BEYOND POINTS: CHEBYSHEV SURROGATES FOR REGION QUERIES ON NEURAL FIELDS

Anonymous authors
Paper under double-blind review

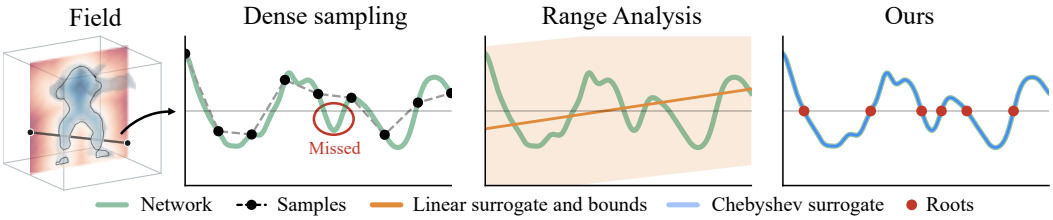


Figure 1: **Reasoning over a region of a neural field.** We query the Signed Distance Function (SDF) of a neural field along a 1 D segment (left). Sampling can miss features. Range analysis (Sharp & Jacobson, 2022) computes a linear surrogate whose bounds are guaranteed but too loose to locate the surface. Our Chebyshev surrogate follows the field closely over the segment and enables efficient estimation of surface intersections.

ABSTRACT

Neural fields provide compact and expressive representations of signals and geometry, mapping coordinates in a continuous input domain to output values. However, many applications require or benefit from reasoning over continuous regions of that domain rather than evaluating the network at individual coordinates. Existing approaches either compute a linear surrogate over each region and reduce it to a pair of output bounds, as in range analysis, or require dedicated architectures and training to answer region queries, such as integrals. Instead, we approximate a trained neural field over a region with a closed-form surrogate, obtained by propagating a Chebyshev polynomial through the network in a single forward pass. The output of each activation is refitted with a discrete cosine transform, which prevents the degree of the surrogate from growing across layers and exposes it as a knob, trading computation against approximation accuracy. Since it describes how the field varies within the region, it enables a broad range of queries directly on the field, without further network evaluations. This lets us locate the sub-regions where the field takes a given value, instead of only accepting, rejecting, or subdividing the region. Moreover, the coefficients of the surrogate give the integral of the field over the region in closed form. We apply our framework to mesh extraction on neural signed distance functions, and to anti-aliased, adaptive-resolution reconstruction of neural images, improving both speed and accuracy over sampling-based and range-analysis methods.

1 INTRODUCTION

Neural fields map coordinates in a continuous input domain to output values through a neural network, providing compact continuous representations of images and 3D scenes (Stanley, 2007; Mescheder et al., 2019; Sitzmann et al., 2020; Mildenhall et al., 2020). Despite the fast development of the field, neural fields share a major shortcoming: they are evaluated at individual points, while many applications require reasoning over entire regions of the input domain. Surface extraction, for example, must identify the cells that intersect a level set. Image reconstruction and volumetric rendering, on the other hand, must integrate the field over a finite pixel footprint or over an extended portion of a ray. These operations are commonly approximated by sampling. However, an

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.

accurate approximation usually requires either a large number of samples, which is computationally expensive, or a restriction on the size of the queried region.

Previous work has addressed region queries in two ways. One line of work replaces or augments the network with a representation in which specific queries are tractable. Some methods design dedicated architectures or training procedures, such as networks whose integrals can be evaluated in closed form (Lindell et al., 2021; Nsampi et al., 2023) or encodings that are integrated over a pixel footprint (Barron et al., 2021; Mujkanovic et al., 2024; Yaldiz et al., 2025). Others rely on complementary representations (Ban & Valasek, 2025) or bypass the neural representation completely (Pujol & Chica, 2024; Lange & Kutz, 2021). These methods are effective for the query they are designed for, but they either require a specific architecture or a precomputed approximation of the whole domain, and none of them builds a high-degree approximation of an arbitrary trained network on demand.

Alternatively, Sharp & Jacobson (2022) showed that affine range analysis can reason directly about regions of an existing network, without evaluating it densely. However, their approach relies on a linear surrogate that cannot track the field when it varies rapidly within the region, as is the case with periodic activation functions (Sitzmann et al., 2020). Moreover, they reduce this surrogate to a pair of output bounds, discarding the local approximation of the field.

This paper proposes a framework that approximates a trained neural field within a region by propagating a polynomial surrogate through the network in a single forward pass. We use Chebyshev polynomials (Mason & Handscomb, 2003) as our surrogate, since they are close to the best polynomial approximation of a given degree over the whole region (Trefethen, 2019). However, naively propagating such polynomials through the network would result in an exploding number of coefficients after a few layers. Therefore, we refit the output of each activation directly to a lower-degree polynomial using the Discrete Cosine Transform (DCT) Trefethen (2019), which avoids instantiating the high-degree intermediate polynomial. This keeps the degree of the surrogate bounded and exposes it as a parameter, trading accuracy against computational cost in a controllable way. Since the framework only requires evaluating the activation functions at a set of nodes, it applies to existing networks without retraining, architectural changes, or precomputation. The surrogate describes how the field varies within the region, and therefore where the values of interest occur. We exploit this to locate the roots of the field within them. This improves speed and accuracy on tasks such as mesh extraction on signed distance functions. Moreover, the coefficients of the surrogate give the integral of the field over the region in closed form, enabling fast, anti-aliased discretization of image fields. Our contributions can be summarized as:

- A framework that approximates trained neural fields over regions with Chebyshev polynomial surrogates, without retraining or architectural changes in a single forward pass.
- A DCT-based refitting of the surrogate after each activation function, which keeps the polynomial degree bounded.
- Queries that exploit the surrogate itself, such as root finding and closed-form integration.
- An evaluation on neural signed distance fields, and neural images where our framework improves speed and accuracy over sampling-based approximations and range analysis methods.

2 Related Work

Neural Fields and Representations. Neural fields represent continuous signals with coordinate-based neural networks and have been used to represent implicit geometry (Park et al., 2019; Mescheder et al., 2019), images (Stanley, 2007; Sitzmann et al., 2020; Martel et al., 2021; Lindell et al., 2022), and radiance fields (Mildenhall et al., 2020). Their ability to represent high-frequency structure has been improved through periodic activations (Sitzmann et al., 2020), Fourier features (Tancik et al., 2020), Gabor wavelets (Saragadam et al., 2023), and Gaussian activations (Chng et al., 2022). A complementary line of work couples a neural decoder with explicit spatial features. Examples include local or grid-based implicit representations (Peng et al., 2020; Chabra et al., 2020; Genova et al., 2020; Takikawa et al., 2021), sparse voxel structures (Liu et al., 2020), multiresolution hash encodings (Müller et al., 2022), and factorized tensor or plane representations (Chen et al., 2022; Fridovich-Keil et al., 2023). This often trades memory for faster evaluation and has, in turn,

motivated dedicated compression methods (Girish et al., 2023). We primarily consider fields whose continuous representation resides in the coordinate network. All these designs, whether purely neural or hybrid, are by construction queried at individual points, whereas our method describes the field over an entire region directly from the network.

Continuous Queries on Neural Fields. Region queries on neural fields are essential for applications such as mesh extraction, rendering, or discretization. Classical mesh extraction methods, such as Marching Cubes (Lorensen & Cline, 1987), rely on discrete samples. Recent approaches (Lei & Jia, 2020; Berzins, 2023; Stippel et al., 2025) instead exploit the partition of the input domain induced by the neurons of ReLU networks to extract meshes analytically, but they are restricted to piecewise-linear activations. For neural radiance fields, Mip-NeRF (Barron et al., 2021) integrates the positional encoding over a Gaussian approximation of each pixel's conical frustum, while Zip-NeRF (Barron et al., 2023) approximates this integral by multisampling the frustum. Other works design dedicated architectures or training procedures such that integrals of the network can be evaluated efficiently (Lindell et al., 2021; Nsampi et al., 2023; Mujkanovic et al., 2024; Yaldiz et al., 2025). All these methods, however, must be trained with the corresponding scheme and cannot be applied to an existing neural field. Another line of work bypasses the neural representation and converts the field into piecewise polynomials stored in an octree or grid, where rays can be intersected analytically (Pujol & Chica, 2024; Lange & Kutz, 2021). Generalized Lipschitz Tracing (Ban & Valasek, 2025) similarly fits local Bernstein polynomials to black-box implicit functions in a preprocessing step, and stores bounds on their directional derivatives in a volumetric Lipschitz field used for robust surface tracing. Range analysis, on the other hand, operates directly on the network: Spelunking the Deep (Sharp & Jacobson, 2022) computes certified bounds over a region from a linear surrogate obtained in a single forward pass. However, since this surrogate is linear, it provides only a coarse description of the field, and its bounds become loose over large regions. Complementary work (Ludwig & Campen, 2026) guarantees conservative neural distance fields by combining Lipschitz-constrained architectures with a constrained training strategy. These works motivate reasoning over finite regions rather than isolated coordinates, but they either depend on a specific architecture or training procedure, require a precomputed approximation of the whole domain, or rely on a coarse linear description of the field. Our method, in contrast, computes a surrogate with low approximation error for any given region of a trained Multi-layer Perceptron (MLP)-based neural field, without retraining or precomputation.

Neural Network Verification. A closely related field is neural network verification, where methods certify properties of a network's output by propagating a surrogate through the network and bounding it. Most methods rely on affine surrogates, such as zonotopes and relational abstract domains (Mirman et al., 2018; Singh et al., 2019) or the linear bounds of CROWN and its descendants (Zhang et al., 2018; Xu et al., 2020; 2021; Wang et al., 2021). These bounds are often refined through branch-and-bound (Shi et al., 2025) or by clipping the input domain (Zhou et al., 2025). Higher-order methods instead propagate polynomial models, based on Taylor (Huang et al., 2022; Wang et al., 2023; Schilling et al., 2022; Entesari & Fazlyab, 2026), Bernstein (Huang et al., 2019; Fan et al., 2020; Fatnassi et al., 2023; 2024), or Fourier-based approximations (Wang et al., 2026), as well as polynomial zonotopes (Kochdumper et al., 2023; Ladner & Althoff, 2024). While these methods rely on certified affine and polynomial models, their primary use is to prove output properties or compute reachable sets. Our focus is instead on efficiently approximating a neural field while retaining the resulting local surrogate as a reusable representation for region queries.

3 METHOD

In this section, we describe our main approach. First, we describe how neural fields are constructed, Sec. 3.1, and present our polynomial surrogate for the 1 D case, Sec. 3.2. We then explain how it is propagated through the network, Sec. 3.3, and describe how such a surrogate can be exploited by various applications, Sec. 3.4. Lastly, Sec. 4 introduces how this formulation naturally extends to higher dimensions. An overview of our approach is presented in Fig. 2.

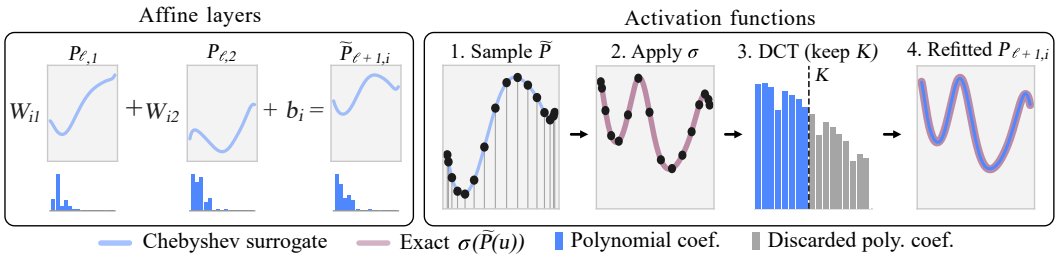


Figure 2: **Method overview.** Propagation of a Chebyshev surrogate through one layer of the network. Left: an affine layer combines the surrogates of the previous neurons with the layer's weights and bias. Since the same weights apply to every coefficient (bars below each curve), the result is exact and keeps the same degree. Right: at an activation function σ, the pre-activation surrogate $\widetilde{P}$ is sampled at Chebyshev nodes (1), and σ is applied to the sampled values (2). A DCT then recovers the Chebyshev coefficients of the result, of which only those up to degree K are kept (3). The refitted surrogate closely matches the exact activation output (4).

3.1 Neural Fields

Neural fields, or Implicit neural representations (INRs), represent continuous signals by coordinate-based neural networks: $f_\theta : \Omega \subseteq \mathbb{R}^d \to \mathbb{R}^c$. The network f_θ maps an input coordinate of dimension d to an output value of dimension c, and therefore defines a continuous field over the input domain. Depending on the application, this field may represent image intensities or signed distances to a surface (Sitzmann et al., 2020; Park et al., 2019), occupancy or density of a volumetric scene (Mescheder et al., 2019), as well as radiance information (Mildenhall et al., 2020). Internally, f_θ is an MLP, where each layer applies an affine transformation followed by an activation function:

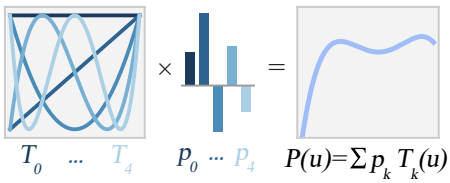
$$z_{\ell+1} = W_\ell a_\ell + b_\ell \qquad a_\ell = \sigma(z_\ell) \tag{1}$$

However, all these architectures share a common limitation: although the represented signal is continuous, the network is always evaluated at individual coordinates.

3.2 Polynomial Surrogate

To retain nonlinear spatial variation, we use Chebyshev polynomials (Trefethen, 2019; Mason & Handscomb, 2003) over local coordinates $u \in [-1, 1]$, which are defined as:

$$P(u) = \sum_{k=0}^{K} p_k T_k(u), \qquad T_k(u) = \cos(k \arccos u), \tag{2}$$



where $T_k(u)$ is the Chebyshev polynomial of degree k, and p_k is its coefficient. Chebyshev polynomials form an orthogonal basis on $[-1, 1]$ with respect to a weighted inner product, so each coefficient captures a distinct component of the approximated function. Moreover, since $|T_k(u)| \leq 1$ on $[-1, 1]$, the coefficient magnitudes directly bound the contribution of each term, and therefore the error introduced by truncation. Our surrogate of a quantity x over the region is thus a Chebyshev polynomial, $\hat{x}(u) = P(u)$.

3.3 Surrogate Propagation

Input. Instead of propagating a single point x through the network, we propagate an entire region of the input domain, $[x^-, x^+]$. To this end, we convert the input into a polynomial surrogate that represents this region. In one dimension, an input region $[x^-, x^+]$ is represented as:

$$\hat{a}_0(u) = P_0(u), \qquad P_0(u) = p_{0,0} T_0(u) + p_{0,1} T_1(u), \tag{3}$$

where, $p_{0,0} = (x^- + x^+)/2$, and $p_{0,1} = (x^+ - x^-)/2$. This surrogate is exact, since P_0 maps $u = \pm 1$ to $x^\pm$, and is then propagated layer by layer through the network.

Affine layers. An affine layer computes a weighted sum of the previous layer's post-activations, a_ℓ, to produce the current layer's pre-activations, $z_{\ell+1}$. When propagating polynomial surrogates, as illustrated in Fig. 2, this weighted sum is applied to the previous layer's surrogates. For neuron i at layer $\ell + 1$, the resulting surrogate is computed as:

$$\hat{z}_{\ell+1,i}(u) = \widetilde{P}_{\ell+1,i}(u) \qquad \widetilde{P}_{\ell+1,i}(u) = \sum_j W_{\ell+1,ij} P_{\ell,j}(u) + b_{\ell+1,i}, \tag{4}$$

where $\hat{a}_{\ell,j}(u) = P_{\ell,j}(u)$ is the post-activation surrogate of neuron j at layer ℓ. This polynomial is computed exactly, and its degree does not increase.

Activation functions. Activation functions compute the post-activation of a layer, a_ℓ, by transforming pre-activations, z_ℓ, using a non-linear function σ. Computing this operation in a standard neural network usually requires only a few operations. However, passing a surrogate through an activation function is not as straightforward. If σ can be approximated by a Chebyshev polynomial S of degree m over the range of $\widetilde{P}_{\ell,i}$, the post-activation surrogate can be obtained by composition, $S(\widetilde{P}_{\ell,i}(u))$. However, composing a polynomial of degree m with a surrogate of degree K produces a polynomial of degree mK, which must be computed before it can be truncated back to degree K. When the pre-activation varies strongly within the queried region, m has to be large to approximate σ over its whole range, and the number of intermediate coefficients quickly becomes prohibitive, especially in higher dimensions.

Instead, we directly refit the post-activation using the DCT. We evaluate the pre-activation polynomial at M Chebyshev–Lobatto nodes $u_j = \cos(j\pi/(M-1))$, $j = 0, \dots, M-1$, apply the activation function to the resulting values, and compute the coefficients $\widehat{p}_{\ell,i,k}$ of the degree-$(M-1)$ interpolant of $\sigma(\widetilde{P}_{\ell,i}(u))$ with a type-I DCT. Fig. 2 illustrates this procedure. Retaining only the coefficients up to degree $K < M-1$ gives the polynomial part of the post-activation surrogate:

$$P_{\ell,i}(u) = \sum_{k=0}^{K} \widehat{p}_{\ell,i,k}\, T_k(u). \tag{5}$$

This requires only M evaluations of σ and a single transform, independently of the activation function, and avoids both the approximation of σ and the high-degree composition. Moreover, using M nodes with $M - 1 > K$ shifts the lowest degree that can alias into the retained coefficients to $2(M-1) - K$, reducing aliasing when the Chebyshev coefficients decay with degree.

Output layer. For a scalar neural field, the final layer is usually affine, without an activation function. Applying equation 4 to the last layer L therefore gives the output surrogate, denote by P_{out}.

3.4 Queries on the surrogate

Once the surrogate has traversed the entire network, it describes how the neural field behaves within the queried region. This surrogate can then be used to answer several queries without further evaluations of the network.

Value range. Many applications on neural fields require an estimate of the range of values the field takes within a region. Since $|T_k(u)| \le 1$ on $[-1, 1]$, the surrogate is contained in:

$$P_{\text{out}}(u) \in [p_{\text{out},0} - R,\, p_{\text{out},0} + R], \qquad R = \sum_{k=1}^{K} |p_{\text{out},k}|. \tag{6}$$

This interval approximates the range of the field up to the approximation error of the surrogate, and can be used to quickly discard regions where a value of interest does not occur.

Root finding. We approximate the locations where the field takes a value of interest v by finding the roots of $P_{\text{out}}(u) - v$ in $[-1, 1]$. These polynomial roots can be computed through the eigenvalues of the corresponding colleague matrix Trefethen (2019). Since the surrogate approximates the network only up to its approximation error, the resulting roots provide approximate locations of the corresponding level-set crossings.

where the tolerance τ accounts for the approximation error of the surrogate. The set $\mathcal{C}$ defines the sub-regions where the value v is expected to occur, which are then processed further, while the rest of the region is discarded.

Integral approximation. Other applications require the integral of the field over a region rather than its value at individual points. The surrogate provides this integral in closed form, since the integrals of the Chebyshev polynomials over $[-1, 1]$ are known analytically:

$$\int_{-1}^{1} T_k(u)\,du = w_k = \begin{cases} \dfrac{2}{1-k^2}, & k \text{ even}, \\ 0, & k \text{ odd}. \end{cases} \tag{7}$$

For a region $[x^-, x^+]$ with half-width $h = (x^+ - x^-)/2$, the integral is approximated by:

$$\int_{x^-}^{x^+} f_\theta(x)\,dx \approx h \sum_{k=0}^{K} w_k\, p_{\mathrm{out},k}. \tag{8}$$

This corresponds to Clenshaw–Curtis quadrature (Trefethen, 2019), applied to the surrogate of the network. The integral is thus obtained from a weighted sum of the coefficients, without any additional evaluation of the network, and its accuracy is given by the approximation accuracy of the surrogate over the region.

4 Extension to Higher Dimensions

The method discussed in the previous sections was derived for the 1D case. However, the same construction extends to higher-dimensional query regions. For a region in a d-dimensional input domain, we use local coordinates $\mathbf{u} = (u_1, \ldots, u_d) \in [-1, 1]^d$, and the univariate Chebyshev expansion is replaced by a multivariate one:

$$P(\mathbf{u}) = \sum_{\boldsymbol{\alpha} \in \mathcal{I}_K} p_{\boldsymbol{\alpha}} \prod_{i=1}^{d} T_{\alpha_i}(u_i), \qquad \mathcal{I}_K = \{\boldsymbol{\alpha} \in \mathbb{N}_0^d : |\boldsymbol{\alpha}|_1 \le K\}. \tag{9}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ is a multi-index whose entry α_i gives the degree of the Chebyshev polynomial in dimension i, and $|\boldsymbol{\alpha}|_1 = \sum_i \alpha_i$ is the total degree of the corresponding term. A surrogate of total degree K therefore has $|\mathcal{I}_K| = \binom{K+d}{d}$ coefficients per neuron.

Input. As in the one-dimensional case, each input coordinate is represented exactly by a surrogate of degree one. For an axis-aligned box $[x_1^-, x_1^+] \times \cdots \times [x_d^-, x_d^+]$, each coordinate depends only on its own local coordinate:

$$\hat{a}_{0,i}(\mathbf{u}) = c_i + h_i\, T_1(u_i), \qquad c_i = \frac{x_i^- + x_i^+}{2}, \quad h_i = \frac{x_i^+ - x_i^-}{2}. \tag{10}$$

Regions of lower dimension are handled in the same way. For a segment along a ray, for example, all input coordinates depend on the same local coordinate u along the ray.

Propagation. Affine layers act on the coefficients exactly as in the one-dimensional case. At activation functions, we evaluate the pre-activation surrogate on a tensor-product grid of M^d Chebyshev–Lobatto nodes and apply a type-I DCT along each dimension, since the transform is separable. This yields the coefficients of the tensor-product interpolant, from which we retain those in $\mathcal{I}_K$.

Queries. All queries of Sec. 3.4 extend directly. The value range follows from $|\prod_i T_{\alpha_i}(u_i)| \le 1$, so that R becomes the sum of $|p_{\mathrm{out},\boldsymbol{\alpha}}|$ over all $\boldsymbol{\alpha} \neq \mathbf{0}$. Root finding applies in the same way, but the set of solutions is now a curve or a surface rather than isolated points. Finally, the integration weights become products of the one-dimensional weights of equation 7, and the integral over the box is approximated by:

$$\int_B f_\theta(\mathbf{x})\,d\mathbf{x} \approx \prod_{i=1}^{d} h_i \sum_{\boldsymbol{\alpha} \in \mathcal{I}_K} p_{\mathrm{out},\boldsymbol{\alpha}} \prod_{i=1}^{d} w_{\alpha_i}. \tag{11}$$

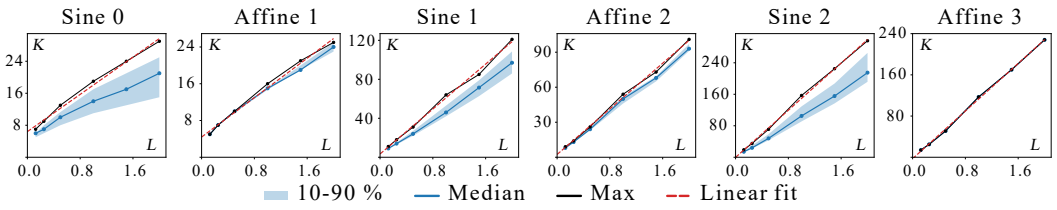


Figure 3: **Required degree** K versus region size L for an INR (`d3_w128`, $\varepsilon = 10^{-6}$).

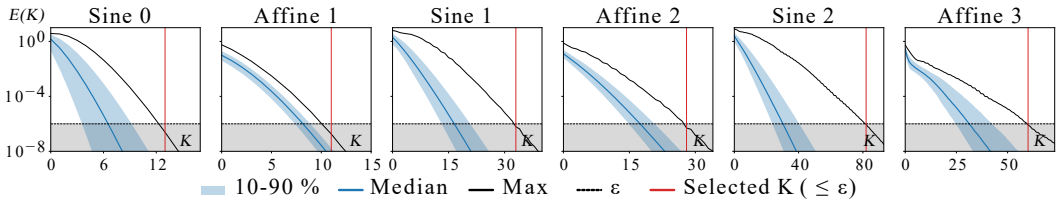


Figure 4: **Error** $E(K)$, for different degrees, K, across stages of an INR (`d3_w128`, $\varepsilon = 10^{-6}$).

5 Evaluation

In this section, we first, in Sec. 5.2, analyze how the degree, the number of nodes, and the size of the queried region affect the error and cost of our surrogate, and derive rules to choose them from a target error. We then, in Sec. 5.3, we apply these rules to mesh extraction and neural image discretization.

5.1 Experimental Setup

Network Architecture. We use two different network sizes in our experiments: a 3-layer MLP with 128 (`d3_w128`) and a 4-layer MLP with 256 neurons (`d4_w256`). We use SIREN (Sitzmann et al., 2020) as the activation function in our experiments, since it can capture high-frequency signals. However, in Sec. 5.4, we test how our method generalizes to different activations.

Datasets. For the SDF experiments, we use 10 shapes from the dataset of Stippel et al. (2025), retaining only watertight meshes with no duplicate faces. For the image experiments, we use two images with a resolution of 4096^2, *pearl* and *pluto* (see Fig. 5).

Baselines. In our applications, we compare our approach to a dense sampling approach, where all queries within a region are estimated from samples. For SDF experiments, we also compare to Spelunking the Deep (Sharp & Jacobson, 2022).

5.2 Surrogate Analysis and Parameter Selection

Our surrogate has two parameters at each activation layer ℓ: its degree K_ℓ, and the number of nodes M_ℓ at which the activation is sampled. A higher degree gives a more accurate surrogate, but also increases the cost. Instead of tuning these parameters for each region, we analyze how they relate to the size of the region and the approximation error, and derive rules that choose them directly from a target error. We perform this analysis on 1 D segments and 3 D regions of a `d3_w128` MLP representing the SDF of the armadillo shape.

What is the ideal degree? For a given local truncation tolerance ε, we define the ideal degree K_ℓ of layer ℓ as the smallest degree for which the sum of the absolute values of the discarded coefficients is below ε for every neuron in the layer. We use $\varepsilon = 10^{-6}$ in this analysis. As shown in Fig. 3, the ideal degree grows approximately linearly with the size L of the region, $K_\ell(L) \approx a_\ell L + b_\ell$, where the slope a_ℓ and offset b_ℓ are obtained by a linear fit for each layer. The slope increases with depth, so deeper layers require considerably higher degrees.

What happens if the degree is not ideal? Choosing a degree different from K_ℓ changes the error of the surrogate, and in Fig. 4 we evaluate by how much. Beyond a small degree, the magnitude of the discarded coefficients decreases by a roughly constant factor $\rho_\ell > 1$ with every additional degree, $E_\ell(K) \approx A_\ell \, \rho_\ell^{-K}$, where $E_\ell(K)$ is the error of truncating layer ℓ at degree K, and A_ℓ and ρ_ℓ are fitted for each layer. A degree lower than K_ℓ therefore increases the error by a factor ρ_ℓ for every missing degree. Conversely, increasing the degree reduces the error quickly: each additional digit of accuracy only requires a fixed number of additional degrees, $\log 10 / \log \rho_\ell$. We therefore add a safety margin to the fitted degree to reduce the local approximation error. The resulting output accuracy also depends on the amplification and accumulation of errors across layers.

How many nodes are needed on the DCT? Once the degree is chosen, the number of nodes M_ℓ is chosen to limit aliasing. Sampling at M Chebyshev–Lobatto nodes causes modes of degree higher than $M - 1$ to alias into lower-degree modes, potentially affecting the retained coefficients. Empirically, setting $M_\ell = K_\ell + 2$ yielded sufficiently small approximation errors in our evaluation, and we use this choice in all experiments.

How expensive is the surrogate? Evaluating an MLP at a single point is dominated by its affine layers. In the surrogate, these layers act on the $K + 1$ coefficients of each neuron, which costs as much as evaluating the network at $K + 1$ points. The activation function is evaluated at $M = K + 2$ nodes per neuron, as it would be for the same number of points. The only additional cost comes from the DCT, which we implement as products with precomputed matrices. Their cost relative to the affine layers scales with the ratio of the degree and the width of the network. For the degrees used in practice for short segments, between 20 and 40 for networks of width 128, a surrogate therefore costs roughly as much as evaluating the network at K to $2K$ points.

How does our layer-wise surrogate compare to fitting a polynomial on network samples? We compare both approaches over different query lengths and record the largest pointwise error observed at each length. Across all tested settings, propagating the surrogate through the network gives a lower worst-case error than fitting a polynomial only to the final network samples. For coarse rays, the error drops from 2.360×10^{-2} to 1.840×10^{-2}, a reduction of 22.0%. For fine rays, it decreases from 3.833×10^{-7} to 3.603×10^{-7} (6.0%). We see the same behavior for 3D cubes: the worst-case error drops from 2.342×10^{-3} to 1.665×10^{-3} for coarse cubes (28.9%), and from 7.133×10^{-4} to 4.561×10^{-4} for fine cubes (36.1%).

Summary. Given a region of size L and a target error ε, we set the degree of each layer from the linear fit, $K_\ell = a_\ell L + b_\ell$, plus a small safety margin, and use $M_\ell = K_\ell + 2$ nodes. This choice requires no search per region, and we use it in all subsequent experiments.

5.3 Applications

Adaptive Mesh Extraction. For surface extraction, the same two-stage strategy is applied spatially. Inexpensive surrogates first filter regions away from the surface. Accurate surrogates are then constructed only for the surviving cells. They accelerate standard extraction methods, such as Marching Cubes (Lorensen & Cline, 1987) or Dual Contouring (Ju et al., 2002), by reliably discarding cells that do not contain the surface. We additionally present results of a hierarchical algorithm, MISE (Mescheder et al., 2019), which hides costly network evaluations by traversing a spatial hierarchical structure. Table 1 shows the results of this experiment for our networks `d4_256` with H being Hausdorff distance. Surrogate filtering reduces the runtime of both Marching Cubes and Dual Contouring by a large margin while preserving or improving reconstruction quality. Our approach, combined with these algorithms, presents similar runtimes as the hierarchical approach of MISE. However, MISE yields a high Hausdorff distance, since coarse cells whose corners all lie on the same side of the surface are never subdivided, even if the surface passes through them.

Table 1: **Mesh extraction.**

Method	T [s] ↓	Tris	H ↓
MC	3.173	1.21 M	4.12
+ Spelunking	7.314	1.21 M	4.12
+ Ours	**0.309**	1.21 M	4.12
DC	3.179	1.21 M	4.20
+ Spelunking	4.727	1.21 M	4.20
+ Ours	**0.442**	1.21 M	2.87
MISE	0.126	1.20 M	25.59

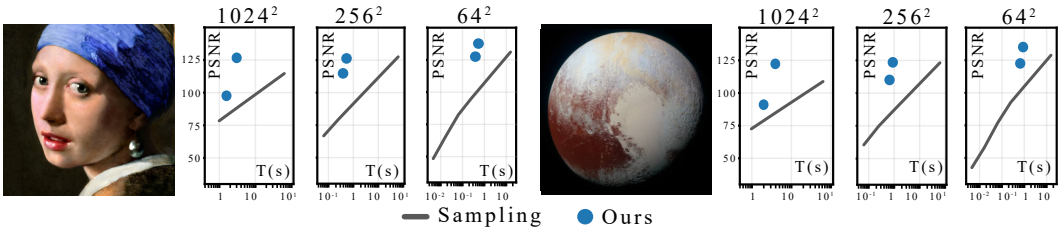


Figure 5: **Image discretization.** PSNR versus computation time for two images at three resolutions, relative to the pixel integral of the same network. Our method reaches a higher PSNR in less time.

Neural Image Discretization. We additionally evaluate whether our surrogate can be used to discretize images represented as INRs. In this experiment, given an image represented as an INR, we aim to generate a discrete set of pixel values that are free from aliasing. This is usually achieved via super-sampling. However, we directly integrate the surrogate to obtain the integral over the desired pixel resolution. Fig. 5 presents the results of this experiment, reporting the PSNR with respect to the pixel integral of the same network. Results show that our surrogates strike a good balance between image fidelity and computational cost, achieving a higher PSNR than the dense-sampling baseline.

5.4 Generalization to Other Activation Functions

Lastly, in Table 2 we evaluate whether our approach generalizes to other activation functions through mesh extraction. In particular, we train a `d3_w128` with Gauss (Chng et al., 2022) and GELU (Hendrycks & Gimpel, 2016) on five meshes, and measure error and speed of the renderings. As shown in the accompanying table, our method still outperforms dense sampling for all activation functions tested.

Table 2: **Comparing activation functions.**

Method	T [s] ↓	Tris	H ↓
MC (Siren)	0.878	1.28 M	1.17
+ Ours	0.109	1.28 M	1.17
MC (Gauss)	0.874	1.29 M	2.12
+ Ours	0.118	1.28 M	2.12
MC (GELU)	0.872	1.28 M	2.55
+ Ours	0.109	1.28 M	2.55

6 Conclusion

We presented a framework to approximate a trained neural field over a finite region of its input domain, rather than evaluating it at individual points. Our method propagates a Chebyshev polynomial surrogate through the network, and refits the output of each activation function with a DCT. This refit keeps the degree of the surrogate bounded without instantiating high-degree intermediate polynomials, and exposes the degree as a parameter, trading accuracy against computational cost. The resulting surrogate describes how the field varies within the region, allowing us to locate the subregions and roots where the field takes a given value and to compute its integral over the region in closed form. We further analyzed how the required degree depends on the region size and the target accuracy, which allows us to choose the degree and the number of sampling nodes for each layer without searching for them per region. Applied to neural signed distance functions, our framework speeds up mesh extraction compared to range-analysis methods. At a similar cost, it also provides more accurate results than sampling-based methods on these tasks, as well as on the discretization of neural images.

GENERATIVE AI DISCLOSURE

We used generative AI tools throughout the development and preparation of this work as research and writing assistants. In particular, we used LLMs to brainstorm and refine research ideas, formulate and check mathematical derivations and claims, discuss alternative methodological choices, design and refine experiments, suggest experimental parameters, assist with implementation and debugging, and interpret experimental results. We also used LLMs to search for and summarize relevant literature and to identify connections to related work; all literature and citations included in the paper were subsequently checked against the original sources by the authors.

Generative AI tools were additionally used during manuscript preparation to suggest structure and terminology, draft and revise portions of the text, improve clarity and concision, and assist with figure design and captions. AI-assisted code was inspected and tested by the authors, and mathematical arguments and experimental conclusions were independently checked against our implementation and results. We did not rely on LLM output as evidence for scientific claims or citations.

The authors determined the research direction, made the final methodological and experimental decisions, performed the experiments, and reviewed and edited all AI-assisted material. We take responsibility for the final content of this work, including all text, claims, code, results, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

We will release all code required to reproduce our method and the experiments reported in this paper, together with all trained models and the parameters used for each experiment. The experimental setup, including network architectures, datasets, and baselines, is described in the paper. The evaluated baselines are based on publicly available implementations and require only minor adaptations, such as correcting the propagation of sinusoidal activations and adding Dual Contouring support in the Spelunking baseline.

REFERENCES

Robert Ban and Gabor Valasek. Generalized lipschitz tracing of implicit surfaces. *Computer Graphics Forum*, 2025.

Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Arturs Berzins. Polyhedral complex extraction from relu networks using edge subdivision. In *International conference on machine learning*, 2023.

Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *European Conference on Computer Vision*, 2020.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European conference on computer vision*, 2022.

Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Gaussian activated neural radiance fields for high fidelity reconstruction and pose estimation. In *European Conference on Computer Vision*, 2022.

Taha Entesari and Mahyar Fazlyab. Hierarchical end-to-end taylor bounds for complete neural network verification. *arXiv*, 2026.

Jiameng Fan, Chao Huang, Xin Chen, Wenchao Li, and Qi Zhu. Reachnn*: A tool for reachability analysis of neural-network controlled systems. In *International Symposium on Automated Technology for Verification and Analysis*, 2020.

Wael Fatnassi, Haitham Khedr, Valen Yamamoto, and Yasser Shoukry. Bern-nn: Tight bound propagation for neural networks using bernstein polynomial interval arithmetic. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, 2023.

Wael Fatnassi, Arthur Feeney, Valen Yamamoto, Aparna Chandramowlishwaran, and Yasser Shoukry. Bern-nn-ibf: Enhancing neural network bound propagation through implicit bernstein form and optimized tensor operations. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2024.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3d shape. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Sharath Girish, Abhinav Shrivastava, and Kamal Gupta. Shacira: Scalable hash-grid compression for implicit neural representations. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv*, 2016.

Chao Huang, Jiameng Fan, Wenchao Li, Xin Chen, and Qi Zhu. Reachnn: Reachability analysis of neural-network controlled systems. *ACM Transactions on Embedded Computing Systems (TECS)*, 2019.

Chao Huang, Jiameng Fan, Xin Chen, Wenchao Li, and Qi Zhu. Polar: A polynomial arithmetic framework for verifying neural-network controlled systems. In *International Symposium on Automated Technology for Verification and Analysis*, 2022.

Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. *ACM Transactions on Graphics*, 2002.

Niklas Kochdumper, Christian Schilling, Matthias Althoff, and Stanley Bak. Open-and closed-loop neural network verification using polynomial zonotopes. In *NASA Formal Methods Symposium*, 2023.

Tobias Ladner and Matthias Althoff. Exponent relaxation of polynomial zonotopes and its applications in formal neural network verification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Henning Lange and J. Nathan Kutz. FC^2T^2: The fast continuous convolutional taylor transform with applications in vision and graphics. *arXiv*, 2021.

Jiabao Lei and Kui Jia. Analytic marching: An analytic meshing solution from deep implicit surface networks. In *International Conference on Machine Learning*, 2020.

David B. Lindell, Julien N. P. Martel, and Gordon Wetzstein. AutoInt: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2022.

Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 2020.

William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Computer Graphics*, 1987.

I Ludwig and M Campen. Strictly conservative neural distance fields. In *Computer Graphics Forum*, 2026.

Julien NP Martel, David B Lindell, Connor Z Lin, Eric R Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: Adaptive coordinate networks for neural scene representation. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 2021.

John C Mason and David Christopher Handscomb. *Chebyshev polynomials*. Chapman & Hall/CRC Boca Raton, 2003.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pp. 405–421, 2020.

Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, 2018.

Felix Mujkanovic, Ntumba Elie Nsampi, Christian Theobalt, Hans-Peter Seidel, and Thomas Leimkühler. Neural gaussian scale-space fields. *ACM Transactions on Graphics*, 2024.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 2022.

Ntumba Elie Nsampi, Adarsh Djeacoumar, Hans-Peter Seidel, Tobias Ritschel, and Thomas Leimkühler. Neural field convolutions by repeated differentiation. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 2023.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2019.

Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European conference on computer vision*, 2020.

E. Pujol and A. Chica. Rendering piecewise approximations of sdfs through analytic intersections. *Computers & Graphics*, 2024.

Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G Baraniuk. Wire: Wavelet implicit neural representations. In *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2023.

Christian Schilling, Marcelo Forets, and Sebastián Guadalupe. Verification of neural-network control systems by integrating taylor models and zonotopes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Nicholas Sharp and Alec Jacobson. Spelunking the deep: Guaranteed queries on general neural implicit surfaces via range analysis. *ACM Transactions on Graphics (TOG)*, 2022.

Zhouxing Shi, Qirui Jin, Zico Kolter, Suman Jana, Cho-Jui Hsieh, and Huan Zhang. Neural network verification with branch-and-bound for general nonlinearities. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, 2025.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proceedings of the ACM on Programming Languages*, 2019.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 2020.

Kenneth O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines*, 2007.

Christian Stippel, Felix Mujkanovic, Thomas Leimkühler, and Pedro Hermosilla. Marching neurons: Accurate surface extraction for neural implicit shapes. *ACM Transactions on Graphics (TOG)*, 2025.

Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3d shapes. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 2020.

Lloyd N Trefethen. *Approximation theory and approximation practice, extended edition*. SIAM, 2019.

Han Wang, Xuyang Ding, Ying Xie, and Yakun Sheng. Qav-ft: Quadratic approximation-based neural network verification via fourier series and taylor truncation. In *International Conference on Computer Aided Verification*, 2026.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in neural information processing systems*, 2021.

Yixuan Wang, Weichao Zhou, Jiameng Fan, Zhilu Wang, Jiajun Li, Xin Chen, Chao Huang, Wenchao Li, and Qi Zhu. Polar-express: Efficient and precise formal reachability analysis of neural-network controlled systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2023.

Kaidi Xu, Zhouxing Shi, Huan Zhang, Yihan Wang, Kai-Wei Chang, Minlie Huang, Bhavya Kailkhura, Xue Lin, and Cho-Jui Hsieh. Automatic perturbation analysis for scalable certified robustness and beyond. *Advances in Neural Information Processing Systems*, 2020.

Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. *International Conference on Learning Representations (ICLR)*, 2021.

Mustafa B. Yaldiz, Ishit Mehta, Nithin Raghavan, Andreas Meuleman, Tzu-Mao Li, and Ravi Ramamoorthi. Spectral prefiltering of neural fields. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, 2025.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. *Advances in neural information processing systems*, 2018.

Duo Zhou, Jorge Chavez, Hesun Chen, Grani A Hanasusanto, and Huan Zhang. Clip-and-verify: Linear constraint-driven domain clipping for accelerating neural network verification. *Advances in Neural Information Processing Systems*, 2025.

A LIMITATIONS.

Our surrogate is an approximation and does not provide guaranteed bounds on the field. Its accuracy is controlled through the degree and an empirically calibrated error estimate, which is conservative in most, but not all, of the tested cases. Certified bounds could be obtained by propagating a rigorous bound on the interpolation error, which classical results provide for analytic activations, at the cost of looser estimates. Finally, the efficiency of the surrogate also depends on the smoothness of the field. For analytic activations, such as sine or Gaussian functions, the Chebyshev coefficients decay geometrically, whereas non-analytic activations, such as ReLU, need not exhibit this decay and may require considerably higher degrees.