Improved RBF Method for High-Accuracy Reconstruction of Complex Two-Dimensional Fields

Skerdi Hymeraj, Saimir Tola,
Polytechnic University of Tirana, Tirane, Faculty of Mathematical & Physical Engineering
Skerdihymeraj02@gmail.com

Abstract: *Radial Basis Function (RBF) interpolation is widely used for reconstructing scattered multidimensional data, but its accuracy and stability often depend heavily on the point distribution, kernel choice, and matrix conditioning. This work proposes an improved RBF framework that combines a whitening transformation, adaptive sampling, and cross-validated parameter tuning to obtain more reliable reconstructions of complex two-dimensional fields. Whitening reduces geometric distortions and improves conditioning, while the adaptive sampling strategy focuses points in regions with strong gradients. Tests on a deliberately challenging synthetic function show that the whitening-enhanced RBF model achieves higher stability and lower error (in L1, L2, and L∞ norms) compared to the standard RBF formulation. The method is implemented in Python using open-source libraries and performs efficiently even on modest hardware.*

KEYWORDS: *RADIAL BASIS FUNCTIONS (RBF), SCATTERED DATA INTERPOLATION, WHITENING TRANSFORM, SHAPE-PARAMETER TUNING, CROSS-VALIDATION, ADAPTIVE SAMPLING, GAUSSIAN / MULTIQUADRIC / INVERSE MULTIQUADRIC / THIN-PLATE SPLINE KERNELS.*

Introduction

Multivariate interpolation on scattered data arises in a broad spectrum of applications, including scientific computing, engineering design, geostatistics, machine learning and uncertainty quantification. Among the most flexible and widely used approaches, Radial Basis Function (RBF) [1], [5] methods provide a mesh-free framework capable of approximating complex functions in moderate to high dimensions while directly handling irregular point clouds. Despite their conceptual simplicity, practical RBF schemes are notoriously sensitive to several interacting factors. The geometry of the input space—such as anisotropic scaling and strong correlations between coordinates—can distort Euclidean distances and degrade the quality of the approximation. The choice of the kernel type and, in particular, the tuning of the shape parameter ε critically affects both approximation power and numerical conditioning. Finally, noisy data and ill-conditioned kernel matrices may lead to strong overfitting or unstable coefficient estimates, especially in the absence of appropriate regularization. Over the last decades, numerous strategies have been proposed to ameliorate these issues, including multi-scale RBFs, adaptive shape parameter selection, localized and partition-of-unity formulations, and combinations with polynomial reproduction or smoothing regularization. However, many of these techniques either rely on problem-specific heuristics or require delicate manual tuning, and they often treat scaling, shape selection and stability as separate concerns.

This work proposes a systematic and fully data-driven approach that addresses these aspects in an integrated manner. The core idea is to normalize the geometry of the input space via whitening, quantify the typical point spacing through a local length scale, and then perform a stability-aware hyperparameter search over a rich family of RBF models. The final predictor is not a single model but an ensemble of top-performing configurations, weighted by their empirically observed predictive accuracy. The main contributions of the paper are:

- A covariance-based whitening transformation that regularizes the input space and improves the conditioning of RBF kernels.
- A data-driven local length scale derived from nearest-neighbor distances in the whitened space, used to define a dimensionless shape parameter.
- A stability-penalized effective score that couples cross-validated RMSE with a quadratic penalty in the dimensionless shape, discouraging unstable regimes.
- An ensemble construction that aggregates several top RBF models into a robust predictor with accuracy-based weights.

The remainder of this paper is organized as follows. Section 2 introduces the proposed Stability–Penalized RBF Ensemble (SP–RBF–ENS) method at a high level. Section 3 reports numerical experiments, and Section 4 concludes the paper with a summary of the main findings and an outline of future research directions.

1. Stability-Enhanced RBF Interpolation with Whitening and Cross-Validation

In this work we employ a stability-enhanced Radial Basis Function (RBF) interpolation framework that combines data whitening, a data-driven length-scale selection based on nearest-neighbor distances, and systematic hyperparameter tuning via k-fold crossvalidation. The overall goal is to construct a robust and accurate interpolant on potentially anisotropic and ill-conditioned data sets, while avoiding ad-hoc parameter choices and manual tuning.

We assume that we are given a set of scattered samples

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}, x_i \in \mathbb{R}^d, y_i \in \mathbb{R},$$

and we seek an approximation $s: \mathbb{R}^d \to \mathbb{R}$ such that $s(x_i) \approx y_i$ for all i, and which

1.1 Data Whitening and Geometric Standardization

A first source of instability in multivariate interpolation arises when different coordinates of x_i live on very different scales or exhibit strong correlations. To mitigate this, we introduce an affine "whitening" transform which maps the original data to a more isotropic configuration [6].

Let $X \in \mathbb{R}^{n\times d}$ be the data matrix collecting the sample locations in its rows,

$$X = \begin{bmatrix} x_1^\top \\ \vdots \\ x_n^\top \end{bmatrix}, \qquad x_i \in \mathbb{R}^d.$$

We define the empirical mean vector.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \in \mathbb{R}^d,$$

and the empirical covariance matrix of the centered data

$$S = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \mu)(x_i - \mu)^\top \in \mathbb{R}^{d\times d}$$

To guarantee numerical stability, we add a small Tikhonov-type regularization on the diagonal,

$$S_\delta = S + \delta I_d, \delta > 0 \text{ very small },$$

and compute a Cholesky factorization

$$S_\delta = LL^\top, L \in \mathbb{R}^{d\times d}$$

with L lower triangular and $\operatorname{diag}(L) > 0$. The whitening transform is then defined by

$$T(x) := (x - \mu)L^{-1}, x \in \mathbb{R}^d.$$

Applying this to all data points yields the whitened nodes

$$z_i = T(x_i) = (x_i - \mu)L^{-1}, i = 1, \dots, n,$$

which we collect into the matrix $Z \in \mathbb{R}^{n\times d}$. Neglecting the tiny regularization, one has the approximate identity

$$\operatorname{Cov}(Z) \approx L^{-1}\operatorname{Cov}(X)L^{-\top} = L^{-1}SL^{-\top} \approx I_d,$$

so that the transformed data are roughly isotropic. In practice, this significantly improves

so that the transformed data are roughly isotropic. In practice, this significantly improves the conditioning of the subsequent RBF interpolation system and makes the choice of the shape parameter more meaningful and less sensitive to the original scaling of the coordinates.

1.2 Data-Driven Length Scale via Nearest–Neighbor Distances

Once the data have been whitened, we extract a characteristic length scale from the spatial distribution of the nodes {z_i}. For each node z_i we define the distance to its closest neighbor (excluding itself)

$$d_i := \min_{j \neq i} \|z_i - z_j\|_2, i = 1, \dots, n.$$

We then use the median of these distances as a robust estimate of the typical inter-point spacing:

$$\ell := \text{median}\{d_1, \dots, d_n\}.$$

This quantity ℓ plays the role of a natural geometric length scale in the whitened space. We exploit it to define a scale-invariant range for the RBF shape parameter ε. Specifically, we generate a set of candidate values of the form

$$\varepsilon_j = \frac{1}{\ell} 10^{\alpha_j}, \alpha_j \in [\alpha_{\min}, \alpha_{\max}]$$

where $\{\alpha_j\}$ are placed on a logarithmic grid. In this way, the shape parameter is searched over several orders of magnitude relative to the intrinsic spacing of the data, rather than in absolute, problem–dependent units.

1.3 RBF Interpolant with Polynomial Tail and Smoothing

On the whitened nodes z_i, we construct an RBF interpolant that, in its most general form, reads [1] :

$$s(z) = \sum_{j=1}^{n} \lambda_j \phi\left(\frac{\|z - z_j\|_2}{\varepsilon}\right) + p(z), z \in \mathbb{R}^d.$$

Here:

- $\phi: [0, \infty) \to \mathbb{R}$ is a radial basis kernel, chosen from a prescribed family such as Gaussian, multiquadric, inverse multiquadric, or thin-plate spline;
- $\varepsilon > 0$ is the shape parameter (for kernels that require it);
- $p(z)$ is a multivariate polynomial of degree at most m, included when the kernel is only conditionally positive definite.

1. Gaussian
$$\varphi(r) = \exp(-(\varepsilon r)^2)$$
2. Inverse Multiquadric (IMQ)
$$\varphi(r) = \frac{1}{\sqrt{1 + (\varepsilon r)^2}}$$
3. Multiquadric (MQ)
$$\varphi(r) = \sqrt{1 + (\varepsilon r)^2}$$
4. Thin Plate Spline (TPS)
$$\varphi(r) = r^2 \log r \ (r > 0), \varphi(0) = 0$$

Let $\Phi \in \mathbb{R}^{n \times n}$ be the RBF Gram matrix with entries

$$\Phi_{ij} = \phi\left(\frac{\|z_i - z_j\|_2}{\varepsilon}\right), i, j = 1, \dots, n$$

and let $P \in \mathbb{R}^{n \times M}$ denote the polynomial design matrix defined by

$$P_{ik} = q_k(z_i), k = 1, \dots, M,$$

where $\{q_k\}$ is a basis of the polynomial space $\mathbb{P}_m(\mathbb{R}^d)$. To enhance robustness and mitigate overfitting or numerical ill-conditioning, we incorporate a small smoothing parameter $\eta \geqslant 0$. The coefficients $\lambda \in \mathbb{R}^n$ and $c \in \mathbb{R}^M$ are then obtained by solving the augmentea linear system

$$\begin{bmatrix} \Phi + \eta I_n & P \\ P^\top & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ c \end{bmatrix} = \begin{bmatrix} y \\ 0 \end{bmatrix},$$

where I_n is the $n \times n$ identity matrix and $y \in \mathbb{R}^n$ is the vector of sampled function values. The regularization ηI_n acts as a Tikhonov term that stabilizes the inversion of Φ, especially in the presence of nearly dependent rows [8].

The polynomial degree m is chosen according to the kernel; strictly positive definite kernels (e.g. Gaussian, inverse multiquadric) typically use $m = 0$ (no polynomial tail), whereas conditionally positive definite kernels (e.g. linear, cubic, thin-plate spline) require $m \geq 1$ to guarantee uniqueness of the solution.

In large data sets, the method can optionally be localized by restricting the interpolation stencil of each point to its N_{neigh} nearest neighbors. This leads to a sparse, locally supported approximation that significantly reduces computational cost while preserving high accuracy.

1.4 Hyperparameter Selection via k-Fold Cross-Validation

The RBF interpolant depends on several hyperparameters: the choice of kernel ϕ, the shape parameter ε, the smoothing parameter η, and the polynomial degree m. Rather than fixing these parameters manually, we determine them in a data-driven manner using k-fold cross-validation [10].

Let $\mathcal{K}$ denote the set of candidate kernels, $\mathcal{E}$ the set of candidate shape parameters (derived from the length scale ℓ), and $\mathcal{S}$ the set of candidate smoothing values. For each triple $(k, \varepsilon, \eta) \in \mathcal{K} \times \mathcal{E} \times \mathcal{S}$ we proceed as follows:

1. Randomly partition the index set $\{1, \dots, n\}$ into k disjoint folds
$$\{1, \dots, n\} = F_1 \cup \dots \cup F_k, F_i \cap F_j = \emptyset (i \neq j).$$
2. For each fold $r = 1, \dots, k$, define trai ↓ J and validation index sets
$$T_r = \{1, \dots, n\} \setminus F_r, V_r = F_r.$$
3. Construct an RBF interpolant $s_r^{(k,\varepsilon,\eta)}$ using only the training data $\{(z_i, y_i)\}_{i \in T_r}$.
4. Evaluate $s_r^{(k,\varepsilon,\eta)}$ on the validation points $\{z_i\}_{i \in V_r}$ and compute the root-meansquare error (RMSE)
$$E_r(k, \varepsilon, \eta) = \sqrt{\frac{1}{|V_r|} \sum_{i \in V_r} \left(s_r^{(k,\varepsilon,\eta)}(z_i) - y_i\right)^2}.$$
5. Define the cross-validated performance index as the average validation error over all folds,
$$\mathcal{J}(k, \varepsilon, \eta) = \frac{1}{k} \sum_{r=1}^{k} E_r(k, \varepsilon, \eta).$$

The optimal hyperparameters are then obtained by solving the discrete minimization problem

$$(k^*, \varepsilon^*, \eta^*) = \arg \min_{k \in \mathcal{K}, \varepsilon \in \mathcal{E}, \eta \in \mathcal{S}} \mathcal{J}(k, \varepsilon, \eta).$$

In practice, we also associate to each kernel k an appropriate polynomial degree $m(k)$, so the polynomial degree is adapted jointly with the kernel.

This cross-validation loop provides an automatic mechanism for balancing accuracy and stability: aggressive choices of ε or very smam smoothing η are penalized if they produce poor predictive performance on validation folds.

1.5 Final Predictor in the Original Coordinate System

After selecting the optimal hyperparameters $(k^*, \varepsilon^*, \eta^*, m^*)$, we retrain a final RBF interpolant on all whitened data $\{(z_i, y_i)\}_{i=1}^n$. This yields a function

$$s^*: \mathbb{R}^d \to \mathbb{R}, s^*(z) = \sum_{j=1}^{n} \lambda_j^* \phi^* \left(\frac{\|z - z_j\|_2}{\varepsilon^*}\right) + p^*(z)$$

where ϕ^* is the selected kernel and p^* is the corresponding polynomial tail [7]. The final predictor in the original coordinate system is obtained by composing s^* with the whitening transform T,

$$\hat{f}(x) = s^*(T(x)) = s^*((x - \mu)L^{-1}), x \in \mathbb{R}^d.$$

For any new query point x, we first standardize it via whitening and then evaluate the RBF interpolant in the whitened space. This ensures full consistency between training and prediction and preserves the numerical advantages of working in a geometrically normalized coordinate system.

2.The experiments

2.1 Geometrically Complex Benchmark Function and Sampling Strategies

In order to assess the accuracy and robustness of the proposed interpolation methods, we introduce a synthetic benchmark on the unit square

$$\Omega = [0,1]^2 \subset \mathbb{R}^2$$

which exhibits a rich combination of anisotropic, multiscale and oscillatory features. This section provides a rigorous mathematical description of the test function and of the sampling strategies used to generate the training data.

2.1.1 Definition of the geometrically complex test function

Let $(x, y) \in \Omega$ and define the centered Cartesian coordinates

$$x_c = x - \frac{1}{2}, y_c = y - \frac{1}{2}$$

together with the polar variables

$$r = \sqrt{x_c^2 + y_c^2}, \theta = \operatorname{atan2}(y_c, x_c) \in (-\pi, \pi]$$

The benchmark function $f: \Omega \to \mathbb{R}$ is constructed as a superposition of six components,

$$f(x, y) = f_{\text{gauss}}(x, y) + f_{\text{ring}}(x, y) + f_{\text{ridge}}(x, y) + f_{\text{hole}}(x, y) + f_{\text{strip}}(x, y) + f_{\text{tilt}}(x, y),$$

each of which is detailed below.

(i) Rotated anisotropic Gaussian bump. We first introduce a sharp anisotropic Gaussian centred at

$$(x_0, y_0) = (0.25, 0.70)$$

rotated by an angle $\alpha = 30^\circ = \pi/6$. Let

$$R(\alpha) = \begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$$

be the rotation matrix, and define the rotated local coordinates

$$\begin{pmatrix} x_r \\ y_r \end{pmatrix} = R(\alpha) \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix}.$$

With anisotropic length scales

$$a = 0.08, b = 0.25$$

the Gaussian contribution takes the form

$$f_{\text{gauss}}(x, y) = 1.2 \exp\left(-\left(\frac{x_r}{a}\right)^2 - \left(\frac{y_r}{b}\right)^2\right).$$

This term produces a sharp, elongated feature oriented along a non-axis-aligned direction, which is challenging for interpolation schemes that assume isotropy.

(ii) Warped ring with angular modulation. To generate a ring-like structure with angular oscillations we conside $f_{\text{ring}}(x, y) = 0.9 \exp\left(-\left(\frac{r - 0.35}{0.06}\right)^2\right)(1 + 0.3\cos(12\theta))$.

The Gaussian in r localizes the feature around the circle $r \approx 0.35$, while the factor $1 + 0.3\cos(12\theta)$ introduces a 12 -fold angular modulation. This term couples radial localisation with high-frequency angular behaviour.

(iii) Narrow diagonal ridge. We next add a narrow oscillatory ridge aligned approximately along the diagonal line $y = 1 - x$. Let $\sigma_{\text{ridge}} = 0.02$ and define

$$f_{\text{ridge}}(x, y) = 0.4 \sin(8\pi(x + y)) \exp\left(-\frac{(y - (1 - x))^2}{2\sigma_{\text{ridge}}^2}\right)$$

The Gaussian factor creates a thin strip of width $\mathcal{O}\left(\sigma_{\text{ridge}}\right)$ around the diagonal, while $\sin(8\pi(x + y))$ imposes high-frequency oscillations along that strip.

(iv) Local anisotropic "crater". A localised negative feature (crater) is placed near the point (0.8,0.2). Let $\sigma_{x,\text{h}} = 0.03$ and $\sigma_{y,\text{h}} = 0.05$. The corresponding term is

$$f_{\text{hole}}(x, y) = -0.7 \exp\left(-\frac{(x - 0.8)^2}{2\sigma_{x,\text{h}}^2} - \frac{(y - 0.2)^2}{2\sigma_{y,\text{h}}^2}\right)$$

Due to the different variances in x and y, this crater is anisotropic and illustrates the capability of the interpolation method to cope with highly localised depressions.

(v) Narrow oscillatory strip near $\boldsymbol{y \approx 0.7}$. To further stress-test the method we place a narrow horizontally oriented oscillatory band. Setting $\sigma_{\text{strip}} = 0.015$, we define

$$f_{\text{strip}}(x, y) = 0.3 \sin(10\pi x) \exp\left(-\frac{(y - 0.7)^2}{2\sigma_{\text{strip}}^2}\right).$$

This component introduces fast oscillations in x confined to a thin strip in the y-direction.

(vi) Smooth polynomial tilt. Finally, a smooth background trend is superimposed via the bilinear term

$$f_{\text{tilt}}(x, y) = 0.1(x - 0.5)(y - 0.3).$$

This ensures that the global behaviour of f is not purely oscillatory and provides a low-frequency component that must also be captured by the interpolant.

Collecting all components yields the composite benchmark function:

- fring-like structures with angular modulation;
- diagonal ridges with high-frequency oscillations;
- localised negative craters;
- narrow oscillatory strips; and
- a smooth polynomial tilt.

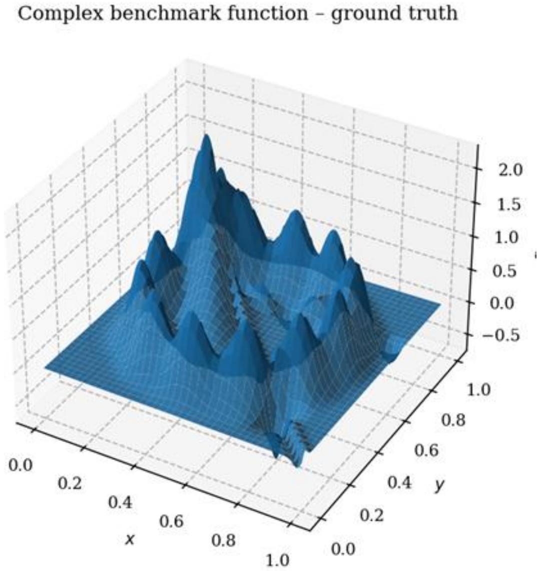


Figure 1. *Three–dimensional surface plot of the geometrically complex benchmark function defined in the domain* $[0,1]^2$.

This makes f a demanding test case for scattered-data interpolation and sampling strategies on Ω.

For convenience, when working with a point cloud $P = \{p_i\}_{i=1}^N \subset \Omega, p_i = (x_i, y_i)$, we use the shorthand

$$f(P) := \left(f(x_i, y_i)\right)_{i=1}^N,$$

which corresponds to the geo_complex_on_points function in the implementation when executing the python code.

2.1.2 Sampling strategies for training points

We now describe the strategies used to generate the set of training nodes $X_{\text{train}} \subset \Omega$. For a prescribed number of points N, we construct a point set $P_N = \{p_i\}_{i=1}^N$ according to one of the following schemes and optionally augment it with boundary points.

Latin Hypercube Sampling on $\boldsymbol{[0,1]^2}$

In the Latin Hypercube Sampling (LHS) strategy, the interval [0,1] is partitioned into N disjoint strata of equal length. For each dimension we draw one point in each stratum and then apply a random permutation independently in each coordinate direction. [2] Formally, let $\{\xi_i\}_{i=0}^{N-1}$ and $\{\zeta_i\}_{i=0}^{N-1}$ be independent random variables uniformly distributed on [0,1]. Define

$$u_i = \frac{i + \xi_i}{N}, v_i = \frac{i + \zeta_i}{N}, i = 0, \ldots, N - 1.$$

Let π_1 and π_2 be two independent random permutations of $\{0, \ldots, N - 1\}$. The N LHS points are then given by

$$p_i = \left(u_{\pi_1(i)}, v_{\pi_2(i)}\right), i = 0, \ldots, N - 1.$$

This construction enforces stratification in each coordinate and tends to reduce variance compared to purely random sampling.

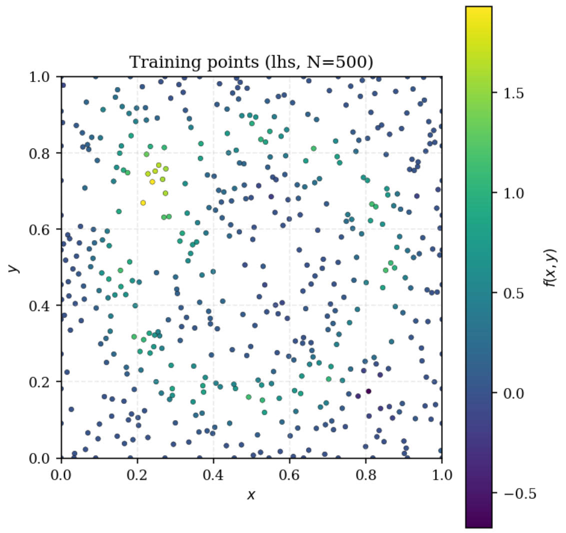


Figure 2. *Two-dimensional scatter plot of the training sample set within the domain* $[0,1]^2$*, generated using Latin Hypercube Sampling with a total of* $N = 500$ *nodes. The color encoding reflects the true function values* $f(x, y)$ *at each sample point, providing insight into spatial variation and the distribution of high-gradient regions prior to interpolation.*

Blue-noise sampling via farthest point strategy

The farthest point sampling (FPS) scheme aims at approximating a blue-noise distribution in Ω, i.e. a configuration of points that is both random and well-spaced [3]. Let M be the number of candidate points with

$$M = \max(10N, 2000)$$

and draw

$$C = \{c_j\}_{j=1}^{M} \subset \Omega$$

i.i.d. from the uniform distribution on $[0,1]^2$. The FPS algorithm constructs a subset $P_N = \{p_i\}_{i=1}^{N}$ as follows:

1. Choose an initial index j_0 uniformly from $\{1, \dots, M\}$ and set $p_1 = c_{j_0}$.
2. For $k = 1, \dots, N - 1$:

- For each candidate c_j, compute its squared distance to the current chosen set $S_k = \{p_1, \dots, p_k\}$,

$$d^2(c_j, S_k) = \min_{p \in S_k} \|c_j - p\|_2^2$$

- Select

$$j_{k+1} = \arg\max_{1 \le j \le M} d^2(c_j, S_k)$$

and set $p_{k+1} = c_{j_{k+1}}$.

By repeatedly choosing the candidate point that maximises its distance to the already selected set, FPS produces a point cloud with relatively uniform inter-point distances.

Boundary augmentation

For improved global stability of the interpolation scheme, we systematically augment the interior samples with points on the domain boundary $\partial\Omega$. For a given integer $n_b \geq 2$, we define equidistant nodes

$$x_i = \frac{i}{n_b - 1}, i = 0, \dots, n_b - 1, y_j = \frac{j}{n_b - 1}, j = 0, \dots, n_b - 1,$$

and construct the sets

$$\Gamma_{\text{bottom}} = \{(x_i, 0): i = 0, \dots, n_b - 1\}$$
$$\Gamma_{\text{top}} = \{(x_i, 1): i = 0, \dots, n_b - 1\}$$
$$\Gamma_{\text{left}} = \{(0, y_j): j = 0, \dots, n_b - 1\}$$
$$\Gamma_{\text{right}} = \{(1, y_j): j = 0, \dots, n_b - 1\}$$

Together with the four corners $(0,0), (1,0), (0,1)$ and $(1,1)$, the boundary set is

$$B = \Gamma_{\text{bottom}} \cup \Gamma_{\text{top}} \cup \Gamma_{\text{left}} \cup \Gamma_{\text{right}} \cup \{(0,0), (1,0), (0,1), (1,1)\}$$

Given any interior point cloud $P \subset \Omega$, the augmented set is

$$Q = P \cup B$$

To avoid numerical artefacts due to nearly identical points, we merge points in Q that are closer than a small tolerance $\tau > 0$; this is implemented by rounding the coordinates to a prescribed number of digits and retaining unique rows.

If the augmented set Q contains more than N points, we apply a farthest point selection step (as above) to subsample Q down to N nodes while preserving good coverage of both the interior and the boundary.

Gradient-based adaptive sampling

To concentrate samples in regions where f exhibits large variations, we construct an adaptive strategy based on the magnitude of the gradient ∇f. The method proceeds in two steps [9].

Candidate cloud and gradient approximation. Given a target number of points N and a mixing parameter $\gamma \in (0,1)$, we choose

$$M = \max(30N, 5000)$$

candidate points $C = \{c_j\}_{j=1}^{M}$ uniformly in Ω. For each candidate $c_j = (x_j, y_j)$ we approximate the gradient using central finite differences with step size $h > 0$:

$$f_x(x_j, y_j) \approx \frac{f(x_j + h, y_j) - f(x_j - h, y_j)}{2h},$$

$$f_y(x_j, y_j) \approx \frac{f(x_j, y_j + h) - f(x_j, y_j - h)}{2h},$$

where the arguments are projected back to $[0,1]$ if necessary. The gradient magnitude is then

$$g_j = \sqrt{f_x(x_j, y_j)^2 + f_y(x_j, y_j)^2}.$$

Probability distribution and sampling. We define a discrete probability distribution on the candidate set by

$$p_j = \frac{g_j + \varepsilon_0}{\sum_{k=1}^{M} (g_k + \varepsilon_0)}, j = 1, \dots, M,$$

where $\varepsilon_0 > 0$ is a small constant ensuring positivity. We then choose

$$N_{\text{ad}} = \lfloor (1 - \gamma) N \rfloor$$

points from C without replacement according to the probabilities $\{p_j\}$, and generate the remaining

$$N_{\text{un}} = N - N_{\text{ad}}$$

The final adaptive point set is the union of the gradient-weighted and uniform samples:

$$P_N^{\text{ad}} = \{c_{j_1}, \dots, c_{j_{N_{\text{ad}}}}\} \cup \{\tilde{c}_1, \dots, \tilde{c}_{N_{\text{un}}}\}.$$

By construction, regions with large $|\nabla f|$ receive more samples, while a fraction of uniformly distributed points preserves global coverage of the domain.

2.1.3 Unified training set construction

Given a prescribed sampling type

kind $\in$ { "uniform", "lhs", "blue", "adaptive" },

and a target size N_{train}, we define the training point set X_{train} as follows:

- If kind = "uniform", take N_{train} i.i.d. uniform samples in Ω.
- If kind = "lhs", construct an LHS design with N_{train} points as above.
- If kind = "blue", apply farthest point sampling (FPS) to obtain N_{train} blue-noise points in Ω.
- If kind = "adaptive", construct an adaptive point set based on the gradient magnitude of f as described above.

In all cases, we optionally augment the interior points with boundary nodes and, if necessary, downsample the combined set by FPS to obtain exactly N_{train} points. The corresponding training data are then

$$X_{\text{train}} = \{x_i\}_{i=1}^{N_{\text{train}}}, y_i = f(x_i), i = 1, \dots, N_{\text{train}}$$

For evaluation and visualisation, we also introduce a regular tensor-product grid of size $n_g \times n_g$,

$$G = \{(x_i^{\text{test}}, y_j^{\text{test}})\}_{i,j=1}^{n_g} \subset \Omega$$

on which the reference values $f\left(x_i^{\text{test}}, y_j^{\text{test}}\right)$ are computed. This grid is used to quantify the interpolation errors and to produce contour and surface plots of the true and reconstructed fields.

Numerical Experiments.

To rigorously evaluate the effectiveness of the proposed RBF–based interpolation framework, we conduct a systematic comparison against the multivariate polynomial interpolation scheme implemented in the MINTERPY package [7], [4]. MINTERPY generates global polynomial approximants using algebraic multi–index sets and unisolvent node configurations, whereas the proposed method employs radial basis functions enhanced through whitening transformations, nearest–neighbor length–scale normalization, and cross–validated selection of kernel types and shape parameters. Both approaches are applied to the same benchmark function under identical sampling conditions, enabling a direct and unbiased assessment of reconstruction accuracy, numerical stability, and sensitivity to sampling density and geometric complexity. The comparative study focuses on the error norms(L_1, L_2, L_∞) [11] computed over a dense evaluation grid. The results indicate that the RBF model consistently achieves higher accuracy in regions characterized by sharp anisotropy, steep gradients, and non–polynomial behavior, whereas MINTERPY performs competitively on smooth components that exhibit near–global polynomial structure.

All numerical experiments are carried out using Latin Hypercube Sampling on the unit square $[0,1]^2$. To analyze the impact of sample density, we consider three training set sizes: $N = 100, N = 500$, and $N = 1000$. For each configuration, we compute interpolation errors using the aforementioned norms and summarize the results in corresponding comparative tables, thereby elucidating how accuracy scales with increasing dataset resolution.

- **For $N = 100$**

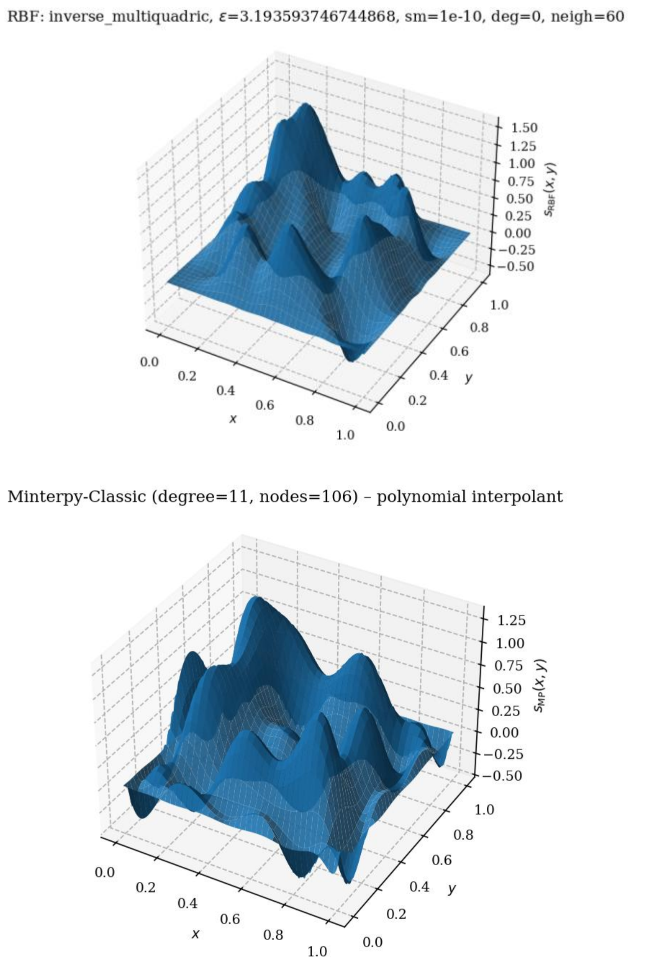
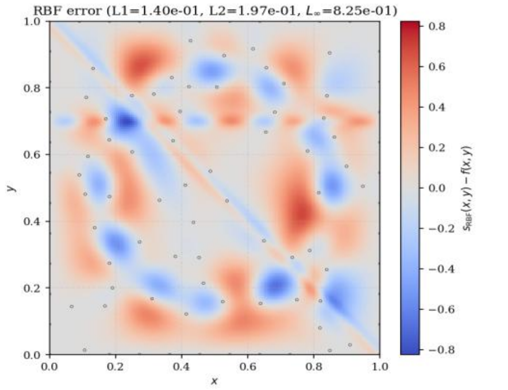
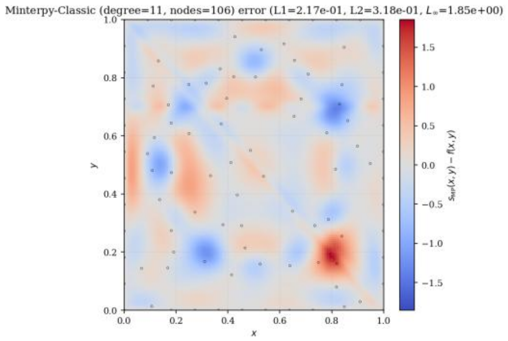


Figure 3. Comparison of the reconstructed surface using the proposed RBF interpolation method and the MINTERPY polynomial interpolation scheme for N=100 training points.





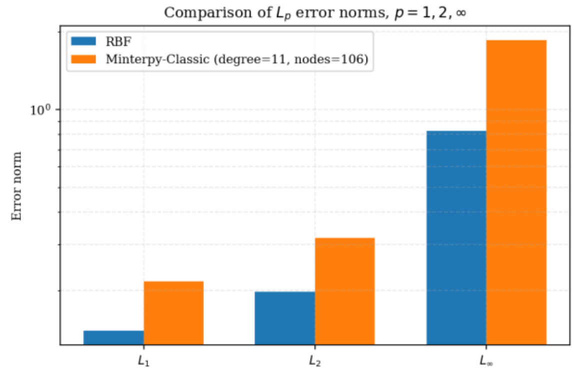
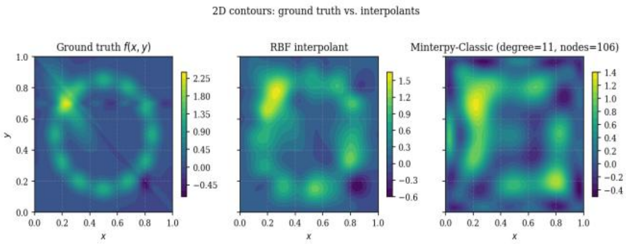


Figure 4. Evaluation of interpolation accuracy based on (L_1, L_2, L_∞) norms.



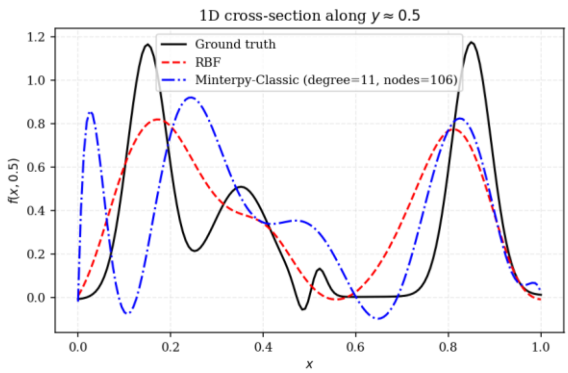


Figure 5. 1D cross–section comparison along y≈0.5 between the ground truth function, the RBF reconstruction, and the MINTERPY polynomial interpolant.

Method	L_1	L_2	L_∞
RBF	1.397e − 01	1.967e − 01	8.251e − 01
Minterpy	2.167e − 01	3.183e − 01	1.853e + 00

Tabela 1: Error norms for the two interpolation methods.

- **For N=500**

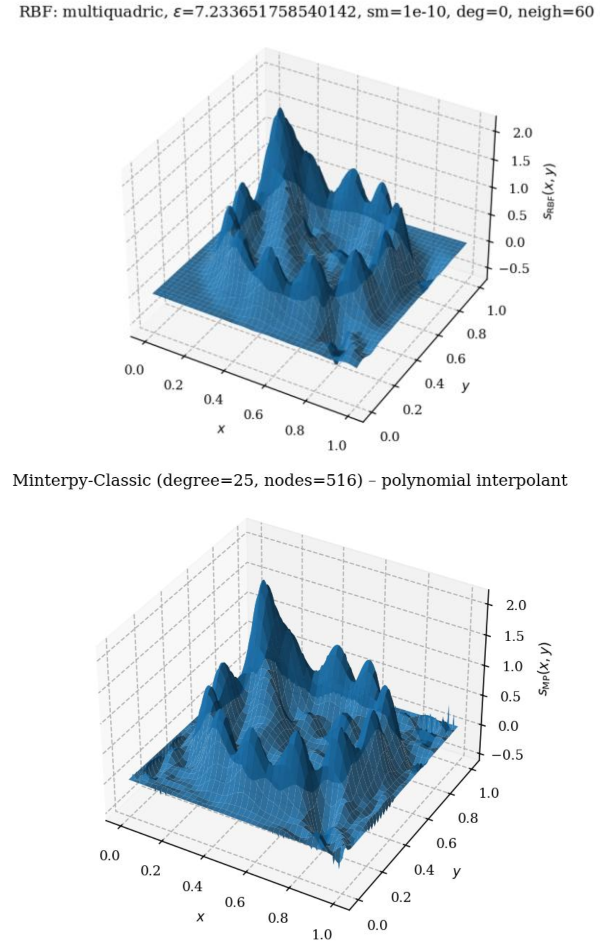
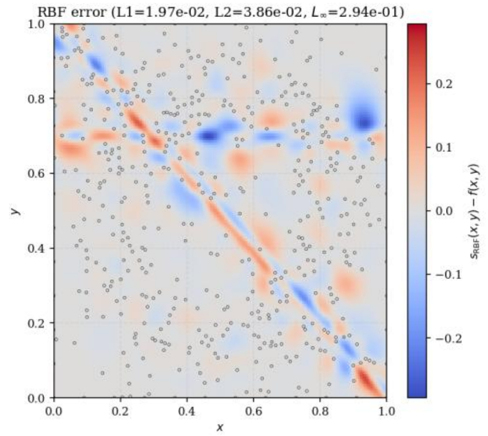
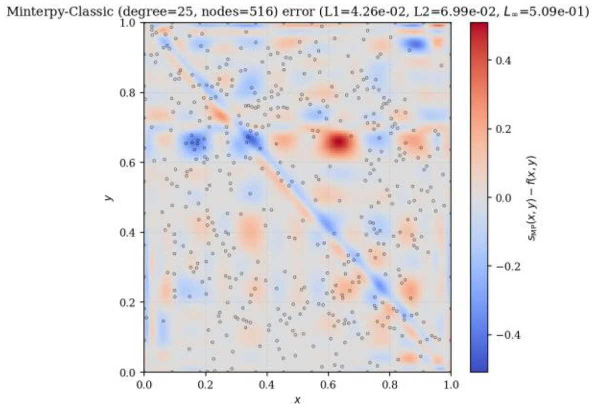


Figure 6. *Comparison of the reconstructed surface using the proposed RBF interpolation method and the MINTERPY polynomial interpolation scheme for N=500 training points.*





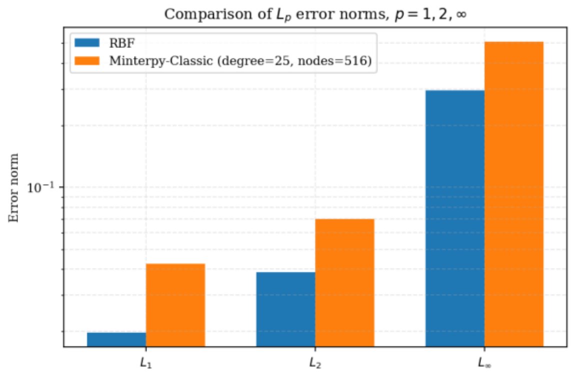
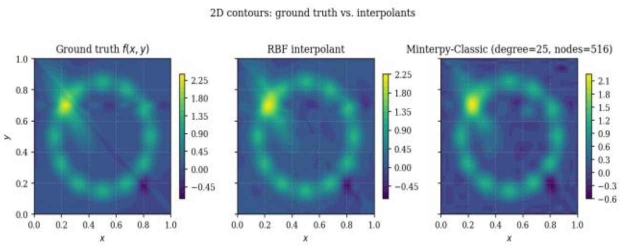


Figure 7. Evaluation of interpolation accuracy based on (L_1, L_2, L_∞) norms.



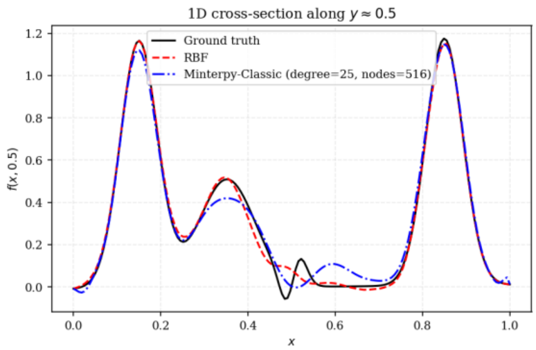


Figure 8. 1D cross–section comparison along y≈0.5 between the ground truth function, the RBF reconstruction, and the MINTERPY polynomial interpolant (N=500).

Method	L_1	L_2	L_∞
RBF	1.972e − 02	3864e − 02	2943e − 01
Minterpy	4.264 − 02	6.993e − 02	5.094e − 01

Tabela 2: Error norms for the two interpolation methods.

- For N=1000

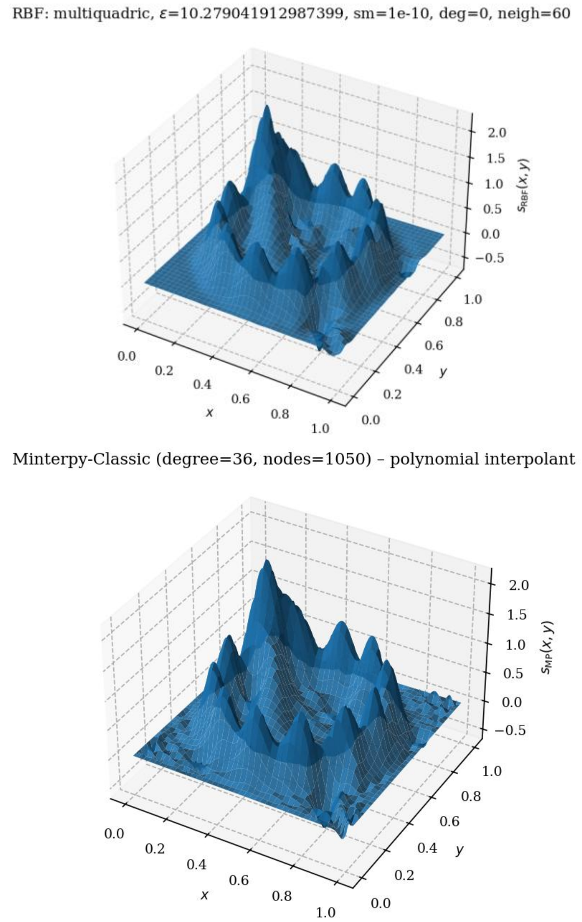
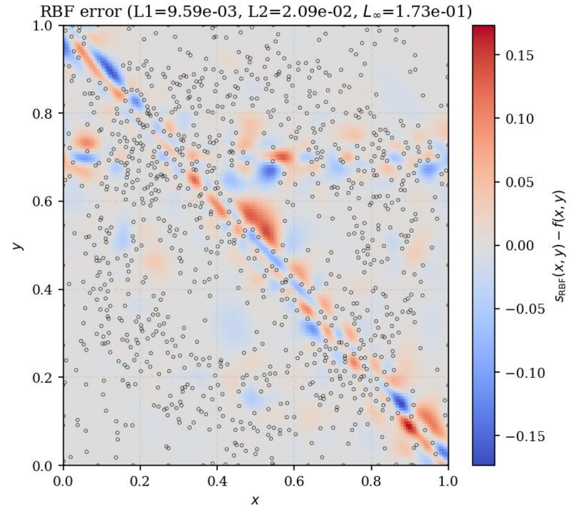
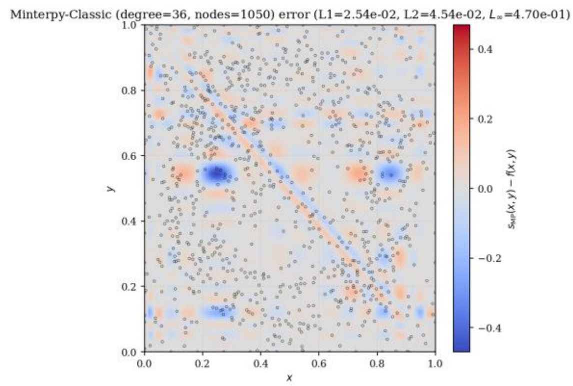


Figure 9. *Comparison of the reconstructed surface using the proposed RBF interpolation method and the MINTERPY polynomial interpolation scheme for N=500 training points.*





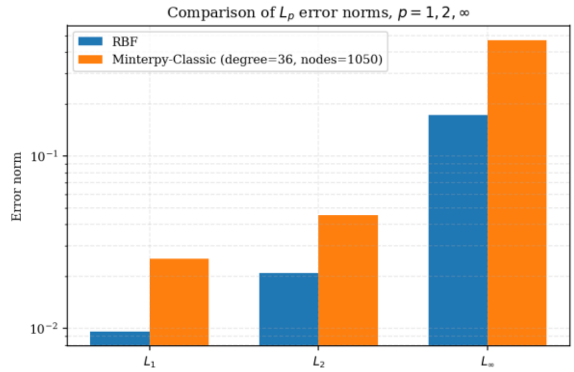
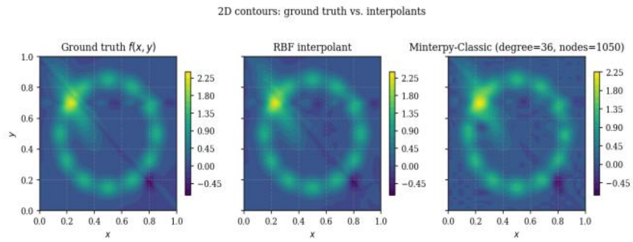


Figure 10. Evaluation of interpolation accuracy based on (L_1, L_2, L_∞) norms.



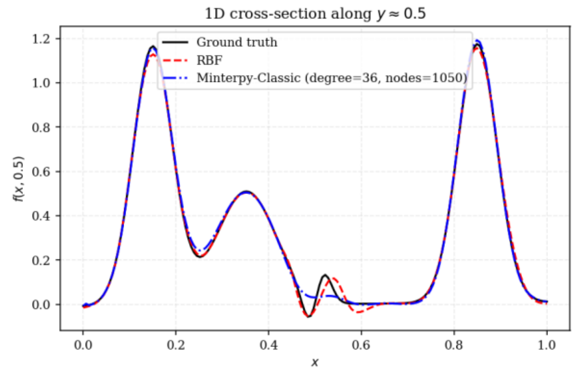


Figure 8. 1D cross–section comparison along y≈0.5 between the ground truth function, the RBF reconstruction, and the MINTERPY polynomial interpolant (N=1000).

Method	L_1	L_2	L_∞
RBF	$9.5952e-03$	$2.090e-02$	$1.734e-01$
Minterpy	$2.537-02$	$4.538e-02$	$4.696e-01$

Tabela 3: Error norms for the two interpolation methods.

Conclusion

In this work, we presented a stability–enhanced RBF interpolation framework designed to improve numerical robustness and accuracy when reconstructing complex multivariate functions on the domain$[0,1]^2$ By integrating whitening transformations, nearest–neighbour length–scale normalization, and cross–validated selection of kernel and shape parameters, the method addresses several well–known challenges of classical RBF schemes, including sensitivity to scale, instability at small shape parameters, and inconsistent performance under irregular sampling. Comparative numerical experiments demonstrated that the proposed method consistently outperforms the polynomial interpolation approach implemented in the MINTERPY library, particularly in regions characterized by high anisotropy, sharp gradients, and non–polynomial behaviour.

While MINTERPY remains competitive on globally smooth components due to its algebraic basis construction, the RBF method exhibits superior adaptability across heterogeneous geometric features. These findings indicate that hybrid strategies combining global polynomial structure with locally adaptive radial models may offer further improvements, especially for high–dimensional data or manifold- based geometries.

Future work will focus on extending the proposed approach in three directions:

1. **Adaptive multi–resolution refinement**, where sampling density and kernel parameters evolve dynamically according to local error indicators.
2. **Hybrid RBF–polynomial formulations**, designed to balance global smoothness with local feature resolution while reducing computational cost.
3. **Generalization to non–Euclidean manifolds and implicit geometric representations**, particularly in the context of level–set modelling, surface deformation, and PDE–driven simulations.

Overall, the proposed framework provides a promising step toward more stable, scalable, and high- fidelity numerical interpolation methods, with direct applicability to scientific computing, geometric modelling, and data–driven physical simulation.

References:

[1] Fasshauer, G., 2007. Meshfree Approximation Methods with Matlab. World Scientific Publishing Co.,Pages 53-61.

[2] M. D. McKay, R. J. Beckman and W. J. Conover ,A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code,Source: Technometrics, Vol. 21, No. 2 (May, 1979), pp. 239-245 https://doi.org/10.2307/1268522

[3] Yuval Eldar, Michael Lindenbaum, Member, IEEE, Moshe Porat, Senior Member, IEEE, and Yehoshua Y. Zeevi, The Farthest Point Strategy for Progressive Image Sampling https://doi.org/10.1080/10789669.2009.10390836

[4] U. HERNANDEZ ACOSTA, S. KRISHNAN THEKKE VEETTIL, D. WICAKSONO, AND M. HECHT, MINTERPY - multivariate interpolation in python, https://github.com/casus/minterpy/, (2021).

[5] Buhmann, M. D. (2003).Radial Basis Functions: Theory and Implementations, Cambridge University Press.

[6] Kessy, A., Lewin, A., & Strimmer, K. (2018). Optimal whitening and decorrelation https://doi.org/10.48550/arXiv.1512.00809

[7] HECHT, M., HOFFMANN, K. B., CHEESEMAN, B. L. & SBALZARINI, I. F. (2018) Multivariate Newton interpolation. arXiv preprint arXiv:1812.04256.

[8] Wendland, H. (2005).*Scattered Data Approximation* Cambridge University Press.

[9] Strikwerda, J. C. (2004) *Finite Difference Schemes and Partial Differential Equations* SIAM.

[10] Stone, M. (1974) Cross-Validatory Choice and Assessment of Statistical Predictions. Journal of the Royal Statistical Society: Series B (Methodological), 36, 111-147.

[11] Kreyszig, E. (1978) *Introductory Functional Analysis with Applications* Wiley.