simulations for machine learning. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks Track)*, 2024.

Joost A. A. Opschoor, Philipp C. Petersen, and Christoph Schwab. Deep ReLU networks and high-order finite element methods. *Analysis and Applications*, 18(05):715–770, 2020.

Joost A.A. Opschoor and Christoph Schwab. Deep relu networks and high-order finite element methods ii: Chebyšev emulation. *Computers amp; Mathematics with Applications*, 169:142–162, 2024. ISSN 0898-1221. doi: 10.1016/j.camwa.2024.06.008. URL http://dx.doi.org/10.1016/j.camwa.2024.06.008.

Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

T. Mitchell Roddenberry, Vishwanath Saragadam, Maarten V. de Hoop, and Richard G. Baraniuk. Implicit neural representations and the algebra of complex wavelets, 2023. URL https://arxiv.org/abs/2310.00545.

Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G. Baraniuk, and Ashok Veeraraghavan. Miner: Multiscale implicit neural representations, 2022. URL https://arxiv.org/abs/2202.03532.

Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G. Baraniuk. Wire: Wavelet implicit neural representations, 2023. URL https://arxiv.org/abs/2301.05187.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions, 2020. URL https://arxiv.org/abs/2006.09661.

Jianxin Sun, David Lenz, Hongfeng Yu, and Tom Peterka. Adaptive multi-resolution encoding for interactive large-scale volume visualization through functional approximation. In *2024 IEEE 14th Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 33–42, 2024. doi: 10.1109/LDAV64567.2024.00006.

Jianxin Sun, David Lenz, Hongfeng Yu, and Tom Peterka. F-hash: Feature-based hash design for time-varying volume visualization via multi-resolution tesseract encoding. *IEEE Transactions on Visualization and Computer Graphics*, 32(1):396–406, 2026. doi: 10.1109/TVCG.2025.3634812.

S. Weiss, P. Hermüller, and R. Westermann. Fast neural representations for direct volume rendering. *Computer Graphics Forum*, 41(6):196–211, 2022. ISSN 1467-8659. doi: 10.1111/cgf.14578. URL http://dx.doi.org/10.1111/cgf.14578.

Holger Wendland. *Scattered Data Approximation*. Cambridge University Press, 2004.

Skylar W. Wurster, Hanqi Guo, Tom Peterka, and Han-Wei Shen. Neural stream functions. In *2023 IEEE 16th Pacific Visualization Symposium (PacificVis)*, pp. 132–141, 2023. doi: 10.1109/PacificVis56936.2023.00022.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

Gizem Yüce, Guillermo Ortiz-Jiménez, Beril Besbinar, and Pascal Frossard. A structured dictionary perspective on implicit neural representations, 2022. URL https://arxiv.org/abs/2112.01917.

Fangcheng Zhong, Kyle Fogarty, Param Hanji, Tianhao Wu, Alejandro Sztrajman, Andrew Spielberg, Andrea Tagliasacchi, Petra Bosilj, and Cengiz Oztireli. Neural fields with hard constraints of arbitrary differential order. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Sophia Zorek, Kushal Vyas, Yuhao Liu, David Lenz, Tom Peterka, and Guha Balakrishnan. Fast amortized fitting of scientific signals across time and ensembles via transferable neural fields, 2026. URL https://arxiv.org/abs/2604.19979.

A Appendix

Appendix Contents

B Proofs of Approximation Results **14**

B.1 Notation and Auxiliary Results 14

B.2 Proof of Lemma 1 15

B.3 Proof of Theorem 1 15

B.4 Proof of Corollary 1 16

B.5 Corollary 2 Statement and Proof 17

C Analytic Test Functions **18**

C.1 Dataset Details 18

C.2 Additional Convergence Results 19

C.3 Approximation Order at Fixed Resolution 20

C.4 Neighborhood Truncation: Accuracy vs. Compute Cost 22

C.5 Role of the Neural Decoder 23

D Large-Scale Scientific Signals **24**

D.1 Dataset Description 24

D.2 Scientific Visualization Fitting 24

D.3 Anchor Relocation 26

D.4 Gradient-Based Scientific Analysis 28

E Extended Related Works **28**

F Reproducibility Statement **29**

G Ethics Statement **29**

H GenAI Statement **29**

B Proofs of Approximation Results

B.1 Notation and Auxiliary Results

Throughout this appendix, $C > 0$ denotes a generic constant independent of the refinement scale h, x, or anchor a_i and its value may change from line to line.

Taylor remainder. Let $T_{i,p}f$ denote the degree-p Taylor polynomial of $f \in C^{p+1}(\overline{\Omega})$ about $\mathbf{a}_i$. For any multi-index $\boldsymbol{\beta}$ with $|\boldsymbol{\beta}| \le p$,

$$\left|D^{\boldsymbol{\beta}}\big(f - T_{i,p}f\big)(\mathbf{x})\right| \le C\|\mathbf{x} - \mathbf{a}_i\|_2^{p+1-|\boldsymbol{\beta}|}. \tag{14}$$

The constant may be chosen uniformly over $\mathbf{x}, \mathbf{a}_i \in \Omega$ because Ω is bounded and convex and the derivatives of f of order $p+1$ are bounded on $\overline{\Omega}$.

Gaussian moment bound. Let $\{\mathcal{A}_h\}_{h>0}$ be a quasi-uniform family of anchor sets and consider the untruncated normalized Gaussian weights

$$w_i(\mathbf{x}) = \frac{\psi_i(\mathbf{x})}{\sum_j \psi_j(\mathbf{x})},$$

with ψ_i as in equation 4 and $\Sigma_i = \kappa^2 h^2 I$, $\kappa > 0$. For every $\lambda \ge 0$,

$$\sup_{\mathbf{x}\in\Omega} \sum_i w_i(\mathbf{x})\|\mathbf{x} - \mathbf{a}_i\|_2^{\lambda} \le Ch^{\lambda}. \tag{15}$$

Proof. For each $\mathbf{x} \in \Omega$, the definition of fill distance guarantees an anchor $\mathbf{a}_{i_*}$ satisfying $\|\mathbf{x} - \mathbf{a}_{i_*}\|_2 \le h$. Hence the Gaussian normalization factor is bounded below by

$$\sum_j \psi_j(\mathbf{x}) \ge \psi_{i_*}(\mathbf{x}) \ge \exp\left(-\frac{1}{2\kappa^2}\right).$$

Partition the anchors into annuli

$$\mathcal{R}_j(\mathbf{x}) = \{i : jh \le \|\mathbf{x} - \mathbf{a}_i\|_2 < (j+1)h\}, \qquad j = 0, 1, \ldots.$$

Quasi-uniformity implies that the number of anchors in $\mathcal{R}_j(\mathbf{x})$ is bounded by $C(j+1)^d$, uniformly in h and $\mathbf{x}$. Therefore,

$$\begin{aligned}\sum_i \psi_i(\mathbf{x})\|\mathbf{x} - \mathbf{a}_i\|_2^{\lambda} &\le Ch^{\lambda} \sum_{j=0}^{\infty} (j+1)^{\lambda+d} \exp\left(-\frac{j^2}{2\kappa^2}\right) \\ &\le Ch^{\lambda},\end{aligned}$$

since the Gaussian decay dominates the polynomial factor. Dividing by the uniform lower bound on the normalization factor gives equation 15. □

Derivative Gaussian moment bound. Under the same assumptions, for any multi-index $\boldsymbol{\gamma}$ and any $\lambda \ge 0$,

$$\sup_{\mathbf{x}\in\Omega} \sum_i |D^{\boldsymbol{\gamma}} w_i(\mathbf{x})|\, \|\mathbf{x} - \mathbf{a}_i\|_2^{\lambda} \le Ch^{\lambda - |\boldsymbol{\gamma}|}. \tag{16}$$

Proof. For every multi-index $\boldsymbol{\alpha}$, derivatives of the unnormalized Gaussian satisfy

$$|D^{\boldsymbol{\alpha}}\psi_i(\mathbf{x})| \le Ch^{-|\boldsymbol{\alpha}|}\left(1 + \left(\frac{\|\mathbf{x} - \mathbf{a}_i\|_2}{h}\right)^{|\boldsymbol{\alpha}|}\right)\psi_i(\mathbf{x}).$$

The annulus argument above with exponents λ and $\lambda + |\boldsymbol{\alpha}|$ gives

$$\sum_i |D^{\boldsymbol{\alpha}} \psi_i(\mathbf{x})| \, \|\mathbf{x} - \mathbf{a}_i\|_2^{\lambda} \le C h^{\lambda - |\boldsymbol{\alpha}|}.$$

Writing

$$Z(\mathbf{x}) = \sum_j \psi_j(\mathbf{x}), \qquad w_i(\mathbf{x}) = \psi_i(\mathbf{x}) Z(\mathbf{x})^{-1},$$

the preceding bounds together with $Z(\mathbf{x}) \ge \exp(-1/(2\kappa^2))$ imply

$$\left| D^{\boldsymbol{\alpha}} Z^{-1}(\mathbf{x}) \right| \le C h^{-|\boldsymbol{\alpha}|}.$$

Leibniz's rule applied to $w_i = \psi_i Z^{-1}$ then yields

$$\begin{aligned} \sum_i |D^{\boldsymbol{\gamma}} w_i(\mathbf{x})| \|\mathbf{x} - \mathbf{a}_i\|_2^{\lambda} &\le C \sum_{\boldsymbol{\alpha} \le \boldsymbol{\gamma}} h^{-|\boldsymbol{\gamma} - \boldsymbol{\alpha}|} h^{\lambda - |\boldsymbol{\alpha}|} \\ &\le C h^{\lambda - |\boldsymbol{\gamma}|}, \end{aligned}$$

which proves equation 16. □

Near-identity decoder construction. Let $B > 0$ and consider inputs $\mathbf{z} = (z_1, \dots, z_F)$ satisfying $|z_1| \le B$. For $\eta > 0$, a two-hidden-layer $\tanh$ MLP with linear output can realize

$$D_\eta(\mathbf{z}) = \eta^{-2} \tanh(\eta \, \tanh(\eta z_1)),$$

by setting all weights associated with $z_2, \dots, z_F$ to zero. Using the Taylor expansion of $\tanh$ at the origin,

$$D_\eta(\mathbf{z}) = z_1 + \mathcal{O}(\eta^2)$$

uniformly for $|z_1| \le B$. Moreover, for every fixed derivative order ν, the same construction converges to the linear readout in C^ν:

$$\max_{0 \le r \le \nu} \sup_{|z_1| \le B} \left| \frac{d^r}{dz_1^r} \big(D_\eta(\mathbf{z}) - z_1 \big) \right| \longrightarrow 0 \qquad \text{as } \eta \to 0.$$

B.2 Proof of Lemma 1

Proof. Because $S_h f$ is realizable by one LBNF feature channel, there exists a choice of encoder parameters for which

$$\mathbf{z}_h(\mathbf{x}) = \big(S_h f(\mathbf{x}), 0, \dots, 0 \big).$$

Fix this encoder. For any decoder parameters θ, the triangle inequality gives

$$\|f - D_\theta(\mathbf{z}_h)\|_{L^\infty(\Omega)} \le \|f - S_h f\|_{L^\infty(\Omega)} + \|S_h f - D_\theta(\mathbf{z}_h)\|_{L^\infty(\Omega)}.$$

Taking the infimum over decoder parameters yields

$$\inf_\theta \|f - D_\theta(\mathbf{z}_h)\|_{L^\infty(\Omega)} \le \|f - S_h f\|_{L^\infty(\Omega)} + \varepsilon_D(h).$$

The full LBNF hypothesis class contains this restricted family obtained by fixing the encoder and varying only the decoder, so

$$\inf_\Theta \|f - \hat{f}_\Theta\|_{L^\infty(\Omega)} \le \inf_\theta \|f - D_\theta(\mathbf{z}_h)\|_{L^\infty(\Omega)},$$

which, combined with the previous display, gives equation 11. □

B.3 Proof of Theorem 1

Proof. Let $T_{i,p} f$ denote the degree-p Taylor polynomial of f about $\mathbf{a}_i$, and define the polynomial PoU approximant

$$S_h f(\mathbf{x}) = \sum_i w_i(\mathbf{x}) T_{i,p} f(\mathbf{x}).$$

The argument follows the local Taylor/partition-of-unity mechanism of Lee et al. (2021), with their compact-support diameter bound replaced by the Gaussian moment bound equation 15. Since the normalized Gaussian weights are nonnegative and satisfy $\sum_i w_i(\mathbf{x}) = 1$,

$$\begin{aligned}|f(\mathbf{x}) - S_h f(\mathbf{x})| &= \left|\sum_i w_i(\mathbf{x})\big(f(\mathbf{x}) - T_{i,p} f(\mathbf{x})\big)\right| \\ &\le \sum_i w_i(\mathbf{x})\,|f(\mathbf{x}) - T_{i,p} f(\mathbf{x})| \\ &\le C \sum_i w_i(\mathbf{x}) \|\mathbf{x} - \mathbf{a}_i\|_2^{p+1},\end{aligned}$$

where the last inequality follows from equation 14 with $\boldsymbol{\beta} = 0$. Applying equation 15 with $\lambda = p+1$ and taking the supremum over $\mathbf{x} \in \Omega$ gives

$$\|f - S_h f\|_{L^\infty(\Omega)} \le C h^{p+1}.$$

The approximant $S_h f$ is realizable by one LBNF feature channel under summation aggregation by taking the local polynomial at anchor $\mathbf{a}_i$ to be $T_{i,p} f$. By Lemma 1 and the near-identity decoder construction of Sec. B.1, applied on the bounded range of $S_h f$, for every $\epsilon > 0$ there exists a choice of LBNF parameters such that

$$\|f - \hat{f}_\Theta\|_{L^\infty(\Omega)} \le C h^{p+1} + \epsilon.$$

For each h, choose $\epsilon = h^{p+1}$. This defines a sequence of parameter sets $\{\Theta_h\}$ satisfying

$$\|f - \hat{f}_{\Theta_h}\|_{L^\infty(\Omega)} \le C h^{p+1},$$

after absorbing the additional term into C, which proves the result. □

B.4 Proof of Corollary 1

Proof. Let

$$S_h f(\mathbf{x}) = \sum_i w_i(\mathbf{x}) T_{i,p} f(\mathbf{x}),$$

as in the proof of Theorem 1. Since $\sum_i w_i(\mathbf{x}) = 1$,

$$f - S_h f = \sum_i w_i \big(f - T_{i,p} f\big).$$

Let $\boldsymbol{\beta}$ be a multi-index with $|\boldsymbol{\beta}| \le \nu$. By the multi-index Leibniz rule,

$$D^{\boldsymbol{\beta}}(f - S_h f) = \sum_i \sum_{\boldsymbol{\gamma} \le \boldsymbol{\beta}} \binom{\boldsymbol{\beta}}{\boldsymbol{\gamma}} D^{\boldsymbol{\gamma}} w_i \, D^{\boldsymbol{\beta}-\boldsymbol{\gamma}}(f - T_{i,p} f),$$

where $\boldsymbol{\gamma}$ indexes the derivatives falling on the weight. The Taylor remainder bound gives

$$\big|D^{\boldsymbol{\beta}-\boldsymbol{\gamma}}(f - T_{i,p} f)(\mathbf{x})\big| \le C \|\mathbf{x} - \mathbf{a}_i\|_2^{p+1-|\boldsymbol{\beta}|+|\boldsymbol{\gamma}|}.$$

Therefore,

$$\begin{aligned}\big|D^{\boldsymbol{\beta}}(f - S_h f)(\mathbf{x})\big| &\le C \sum_{\boldsymbol{\gamma} \le \boldsymbol{\beta}} \sum_i |D^{\boldsymbol{\gamma}} w_i(\mathbf{x})| \|\mathbf{x} - \mathbf{a}_i\|_2^{p+1-|\boldsymbol{\beta}|+|\boldsymbol{\gamma}|} \\ &\le C h^{p+1-|\boldsymbol{\beta}|},\end{aligned}$$

where the final inequality follows from equation 16 with $\lambda = p+1-|\boldsymbol{\beta}|+|\boldsymbol{\gamma}|$. Taking the supremum over $\mathbf{x} \in \Omega$ gives

$$\|D^{\boldsymbol{\beta}}(f - S_h f)\|_{L^\infty(\Omega)} \le C h^{p+1-|\boldsymbol{\beta}|}.$$

Since this holds for every $|\boldsymbol{\beta}| \le \nu$, and $h \le 1$ may be assumed along the refining sequence,

$$\|f - S_h f\|_{W^{\nu,\infty}(\Omega)} \le C h^{p+1-\nu}.$$

As in Theorem 1, $S_h f$ is realizable by one LBNF feature channel. The near-identity decoder construction of Sec. B.1 can be chosen arbitrarily close to the linear readout together with its derivatives through order ν. Since $S_h f$ and its derivatives through order ν are bounded on Ω, the chain rule implies that, for every $\epsilon > 0$, the decoder parameters can be chosen so that

$$\|S_h f - \hat{f}_\Theta\|_{W^{\nu,\infty}(\Omega)} < \epsilon.$$

Choosing $\epsilon = h^{p+1-\nu}$ for each h and applying the triangle inequality yields a sequence $\{\Theta_h\}$ satisfying

$$\|f - \hat{f}_{\Theta_h}\|_{W^{\nu,\infty}(\Omega)} \le Ch^{p+1-\nu},$$

after absorbing the additional term into C. □

B.5 Corollary 2 Statement and Proof

Corollary 2 (Approximation-order inheritance under K-truncation)**.** *Under the smoothness and quasi-uniformity assumptions of Theorem 1, consider the LBNF formulation restricted to the K nearest anchors at each query point. Suppose K is fixed and the corresponding neighborhood radius satisfies*

$$\max_{i \in \mathcal{N}_K(\mathbf{x})} \|\mathbf{x} - \mathbf{a}_i\|_2 \le C_K h$$

uniformly for $\mathbf{x} \in \Omega$. Then there exists a sequence of LBNF parameter sets $\{\Theta_h\}$ such that

$$\|f - \hat{f}_{\Theta_h}\|_{L^\infty(\Omega)} \le Ch^{p+1}. \tag{17}$$

Proof. Let

$$S_h^K f(\mathbf{x}) = \sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i^{(K)}(\mathbf{x}) T_{i,p} f(\mathbf{x})$$

denote the degree-p polynomial PoU approximant formed from the K nearest anchors. Since the truncated weights are nonnegative and satisfy

$$\sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i^{(K)}(\mathbf{x}) = 1,$$

the Taylor remainder bound gives

$$\begin{aligned} |f(\mathbf{x}) - S_h^K f(\mathbf{x})| &\le \sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i^{(K)}(\mathbf{x}) |f(\mathbf{x}) - T_{i,p} f(\mathbf{x})| \\ &\le C \sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i^{(K)}(\mathbf{x}) \|\mathbf{x} - \mathbf{a}_i\|_2^{p+1}. \end{aligned}$$

By the assumed neighborhood-radius bound,

$$\|\mathbf{x} - \mathbf{a}_i\|_2 \le C_K h \qquad \text{for all } i \in \mathcal{N}_K(\mathbf{x}),$$

and therefore

$$|f(\mathbf{x}) - S_h^K f(\mathbf{x})| \le C(C_K h)^{p+1} \sum_{i \in \mathcal{N}_K(\mathbf{x})} w_i^{(K)}(\mathbf{x}) \le Ch^{p+1},$$

after absorbing C_K^{p+1} into the constant. Taking the supremum over $\mathbf{x} \in \Omega$ gives

$$\|f - S_h^K f\|_{L^\infty(\Omega)} \le Ch^{p+1}.$$

Since $S_h^K f$ is realizable by one LBNF feature channel under summation aggregation, Lemma 1 and the near-identity decoder construction, applied on the bounded range of $S_h^K f$, imply that, for every $\epsilon > 0$, there exists a choice of LBNF parameters satisfying

$$\|f - \hat{f}_\Theta\|_{L^\infty(\Omega)} \le Ch^{p+1} + \epsilon.$$

Choosing $\epsilon = h^{p+1}$ for each h and absorbing the additional factor into C yields a sequence $\{\Theta_h\}$ such that

$$\|f - \hat{f}_{\Theta_h}\|_{L^\infty(\Omega)} \le Ch^{p+1}.$$

□

Remark on truncation. For hard K-nearest-neighbor truncation, the same derivative estimates hold within regions on which the active neighbor set is fixed. Changes in $\mathcal{N}_K(\mathbf{x})$ can introduce switching interfaces, so the global derivative result in 1 is stated for the untruncated formulation.

C Analytic Test Functions

This section outline definitions, training protocol, and additional results for the controlled analytical benchmarks of Sec. 4.2.

C.1 Dataset and Training Details

Each target is defined on its native domain $[a, b]^2$ and evaluated on the normalized domain $\Omega = [0, 1]^2$ and function values are rescaled to $[-1, 1]$. Derivatives are taken with respect to the normalized coordinates and are not rescaled, so a p-th order derivative carries some factor from the chain rule relative to the native units. We choose these function to test how LBNFs handle specific signal complexities often found in scientific data.

Table 3 summarizes the targets and Fig. 7 shows each target together with its first and second derivatives.

Sinusoid:

$$f_{\text{raw}}(x, y) = \sin(\omega x)\, \sin(\omega y), \qquad \omega = n\pi, \quad n \geq 2. \tag{18}$$

Ackley:

$$f_{\text{raw}}(x, y) = -A \exp\Big(- B\sqrt{\tfrac{1}{2}(x^2 + y^2)}\Big) - \exp\Big(\tfrac{1}{2}\big[\cos(2\pi x) + \cos(2\pi y)\big]\Big) + A + e, \tag{19}$$

with $A = 20$ and $B = 0.2$, on a domain of$[-32.768,\, 32.768]^2$. The function is highly oscillatory and contains a center cusp.

Holder Table:

$$f_{\text{raw}}(x, y) = -\left|\sin(x)\cos(y)\exp\left(\left|1 - \frac{\sqrt{x^2 + y^2}}{\pi}\right|\right)\right| \tag{20}$$

on $[-10, 10]^2$. The function has derivative discontinuities along the grid lines.

RidgeMax:

$$f_{\text{raw}}(x, y) = \max\big(g_1(x, y),\, g_2(x, y)\big), \qquad g_i(x, y) = A_i \exp\left(-\frac{(x - c_{x,i})^2 + (y - c_{y,i})^2}{2s_i^2}\right), \tag{21}$$

on $[0, 1]^2$, with $(c_{x,1}, c_{y,1}, s_1, A_1) = (0.35,\, 0.65,\, 0.18,\, 1.0)$ and $(c_{x,2}, c_{y,2}, s_2, A_2) = (0.68,\, 0.30,\, 0.22,\, 0.9)$. The field is smooth everywhere except on the curved interface $g_1 = g_2$, where the first derivative jumps. The bumps have different widths and heights, the interface is neither straight nor aligned with the anchor grid.

Multi-scale Motifs:

$$f_{\text{raw}}(x, y) = \sum_{k=1}^{100} a_k\, \max\big(0,\, 1 - \rho_k(x, y)\big), \qquad \rho_k = \sqrt{u_k^2/\alpha_k^2 + v_k^2/\beta_k^2}, \tag{22}$$

where constants can be found in the codebase. The function generates cones across three scale groups. Each cone has a kink on its boundary ellipse and a cusp at its centre, giving derivative discontinuities at many positions, scales and orientations.

Training protocol. Training and evaluation coordinates are sampled on a 512×512 grid over Ω. All models are trained solely from sampled function values with the squared ℓ_2 reconstruction objective; analytic derivatives are used only for evaluation and are never provided during training. Unless stated otherwise, models use fixed uniform anchors, isotropic Gaussian weights, summation aggregation and a $\tanh$ decoder, and are optimized with Adam for 100k iterations at a learning rate of 10^{-3}. Individual sweeps deviate from these defaults where noted, since the optimization requirements vary with approximation order and model configuration. We use fp64 for the analytic benchmarks.

Table 3: Summary of the analytic targets.

Target	Native domain	Regularity	Non-smooth set
Sinusoid	$[0,1]^2$	C^∞	none
Ackley	$[-32.768,\ 32.768]^2$	C^0	cusp at the origin
Holder Table	$[-10,10]^2$	C^0	13 crease lines and one circle
RidgeMax	$[0,1]^2$	C^0	one curved interface
Multi-scale Motifs	$[0,1]^2$	C^0	100 ellipses and cone centres

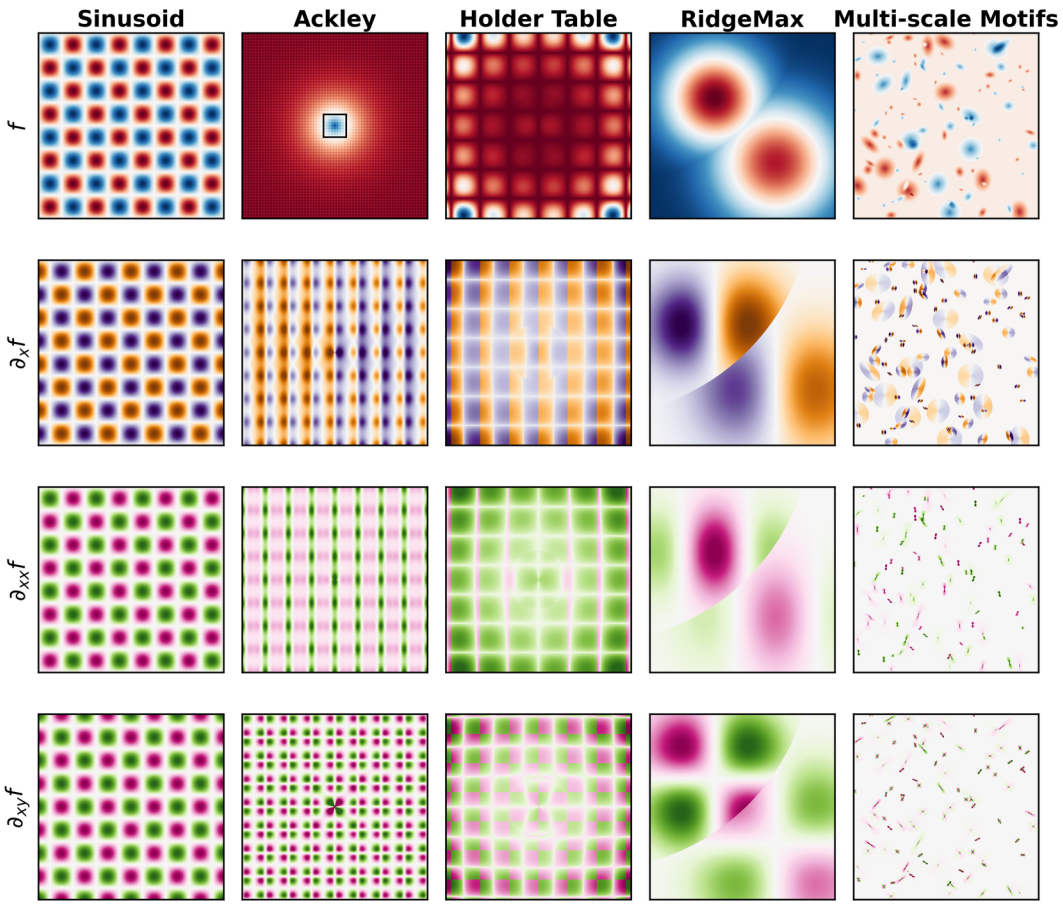

Figure 7: Exact value f, first derivative $\partial_x f$, and second derivatives $\partial_{xx} f$ and $\partial_{xy} f$ of the five analytic targets on $\Omega = [0,1]^2$. The y-derivatives behave analogously and are omitted. Ackley's derivative panels show the boxed central window $[-4,4]^2$ in native units, since its fine-scale oscillation is not resolvable at print size over the full domain. Each panel's colour range is normalized per function per order for visualization purposes.

C.2 Additional Convergence Results

Experimental design. Theorem 1 and Corollary 1 are statements about approximation capacity rather than about training, so we test whether trained LBNFs achieve at least the predicted rates. We refine a uniform anchor grid over $N \in \{64, 144, 256, 576, 1024\}$ anchors, with fill distances $h \in \{0.125, 0.0833, 0.0625, 0.0417, 0.0312\}$, and train LBNFs with $p \in \{0, \dots, 4\}$ on the **Sinusoid** target at each resolution. We fix $\omega = 4\pi$. Adam alone can struggle to learn model parameters that realize the rates, for the convergence experiments only, we take a hybrid approach. We train the decoder with Adam on the coarser anchor grid, then freeze it and solve for the anchor coefficients on each finer grid by Gauss–Newton.

Gradient Convergence. Value convergence is shown in the main paper (Fig. 6). The corresponding gradient convergence is shown for **Sinusoid** in Fig. 8 for both untruncated and truncated regimes. Across all orders, trained LBNFs converge at least as fast as predicted. The legend reports the observed rate for each order.

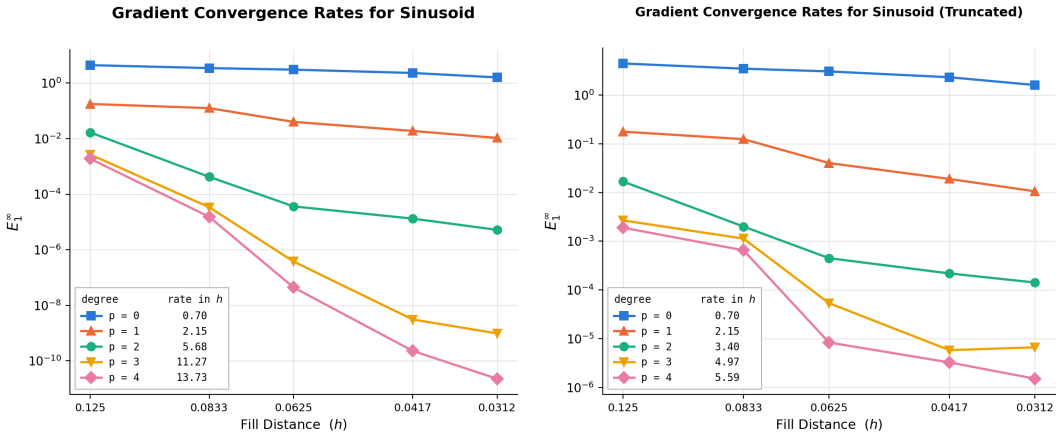

Figure 8: Gradient convergence of LBNFs ($K = 32$) under anchor refinement on the **Sinusoid** (4ω) target for polynomial orders $p = 0$–4. The legend reports the observed rate in h for each order.

C.3 Approximation Order at Fixed Resolution

Experimental design. To isolate the effect of the polynomial order from that of the resolution, we fix an untruncated grid of $N = 4096$ anchors ($K = N$) and vary $p \in \{0, 1, 2\}$ on two targets whose derivative discontinuities have different geometry: **Ackley** and **RidgeMax**. We report E^∞ separately over a non-smooth region (a box enclosing the cusp, or a band enclosing the ridge) and over the smooth remainder. Values, first derivatives and second derivatives are shown for **Ackley** and for **RidgeMax**. In both cases increasing p reduces the smooth-region value error, by roughly an order of magnitude on Ackley and by more than a factor of two on RidgeMax, while the error at the cusp or along the ridge is essentially flat in p.

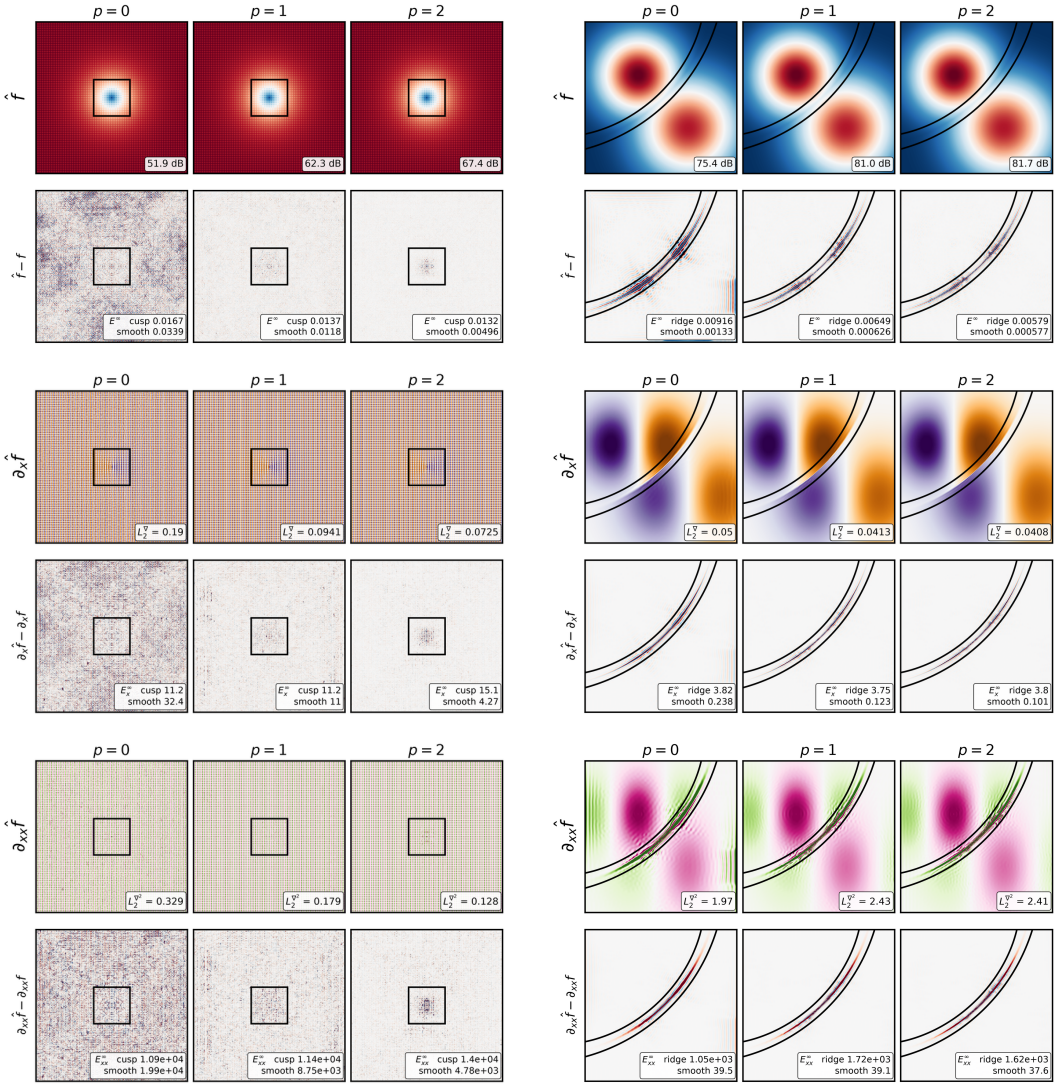

Figure 9: Order sweep $p = 0, 1, 2$ ($N = 4096$, $K = N$) on **Ackley** (left) and **RidgeMax** (right). Rows: $\hat{f}$, $\partial_x \hat{f}$, $\partial_{xx} \hat{f}$. Raising p cuts smooth-region error at every order, while error at the cusp or ridge stays flat.

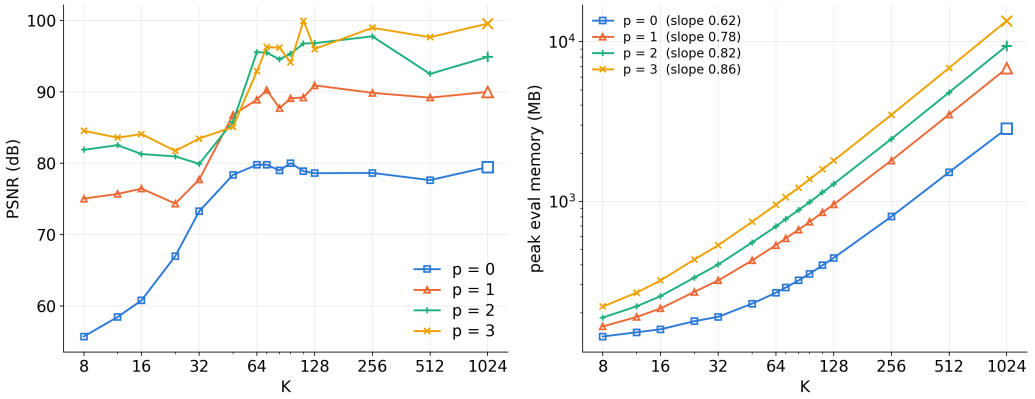

Figure 10: Reconstruction PSNR (left) and peak evaluation memory (right) on the **Ackley** target as the neighborhood truncation K varies, for polynomial orders $p = 0$–3. Memory grows nearly linearly with K, whereas PSNR plateaus beyond roughly $K = 64$. Moderately truncated neighborhoods therefore retain most of the accuracy of the untruncated model at a fraction of its memory cost.

C.4 Neighborhood Truncation: Accuracy vs. Compute Cost

Experimental design. We vary the neighborhood size K to characterize the accuracy–compute tradeoff introduced by local aggregation, for LBNFs with $N = 1024$ anchors on a uniform grid and K from 8 to 1024 (untruncated). Table 2 of the main paper. Fig. 10 repeats the sweep for $p = 0, 1, 2, 3$ on the **Ackley** function and Fig. 11 shows the effect of varying neighborhod size on the **RidgeMax**. We report value PSNR and E_0^∞, the first-derivative errors L_2^∇ and E_1^∞, and peak evaluation memory. Memory is measured with our KeOps (Charlier et al., 2021) implementation, which evaluates truncated neighborhoods without materializing the full query–anchor distance matrix, so it depends only on K and is shared across targets.

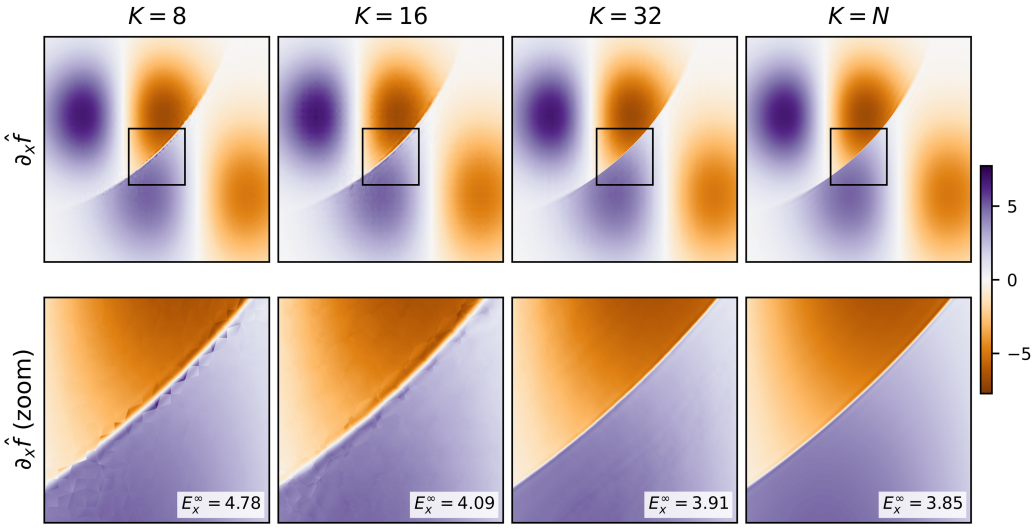

Figure 11: Predicted first derivative $\partial_x \hat{f}$ of the **RidgeMax** target as the neighborhood truncation K increases ($p = 2$, $N = 1024$). Top: full domain; the black box marks the region shown enlarged in the bottom row. Both rows share one color scale. E_x^∞ is the max first derivative error with respect to x. Truncation artifacts are imperceptible at full scale but appear as a jagged interface and blocky patches at small K, largely vanishing by $K = 32$.

C.5 Role of the Neural Decoder

Experimental design. LBNF and PoU are compared at matched total parameter counts (20k and 80k) and a fixed neighborhood size $K = 128$, so any parameters spent on the decoder are taken from the anchors and their coefficients. We report the PSNR difference and the ratios of relative ℓ_2 first- and second-derivative error, LBNF over PoU, for $p = 2$ (Table 4), on the four non-smooth targets. Fig. 12 shows both models in an under-resolved, heavily truncated regime ($N = 64$, $K = 16$, $p = 3$) on the Sinusoid (left) and RidgeMax (right) targets.

Results. For both parameter budgets the decoder improves value PSNR (by 0.6 to 25 dB) and reduces the first-derivative error, with the largest gains on the oscillatory **Ackley** and **Holder Table** targets and the smallest on **RidgeMax**, whose single interface the local polynomials already capture well. In the truncated regime the decoder also smooths over neighborhood-switching interfaces, which is most visible in $\partial_x \hat{f}$, permitting fewer and more truncated bases than the classical construction.

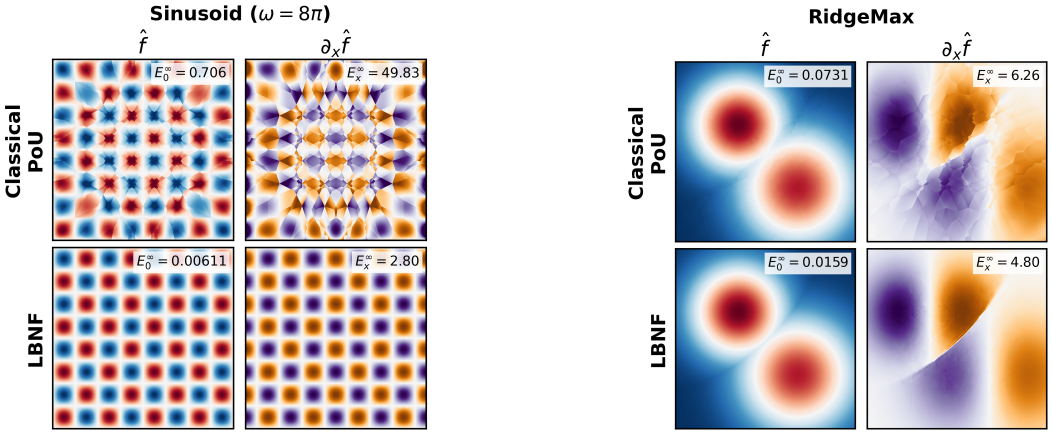

Figure 12: Classical PoU (top) vs. LBNF (bottom) in an under-resolved, truncated regime ($N = 64$, $K = 16$, $p = 3$). Left: **Sinusoid** ($\omega = 8\pi$). Right: **RidgeMax**. Where the decoder-free PoU breaks down, the decoder restores accuracy, most visibly in $\partial_x \hat{f}$, permitting fewer and more truncated bases.

Table 4: LBNF vs. Classical PoU at matched parameter counts ($p = 2$). ΔPSNR is LBNF $-$ PoU; L_2^∇ and $L_2^{\nabla^2}$ are ratios of relative ℓ_2 first- and second-derivative error, LBNF / PoU (raw values in parentheses). Bold marks entries where LBNF outperforms PoU.

Function	ΔPSNR $\uparrow$ (dB)	L_2^∇ ratio $\downarrow$	$L_2^{\nabla^2}$ ratio $\downarrow$
20k parameters			
Ackley	**+21.9** (54.8 − 32.9)	**0.31** (0.294 / 0.964)	**0.54** (0.566 / 1.045)
Holder Table	**+16.8** (65.4 − 48.7)	**0.35** (0.122 / 0.349)	1.67 (5.058 / 3.027)
RidgeMax	**+3.3** (75.1 − 71.8)	**0.76** (0.051 / 0.066)	**0.77** (1.375 / 1.777)
Multi-scale Motifs	**+11.6** (54.2 − 42.6)	**0.45** (0.312 / 0.701)	1.23 (1.280 / 1.043)
80k parameters			
Ackley	**+16.4** (61.0 − 44.6)	**0.81** (0.387 / 0.479)	**0.84** (0.659 / 0.785)
Holder Table	**+13.3** (73.8 − 60.5)	**0.49** (0.115 / 0.236)	**0.71** (2.486 / 3.526)
RidgeMax	**+0.6** (81.9 − 81.3)	**0.78** (0.040 / 0.051)	1.09 (2.846 / 2.616)
Multi-scale Motifs	**+13.3** (66.3 − 53.0)	**0.48** (0.217 / 0.450)	1.16 (1.286 / 1.105)

D LARGE-SCALE SCIENTIFIC SIGNALS

This section describes the volumetric datasets of Sec. 4.1.1, the fitting experiments, the anchor relocation procedure, and gradient-based analysis.

D.1 DATASET DESCRIPTION

We use four volumetric scalar fields from scientific simulations, summarized in Table 5. All four lie on regular grids. We fit each one on its full grid and compute the ground-truth derivatives by second-order central finite differences on that grid. Accuracy is evaluated on the interior of each volume, with 8 voxels trimmed from every face. We select a single timestep for each signal.

Table 5: Summary of the scientific visualization datasets.

Dataset	Domain	Field	Grid	Grad Eval. interior	Voxels
Rayleigh–Taylor	Fluid mixing	density ($t = 50$)	128^3	112^3	2.1M
Argon bubble	Shock physics	intensity	$128 \times 128 \times 256$	$112 \times 112 \times 240$	4.2M
Supernova	Astrophysics	|angular momentum|	216^3	200^3	10.1M
Asteroid impact	Water Impact	water fraction v_{02}	300^3	284^3	27.0M

D.2 SCIENTIFIC VISUALIZATION FITTING

Model and training configuration. All models are trained only on the grid values with the squared ℓ_2 reconstruction loss and no derivative supervision; derivatives are obtained by automatic differentiation at inference. LBNFs use cubic local polynomials ($p = 3$), a neighborhood of $K = 32$ anchors, learned anisotropic Gaussian weights, concatenation aggregation and a $\tanh$ decoder. Anchors are initialized uniformly at random in the domain and relocated with an early training split-and-merge schedule detailed in App. D.3, after which the anchors are trained by gradient descent only. Models are optimized with Adam. All methods, including every baseline and LBNF, use a 1M-parameter model for **Rayleigh–Taylor** and **Argon bubble**, 2M for **Supernova**, and 3M for **Asteroid impact**. Baselines use the default implementations. We use PyTorch re-implementations for the "(B-spline)" variants which replace trilinear interpolation with cubic B-spline interpolation and use a $\tanh$ decoder, so that they are C^2. Additional results can be found below.

Table 6: **Scientific signal fitting** results on **Supernova** and **Asteroid impact**.

		Value		First order		Second order		Peak memory (GB)		
	Method	PSNR ↑	E_0^∞ ↓	NRMSE_1 ↓	E_1^∞ ↓	NRMSE_2 ↓	E_2^∞ ↓	Train	Value inf.	Deriv. inf.
Supernova	**LBNF (ours)**	57.22	4.08e-2	0.02	200	0.19	2.06e6	0.33	0.26	3.35
	SIREN	48.61	9.46e-2	0.01	144	0.07	2.03e5	4.23	1.11	0.78
	WIRE	48.44	7.99e-2	0.02	160	0.23	2.37e6	6.29	3.86	1.14
	NeuRBF†	55.58	6.26e-2	0.20	5.14e3	—	—	0.62	0.57	0.40
	K-Planes†	45.60	1.26e-1	0.11	2.01e3	—	—	0.34	0.16	0.08
	K-Planes (B-spline)	44.82	1.26e-1	0.04	543	0.14	7.96e5	0.58	0.25	0.18
	Instant-NGP†	51.77	6.02e-2	0.19	1.14e3	—	—	0.24	0.15	0.08
	Instant-NGP (B-spline)	52.61	5.17e-2	0.14	637	1.44	2.40e6	5.13	5.11	0.42
	FHash†	40.97	3.07e-1	0.13	2.60e3	—	—	0.35	0.14	0.09
	FHash (B-spline)	40.26	2.68e-1	0.01	213	0.06	7.41e5	0.52	0.31	0.13
Asteroid impact	**LBNF (ours)**	55.83	1.65e-1	0.0369	844	0.9020	6.06e8	0.29	0.21	9.33
	SIREN	34.08	9.21e-1	0.0336	769	0.0786	1.51e6	5.16	1.34	0.97
	WIRE	34.16	8.3e-1	0.0419	960	0.2661	3.21e7	7.71	4.72	1.40
	NeuRBF†	52.19	3.4e-1	1.4153	2.57e6	—	—	1.14	1.08	0.90
	K-Planes†	47.42	3.61e-1	0.0546	2.26e3	—	—	0.53	0.24	0.10
	K-Planes (B-spline)	47.58	4.19e-1	0.0461	2.13e3	0.1577	1.28e7	0.79	0.41	0.25
	Instant-NGP†	49.73	2.49e-1	0.2035	3.15e3	—	—	0.26	0.16	0.09
	Instant-NGP (B-spline)	49.39	3.71e-1	0.1320	1.80e3	0.8774	8.13e6	8.28	8.25	0.63
	FHash†	36.73	7.59e-1	0.0562	3.63e3	—	—	0.29	0.14	0.09
	FHash (B-spline)	37.94	7.57e-1	0.0663	3.36e3	3.9499	3.46e8	0.85	0.44	0.19

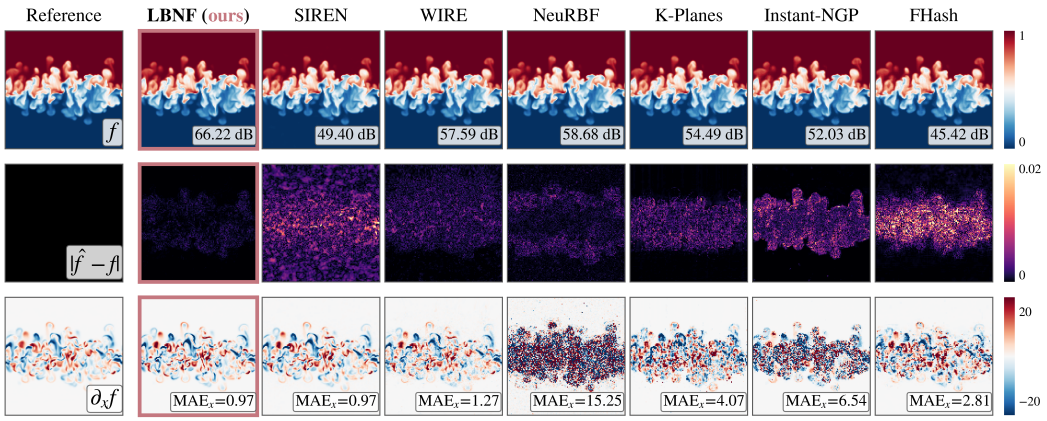

Figure 13: **Rayleigh-Taylor signal fitting.** Rows show reconstructed value, absolute error, and first derivative $\partial_x f$ for a central x-slice. Insets report value **PSNR** and **mean absolute error** of $\partial_x f$, evaluated over the interior of the volume.

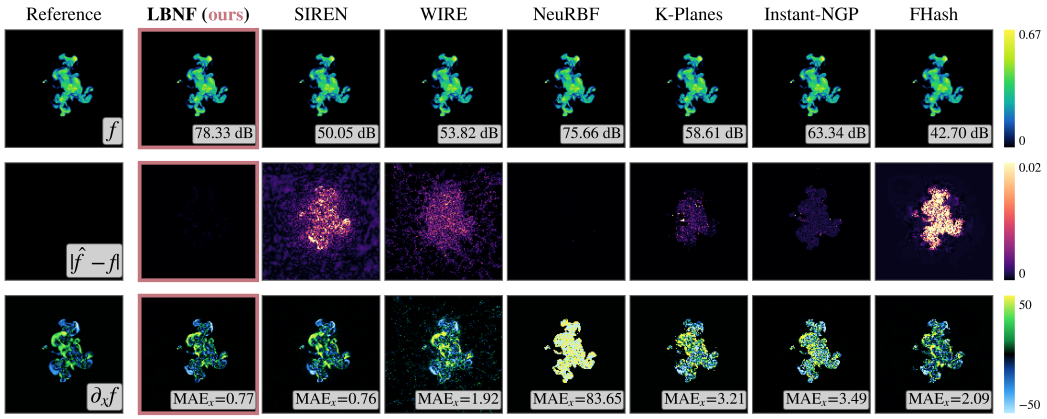

Figure 14: **Argon bubble signal fitting.** Rows show reconstructed value, absolute error, and first derivative $\partial_x f$ for a central x-slice. Insets report value **PSNR** and **mean absolute error** of $\partial_x f$, evaluated over the interior of the volume.

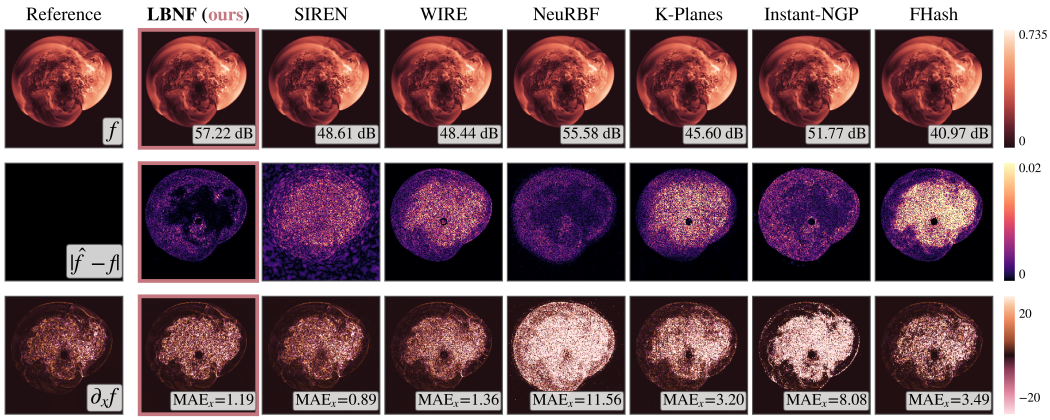

Figure 15: **Supernova signal fitting.** Rows show reconstructed value, absolute error, and first derivative $\partial_x f$ for a central x-slice. Insets report value **PSNR** and **mean absolute error** of $\partial_x f$, evaluated over the interior of the volume.

D.3 Anchor Relocation

This section gives the details of split-and-merge relocation that are omitted from Section 3.2, and defines the effective order used to visualize anchors in Figure 2. Relocation does not depend on how the anchors are initialized: Figure 2 starts from a uniform grid, whereas the scientific models of App. D.2 start from anchors drawn uniformly at random. Figure 16 shows that the random layouts concentrate on the signal within early stages of training.

Relocation schedule. We empirically find that best practice includes a warm-start with fixed anchors, then a period of splitting and merging early in training, eventually tailing off the number of anchors being relocated mid-train. We find that continuing the subtle anchor updates via gradient descent after the split and merge epochs further improves results. Large early events move capacity quickly while the layout is furthest from the signal, smaller later ones limit disruption to the current fit. (Figure 16) visualizes the evolution of anchor location throughout training.

Effective order. To visualize how anchors use their local polynomials (Figure 2), we group each anchor's coefficients by degree, and measure the size of each group by its root mean square. In our implementation the offset is divided by a fixed length scale on the order of the anchor spacing before the monomials are evaluated. This only rescales the coefficients, but keeps every monomial of order one near the anchor, so coefficient magnitudes are comparable across degrees. A degree-p group whose coefficients are close to zero contributes little to the anchor's polynomials and thus the anchor behaves as a lower-degree patch. Those coefficients are not doing as much to represent the signal, and so we define the effective order of anchor i as the highest degree whose group is not negligible. The effective order is purely a diagnostic study. With fixed anchors, most anchors fall in regions where the signal is constant or empty and have effective order zero, while relocated anchors sit almost entirely on the signal and use their full cubic polynomials.

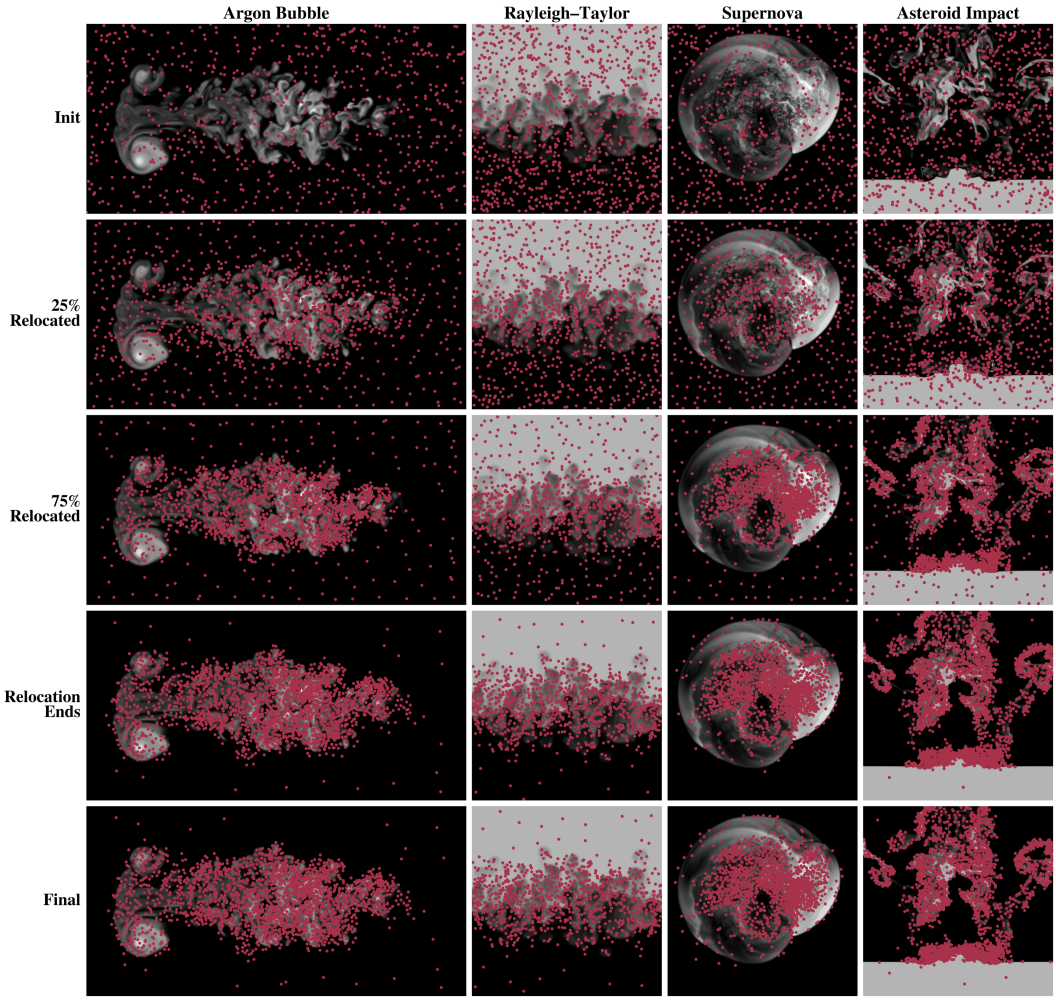

Figure 16: Evolution of the relocated anchors during training for the LBNF of each dataset in Table 1 and Table 6. Each panel shows a mid-slice of the signal with the anchors lying within a thin slab around it. Rows show the initial random anchors, the layouts after 25% and 75% of the change in the fraction of anchors on the signal's structure throughout early training, the layout when relocation ends in training, and the final model. Starting from a random layout rather than the uniform grid of Figure 2 is sufficient as well.

D.4 Gradient-Based Scientific Analysis

This section states how some of the quantities in Sec. 4.1.1 are computed from a fitted field $\hat{f}$ and its automatic-differentiation gradient $\nabla \hat{f}$.

D.4.1 Bubble and Spike Detection on the Rayleigh–Taylor Interface

For each model we extract the $\rho = 0.5$ isosurface of its reconstructed density by marching cubes, which yields a triangle mesh, and evaluate its gradient at every mesh vertex via autograd. The ground truth uses the data's own isosurface with the finite-difference gradient as the reference. Tips are points on the interface where the horizontal gradient $\mathbf{w} = (\partial_x \rho, \partial_y \rho)$ vanishes. To find them we compute, on each mesh triangle, the winding number of $\mathbf{w}$: the net number of turns the vector makes as one walks around the triangle's three vertices. A winding number of $+1$ means the triangle contains a tip, -1 a saddle, and 0 no critical point. A tip is a bubble or a spike according to whether it is a local maximum or minimum of height. Predicted tips are matched to ground-truth tips within 2 voxels. Every model's isosurface lies within 0.08 voxels of the ground truth's, so the differences come from the gradients.

D.4.2 Gradient-Weighted Rendering of the Supernova Dataset

We ray-cast an orthographic image of each model's volume with the front half cut away, so every ray starts at the central plane. Each sample's opacity is proportional to its gradient magnitude $|\nabla f|$ (Levoy, 1988) and its colour to its value. Camera and transfer function are identical for every model. A model with spurious gradient between level sets therefore deposits opacity where the ground truth is transparent. We record for each pixel the depth at which the accumulated opacity first reaches 0.5 and report the mean absolute difference to the ground truth. LBNF's error is 5.6 voxels against 14.1–22.7 for the baselines: the baselines' spurious gradients add surface in front of the true one (Fig. 4).

E Extended Related Work

Grid encodings and Derivative accuracy of neural fields. This section complements Sec. 2 and focuses on work not discussed there. Beyond hash and planar encodings, sparse voxel grids (Fridovich-Keil et al., 2022) and tensor factorizations (Chen et al., 2022) also store features explicitly and recover them by multilinear interpolation, so they share the same piecewise-smooth derivative structure. Neuralangelo (Li et al., 2023) replaces analytic gradients of a hash-grid SDF with finite-difference gradients, since analytic gradients are only locally defined within grid cells. Chetan et al. (2025) show that autodiff derivatives of hybrid fields carry high-frequency noise and repair them with a post hoc local polynomial fitting operator or self-supervised fine-tuning. Constrained neural fields (Zhong et al., 2023) enforce hard linear constraints on a field and its derivatives, and Sobolev training (Czarnecki et al., 2017) supervises derivatives directly. LBNFs differ from these approaches by building local polynomials into the representation itself, so derivative accuracy follows from the same approximation order that governs values rather than from post hoc correction or extra supervision.

Localized and adaptive primitives. KiloNeRF (Reiser et al., 2021) distributes thousands of small MLPs over a regular grid, and local implicit grids (Jiang et al., 2020) decode grid-anchored latent codes with a shared network. 3D Gaussian Splatting (Kerbl et al., 2023) represents scenes with explicit Gaussian primitives that are adaptively cloned, split, and pruned during training, and Scaffold-GS (Lu et al., 2024) organizes such primitives around learned anchors. LBNF's split-and-merge relocation (Sec. 3.2) is closest in spirit to these density-control schemes, but it selects anchors by weighted reconstruction residual at fixed parameter count, and each anchor carries a degree-p polynomial rather than a constant or low-order attribute.

Meshless and adaptive approximation. Moving least squares (Lancaster & Salkauskas, 1981; Levin, 1998) and scattered-data approximation (Wendland, 2004) are the meshless counterparts of partition-of-unity methods, attaining $O(h^{p+1})$ rates under degree-p polynomial reproduction. Nonlinear approximation (DeVore, 1998) and moving-mesh (r-adaptive) methods (Huang & Russell,

2011) allocate a fixed budget of degrees of freedom according to local signal complexity, which motivates LBNF's trainable anchor positions. On the neural side, finite-basis PINNs (Moseley et al., 2023) window local subnetworks with overlapping partition-of-unity functions for domain decomposition. LBNFs instead localize polynomial features and share a single nonlinear decoder across the domain.

Approximation theory for neural networks. Classical rates for deep networks include ReLU error bounds in L^∞ (Yarotsky, 2017) and Sobolev norms (Gühring et al., 2020), ReLU emulation of hp-finite elements (Opschoor et al., 2020), and Sobolev-norm bounds for tanh networks (De Ryck et al., 2021). These analyze MLP-only architectures. Our analysis (Sec. 3.3) extends to hybrid-neural representations.

F Reproducibility Statement

Code is available at `https://anonymous.4open.science/r/LBNFs-anon`. We provide two implementations of LBNFs, a pure PyTorch implementation supporting derivatives of all orders via automatic differentiation, and a hybrid PyTorch/CUDA implementation built on the KeOps library (Charlier et al., 2021) for fast fitting and evaluation. All experiments involving derivatives use the PyTorch implementation, since the hybrid implementation does not yet support higher-order backward passes. All experiments were run on NVIDIA H200 SXM GPUs.

G Ethics Statement

This work develops approximation theory and architectures for neural field representations of continuous signals. It involves no human subjects, personally identifiable information, or clinical data. Experiments use synthetic functions and publicly available [or: institutionally provided] scientific simulation data. We do not foresee direct negative societal impacts. As with any function approximator applied to scientific data, reconstructions may contain artifacts. Downstream scientific or engineering use should validate outputs.

H GenAI Statement

Large language models were used for formatting and editing text for clarity and for debugging the more intricate parts of the code base (CUDA implementation of LBNFs). LLMs did not generate the research ideas, main theoretical results, or experimental results. All proofs, code, and claims are verified by the authors, who take full responsibility for the work in this paper.