LDD-RFM: Learnable Domain Decomposition for Random Feature Models via Variable Projection

Zhaohui Fu[*]
Department of Mathematics
National University of Singapore
Singapore

Duanyu Feng[*]
Sichuan University
China

Yangshuai Wang[†]
Department of Mathematics
National University of Singapore
Singapore
yswang@nus.edu.sg

Abstract

Random Feature Models (RFMs) are attractive PDE surrogates because their linear-in-features form reduces training to regularized least squares. Standard RFMs, however, use one stationary spectrum over the whole domain, creating a bandwidth dilemma on heterogeneous PDE solutions: fine scales resolve sharp local gradients but oscillate in smooth regions, while coarse scales miss interface-dominated derivatives. We propose **LDD-RFM**, a learnable domain-decomposition framework whose local experts remain linear RFMs and whose nonlinear variables specify compact-support partition-of-unity (PoU) gates and local spectral scales. For fixed gates and spectra, expert coefficients are eliminated by ridge regression through variable projection, yielding a differentiable reduced objective with explicit linear-solve and partition-smoothness controls. We prove an H^1 error decomposition in which PoU sharpness enters through a product-rule term. Controlled regression and PDE experiments show that LDD-RFM reduces H^1 errors by $2\times$–$13\times$ over global and fixed-partition RFM baselines at equal feature counts. We report the corresponding runtimes. In separate tests, the regression error gap persists with a direct ridge solve, and derivative errors stabilize as the evaluation grid is refined.

1 Introduction

Random Feature Models (RFMs) [Rahimi and Recht, 2007, 2008, Chen et al., 2022, Nelsen and Stuart, 2024, Deng and He, 2025, Kuvakin et al., 2025] approximate partial differential equation (PDE) solution fields by linear models in randomized basis functions. By replacing exact kernel matrices with finite feature maps, RFMs reduce training to regularized least squares, avoid cubic kernel solves, and allow the number of features to be set directly [Carratino et al., 2018]. These properties make them useful mesh-free surrogates for physics-informed PDE solvers [Karniadakis et al., 2021, Brunton and Kutz, 2024].

Despite their efficiency, standard RFMs can struggle on elliptic and interface PDEs with localized boundary layers, thin transition zones, or material interfaces [Karniadakis et al., 2021, Quarteroni et al., 2025]. A single global spectral scale creates a bandwidth dilemma: fine scales resolve sharp local features but can induce non-physical oscillations in smooth regions, whereas coarse scales miss critical gradients (Fig. 1). Partition-of-unity (PoU) localization [Babuška and Melenk, 1997, Wendland, 2002] addresses this by blending local experts with region-specific bandwidths [Gramacy and Lee, 2008, Park et al., 2011], but fixed patch geometries are brittle. If an unknown interface cuts through a prescribed patch, the local expert must fit incompatible behavior on one support, often amplifying derivative error and making H^1 accuracy sensitive to manual subdomain placement.

[*]These authors contributed equally to this work and share first authorship.
[†]Corresponding author: yswang@nus.edu.sg.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

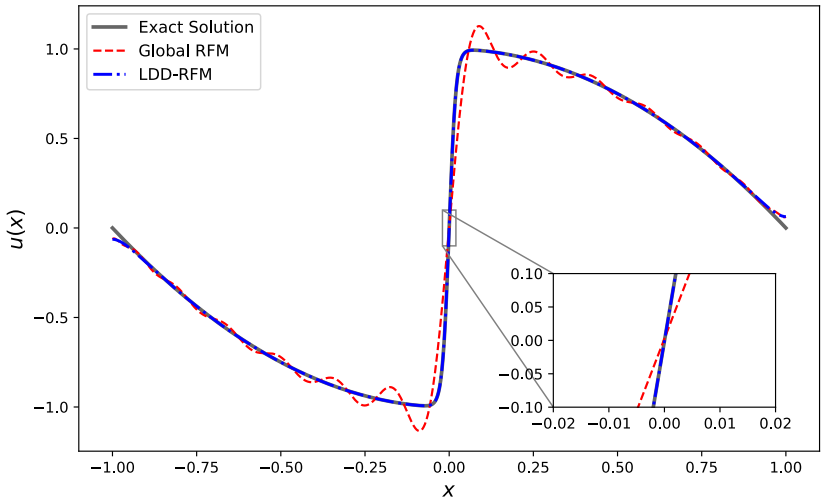


Figure 1: **Reaction–diffusion with a sharp layer.** The global RFM (red dashed) oscillates near the $x = 0$ layer and under-resolves the $\mathcal{O}(\sqrt{\epsilon})$ transition. LDD-RFM adapts local scales and fits both the layer and smooth profile. The vertical marker indicates the layer center.

We propose **LDD-RFM**, a Learnable Domain Decomposition for Random Feature Models. Compact-support PoU gates parameterize *where* capacity is localized, while region-wise RFMs set *which spectral scale* each region uses. Its distinctive choice is to pair learnable PoU geometry with *linear* local random-feature experts and variable projection: for fixed gates and spectra, the expert coefficients are eliminated by a ridge solve, leaving only gate geometry and spectral scales in the reduced nonlinear problem. This keeps the model in the RFM regime while making approximation quality, partition smoothness, and inexact linear algebra explicit in the analysis.

We provide an H^1 error decomposition showing that global gradient accuracy balances local approximation quality against an interface-consistency term controlled by PoU smoothness. This explains why overly sharp partitions can degrade derivative accuracy and motivates PoU regularization. We test both controlled regression targets that isolate stationary-bandwidth failure and residual-trained PDE tasks with sharp layers, material interfaces, and moving fronts. Across these settings, LDD-RFM learns partitions aligned with latent irregularities (Appendix D.7) and recovers sharp PDE layers without the global oscillations seen in stationary RFMs (Fig. 1), reducing H^1 errors by $2\times$–$13\times$ for RFM comparisons at equal feature counts. LDD-RFM complements classical one-shot solvers in related inverse or repeated-query problems where learned geometry can be reused. The inverse-loop comparisons include a finite-difference baseline (Table 14) and a matched adjoint FEM baseline (Appendix D.14).

Our main contributions are as follows:

1. We introduce LDD-RFM, a PoU-gated mixture of linear RFMs whose region-specific spectral scales and gates are trained by variable projection, eliminating expert coefficients at every outer step.
2. We derive an H^1 decomposition linking derivative accuracy to local approximation and PoU interface consistency, and analyze how compact support makes the linear-solve cost scale with local overlap.
3. We show on heterogeneous regression and residual-trained PDE benchmarks that LDD-RFM learns useful partitions and improves H^1 accuracy at matched feature counts, especially for sharp layers, material interfaces, and moving fronts.

2 Related work

Positioning relative to multi-scale RFMs and MscaleDNN. Standard RFMs approximate shift-invariant kernels with Monte Carlo features and scalable ridge solves [Rahimi and Recht, 2007, 2008,

Segata and Blanzieri, 2010, Carratino et al., 2018, Chen et al., 2022, Nelsen and Stuart, 2024, Dong and Li, 2021, Kuvakin et al., 2025, Hu et al., 2025], but stationary spectra mismatch spatially varying regularity. One response is to give every point access to many bandwidths: global multi-resolution encodings use Fourier features, periodic activations, band-limited filters, or hash grids [Tancik et al., 2020, Sitzmann et al., 2020, Lindell et al., 2022, Müller et al., 2022], while MscaleDNN-style frequency-grouped networks [Liu et al., 2020, Wang et al., 2020] feed a fixed bank of radial frequency scales into one global MLP. LDD-RFM instead localizes the domain first, then assigns each ridge-solved expert a learned local scale on its own support. Domain-decomposition PDE solvers, including XPINNs, FBPINNs, ELM-FBPINNs, and local ELM methods [Jagtap et al., 2020, Moseley et al., 2023, Anderson et al., 2024, Dong and Li, 2021, Bonfanti et al., 2025], usually use prescribed subdomains or nonlinear local networks. Mixture-of-experts and neural-operator models use gating for conditional computation or amortized solution maps [Jacobs et al., 1991, Shazeer et al., 2017, Hu et al., 2023, Li et al., 2021, Hao et al., 2023, Takamoto et al., 2022]. Neural operators learn maps across many PDE instances, whereas our RFM setting targets derivative-sensitive approximation of a single heterogeneous solution field or a small family with a fixed number of features.

Partition of unity, APINN-style gating, and variable projection. Classical PoU methods assemble global approximants from local pieces [Babuška and Melenk, 1997, Wendland, 2002, Toselli and Widlund, 2005]. We use this principle as a learnable smooth decomposition for RFMs rather than a fixed mesh-free cover. APINN and related learned-decomposition PINNs [Hu et al., 2023] already learn gates, while ELM-FBPINN and local ELM solvers [Anderson et al., 2024, Dong and Li, 2021] already use random-feature local solvers on prescribed windows. LDD-RFM combines learnable compact-support PoU geometry with ridge-solved linear experts, producing a reduced objective over geometry and spectra only. This separable nonlinear least-squares view [Golub and Pereyra, 2003] makes PoU sharpness, active overlap, and inexact linear-solve effects explicit.

3 Background: random feature models and partition of unity

RFMs approximate a PDE solution field by $u(\boldsymbol{x}) \approx \Phi(\boldsymbol{x})^\top W$, but a single stationary basis struggles when smooth regions coexist with sharp localized irregularities. Localized PoU schemes rescale each region and blend local experts. For a patch $B_m = \prod_i [a_i^{(m)}, b_i^{(m)}]$, the affine normalization

$$\tilde{x}_i^{(m)} = \frac{2x_i - (a_i^{(m)} + b_i^{(m)})}{b_i^{(m)} - a_i^{(m)}}, \qquad i = 1, \dots, d. \tag{1}$$

keeps spectral scales dimensionless and improves conditioning. Given a fixed PoU $\{\psi_m\}$ with $\sum_m \psi_m(\boldsymbol{x}) \equiv 1$, local RFM features are

$$\phi_{m,j}(\boldsymbol{x}) = \sigma_{\text{feat}}\left(\boldsymbol{k}_j^{(m)} \cdot \tilde{\boldsymbol{x}}^{(m)} + \beta_j^{(m)}\right), \quad j = 1, \dots, K_m,$$

where σ_{feat} is the feature activation and frequencies are sampled with local scale $R_m > 0$. The fixed-PoU predictor is trained by ridge regression:

$$u(\boldsymbol{x}) = \sum_{m=1}^{M} \psi_m(\boldsymbol{x}) \Phi_m(\boldsymbol{x})^\top W_m. \tag{2}$$

$$\min_{\{W_m\}} \frac{1}{n} \sum_{i=1}^{n} \left(u(\boldsymbol{x}_i) - y_i\right)^2 + \lambda \sum_{m=1}^{M} \|W_m\|_2^2. \tag{3}$$

Bounded overlap yields a block-structured linear system, but fixed geometries and manually tuned R_m are brittle when latent interfaces are unknown.

4 Method: LDD-RFM

Fixed PoU localization relies on a prescribed partition, which is brittle when interface geometry is unknown. **LDD-RFM** instead learns differentiable PoU gates and region-wise spectral scales with a fixed total number of features while keeping each expert linear and ridge-solved. This spectral–geometric parameterization lets the model allocate bandwidth where the solution is least regular while retaining the linear-solver structure of RFMs.

4.1 Compact-support PoU

Rather than prescribing a fixed tessellation, we learn a *compact-support* PoU with structurally bounded overlap. Let $\chi : [0, \infty) \to [0, 1]$ be a C^1 bump function with unit support, e.g.,

$$\chi(r) = \begin{cases} \frac{1}{2}\left(1 + \cos(\pi r)\right), & 0 \le r < 1, \\ 0, & r \ge 1. \end{cases} \tag{4}$$

We parameterize M learnable patches by centers $\{\mu_m\}$ and radii $\{\rho_m \in \mathbb{R}^d_+\}$, collected in θ. On rectangular domains $\Omega = \prod_i [\ell_i, u_i]$, we use

$$\mu_{m,i} = \ell_i + (u_i - \ell_i)\, \sigma(\hat{\mu}_{m,i}), \qquad i = 1, \dots, d,$$

and prevent vanishing radii by

$$\rho_m \;=\; \rho_{\min}\mathbf{1} + \mathrm{softplus}(\hat{\rho}_m), \qquad \rho_{\min} > 0,$$

where σ is the logistic map and hatted variables are unconstrained. We initialize centers by Sobol or farthest-point sampling and radii large enough to avoid early collapse. When the input domain is space-time (evolution PDEs), we treat $(\boldsymbol{x}, t)$ jointly as the d-dimensional PoU input. The centers and radii then span space and time, allowing experts to specialize to spatial regions and temporal phases simultaneously.

For $m = 1, \dots, M$, we define unnormalized nonnegative weights

$$w_m(\boldsymbol{x}; \theta) = \chi\big(\big\|\mathrm{diag}(\rho_m)^{-1}(\boldsymbol{x} - \mu_m)\big\|_2\big)\,.$$

With background weight $w_0(\boldsymbol{x}) \equiv 1$, normalized gates

$$\alpha_m(\boldsymbol{x}; \theta) = \frac{w_m(\boldsymbol{x}; \theta)}{\sum_{k=0}^{M} w_k(\boldsymbol{x}; \theta)}, \qquad m = 0, 1, \dots, M, \tag{5}$$

are nonnegative, sum to one, and are uniformly well-posed because the denominator is at least one. Compact support makes each local gate vanish outside its patch. The construction is C^1 in $\boldsymbol{x}$ and differentiable in θ on bounded parameter sets. The background expert has the constant feature $\Phi_0 \equiv 1$, but its normalized gate α_0 is spatially varying through the denominator in (5). Since $w_0 \equiv 1$, this background gate is active on all of Ω. In the analysis, f_0 is the best constant on Ω, making the corresponding local-error term conservative.

Bounded overlap. To quantify the number of *local* (non-background) experts activated at any point, define

$$q := \sup_{\boldsymbol{x} \in \Omega} \left|\{m \in \{1, \dots, M\} :\; w_m(\boldsymbol{x}; \theta) > 0\}\right|. \tag{6}$$

Throughout the cost analysis, q counts only non-background experts. Each sample therefore touches at most q local feature blocks plus the single background column, inducing a row block-sparse ridge design (Sec. 4.3). With full local overlap, $q = M$.

Unused experts. When M exceeds the number of latent regimes, redundant experts are typically visible through small integrated gate mass $\bar{\alpha}_m = N^{-1} \sum_i \alpha_m(\boldsymbol{x}_i)$ and ridge-shrunk coefficients. We retain all experts in the reported experiments, including those with small gate mass.

Sharpness control. The H^1 bound in our main Theorem (cf. Theorem 5.3) contains an interface-consistency term amplified by $\kappa_\infty^2 = \left\| \sum_{m=0}^{M} \|\nabla \alpha_m(\cdot)\|_2^2 \right\|_{L^\infty(\Omega)}$. To control this sharpness, we penalize an L^p surrogate with $p > d$. We use $p = 8$ in the experiments. The choice $p = 2$ gives the standard gradient-energy penalty. The regularizer is

$$\mathcal{R}_{\mathrm{PoU}}(\theta) = \sum_{m=0}^{M} \|\nabla_{\boldsymbol{x}} \alpha_m(\cdot; \theta)\|^2_{L^p(\Omega)}. \tag{7}$$

We estimate (7) by Monte Carlo over collocation points and absorb domain-volume constants into γ. Overly sharp transitions inflate this penalty and degrade derivative-sensitive accuracy unless supported by accurate local fits.

4.2 Spectral-adaptive local experts

Each learned region has a local random feature expert with its own spectral scale R_m, mitigating the single-spectrum bandwidth dilemma. Using normalized coordinates $\tilde{\boldsymbol{x}}^{(m)}$, the m-th feature map $\Phi_m(\boldsymbol{x}) \in \mathbb{R}^{K_m}$ is

$$\phi_{m,j}(\boldsymbol{x}) = \sigma_{\text{feat}}\left(\boldsymbol{k}_{m,j}^\top \tilde{\boldsymbol{x}}^{(m)} + \beta_{m,j}\right), \quad j = 1, \dots, K_m,$$
$$\boldsymbol{k}_{m,j} = R_m\, \boldsymbol{\xi}_{m,j}, \quad \boldsymbol{\xi}_{m,j} \sim p_0, \quad \beta_{m,j} \sim \mathcal{U}[0, 2\pi],$$

where p_0 is a base distribution (e.g., $\mathcal{N}(\mathbf{0}, I)$) and $R_m > 0$ is learnable. Equivalently, $\boldsymbol{k}_{m,j} \sim p_{R_m}$, the pushforward of p_0 under scaling by R_m. The reparameterization keeps $\{\boldsymbol{\xi}_{m,j}, \beta_{m,j}\}$ fixed during training, enabling gradients in R_m without resampling noise.

This yields a *spectral–geometric decoupling*: gates $\alpha_m(\cdot;\theta)$ capture *where* heterogeneity occurs, while scales $\{R_m\}$ control *which frequencies* each expert represents. Sharp-transition regions may prefer larger R_m, and smooth regions smaller R_m, reducing unnecessary H^1 oscillations.

The global predictor is assembled by PoU superposition of local linear experts:

$$u(\boldsymbol{x}; \mathcal{C}, \Theta) = \sum_{m=0}^{M} \alpha_m(\boldsymbol{x}; \theta)\, \Phi_m(\boldsymbol{x})^\top \boldsymbol{c}_m, \tag{8}$$

where $\Phi_0(\boldsymbol{x}) \equiv 1$, $\mathcal{C} = \{\boldsymbol{c}_m\}_{m=0}^M$, and $\Theta = \{\theta, \{R_m\}_{m=1}^M\}$.

4.3 Hybrid optimization via variable projection

Given data $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, we minimize a regularized empirical risk over linear expert weights $\mathcal{C} = \{\boldsymbol{c}_m\}_{m=0}^M$ and nonlinear parameters $\Theta = \{\theta, \{R_m\}_{m=1}^M\}$:

$$\min_{\mathcal{C},\Theta}\ \mathcal{J}(\mathcal{C}, \Theta) := \frac{1}{n}\sum_{i=1}^{n} \big(u(\boldsymbol{x}_i; \mathcal{C}, \Theta) - y_i\big)^2 + \lambda \sum_{m=0}^{M} \|\boldsymbol{c}_m\|_2^2 + \gamma\, \mathcal{R}_{\text{PoU}}(\theta). \tag{9}$$

Here $\lambda > 0$ is ridge regularization and $\gamma > 0$ weights the PoU sharpness penalty. For linear PDE residual objectives, the same ridge form is obtained by replacing the value design matrix below with the corresponding derivative/residual feature matrix and taking $\mathbf{y}$ to be the sampled forcing and boundary data.

For fixed Θ, (9) is a strongly convex quadratic in $\mathcal{C}$ with unique minimizer $\mathcal{C}^\star(\Theta)$. We define the reduced objective $\tilde{\mathcal{J}}(\Theta) := \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta)$, and optimize it by alternating a ridge solve and a nonlinear update (Algorithm 1).

For any Θ, let $\boldsymbol{\Psi}(\Theta) \in \mathbb{R}^{n \times K_{\text{tot}}}$ denote the gated design matrix whose i-th row is

$$[\alpha_0(\boldsymbol{x}_i)\Phi_0(\boldsymbol{x}_i)^\top, \dots, \alpha_M(\boldsymbol{x}_i)\Phi_M(\boldsymbol{x}_i)^\top],$$

with $K_{\text{tot}} = 1 + \sum_{m=1}^M K_m$ after including the background constant.

Algorithm 1 LDD-RFM training via variable projection

Require: Data $\mathcal{D}$, experts M, features $\{K_m\}$, ridge λ, PoU weight γ, init. $\Theta^{(0)}$, tolerance ε, iterations T
1: **for** $t = 0, \dots, T-1$ **do**
2: Evaluate the gates and local features to assemble the gated design $\Psi(\Theta^{(t)})$
3: **C-step:** solve $(\Psi^\top\Psi + n\lambda I)c_{\text{flat}} = \Psi^\top y$ to tolerance ε
4: Unflatten c_{flat} into $\mathcal{C}^{(t+1)} = \{c_m^{(t+1)}\}_{m=0}^M$
5: **Θ-step:** take an Adam step using $\nabla_\Theta \mathcal{J}(\mathcal{C}^{(t+1)}, \Theta)\big|_{\Theta=\Theta^{(t)}}$, including $\gamma\mathcal{R}_{\text{PoU}}$
6: **end for**
7: **return** $\Theta^{(T)}, \mathcal{C}^{(T)}$

Algorithm 1 uses inexact CG for the coefficients and Adam for the outer step, avoiding the extra ridge solves required by a line search. Theorem 5.5 assumes exact coefficient solves and sufficient decrease in the outer step. Its proof does not cover Adam with inexact CG. Proposition B.4 bounds the coefficient and reduced-gradient errors in terms of the CG residual.

Linear ridge solve ($\mathcal{C}$-step). For fixed Θ, the concatenated coefficients satisfy the SPD ridge system

$$\left(\mathbf{\Psi}(\Theta)^\top \mathbf{\Psi}(\Theta) + n\lambda \mathbf{I}\right) \mathbf{c}_{\text{flat}} = \mathbf{\Psi}(\Theta)^\top \mathbf{y}. \tag{10}$$

We solve (10) mainly by CG using row block-sparsity from bounded PoU overlap. Appendix D.5 compares CG with a direct Cholesky solve.

Cost of the $\mathcal{C}$-step. Let $K_{\text{loc}} = \max_m K_m$ and let q denote the local-expert overlap in (6), excluding the background constant. Feature construction costs $O(nqK_{\text{loc}}d)$ plus gate evaluation, while each CG matvec costs $O(nqK_{\text{loc}})$ and stores $O(nqK_{\text{loc}} + K_{\text{tot}})$. Thus the favorable regime depends on active local overlap q, with dense local overlap recovering $q = M$. In practice, keep overlap moderate, avoid unnecessary experts, and warm-start gate parameters only when successive problems plausibly share interface structure.

Geometry and spectral update (Θ-step). We update Θ on the reduced objective. With an exact $\mathcal{C}$-step, the envelope theorem gives $\nabla_\Theta \tilde{\mathcal{J}}(\Theta) = \nabla_\Theta \mathcal{J}(\mathcal{C}, \Theta)\big|_{\mathcal{C}=\mathcal{C}^\star(\Theta)}$, i.e., one differentiates $\mathcal{J}$ while treating $\mathcal{C}^\star(\Theta)$ as constant. With inexact solves, we use the same expression with approximate $\mathcal{C}$. During the Adam update, the PoU penalty discourages steep gates.

Inexact variable projection. Let $A(\Theta) = \Psi(\Theta)^\top \Psi(\Theta) + n\lambda I$ and $b(\Theta) = \Psi(\Theta)^\top y$. If CG returns $\widehat{c}$ with residual $r = A(\Theta)\widehat{c} - b(\Theta)$, then

$$\|\widehat{c} - c^\star(\Theta)\|_2 \le \frac{\|r\|_2}{n\lambda}, \qquad \|\nabla_\Theta \mathcal{J}(\widehat{c}, \Theta) - \nabla_\Theta \tilde{\mathcal{J}}(\Theta)\|_2 \le \frac{L_{\Theta c}}{n\lambda}\|r\|_2, \tag{11}$$

whenever $\nabla_\Theta \mathcal{J}(c, \Theta)$ is Lipschitz in c with constant $L_{\Theta c}$ on the relevant bounded parameter set. Thus the CG tolerance controls the deterministic reduced-gradient bias. Appendix B.2 gives a condition for these errors to be summable.

5 Theoretical analysis

We analyze approximation error under PoU localization (Sec. 5.1) and optimization guarantees for variable projection (Sec. 5.2).[3]

5.1 Approximation theory

We derive a global H^1 bound for a PoU-assembled predictor, decomposing error into local H^1 approximation and a product-rule interface-consistency term. The only PoU-induced amplification factor in the bound is κ_∞, and Lemma B.2 relates it to the actual overlap and bump smoothness through $(q_\nabla + 1)L_w$.

Assumption 5.1 (PoU regularity and a sharpness constant)**.** Let $\Omega \subset \mathbb{R}^d$ be a bounded Lipschitz domain. The gating functions $\{\alpha_m\}_{m=0}^M$ form a partition of unity on Ω: $0 \le \alpha_m(\boldsymbol{x}) \le 1$ and $\sum_{m=0}^M \alpha_m(\boldsymbol{x}) \equiv 1$ for all $\boldsymbol{x} \in \Omega$. Assume $\alpha_m \in W^{1,\infty}(\Omega)$ for all m and define the PoU sharpness constant

$$\kappa_\infty^2 \;:=\; \left\| \sum_{m=0}^M \|\nabla \alpha_m(\cdot)\|_2^2 \right\|_{L^\infty(\Omega)} \;<\; \infty. \tag{12}$$

For the compact-support bump construction with $w_0 \equiv 1$, κ_∞ admits an explicit bound in terms of bump smoothness and patch radii (Lemma B.2).

[3]We provide proof details in Appendix B.

Assumption 5.2 (Local approximation on supports)**.** Let $u^* \in H^1(\Omega)$. For each $m \in \{0, 1, \dots, M\}$, there exists a local expert f_m such that on $\mathrm{supp}(\alpha_m)$,

$$\begin{aligned} \|u^* - f_m\|_{L^2(\mathrm{supp}(\alpha_m))} \leq & \epsilon_{m,0}, \\ \|\nabla(u^* - f_m)\|_{L^2(\mathrm{supp}(\alpha_m))} \leq & \epsilon_{m,1}. \end{aligned} \tag{13}$$

For the background gate $m = 0$ (with $\Phi_0(\boldsymbol{x}) \equiv 1$), take f_0 as the best constant approximation of u^* on $\mathrm{supp}(\alpha_0)$, defining $(\epsilon_{0,0}, \epsilon_{0,1})$.

Assumption 5.2 decouples local approximation quality from PoU geometry. For local RFMs, $\epsilon_{m,0}, \epsilon_{m,1}$ depend on the number of local features and their spectral scales. Combining Assumptions 5.1–5.2 gives:

Theorem 5.3 (Global H^1 error bound)**.** *For any dimension d, let $u(x) = \sum_{m=0}^{M} \alpha_m(x)\, f_m(x)$, and define local errors*

$$\begin{aligned} \epsilon_{m,0} &:= \|f_m - u^*\|_{L^2(\mathrm{supp}(\alpha_m))}, \\ \epsilon_{m,1} &:= \|\nabla(f_m - u^*)\|_{L^2(\mathrm{supp}(\alpha_m))}. \end{aligned} \tag{14}$$

Under Assumptions 5.1-5.2, we have

$$|u - u^*|_{H^1(\Omega)} \leq \Big(\sum_{m=0}^{M} \epsilon_{m,1}^2 \Big)^{1/2} \; + \; \kappa_\infty \Big(\sum_{m=0}^{M} \epsilon_{m,0}^2 \Big)^{1/2}, \tag{15}$$

where $|v|_{H^1(\Omega)} := \|\nabla v\|_{L^2(\Omega)}$ denotes the H^1 seminorm and κ_∞ is defined by (12)*.*

Remark 5.4. For either a prescribed or a learned partition, Theorem 5.3 isolates *local approximation* from a product-rule *interface-consistency* term scaled by κ_∞. For compact-support bumps, sharper partitions increase the pointwise gate-gradient energy as the effective radius decreases, amplifying L^2 mismatch near interfaces. We control this worst-case sharpness with $\kappa_p^2 := \sum_{m=0}^{M} \|\nabla \alpha_m\|_{L^p(\Omega)}^2$ for $p > d$ (implemented by (7)). We use $p = 8$ throughout the experiments. On a grid of size N, $\|\cdot\|_\infty \leq N^{1/p} \|\cdot\|_p$ bounds the maximum by the empirical norm times a grid factor. This factor is $N^{1/8} = 2.83$ for $N = 4097$ in 1D and 3.37 for $N = 129^2$ in 2D. Bounds on local random-feature approximation improve as the number of features increases [Ming and Yu, 2025].

Patch size and feature allocation. One-dimensional local RFM bounds [Ming and Yu, 2025] relate $R_m \rho_m$ to the bandwidth to be resolved and K_m to the derivative sampling error. Shrinking ρ_m can reduce local bias around an irregularity, but it also increases derivative scaling and PoU sharpness. A smaller patch helps only when the local-error reduction exceeds the interface penalty. Sharp dimension-dependent local RFM rates for $d \geq 2$ remain open.

5.2 Optimization dynamics: variable projection and convergence

The objective $\mathcal{J}(\mathcal{C}, \Theta)$ is *separable*: for fixed Θ, it is strongly convex quadratic in $\mathcal{C}$ (since $\lambda > 0$), while generally nonconvex in Θ. Variable projection eliminates the linear coefficients. The convergence theorem assumes an exact inner solve and sufficient decrease at each outer step.

For any fixed Θ, let $\mathcal{C}^\star(\Theta)$ denote the unique minimizer of the ridge subproblem, obtained by solving the SPD linear system (10). Define the reduced objective

$$\tilde{\mathcal{J}}(\Theta) := \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta). \tag{16}$$

With an exact $\mathcal{C}$-step, envelope/variable-projection results compute $\nabla_\Theta \tilde{\mathcal{J}}(\Theta)$ by differentiating $\mathcal{J}$ while treating $\mathcal{C}^\star(\Theta)$ as fixed.

Theorem 5.5 (Convergence of exact variable-projection descent)**.** *Consider the alternating procedure where (i) the $\mathcal{C}$-step is solved exactly to obtain $\mathcal{C}^\star(\Theta^k)$, and (ii) the Θ-step performs a gradient method on $\tilde{\mathcal{J}}$ with a stepsize rule that ensures sufficient decrease in the sense of* (34) *(e.g., backtracking line search). Assume $\mathcal{C}^\star$ is C^1 on an open neighborhood U of $\overline{\{\Theta^k\}}$, and $\mathcal{J}$ is C^1 near the graph of $\mathcal{C}^\star|_U$. Require also $\|\nabla \tilde{\mathcal{J}}(\Theta^{k+1})\| \leq b\|\Theta^{k+1} - \Theta^k\|$ for a fixed $b > 0$ and all k. If $\mathcal{J}$ satisfies the Kurdyka–Łojasiewicz (KL) property and the iterates remain in a bounded set, then the iterates have finite length and converge to a critical point of $\mathcal{J}$.*

Table 1: **Accuracy on four synthetic targets.** Relative L^2 error and a discrete H^1 seminorm proxy (mean±std over 5 random seeds). **Bold** denotes the best *learned* method (excluding oracles). An underline denotes the best non-oracle baseline. Each entry reports $\mathcal{E}_{L^2}$ (top) and $\mathcal{E}_{H^1}$ (bottom). ORACLE-H reports the derivative cost of discontinuous gates. ORACLE-S uses smoothed ground-truth gates.

Method	**(i)** Spectral jump	**(ii)** Multi-regime	**(iii)** Windowed interface	**(iv)** Anisotropic partition
Nyström-KRR	3.27e−1±0.02e−1 5.55e−1±0.04e−1	4.33e−2±0.11e−2 2.34e−1±0.05e−1	8.79e−2±0.15e−2 2.52e−1±0.08e−1	1.18e−2±0.03e−2 4.81e−2±0.12e−2
Gaussian FFN	1.12e−1±0.05e−1 4.80e−1±0.12e−1	3.15e−2±0.08e−2 2.90e−1±0.10e−1	5.40e−2±0.21e−2 1.88e−1±0.09e−1	1.55e−2±0.06e−2 6.10e−2±0.15e−2
SIREN	8.50e−2±0.41e−2 3.20e−1±0.25e−1	2.10e−2±0.15e−2 1.55e−1±0.08e−1	4.22e−2±0.30e−2 1.45e−1±0.11e−1	9.80e−3±0.55e−3 3.20e−2±0.20e−2
MscaleDNN	9.20e−2±0.50e−2 3.05e−1±0.18e−1	2.35e−2±0.14e−2 1.42e−1±0.08e−1	4.10e−2±0.28e−2 1.42e−1±0.10e−1	1.01e−2±0.07e−2 2.85e−2±0.17e−2
Mixture RFF	1.05e−1±0.03e−1 2.80e−1±0.10e−1	1.80e−2±0.05e−2 1.20e−1±0.04e−1	4.50e−2±0.15e−2 1.35e−1±0.05e−1	1.10e−2±0.02e−2 2.50e−2±0.10e−2
Global RFM	1.35e−1±0.01e−1 5.58e−1±0.02e−1	5.50e−2±0.03e−2 3.50e−1±0.05e−1	3.14e−2±0.05e−2 1.52e−1±0.04e−1	2.00e−2±0.02e−2 6.86e−2±0.05e−2
PoU Grid	9.70e−2±0.05e−2 1.15e0±0.02e0	1.75e−2±0.02e−2 1.79e−1±0.03e−1	8.25e−2±0.10e−2 3.83e−1±0.05e−1	1.05e−2±0.01e−2 4.98e−2±0.08e−2
PoU Oracle-H	1.20e−2±0.01e−2 2.16e0±0.01e0	1.50e−3±0.01e−3 1.85e0±0.01e0	9.50e−3±0.05e−3 1.92e0±0.02e0	5.20e−4±0.05e−4 1.10e0±0.01e0
PoU Oracle-S	7.50e−3±0.08e−3 8.80e−2±0.15e−2	1.02e−3±0.02e−3 3.10e−2±0.05e−2	7.90e−3±0.12e−3 5.50e−2±0.10e−2	4.50e−4±0.08e−4 5.20e−3±0.12e−3
LDD-RFM	**8.00e−3±0.12e−3** **9.00e−2±0.35e−2**	**1.08e−3±0.05e−3** **3.30e−2±0.15e−2**	**8.11e−3±0.22e−3** **5.89e−2±0.18e−2**	**4.76e−4±0.10e−4** **5.52e−3±0.25e−3**

Remark 5.6. Theorem 5.5 assumes the KL property, which holds for broad classes of analytic objectives. Our implementation uses Adam, optional coarse-to-fine radius scheduling, and inexact linear solves. The experiments report inner-solve tolerances and convergence traces. Thus the theorem justifies the reduced-objective formulation under exact sufficient-decrease updates, while Proposition B.4 bounds the gradient error due to the inner solve, whose tolerance we vary in the experiments. On Task (ii), the reduced objective decreased monotonically in all five exact-VP runs with Armijo backtracking (Appendix D.14).

6 Experiments

We evaluate derivative-sensitive accuracy, gate/geometry alignment, PDE layer/interface performance, and scaling against nearby domain-decomposition and operator baselines under matched protocols.

6.1 Experimental setup

Benchmarks. We use four heterogeneous synthetic targets on $\Omega = [-1, 1]^d$: a 1D frequency jump, a 1D multi-regime signal, a 2D windowed wave packet, and a 2D anisotropic partition. The known targets and their derivatives provide reference values for evaluating error. We also test four PDE benchmarks: singularly perturbed 1D reaction–diffusion with $\epsilon = 10^{-3}$, 2D Poisson with one diffusion jump ($a = 1$ vs. $a = 10$), 2D Poisson with multiple inclusions/interfaces, and space-time Allen–Cahn.[4] Tasks (i)–(iv) isolate the effect of a single stationary bandwidth. Tasks (v)–(vii) use residual least squares for linear PDEs. Task (viii) compares spatial and space-time partitions.

Baselines. Baselines cover global RFMs/kernel/FFN models, Mixture RFF, SIREN/MscaleDNN, fixed and oracle PoU variants, classical solvers, XPINN/FBPINN/APINN/ELM-FBPINN, and FNO/GNOT. Appendix C.6 lists sources, model sizes, validation grids, and the stronger PINN-style settings.

Fairness protocol. All learned methods share splits, fixed evaluation grids, validation by $\mathcal{E}_{H^1}$, and five reported seeds. Timing includes feature construction, gates, linear solves, and optimizer updates,

[4] Appendix C describes the tasks. Appendix D reports additional grid, solver-tolerance, and noise robustness studies.

Table 2: **Matched PDE/domain-decomposition baselines.** Common objectives and evaluation grids. Errors are means ± standard deviations across seeds. Times are medians in seconds.

Method	(v) React.–diff.		(vi) 2D interface		(vii) Multi-interface		(viii) Allen–Cahn	
	$\mathcal{E}_{H^1}$	Time	$\mathcal{E}_{H^1}$	Time	$\mathcal{E}_{H^1}$	Time	$\mathcal{E}_{H^1}$	Time
PINN (MLP)	1.8×10^{-1} $\pm1.9\times10^{-2}$	96.4	1.5×10^{-1} $\pm1.7\times10^{-2}$	181.7	2.2×10^{-1} $\pm2.6\times10^{-2}$	242.9	1.6×10^{-1} $\pm2.1\times10^{-2}$	263.5
MscaleDNN	1.1×10^{-1} $\pm1.07\times10^{-2}$	112.6	9.8×10^{-2} $\pm9.4\times10^{-3}$	213.8	1.7×10^{-1} $\pm1.58\times10^{-2}$	282.4	1.2×10^{-1} $\pm1.16\times10^{-2}$	313.1
XPINN	8.0×10^{-2} $\pm8.6\times10^{-3}$	151.3	5.5×10^{-2} $\pm6.3\times10^{-3}$	264.2	8.0×10^{-2} $\pm8.4\times10^{-3}$	343.7	7.2×10^{-2} $\pm8.7\times10^{-3}$	394.6
FBPINN	6.0×10^{-2} $\pm5.4\times10^{-3}$	117.8	3.5×10^{-2} $\pm4.3\times10^{-3}$	244.5	5.6×10^{-2} $\pm5.2\times10^{-3}$	313.2	6.5×10^{-2} $\pm6.4\times10^{-3}$	365.4
APINN	4.5×10^{-2} $\pm5.1\times10^{-3}$	158.9	2.2×10^{-2} $\pm2.7\times10^{-3}$	281.6	3.4×10^{-2} $\pm4.2\times10^{-3}$	366.2	4.8×10^{-2} $\pm5.4\times10^{-3}$	427.3
ELM-FBPINN	5.0×10^{-2} $\pm3.2\times10^{-3}$	18.6	2.6×10^{-2} $\pm2.1\times10^{-3}$	43.9	4.2×10^{-2} $\pm3.3\times10^{-3}$	74.8	5.8×10^{-2} $\pm4.3\times10^{-3}$	103.5
Global RFM	1.2×10^{-1} $\pm4.4\times10^{-3}$	12.4	9.0×10^{-2} $\pm3.2\times10^{-3}$	28.7	1.6×10^{-1} $\pm5.3\times10^{-3}$	41.2	1.1×10^{-1} $\pm4.2\times10^{-3}$	61.5
Fixed-PoU RFM	7.0×10^{-2} $\pm4.1\times10^{-3}$	20.6	4.0×10^{-2} $\pm3.4\times10^{-3}$	46.2	6.5×10^{-2} $\pm4.2\times10^{-3}$	76.8	7.0×10^{-2} $\pm5.3\times10^{-3}$	92.7
LDD-RFM	$\mathbf{3.0\times10^{-2}}$ $\mathbf{\pm1.7\times10^{-3}}$	24.3	$\mathbf{1.2\times10^{-2}}$ $\mathbf{\pm8.6\times10^{-4}}$	55.7	$\mathbf{1.8\times10^{-2}}$ $\mathbf{\pm1.3\times10^{-3}}$	86.4	$\mathbf{3.5\times10^{-2}}$ $\mathbf{\pm2.2\times10^{-3}}$	120.8

but excludes hyperparameter search. The random-feature methods use the same total number of features unless stated otherwise.

Metrics. We report relative L^2 error ($\mathcal{E}_{L^2}$) and a discrete H^1 seminorm ($\mathcal{E}_{H^1}$) on fixed fine grids ($N = 4097$ in 1D, 129^2 in 2D). Unless stated otherwise, LDD-RFM uses $M = 4$ experts. The remaining hyperparameters and ablations are in Appendix D.

6.2 Main results

Table 1 shows lower derivative-sensitive error for LDD-RFM in comparisons at equal feature counts. The oracle rows show why smooth gates matter: ORACLE-H attains low L^2 error but severe H^1 error, while ORACLE-S is close to the best smooth partition. On Task (i), for example, LDD-RFM reduces Global RFM's H^1 error from 5.58×10^{-1} to 9.00×10^{-2}, close to Oracle-S and below SIREN, MscaleDNN, and fixed-grid PoU. Appendix D reports time to accuracy, variation across random seeds, sensitivity to grid resolution, solver tolerance and noise, and timings for the inverse problem.

6.3 PDE results

For the linear PDE tasks, training uses residual least-squares and evaluation is always on fixed grids. In the singularly perturbed reaction–diffusion task, Global RFM oscillates near the $\mathcal{O}(\sqrt{\epsilon})$ layer (Fig. 1), while LDD-RFM concentrates support and spectral scale locally. On the 2D Poisson interface task, LDD-RFM reaches $\mathcal{E}_{H^1} = 1.2\times10^{-2}$, improving over PINN ($1.5\times10^{-1}$), MscaleDNN ($9.8\times10^{-2}$), and APINN ($2.2\times10^{-2}$). The Allen–Cahn row uses the space-time protocol in Appendix C.5 to test moving-front localization. Its nonlinear coefficient residual makes this an empirical extension beyond the exact ridge formulation.

Stronger domain-decomposition comparison. ELM-FBPINN uses fixed windows and runs faster than LDD-RFM in Table 2. LDD-RFM has the smallest $\mathcal{E}_{H^1}$ on all four PDEs. On the 2D interface task it improves over APINN by $1.8\times$ and over ELM-FBPINN by $2.2\times$. For Allen–Cahn, space-time gates reduce $\mathcal{E}_{H^1}$ from 5.1×10^{-2} (spatial-only control) to 3.5×10^{-2}.

Across Table 2, LDD-RFM takes 1.94–2.10 times the Global RFM runtime and 1.16–1.31 times the ELM-FBPINN runtime. APINN takes 3.54–6.54 times the LDD-RFM runtime. Warm-starting the gates reduces LDD-RFM inverse-loop time from 52 ± 7 to 34 ± 5 s (Table 14).

In separate matched runs, warm-start LDD-RFM took 35 ± 5 s and adjoint FEM 620 ± 95 s, with mean recovered-state $\mathcal{E}_{H^1}$ of 0.018 and 0.012, respectively (Table 17).

Controls and robustness. Replacing the inner iterative ridge solve by exact Cholesky, and giving Global RFM twice as many features on Task (iv), reduces its H^1 error only from 6.86×10^{-2} to 6.10×10^{-2}, while LDD-RFM with the original number of features attains 5.52×10^{-3} (Appendix D.5). Varying the CG tolerance from 10^{-3} to 10^{-12} shows that the default 10^{-6} tolerance gives nearly the same error as the tightest CG solve. In a separate test, the H^1 error stabilizes as the grid is refined from 129 to 4097 points. Appendix D reports exact Cholesky controls, scaling/radius sweeps, PoU ablations, and parameterized-PDE/operator comparisons. Table 2 compares full models. Table 10 isolates PoU regularization, and Table 16 tests gate learning, local spectra, and variable projection on Task (vi). The PoU ablation directly tests the interface term in Theorem 5.3. On the 2D windowed-interface task, removing PoU regularization increases the empirical peak-gradient proxy $\hat{\kappa}_\infty$ from 32.4 to 142.5 and more than doubles the H^1 error, from 5.89×10^{-2} to 1.25×10^{-1} (Appendix D.6). Thus learning the partition alone is insufficient: the partition must remain smooth enough for derivative-sensitive accuracy. Noise and radius sweeps show the same pattern: moderate label noise degrades smoothly, while overly small radii create sharp gates and larger H^1 error.

In the 3D comparison, both the expert count M and the mean active overlap $\bar{q}$ are higher for the denser partition. It gives a slightly smaller error but takes approximately $3.7\times$ as much time per iteration as the sparse partition (Appendix D.2). The increase in runtime is consistent with the dependence of CG cost on local overlap. On the 3D Poisson interface problem, the mean error for LDD-RFM was $\mathcal{E}_{H^1} = 0.034$, compared with 0.064 for APINN and 0.074 for ELM-FBPINN (Table 15).

6.4 Mechanism analysis

The learned gates reflect the two mechanisms in the theory. On Task (ii), high-scale experts concentrate on the high-frequency regime while low-frequency experts dominate smooth regions (Appendix D.7), avoiding the spectral leakage of one stationary global scale. On Task (iv), LDD-RFM reconstructs the off-center four-region geometry without boundary labels (Fig. 3). The ORACLE-H row is consistent with Theorem 5.3: discontinuous gates fit values but induce severe derivative error, whereas the PoU penalty in Eq. 7 controls gate sharpness.

7 Conclusion

LDD-RFM pairs learnable compact-support PoU geometry with ridge-solved local RFMs, so the nonlinear search places capacity and bandwidth while expert weights remain a transparent least-squares block. Across controlled regression and PDE tasks, this reduces derivative error near interfaces, layers, and moving fronts without relying on solver-tolerance or grid artifacts. The main lesson is that the hard cases are local changes of regularity: spending resolution everywhere creates oscillations, while spending it nowhere misses the derivative. Our experiments cover synthetic and manufactured benchmarks, including a three-dimensional Poisson interface problem, and leave open whether the observed gains in derivative accuracy persist for high-frequency Helmholtz problems in heterogeneous media.

The broader use case is PDE surrogate construction for design sweeps, inverse loops, and scientific studies with many local evaluations. The reduction in mean inverse-loop time with gate reuse (Table 14) motivates using the partition learned in one solve to initialize the next when successive problems have similar interface geometries. LDD-RFM is not a certified discretization. Remaining bottlenecks are high-dimensional gates and richer interface parameterizations.

Acknowledgments and Disclosure of Funding

This research received no funding. The authors declare no competing interests.

References

Samuel Anderson, Victorita Dolean, Ben Moseley, and Jennifer Pestana. ELM-FBPINNs: An efficient multilevel random feature method. *arXiv preprint arXiv:2409.01949*, 2024.

Hedy Attouch, Jérôme Bolte, and Benar Fux Svaiter. Convergence of descent methods for semi-algebraic and tame problems: proximal algorithms, forward–backward splitting, and regularized Gauss–Seidel methods. *Mathematical Programming*, 137(1):91–129, 2013.

Ivo Babuška and Jan M Melenk. The partition of unity method. *International Journal for Numerical Methods in Engineering*, 40(4):727–758, 1997.

Andrea Bonfanti, Ismael Medina, Roman List, Björn Staeves, Roberto Santana, and Marco Ellero. PINN Balls: Scaling second-order methods for PINNs with domain decomposition and adaptive sampling. *arXiv preprint arXiv:2510.21262*, 2025.

Steven L Brunton and J Nathan Kutz. Promising directions of machine learning for partial differential equations. *Nature Computational Science*, 4(7):483–494, 2024.

Luigi Carratino, Alessandro Rudi, and Lorenzo Rosasco. Learning with SGD and random features. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

Jingrun Chen, Xurong Chi, and Zhouwang Yang. Bridging traditional and machine learning-based algorithms for solving pdes: the random feature method. *Journal of Machine Learning*, 1(3): 268–298, 2022.

Yangtao Deng and Qiaolin He. Runge–kutta random feature method for solving multiphase flow problems of cells. *Physics of Fluids*, 37(2):023350, 2025.

Suchuan Dong and Zongwei Li. Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 387:114129, 2021.

Gene Golub and Victor Pereyra. Separable nonlinear least squares: the variable projection method and its applications. *Inverse problems*, 19(2):R1, 2003.

Robert B. Gramacy and Herbert K. H. Lee. Bayesian treed gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, 103(483): 1119–1130, 2008.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 12556–12569. PMLR, 2023.

Junpeng Hu, Shi Jin, Nana Liu, and Lei Zhang. Quantum random feature method for solving partial differential equations. *arXiv preprint arXiv:2510.07945*, 2025.

Zheyuan Hu, Ameya D Jagtap, George Em Karniadakis, and Kenji Kawaguchi. Augmented physics-informed neural networks (apinns): A gating network-based soft domain decomposition methodology. *Engineering Applications of Artificial Intelligence*, 126:107183, 2023.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

Ameya D. Jagtap, Kenji Kawaguchi, and George Em Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Mikhail Kuvakin, Zijian Mei, and Jingrun Chen. Weak random feature method for solving partial differential equations. *arXiv preprint arXiv:2505.00508*, 2025.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16252–16262, 2022.

Ziqi Liu, Wei Cai, and Zhi-Qin John Xu. Multi-scale deep neural network (MscaleDNN) for solving Poisson–Boltzmann equation in complex domains. *Communications in Computational Physics*, 28 (5):1970–2001, 2020.

Pingbing Ming and Hao Yu. Spectral convergence of random feature method in one dimension. *arXiv preprint arXiv:2507.07371*, 2025.

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 (ToG)*, 41(4): 1–15, 2022.

Nicholas H Nelsen and Andrew M Stuart. Operator learning using random features: A tool for scientific computing. *SIAM Review*, 66(3):535–571, 2024.

Chiwoo Park, Jianhua Z Huang, and Yu Ding. Domain decomposition approach for fast Gaussian process regression of large spatial data sets. *Journal of Machine Learning Research*, 12(47): 1697–1728, 2011.

Alfio Quarteroni, Paola Gervasio, and Francesco Regazzoni. Combining physics-based and data-driven models: advancing the frontiers of research with scientific machine learning. *Mathematical Models and Methods in Applied Sciences*, 35(04):905–1071, 2025.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 20, 2007.

Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. *Advances in neural information processing systems*, 21, 2008.

Nicola Segata and Enrico Blanzieri. Fast and scalable local kernel machines. *Journal of Machine Learning Research*, 11(64):1883–1926, 2010.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, Jeff Dean, et al. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1701.06538.

Vincent Sitzmann, Julien N. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 7462–7473, 2020.

Makoto Takamoto, Timothy Praditia, Raphael Leiteritz, Dan MacKinlay, Francesco Alesiani, Dirk Pflüger, and Mathias Niepert. PDEBench: An extensive benchmark for scientific machine learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

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. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 7537–7547, 2020.

Andrea Toselli and Olof B. Widlund. *Domain Decomposition Methods—Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer, 2005.

Bo Wang, Wenzhong Zhang, and Wei Cai. Multi-scale deep neural network (MscaleDNN) methods for oscillatory Stokes flows in complex domains. *Communications in Computational Physics*, 28 (5):2139–2157, 2020.

Holger Wendland. Fast evaluation of radial basis functions: Methods based on partition of unity. In *Approximation Theory X: Wavelets, Splines, and Applications*, pages 473–483. Vanderbilt University Press, 2002.

LDD-RFM: Learnable Domain Decomposition for Random Feature Models via Variable Projection

Supplementary material

A Mathematical preliminaries and auxiliary lemmas

This appendix collects functional-analytic and matrix-calculus tools used in the proofs of the H^1 error bound and the idealized convergence template.

A.1 Notation and geometric constants

Let $\Omega \subset \mathbb{R}^d$ be a bounded Lipschitz domain. For a measurable $A \subseteq \Omega$, we use

$$\|v\|^2_{H^1(A)} := \|v\|^2_{L^2(A)} + \|\nabla v\|^2_{L^2(A)}.$$

The gating functions $\{\alpha_m\}_{m=0}^M$ form a PoU: $\alpha_m \geq 0$ and $\sum_{m=0}^M \alpha_m \equiv 1$ on Ω with $m = 0$ corresponding to the background gate in our construction.

The H^1 bound in Theorem 5.3 involves the worst-case PoU sharpness constant as defined in (12):

$$\kappa_\infty^2 := \left\| \sum_{m=0}^M \|\nabla \alpha_m(\cdot)\|_2^2 \right\|_{L^\infty(\Omega)} < \infty. \tag{17}$$

For optimization, we also consider the L^p case (with $p > d$)

$$\kappa_p^2 := \sum_{m=0}^M \|\nabla \alpha_m\|^2_{L^p(\Omega)}, \tag{18}$$

where $p = 2$ corresponds to the standard gradient-energy penalty and larger p places more emphasis on peak gate gradients.

A.2 Tools for approximation analysis

We also record a standard H^1 product-rule estimate, which is useful for alternative bounds and for relating PoU smoothness penalties to interface terms.

Lemma A.1. *Let $\beta \in W^{1,\infty}(\Omega)$ and $v \in H^1(\Omega)$. Then $\beta v \in H^1(\Omega)$ and*

$$\|\nabla(\beta v)\|_{L^2(\Omega)} \leq \|\beta\|_{L^\infty(\Omega)} \|\nabla v\|_{L^2(\Omega)} + \|v\|_{L^2(\Omega)} \|\nabla \beta\|_{L^\infty(\Omega)}. \tag{19}$$

Consequently, $\|\beta v\|_{H^1(\Omega)} \leq \|\beta\|_{L^\infty(\Omega)} \|v\|_{H^1(\Omega)} + \|v\|_{L^2(\Omega)} \|\nabla \beta\|_{L^\infty(\Omega)}$.

Proof. The weak product rule gives $\nabla(\beta v) = \beta \nabla v + v \nabla \beta$. Apply the triangle inequality in L^2 and the bounds $\|\beta \nabla v\|_{L^2} \leq \|\beta\|_{L^\infty} \|\nabla v\|_{L^2}$ and $\|v \nabla \beta\|_{L^2} \leq \|v\|_{L^2} \|\nabla \beta\|_{L^\infty}$. □

A.3 Tools for optimization analysis

Lemma A.2 (Differentiability of the regularized inverse)**.** *Let $\mathbf{A}(\Theta) \in \mathbb{R}^{K \times K}$ be symmetric positive definite for all Θ and continuously differentiable in Θ. If $\mathbf{A}(\Theta) \succeq \lambda \mathbf{I}$ for some $\lambda > 0$ uniformly on a set $\mathcal{S}$, then $\mathbf{A}(\Theta)^{-1}$ is continuously differentiable on $\mathcal{S}$ and*

$$\frac{\partial}{\partial \Theta} \mathbf{A}(\Theta)^{-1} = -\mathbf{A}(\Theta)^{-1} \Big(\frac{\partial}{\partial \Theta} \mathbf{A}(\Theta) \Big) \mathbf{A}(\Theta)^{-1}. \tag{20}$$

Proof. Differentiate the identity $\mathbf{A}(\Theta)\mathbf{A}(\Theta)^{-1} = \mathbf{I}$ and use uniform positive definiteness to justify smoothness. □

B Proofs of main theoretical results

In this appendix, we provide detailed proofs for the theoretical results presented in Section 5. We first establish the H^1 approximation bound (Theorem 5.3), then detail the convergence analysis under the Kurdyka-Łojasiewicz framework (Theorem 5.5).

Lemma B.1 (Discrete L^p control bounds a pointwise maximum)**.** *Let $\{a_i\}_{i=1}^N$ be nonnegative numbers and $1 \le p < \infty$. Then*

$$\max_{1\le i\le N} a_i \le \Big(\sum_{i=1}^N a_i^p\Big)^{1/p} = N^{1/p}\Big(\frac{1}{N}\sum_{i=1}^N a_i^p\Big)^{1/p}. \tag{21}$$

In particular, for any point set $\mathcal{X} = \{x_i\}_{i=1}^N \subset \Omega$ and gates $\{\alpha_m\}_{m=0}^M$, this applies to each sequence $a_i = \|\nabla\alpha_m(x_i)\|_2$. Consequently, the point-set approximation $\hat\kappa_\infty$ defined by

$$\hat\kappa_\infty^2 := \max_{x_i\in\mathcal{X}} \sum_{m=0}^M \|\nabla_x\alpha_m(x_i)\|_2^2 \quad \text{satisfies} \quad \hat\kappa_\infty^2 \le N^{2/p}\sum_{m=0}^M \Big(\frac{1}{N}\sum_{i=1}^N \|\nabla\alpha_m(x_i)\|_2^p\Big)^{2/p}. \tag{22}$$

Lemma B.2 (Explicit control of κ_∞ for normalized bump PoU)**.** *Let $w_0(x) \equiv 1$ and $w_m : \Omega \to [0,1]$ be C^1 functions for $m = 1,\dots,M$. Define*

$$W(x) := w_0(x) + \sum_{j=1}^M w_j(x), \quad \alpha_m(x) := \frac{w_m(x)}{W(x)}, \quad m = 0,1,\dots,M,$$

so that $\alpha_m \ge 0$ and $\sum_{m=0}^M \alpha_m \equiv 1$ on Ω. Assume $\|\nabla w_m\|_{L^\infty(\Omega)} \le L_w$ for all $m \ge 1$ and define $\kappa_\infty^2 := \Big\|\sum_{m=0}^M \|\nabla\alpha_m(\cdot)\|_2^2\Big\|_{L^\infty(\Omega)}$. If no overlap information is used, then

$$\|\nabla W\|_{L^\infty(\Omega)} \le \sum_{j=1}^M \|\nabla w_j\|_{L^\infty(\Omega)} \le M L_w.$$

Moreover, pointwise for $m \ge 1$,

$$\nabla\alpha_m = \frac{\nabla w_m}{W} - \frac{w_m \nabla W}{W^2}, \quad \|\nabla\alpha_m(x)\|_2 \le \|\nabla w_m(x)\|_2 + \|\nabla W(x)\|_2, \tag{23}$$

where we used $0 \le w_m \le 1$ and $W \ge 1$. Consequently,

$$\kappa_\infty \le \sqrt{(ML_w)^2 + M\big((M+1)L_w\big)^2} \le (M+1)\sqrt{M+1}\,L_w. \tag{24}$$

If, moreover, the bumps have bounded gradient overlap *in the sense that for every $x \in \Omega$,*

$$\#\{j \in \{1,\dots,M\} : \nabla w_j(x) \ne 0\} \le q_\nabla,$$

and active overlap

$$\#\{j \in \{1,\dots,M\} : w_j(x) > 0\} \le q,$$

then $\|\nabla W(x)\|_2 \le q_\nabla L_w$ pointwise and, since compact C^1 bumps have $\nabla w_m(x) = 0$ whenever $w_m(x) = 0$,

$$\|\nabla\alpha_0(x)\|_2 \le q_\nabla L_w, \quad \|\nabla\alpha_m(x)\|_2 \le (q_\nabla + 1)L_w \ \text{ for at most } q \text{ active local gates}, \tag{25}$$

and therefore

$$\kappa_\infty \le \sqrt{(q_\nabla L_w)^2 + q\big((q_\nabla+1)L_w\big)^2}. \tag{26}$$

This is the bound relevant to our compact-support implementation: it scales with actual local overlap rather than total expert count.

B.1 Proof of Theorem 5.3

Our goal is to prove the global H^1 *seminorm* bound (15) for $u = \sum_{m=0}^{M} \alpha_m f_m$ and target u^*.

Define local residuals $e_m := f_m - u^*$ on $\mathrm{supp}(\alpha_m)$ and the global error $e := u - u^*$. Using $\sum_{m=0}^{M} \alpha_m \equiv 1$,

$$e = \sum_{m=0}^{M} \alpha_m f_m - \sum_{m=0}^{M} \alpha_m u^* = \sum_{m=0}^{M} \alpha_m e_m.$$

By the weak product rule,

$$\nabla e = \sum_{m=0}^{M} \alpha_m \nabla e_m \; + \; \sum_{m=0}^{M} (\nabla \alpha_m)\, e_m. \tag{27}$$

Hence, by the triangle inequality,

$$\|\nabla e\|_{L^2(\Omega)} \le \underbrace{\left\| \sum_{m=0}^{M} \alpha_m \nabla e_m \right\|_{L^2}}_{(A)} + \underbrace{\left\| \sum_{m=0}^{M} (\nabla \alpha_m) e_m \right\|_{L^2}}_{(B)}. \tag{28}$$

Using $\alpha_m \ge 0$ and $\sum_{m=0}^{M} \alpha_m \equiv 1$, convexity gives pointwise

$$\Big| \sum_{m=0}^{M} \alpha_m(x) \nabla e_m(x) \Big|^2 \le \sum_{m=0}^{M} \alpha_m(x) |\nabla e_m(x)|^2.$$

Integrating yields

$$(A)^2 = \int_\Omega \Big| \sum_{m=0}^{M} \alpha_m \nabla e_m \Big|^2 \le \int_\Omega \sum_{m=0}^{M} \alpha_m |\nabla e_m|^2 \le \sum_{m=0}^{M} \|\nabla e_m\|^2_{L^2(\mathrm{supp}(\alpha_m))} \le \sum_{m=0}^{M} \epsilon^2_{m,1}. \tag{29}$$

For each $x \in \Omega$, Cauchy–Schwarz gives

$$\Big| \sum_{m=0}^{M} (\nabla \alpha_m(x)) e_m(x) \Big|^2 \le \Big(\sum_{m=0}^{M} |\nabla \alpha_m(x)|^2 \Big) \Big(\sum_{m=0}^{M} |e_m(x)|^2 \Big). \tag{30}$$

By definition of κ_∞,

$$\sum_{m=0}^{M} |\nabla \alpha_m(x)|^2 \le \kappa_\infty^2,$$

so pointwise

$$\Big| \sum_{m=0}^{M} (\nabla \alpha_m(x)) e_m(x) \Big|^2 \le \kappa_\infty^2 \sum_{m=0}^{M} |e_m(x)|^2.$$

Integrating and using $\|e_m\|_{L^2(\Omega)} = \|e_m\|_{L^2(\mathrm{supp}(\alpha_m))}$ yields

$$(B) \le \kappa_\infty \left(\sum_{m=0}^{M} \|e_m\|^2_{L^2(\mathrm{supp}(\alpha_m))} \right)^{1/2} \le \kappa_\infty \left(\sum_{m=0}^{M} \epsilon^2_{m,0} \right)^{1/2}. \tag{31}$$

Combining (28), (29), and (31) gives

$$|u - u^*|_{H^1(\Omega)} = \|\nabla e\|_{L^2(\Omega)} \le \left(\sum_{m=0}^{M} \epsilon^2_{m,1} \right)^{1/2} + \kappa_\infty \left(\sum_{m=0}^{M} \epsilon^2_{m,0} \right)^{1/2}.$$

This proves the H^1 seminorm bound (15). □

B.2 Proof of Theorem 5.5

Proposition B.3 (Regularity of the reduced objective)**.** *Assume $\lambda > 0$ and that the assembled design map $\Psi(\Theta)$ and PoU regularizer $\mathcal{R}_{\mathrm{PoU}}(\Theta)$ are continuously differentiable on an open neighborhood of the bounded parameter set. For linear PDE residuals, this regularity is required after applying the differential operator. In particular, for the compact-support bump PoU (Sec. 4.1), assume patch radii are bounded below ($\rho_m \succeq \rho_{\min}\mathbf{1}$) and the PoU normalization is well-defined on Ω (e.g., ensured by a background weight so the denominator is uniformly bounded away from zero). Then the map $\Theta \mapsto \mathcal{C}^\star(\Theta)$ is continuously differentiable, and the reduced objective $\tilde{\mathcal{J}}(\Theta)$ is continuously differentiable on bounded parameter sets.*

Proof. Recall the reduced objective $\tilde{\mathcal{J}}(\Theta) := \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta)$, where $\mathcal{C}^\star(\Theta) = \arg\min_{\mathcal{C}} \mathcal{J}(\mathcal{C}, \Theta)$. This proof concerns the *exact* ridge minimizer $\mathcal{C}^\star(\Theta)$ (cf. Remark 5.6 for the inexact-solve setting).

For fixed Θ, the optimization over $\mathcal{C}$ is ridge regression with design matrix $\Psi(\Theta) \in \mathbb{R}^{n \times K_{\mathrm{tot}}}$ which, for regression, has i-th row

$$[\alpha_0(\boldsymbol{x}_i;\theta)\Phi_0(\boldsymbol{x}_i)^\top, \ldots, \alpha_M(\boldsymbol{x}_i;\theta)\Phi_M(\boldsymbol{x}_i;R_M)^\top],$$

where $\Phi_0(\boldsymbol{x}) \equiv 1$, $K_{\mathrm{tot}} = 1 + \sum_{m=1}^{M} K_m$, and $\mathbf{y} := (y_1, \ldots, y_n)^\top \in \mathbb{R}^n$. For linear PDE objectives, the same argument uses the assembled residual and boundary design matrix. The optimal coefficients satisfy the normal equations:

$$\mathcal{C}^\star(\Theta) = \left(\Psi(\Theta)^\top \Psi(\Theta) + n\lambda I\right)^{-1} \Psi(\Theta)^\top \mathbf{y}. \tag{32}$$

Let $A(\Theta) := \Psi(\Theta)^\top \Psi(\Theta) + n\lambda I$.

Since $\lambda > 0$, we have $A(\Theta) \succeq n\lambda I$ for all Θ, hence $A(\Theta)$ is invertible and uniformly SPD on any set of parameters.

We make four assumptions. (i) The PoU gates $\alpha_m(\cdot;\theta)$ induced by the compact-support bump construction (Sec. 4.1) are continuously differentiable in θ on the considered bounded set. In particular, we use a C^1 bump with vanishing derivative at the cutoff so that the resulting $w_m(\cdot;\theta)$ and normalized gates $\alpha_m(\cdot;\theta)$ are differentiable (at least a.e. in $\boldsymbol{x}$) and have bounded θ-derivatives on bounded parameter sets. (ii) Patch radii are bounded below, $\rho_m \succeq \rho_{\min}\mathbf{1}$, which prevents degeneracy and ensures that derivatives involving $\mathrm{diag}(\rho_m)^{-1}$ remain bounded on bounded parameter sets (with bounds depending only on $\rho_{\min}$, the bump smoothness, and the domain diameter). (iii) The PoU normalization is well-defined on Ω with a denominator uniformly bounded away from zero (e.g., ensured by a background weight $w_0 \equiv 1$, so $\sum_{k=0}^{M} w_k(\boldsymbol{x};\theta) \geq 1$ for all $\boldsymbol{x} \in \Omega$). (iv) The feature maps are continuously differentiable in $\{R_m\}$, and the random draws used in the random features are held fixed during training (equivalently, we use the reparameterization $\boldsymbol{k}_{m,j} = R_m \boldsymbol{\xi}_{m,j}$ with fixed $\boldsymbol{\xi}_{m,j}$), so $\Phi_m(\cdot; R_m)$ is a smooth function of R_m.

By the assumed regularity of the assembled design map, $\Theta \mapsto \Psi(\Theta)$ is continuously differentiable on the parameter neighborhood. Therefore, $\Theta \mapsto \Psi(\Theta)^\top \Psi(\Theta)$ is continuously differentiable as a composition of C^1 maps, and hence so is $\Theta \mapsto A(\Theta)$.

By Lemma A.2, the inverse map $A(\Theta)^{-1}$ is continuously differentiable on any set where $A(\Theta)$ remains uniformly positive definite (here ensured by $\lambda > 0$). Therefore, the composition in (32) implies $\Theta \mapsto \mathcal{C}^\star(\Theta)$ is continuously differentiable on bounded parameter sets.

For each fixed Θ, $\mathcal{J}(\cdot, \Theta)$ is strongly convex and continuously differentiable in $\mathcal{C}$, hence it attains a unique minimizer $\mathcal{C}^\star(\Theta)$ satisfying the first-order optimality condition $\nabla_{\mathcal{C}} \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta) = 0$. By the envelope theorem for smooth parametric minimization (equivalently, the classical variable-projection identity),

$$\nabla_\Theta \tilde{\mathcal{J}}(\Theta) = \nabla_\Theta \mathcal{J}(\mathcal{C}, \Theta)\big|_{\mathcal{C}=\mathcal{C}^\star(\Theta)}. \tag{33}$$

Since $\mathcal{J}$ is C^1 in $(\mathcal{C}, \Theta)$ by the regularity assumptions on Ψ and $\mathcal{R}_{\mathrm{PoU}}$ and $\Theta \mapsto \mathcal{C}^\star(\Theta)$ is C^1, it follows that $\tilde{\mathcal{J}}$ is continuously differentiable on bounded parameter sets. □

Proposition B.4 (Perturbation from an inexact ridge solve)**.** *Fix Θ and let $A(\Theta) = \Psi(\Theta)^\top \Psi(\Theta) + n\lambda I$, $b(\Theta) = \Psi(\Theta)^\top y$. Let $c^\star = A^{-1}b$ and suppose an iterative solver returns $\widehat{c}$ with residual*

$r = A\widehat{c} - b$. *Then*

$$\|\widehat{c} - c^\star\|_2 \le \frac{\|r\|_2}{n\lambda}.$$

If, on a bounded parameter set, $\nabla_\Theta \mathcal{J}(c, \Theta)$ is Lipschitz in c with constant $L_{\Theta c}$, then

$$\left\| \nabla_\Theta \mathcal{J}(\widehat{c}, \Theta) - \nabla_\Theta \tilde{\mathcal{J}}(\Theta) \right\|_2 \le \frac{L_{\Theta c}}{n\lambda} \|r\|_2.$$

For fixed $\lambda > 0$ and uniformly bounded $L_{\Theta c}$, summable CG residuals $\sum_k \|r_k\|_2 < \infty$ imply summable reduced-gradient perturbations.

Proof. Since $A \succeq n\lambda I$, we have $\|A^{-1}\|_2 \le (n\lambda)^{-1}$. Moreover,

$$\widehat{c} - c^\star = A^{-1}(A\widehat{c} - b) = A^{-1} r,$$

which gives the coefficient perturbation bound. For the gradient perturbation, use the exact variable-projection identity $\nabla_\Theta \tilde{\mathcal{J}}(\Theta) = \nabla_\Theta \mathcal{J}(c^\star, \Theta)$ and Lipschitz continuity in c:

$$\|\nabla_\Theta \mathcal{J}(\widehat{c}, \Theta) - \nabla_\Theta \mathcal{J}(c^\star, \Theta)\|_2 \le L_{\Theta c} \|\widehat{c} - c^\star\|_2.$$

The summability conclusion follows by summing the preceding perturbation bound over iterations. □

Remark B.5 (Interpreting $L_{\Theta c}$). The constant $L_{\Theta c}$ is a local smoothness constant of the reduced model, not an additional solver assumption. To see why it is finite under the bounded-set hypotheses used above, write the objective in terms of the gated design matrix $\Psi(\Theta)$:

$$\mathcal{J}(c, \Theta) = \frac{1}{n} \|\Psi(\Theta)c - y\|_2^2 + \lambda \|c\|_2^2 + \gamma \mathcal{R}_{\mathrm{PoU}}(\Theta).$$

After vectorizing Θ, the ℓ-th component of the mixed gradient is

$$\partial_{\Theta_\ell} \mathcal{J}(c, \Theta) = \frac{2}{n} \big((\partial_{\Theta_\ell} \Psi)c\big)^\top (\Psi c - y) + \gamma\, \partial_{\Theta_\ell} \mathcal{R}_{\mathrm{PoU}}(\Theta).$$

The ridge and PoU terms do not add any c-dependence to this expression. Hence, on a set where $\|c\|_2 \le C$, $\|\Psi\|_2 \le B$, $\|\partial_{\Theta_\ell} \Psi\|_2 \le G_\ell$, and $\|y\|_2 \le Y$,

$$|\partial_{\Theta_\ell} \mathcal{J}(c, \Theta) - \partial_{\Theta_\ell} \mathcal{J}(c', \Theta)| \le \frac{2G_\ell}{n} (2BC + Y) \|c - c'\|_2.$$

Combining the componentwise bounds gives a finite $L_{\Theta c}$ on bounded parameter regions. Practically, this constant grows when gates or local spectra make $\partial_\Theta \Psi$ large, which is why sharp PoU transitions and very small ridge weights make the inexact-solve tolerance more important.

Lemma B.6. *Assume $\mathcal{J}$ is proper and lower-semicontinuous, and is* definable *in an o-minimal structure (e.g., semialgebraic or globally subanalytic). Then $\mathcal{J}$ satisfies the Kurdyka–Łojasiewicz (KL) property (see, e.g., Attouch et al., 2013 and references therein).*

We analyze the idealized alternating scheme in Theorem 5.5 through the reduced objective

$$\tilde{\mathcal{J}}(\Theta) := \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta), \qquad \mathcal{C}^\star(\Theta) := \arg\min_{\mathcal{C}} \mathcal{J}(\mathcal{C}, \Theta).$$

Consider the iteration

1. $\mathcal{C}$**-step (exact):** $\mathcal{C}^{k+1} = \mathcal{C}^\star(\Theta^k)$,
2. Θ**-step (descent on $\tilde{\mathcal{J}}$):** Θ^{k+1} is obtained by a gradient-type step on $\tilde{\mathcal{J}}$ with a stepsize rule that guarantees sufficient decrease (e.g., backtracking line search).

Step 1: Descent and square-summability of steps. By definition of $\tilde{\mathcal{J}}$ and the exact $\mathcal{C}$-step,

$$\mathcal{J}(\mathcal{C}^{k+1}, \Theta^k) = \tilde{\mathcal{J}}(\Theta^k).$$

Under Proposition B.3, $\tilde{\mathcal{J}}$ is continuously differentiable on bounded sets. Moreover, the Θ-update is chosen to ensure sufficient decrease: there exists $\rho > 0$ such that

$$\tilde{\mathcal{J}}(\Theta^{k+1}) \le \tilde{\mathcal{J}}(\Theta^k) - \frac{\rho}{2} \|\Theta^{k+1} - \Theta^k\|^2. \tag{34}$$

Summing (34) over $k = 0, \dots, K-1$ yields

$$\frac{\rho}{2} \sum_{k=0}^{K-1} \|\Theta^{k+1} - \Theta^k\|^2 \leq \tilde{\mathcal{J}}(\Theta^0) - \tilde{\mathcal{J}}(\Theta^K).$$

Since $\tilde{\mathcal{J}}$ is bounded below (e.g., by 0 for a squared-loss objective with nonnegative regularizers), we obtain

$$\sum_{k=0}^{\infty} \|\Theta^{k+1} - \Theta^k\|^2 < \infty, \quad \text{hence} \quad \|\Theta^{k+1} - \Theta^k\| \to 0.$$

Step 2: Boundedness and existence of cluster points. Assume the iterates $\{(\mathcal{C}^k, \Theta^k)\}$ remain in a bounded set. Then $\{\Theta^k\}$ admits a convergent subsequence $\Theta^{k_j} \to \bar{\Theta} \in U$. Continuity of $\tilde{\mathcal{J}}$ gives $\tilde{\mathcal{J}}(\Theta^{k_j}) \to \tilde{\mathcal{J}}(\bar{\Theta})$.

Step 3: KL property and convergence of $\{\Theta^k\}$. Assume $\mathcal{J}$ satisfies the KL property. Lemma B.6 gives a sufficient condition. Optimality of $\mathcal{C}^\star(\Theta)$ and the envelope identity give $\nabla_{\mathcal{C}} \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta) = 0$ and $\nabla_\Theta \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta) = \nabla \tilde{\mathcal{J}}(\Theta)$. Hence

$$\|\nabla \mathcal{J}(\mathcal{C}^\star(\Theta), \Theta)\| = \|\nabla \tilde{\mathcal{J}}(\Theta)\|.$$

By continuity of $\mathcal{C}^\star$, restricting the KL inequality for $\mathcal{J}$ at $(\mathcal{C}^\star(\bar{\Theta}), \bar{\Theta})$ to this graph gives the KL inequality for $\tilde{\mathcal{J}}$ at $\bar{\Theta}$. The sufficient-decrease bound (34), the relative-error bound assumed in Theorem 5.5, and the subsequence continuity in Step 2 verify conditions H1–H3 of Attouch et al. [2013, Thm. 2.9]. Hence the sequence has *finite length*:

$$\sum_{k=0}^{\infty} \|\Theta^{k+1} - \Theta^k\| < \infty,$$

and consequently Θ^k converges to a critical point $\Theta^\star$ of $\tilde{\mathcal{J}}$.

Step 4: Convergence of the full iterates. Since $\mathcal{C}^{k+1} = \mathcal{C}^\star(\Theta^k)$ and $\mathcal{C}^\star(\cdot)$ is continuous under $\lambda > 0$ (indeed, $\mathcal{C}^\star(\Theta) = (\Psi(\Theta)^\top \Psi(\Theta) + n\lambda I)^{-1} \Psi(\Theta)^\top y$), we have $\mathcal{C}^k \to \mathcal{C}^\star(\Theta^\star)$. Since $\mathcal{C}^\star$ is C^1 on U, it is Lipschitz on a sufficiently small ball about $\Theta^\star$. Thus, for all sufficiently large k and some $L_C < \infty$,

$$\|\mathcal{C}^{k+1} - \mathcal{C}^k\| \leq L_C \|\Theta^k - \Theta^{k-1}\|.$$

The finite length established in Step 3 therefore yields finite length of $\{(\mathcal{C}^k, \Theta^k)\}$. Moreover, $\nabla_{\mathcal{C}} \mathcal{J}(\mathcal{C}^\star(\Theta^\star), \Theta^\star) = 0$ and $\nabla_\Theta \mathcal{J}(\mathcal{C}^\star(\Theta^\star), \Theta^\star) = \nabla \tilde{\mathcal{J}}(\Theta^\star) = 0$. Therefore $(\mathcal{C}^k, \Theta^k)$ converges to a critical point of $\mathcal{J}$. □

C Benchmark and baseline protocols

This appendix provides the exact analytical definitions of the synthetic targets (Tasks (i)–(iv)) and the baseline training/evaluation protocols used in Sec. 6.

C.1 Task (i): spectral jump (1D)

We consider $\Omega = [-1, 1]$ and define

$$u(x) = \begin{cases} \sin(2x), & x \leq 0, \\ \sin(20x), & x > 0. \end{cases}$$

The target is continuous at $x = 0$ but has a sharp derivative change there, isolating the bandwidth dilemma in its simplest one-dimensional form.

C.2 Task (ii): multi-regime process (1D)

We consider $\Omega = [-1, 1]$ with regime interfaces at $c_1 = -0.3$ and $c_2 = 0.5$. Define three regime-local components

$$f_{\text{smooth}}(x) = (1 + x)\big[\sin(2\pi x) + 0.3\cos(\pi x)\big],$$
$$f_{\text{mixed}}(x) = (1 - x^2)\big[0.8\sin(4\pi x) + 0.2\cos(3\pi x)\big],$$
$$f_{\text{high}}(x) = (1 - x)\big[\sin(3\pi x) + 0.5\sin(5\pi x)\big].$$

To enforce global C^0 continuity, we introduce offsets

$$\delta_1 = f_{\text{smooth}}(c_1) - f_{\text{mixed}}(c_1),$$
$$\delta_2 = \big(f_{\text{mixed}}(c_2) + \delta_1\big) - f_{\text{high}}(c_2),$$

and define the target

$$u(x) = \begin{cases} f_{\text{smooth}}(x), & x \le c_1, \\ f_{\text{mixed}}(x) + \delta_1, & c_1 < x \le c_2, \\ f_{\text{high}}(x) + \delta_2, & x > c_2. \end{cases}$$

By construction, u is continuous at c_1 and c_2.

C.3 Task (iii): windowed interface (2D)

Task (iii) embeds a 1D spectral transition into 2D via a compactly supported smooth window. Let $\Omega = [-1, 1]^2$ and define a raised-cosine (Hann-type) window $w : \mathbb{R} \to [0, 1]$ by

$$w(t) = \begin{cases} \frac{1}{2}\big(1 + \cos(\pi t)\big), & |t| \le 1, \\ 0, & |t| > 1. \end{cases} \tag{35}$$

This bump satisfies $w(\pm 1) = 0$ and $w'(\pm 1) = 0$ with

$$w'(t) = \begin{cases} -\frac{\pi}{2}\sin(\pi t), & |t| < 1, \\ 0, & |t| > 1, \end{cases}$$

hence $w \in C^1(\mathbb{R})$. Let $W(x, y) = w(x)w(y)$ be the two-dimensional envelope and let

$$s_\eta(x) = \frac{1}{2}\left(1 + \tanh\frac{x}{\eta}\right), \qquad \eta = 0.03,$$

be the interior transition profile. The exact target used in Task (iii) is

$$u(x, y) = W(x, y)\Big[(1 - s_\eta(x))\Big\{\sin(2\pi x) + 0.25\cos(2\pi y)\Big\} + s_\eta(x)\Big\{0.8\sin(12\pi x + 4\pi y) + 0.2\cos(10\pi x - 2\pi y)\Big\}\Big]. \tag{36}$$

Thus the field is smooth and boundary-compatible on $\partial\Omega$ because both W and its normal derivative vanish at the domain boundary, while the transition layer near $x = 0$ creates a localized high-frequency wave packet in the interior.

C.4 Task (iv): anisotropic asymmetric partition (2D)

We partition $\Omega = [-1, 1]^2$ into four axis-aligned regions using the interfaces $x = -0.2$ and $y = 0.3$, denoted by $\{R_k\}_{k=1}^4$. On each region, the target takes the form

$$u(\boldsymbol{x})\big|_{R_k} = P_k(\boldsymbol{x})\,\sin(\boldsymbol{k}_k^\top \boldsymbol{x}) + C_k, \qquad \boldsymbol{x} = (x, y)^\top.$$

The parameters are:

- **Region 1** ($x \le -0.2$, $y \le 0.3$): $\boldsymbol{k}_1 = (2\pi, 2\pi)^\top$, $P_1(\boldsymbol{x}) = 1$.
- **Region 2** ($x > -0.2$, $y \le 0.3$): $\boldsymbol{k}_2 = (10\pi, 0)^\top$, $P_2(\boldsymbol{x}) = 1 - x^2$.
- **Region 3** ($x \le -0.2$, $y > 0.3$): $\boldsymbol{k}_3 = (0, 8\pi)^\top$, $P_3(\boldsymbol{x}) = 1 - y^2$.
- **Region 4** ($x > -0.2$, $y > 0.3$): $\boldsymbol{k}_4 = (6\pi, -4\pi)^\top$, $P_4(\boldsymbol{x}) = \exp(-(x^2 + y^2))$.

The additive constants $\{C_k\}$ are fixed once (starting from $C_1 = 0$) to keep the value ranges comparable across regimes. Consequently, the challenge comes from heterogeneous directional frequencies and interface-local value/derivative mismatch, with all methods evaluated against the same fixed-grid value and derivative proxies.

C.5 PDE benchmarks

Task (v): singularly perturbed reaction–diffusion. We solve $-\epsilon u_{xx} + u = f$ on $[-1, 1]$ with homogeneous Dirichlet data and $\epsilon = 10^{-3}$. The forcing f is manufactured from a reference solution with an internal layer centered at $x = 0$ and width $\mathcal{O}(\sqrt{\epsilon})$, so that all methods are evaluated against the same closed-form u^* and derivative u_x^*. Interior collocation points are sampled uniformly with an additional validation-only layer window $[-\sqrt{\epsilon}, \sqrt{\epsilon}]$ for measuring the error near the layer.

Task (vi): single-interface Poisson. On $\Omega = [-1, 1]^2$, we solve

$$-\nabla \cdot (a(\boldsymbol{x})\nabla u) = f, \qquad u|_{\partial\Omega} = 0,$$

where $a(\boldsymbol{x}) = 10$ inside a smooth inclusion and $a(\boldsymbol{x}) = 1$ outside. The right-hand side is manufactured so that the reference solution is continuous while the flux satisfies the usual interface condition.

Task (vii): multi-interface Poisson. This task uses the same elliptic operator as Task (vi), but $a(\boldsymbol{x})$ contains four inclusions with alternating contrasts $a \in \{1, 10, 25\}$ and intersecting transition neighborhoods. The evaluation grid is unchanged (129^2), but the residual collocation set is doubled relative to Task (vi) to sample all interfaces. This benchmark tests whether a learned soft partition can allocate multiple local experts without being handed the inclusion layout.

Task (viii): space–time Allen–Cahn. We solve the Allen–Cahn equation

$$u_t - \nu\Delta u + \epsilon^{-2}(u^3 - u) = 0, \qquad (\boldsymbol{x}, t) \in [-1, 1]^2 \times [0, T],$$

with $\nu = 10^{-3}$, $\epsilon = 0.08$, $T = 0.5$, periodic spatial boundary conditions, and a smooth multi-droplet initial condition. All learning-based baselines use the same three-dimensional space-time collocation protocol and initial and boundary sampling. We average the spatial H^1 seminorm over uniformly sampled time slices. Because the Allen–Cahn residual is nonlinear in the coefficients, this benchmark is used as an empirical space-time localization test rather than as an instance of the exact ridge subproblem analyzed in Sec. 4.3. For LDD-RFM and all PoU-based baselines on Task (viii), gates are space-time ($d = 3$), so the partition can adapt across time slices. A purely spatial ($d = 2$) gate control gives $\mathcal{E}_{H^1} = 5.1\times10^{-2}$, higher than the space-time LDD-RFM value in Table 2, because interface drift is not represented by static gates.

C.6 Baseline implementation details

Oracle-Smooth PoU (Fixed-PoU, Oracle-S). For the "Oracle-Smooth" baseline, we construct a ground-truth PoU that aligns with the known interfaces. Let $\tilde{\alpha}_m(\boldsymbol{x}) = \psi(d_m(\boldsymbol{x})/\delta)$, where $d_m(\boldsymbol{x}) \geq 0$ is the (unsigned) distance from $\boldsymbol{x}$ to region R_m, $\delta = 0.1$ controls the overlap width, and

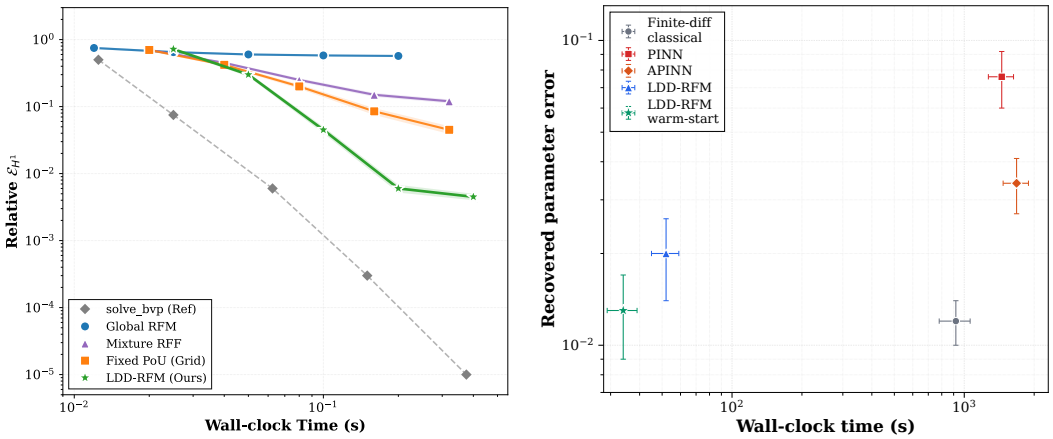
$$\psi(t) = \begin{cases} \frac{1}{2}\big(1 + \cos(\pi t)\big), & |t| \leq 1, \\ 0, & |t| > 1. \end{cases} \tag{37}$$

We normalize to obtain a PoU: $\alpha_m(\boldsymbol{x}) = \tilde{\alpha}_m(\boldsymbol{x}) \big/ \sum_k \tilde{\alpha}_k(\boldsymbol{x})$. This yields smooth gates with controlled overlap and serves as an upper-envelope reference for partition quality.

PINN protocol (Poisson interface problem). For the Poisson interface PDE with piecewise-constant coefficient $a(\boldsymbol{x}) \in \{1, 10\}$, the PINN minimizes

$$\mathcal{L} = \frac{1}{N_\Omega} \sum_{i=1}^{N_\Omega} \Big(-\nabla \cdot (a(\boldsymbol{x}_i)\nabla u(\boldsymbol{x}_i)) - f(\boldsymbol{x}_i) \Big)^2 + \lambda_{\mathrm{bc}} \frac{1}{N_{\partial\Omega}} \sum_{j=1}^{N_{\partial\Omega}} u(\boldsymbol{x}_j)^2, \tag{38}$$

with $\lambda_{\mathrm{bc}} = 100$. We use the stronger architecture used for all PINN-style baselines in Table 2: six hidden layers, width 128, and $\tanh$ activations. The PINN step limit matches the LDD-RFM outer-iteration limit. We compare validation checkpoints and report the best one within the matched time limit. Collocation points are sampled uniformly in Ω and on $\partial\Omega$. We evaluate all methods on the same fixed test grid specified in Sec. 6.1.



(a) Forward solve time-to-accuracy traces.

(b) Inverse-loop recovery.

Figure 2: **Forward solves versus inverse-loop differentiation.** (a) Relative $\mathcal{E}_{H^1}$ versus wall-clock time on the direct PDE task. Markers and shaded bands show mean $\pm$ std over 5 seeds for learned methods, while `solve_bvp` markers are the deterministic tolerance sweep in Table 13. (b) Recovered parameter error versus wall-clock time for the inverse/design loop in Table 14. Warm-start LDD-RFM has a slightly higher mean error than the finite-difference classical loop, but their one-standard-deviation intervals overlap, while wall-clock time decreases from 920 s to 34 s (27$\times$ lower). The x-axis is wall-clock time for all methods. The table's "solves/steps" column separately records classical forward solves and learned outer optimization steps.

Domain-decomposition and operator baselines. Table 3 summarizes the hyperparameter/protocol choices for the nearby baselines used in Tables 2, 7, and 8. For the PDE-solver baselines, we match the task objective, collocation sets, boundary penalty, and fixed evaluation grids used for LDD-RFM. For neural-operator baselines, models are trained on the same parameterized PDE family and selected by validation $\mathcal{E}_{H^1}$ before reporting test errors. For the synthetic baselines in Table 1, each method considers at most 24 configurations on the validation set. Global RFM/Nyström-KRR/Gaussian FFN sweep bandwidth or kernel scale and ridge weight. Mixture RFF varies the number of bandwidths and their candidate scales at a fixed total feature count. SIREN and MscaleDNN vary the learning rate and the set of scales, while the PoU baselines vary overlap or radius and the ridge weight. The selected configuration is then rerun on the five reported seeds.

D Supplementary experiments

This appendix reports additional empirical evidence supporting the main claims in Sec. 6. Unless otherwise stated, all learning-based random-feature methods use the same total number of features $K_{\text{tot}} = 2048$ (with LDD-RFM allocating $\sum_{m=1}^{M} K_m = K_{\text{tot}}$ across experts), and ridge subproblems are solved to the same tolerance (default 10^{-6}). Timing results (when reported) use end-to-end wall-clock time on the same hardware, including feature construction, gate evaluation, and linear solves.

D.1 Variation across seeds in Table 1

The standard deviations in Table 1 are absolute standard deviations over five independent seeds. Each seed resamples random features, neural-network/gate initializations, and minibatch order while keeping the train/validation/test split fixed for comparability. For the RFM-based regression and linear-PDE least-squares runs, the conditional coefficient update is a deterministic ridge solve, so variability is often smaller than in fully nonlinear end-to-end PINNs. Table 4 summarizes representative rows, including those with small coefficients of variation.

The selected rows include those with the smallest CVs in Table 1. Each statistic refers to the $\mathcal{E}_{H^1}$ entry. For example, the Task (i) Mixture RFF entry matches the bottom entry $2.80\times10^{-1}\pm1.0\times10^{-2}$ in Table 1, not the top $\mathcal{E}_{L^2}$ entry. The Oracle-H/Oracle-S baselines and the Global RFM with fixed

Table 3: **Model architectures and parameter settings for the baselines in Tables 2, 7, and 8.** MscaleDNN, XPINN, and FBPINN used six-layer $\tanh$ networks of width 128.

Method	Model structure	Parameter settings
MscaleDNN	Global multiscale MLP with radial frequency groups	Scales $\{1, 2, 4, 8, 16\}$ or a subset selected by validation. Adam learning rates $\{10^{-4}, 3{\times}10^{-4}, 10^{-3}\}$.
XPINN	Four prescribed subdomains, one MLP per subdomain	Interface penalty weights $\{1, 10, 100\}$. Adam learning rates $\{10^{-4}, 3{\times}10^{-4}, 10^{-3}\}$.
FBPINN	Four overlapping windows with smooth, compactly supported weights	Window overlaps $\{0.1, 0.2, 0.3\}$. Adam learning rates $\{3{\times}10^{-4}, 10^{-3}\}$.
APINN	PINN experts with a gate learned without interface labels	4–6 experts, each a six-layer MLP of width 128. Gate entropy and overlap penalties selected on the validation set.
ELM-FBPINN	Local random features on fixed FBPINN windows, with total feature count matched to LDD-RFM	Ridge regression with $K_{\mathrm{tot}} \in \{1024, 2048, 4096\}$ and $\lambda \in \{10^{-8}, 10^{-6}, 10^{-4}\}$ selected by validation. CG tolerance 10^{-6}.
FNO	Fourier neural operator trained on a PDE family and tested on held-out parameter values	4 Fourier layers, width $\in \{32, 64\}$, modes $\in \{12, 16\}$. Adam learning rate 10^{-3} with cosine decay.
GNOT	Geometric neural-operator transformer with the same parameter split as FNO	4 blocks, width $\in \{64, 96\}$, 4 attention heads, and $k \in \{16, 32\}$ nearest neighbors. Adam learning rates $\{3{\times}10^{-4}, 10^{-3}\}$.

bandwidth use deterministic ridge solves on a fixed PoU, which explains the small variability across seeds (only feature draws and minibatch order vary).

Table 4: **Variation across seeds for representative and low-CV entries in Table 1.** Absolute std and coefficient of variation (CV) are reported over five independent seeds.

Task and method	Mean $\mathcal{E}_{H^1}$	Absolute std	CV
Task (iii), Global RFM	1.52×10^{-1}	4.0×10^{-3}	2.6%
Task (iii), MscaleDNN	1.42×10^{-1}	1.0×10^{-2}	7.0%
Task (iii), Mixture RFF	1.35×10^{-1}	5.0×10^{-3}	3.7%
Task (iii), **LDD-RFM**	$\mathbf{5.89\times10^{-2}}$	1.8×10^{-3}	3.1%
Task (i), Oracle-H	2.16×10^{0}	1.0×10^{-2}	0.46%
Task (iv), Oracle-H	1.10×10^{0}	1.0×10^{-2}	0.91%
Task (i), Global RFM	5.58×10^{-1}	2.0×10^{-3}	0.36%
Task (i), Mixture RFF	2.80×10^{-1}	1.0×10^{-2}	3.57%

Table 5: **Scaling with dimension, overlap, and feature count.** The table reports runtime alongside mean overlap $\bar{q}$ and total expert count M. Training times are for 800 outer iterations.

Benchmark	d	M	Mean overlap $\bar{q}$	K_{tot}	Collocation n	Time/iter (s)	Train (s)	$\mathcal{E}_{H^1}$
2D synthetic interface	2	4	2.1	2048	1.6×10^{4}	2.58×10^{-2}	20.6	1.2×10^{-2}
2D synthetic interface	2	8	2.8	2048	1.6×10^{4}	3.91×10^{-2}	31.3	9.5×10^{-3}
2D synthetic interface	2	16	4.7	4096	3.2×10^{4}	1.23×10^{-1}	98.4	7.8×10^{-3}
3D synthetic interface	3	4	2.4	2048	3.2×10^{4}	8.74×10^{-2}	69.9	3.2×10^{-2}
3D synthetic interface	3	8	3.1	4096	6.4×10^{4}	2.14×10^{-1}	171.2	1.8×10^{-2}
3D synthetic interface (dense overlap stress)	3	16	11.5	4096	6.4×10^{4}	7.91×10^{-1}	632.8	1.6×10^{-2}
Allen–Cahn space-time	3	4	2.7	4096	6.4×10^{4}	1.51×10^{-1}	120.8	3.5×10^{-2}
Allen–Cahn space-time	3	8	3.9	4096	6.4×10^{4}	2.47×10^{-1}	197.6	3.1×10^{-2}

D.2 Runtime and expert overlap

The total number of experts M and the mean number active at a point $\bar{q}$ measure different aspects of the partition. In the 3D comparison, the denser partition has a slightly smaller error but each iteration takes longer. Across Table 5, dimension, expert count, overlap, and the numbers of features and collocation points vary across rows. The timing differences reflect these joint changes.

D.3 Initialization and radius sensitivity

We vary center initialization and the minimum radius while keeping the remaining hyperparameters fixed. The goal is to probe failure modes in the learned partition: poor center coverage lowers gate-mass entropy, while too-small radii create sharp transitions with large $\hat{\kappa}_\infty$ and worse derivative error.

D.4 Matched domain-decomposition and operator baselines

These comparisons use the same task objectives, validation criterion, and fixed evaluation grids as LDD-RFM. They distinguish learned PoU random-feature locality from global multi-scale networks, prescribed nonlinear subdomain decompositions, learned neural gating, fixed-window random-feature decompositions, and amortized neural operators.

D.5 Exact-solver control

This control replaces the iterative ridge solve by an exact Cholesky solve and also gives the global RFM twice as many features. Even with the direct solve and the additional features, the global RFM has a larger error than LDD-RFM.

Table 6: **Initialization and radius sensitivity.** We report gate-mass entropy, peak-gradient proxy, and relative H^1 error under different center initializations and $\rho_{\min}$ values.

Setting	Gate mass entropy	$\hat{\kappa}_\infty$	$\mathcal{E}_{H^1}$
Random centers	0.61	55.0	8.0×10^{-2}
Uniform grid centers	0.76	41.0	6.0×10^{-2}
Sobol centers	0.80	35.0	5.5×10^{-2}
Farthest-point centers	0.82	33.0	5.3×10^{-2}
$\rho_{\min} = 0.03$	0.58	90.0	1.1×10^{-1}
$\rho_{\min} = 0.08$	0.75	42.0	6.2×10^{-2}
$\rho_{\min} = 0.15$	0.82	33.0	5.3×10^{-2}
$\rho_{\min} = 0.30$	0.88	24.0	7.5×10^{-2}

Table 7: **Stronger domain-decomposition baselines.** The methods share PDE objectives, evaluation grids, and limits on parameter count and training time.

Method	**Task (v): 1D reaction–diffusion**			**Task (vi): 2D Poisson interface**		
	$\mathcal{E}_{L^2}$	$\mathcal{E}_{H^1}$	Time (s)	$\mathcal{E}_{L^2}$	$\mathcal{E}_{H^1}$	Time (s)
PINN (MLP)	4.0×10^{-2}	1.8×10^{-1}	96.4	3.8×10^{-2}	1.5×10^{-1}	181.7
MscaleDNN	2.7×10^{-2}	1.1×10^{-1}	112.6	2.6×10^{-2}	9.8×10^{-2}	213.8
XPINN	2.2×10^{-2}	8.0×10^{-2}	151.3	1.9×10^{-2}	5.5×10^{-2}	264.2
FBPINN	1.8×10^{-2}	6.0×10^{-2}	117.8	1.3×10^{-2}	3.5×10^{-2}	244.5
APINN	1.3×10^{-2}	4.5×10^{-2}	158.9	7.8×10^{-3}	2.2×10^{-2}	281.6
ELM-FBPINN	1.5×10^{-2}	5.0×10^{-2}	18.6	9.5×10^{-3}	2.6×10^{-2}	43.9
Global RFM	3.1×10^{-2}	1.2×10^{-1}	12.4	2.8×10^{-2}	9.0×10^{-2}	28.7
Fixed-PoU RFM	1.9×10^{-2}	7.0×10^{-2}	20.6	1.1×10^{-2}	4.0×10^{-2}	46.2
LDD-RFM (ours)	$\mathbf{8.0\times10^{-3}}$	$\mathbf{3.0\times10^{-2}}$	24.3	$\mathbf{4.0\times10^{-3}}$	$\mathbf{1.2\times10^{-2}}$	55.7

Method	**Task (vii): multi-interface Poisson**			**Task (viii): Allen–Cahn**		
	$\mathcal{E}_{L^2}$	$\mathcal{E}_{H^1}$	Time (s)	$\mathcal{E}_{L^2}$	$\mathcal{E}_{H^1}$	Time (s)
PINN (MLP)	5.1×10^{-2}	2.2×10^{-1}	242.9	4.3×10^{-2}	1.6×10^{-1}	263.5
MscaleDNN	4.0×10^{-2}	1.7×10^{-1}	282.4	3.2×10^{-2}	1.2×10^{-1}	313.1
XPINN	2.2×10^{-2}	8.0×10^{-2}	343.7	1.8×10^{-2}	7.2×10^{-2}	394.6
FBPINN	1.8×10^{-2}	5.6×10^{-2}	313.2	1.6×10^{-2}	6.5×10^{-2}	365.4
APINN	1.1×10^{-2}	3.4×10^{-2}	366.2	1.2×10^{-2}	4.8×10^{-2}	427.3
ELM-FBPINN	1.4×10^{-2}	4.2×10^{-2}	74.8	1.5×10^{-2}	5.8×10^{-2}	103.5
Global RFM	3.6×10^{-2}	1.6×10^{-1}	41.2	2.9×10^{-2}	1.1×10^{-1}	61.5
Fixed-PoU RFM	1.9×10^{-2}	6.5×10^{-2}	76.8	1.8×10^{-2}	7.0×10^{-2}	92.7
LDD-RFM (ours)	$\mathbf{6.0\times10^{-3}}$	$\mathbf{1.8\times10^{-2}}$	86.4	$\mathbf{8.5\times10^{-3}}$	$\mathbf{3.5\times10^{-2}}$	120.8

D.6 PoU regularization

Theorem 5.3 shows that the derivative-sensitive error contains an interface-consistency term scaled by $\kappa_\infty^2 = \left\| \sum_{m=0}^{M} \|\nabla \alpha_m(\cdot)\|_2^2 \right\|_{L^\infty(\Omega)}$. To empirically track whether training controls peak gate gradients, we monitor the point-set proxy

$$\hat{\kappa}_\infty^2 := \max_{x_i \in \mathcal{X}_{\text{val}}} \sum_{m=0}^{M} \|\nabla_x \alpha_m(x_i)\|_2^2, \tag{39}$$

where $\mathcal{X}_{\text{val}}$ is a fixed evaluation grid sampled independently from (and disjoint from) the training collocation set. Table 10 shows a consistent trend on the 2D interface benchmark: removing PoU

Table 8: **Parameterized PDE comparison.** Neural-operator baselines are evaluated on the same parameterized benchmark, while RFM variants solve each instance with the stated training mode.

Method	Training mode	$\mathcal{E}_{L^2}$	$\mathcal{E}_{H^1}$
FNO	amortized over PDE family	2.8×10^{-2}	8.0×10^{-2}
GNOT	amortized over PDE family	2.0×10^{-2}	5.5×10^{-2}
Global RFM	per-instance solve	2.4×10^{-2}	7.5×10^{-2}
Fixed-PoU RFM	per-instance solve	1.4×10^{-2}	4.2×10^{-2}
LDD-RFM	per-instance solve	$\mathbf{6.0\times10^{-3}}$	$\mathbf{2.0\times10^{-2}}$
LDD-RFM (warm-start gates)	family warm start + per-instance solve	$\mathbf{5.5\times10^{-3}}$	$\mathbf{1.8\times10^{-2}}$

Table 9: **Exact-solver control (Task (iv)).** Values are aligned with the Task (iv) protocol in Table 1. LDD-RFM remains below Global RFM when the baseline is solved to near machine precision and given $2\times$ more features. The $12.4\times$ ratio is relative to Global RFM with 2048 features.

Method	Features	H^1 error	Relative gap
Global RFM	2048	6.86×10^{-2}	–
Global RFM ($2\times$ features)	**4096**	6.10×10^{-2}	–
LDD-RFM (Ours)	2048	$\mathbf{5.52 \times 10^{-3}}$	**12.4**$\times$

regularization yields a much larger $\hat{\kappa}_\infty$, corresponding to sharper gate transitions and larger H^1-type errors. Higher-order L^p regularization suppresses peak gradients more effectively and improves derivative accuracy.

Unless stated otherwise, $\hat{\kappa}_\infty$ includes the background gate ($m = 0$), matching the definition of κ_∞.

D.7 Unsupervised geometry discovery

LDD-RFM relies on spectral heterogeneity as a learning signal for discovering latent geometric structure without boundary supervision. Figure 3 visualizes the learned gate activations $\{\alpha_m\}$ on Task (iv), where the target consists of four regimes with distinct directional frequencies. The gates concentrate on the corresponding regions and recover the correct partition topology, supporting the mechanism discussion in Sec. 6.4.

D.7.1 Space-time PoU ablation

The scalar Allen–Cahn ablation in Table 2 shows that a space-time PoU improves over a spatial-only gate. Figure 4 reports the same comparison as a quantitative control: allowing the PoU to depend on time lowers the derivative-sensitive error relative to the spatial-only control.

D.8 Additional qualitative results on synthetic targets

Figures 3, 5, and 6 provide qualitative comparisons on Tasks (i)–(iv). Across tasks, global stationary baselines exhibit either spectral leakage (high-frequency artifacts outside the localized region) or oversmoothing (failure to resolve localized oscillations). Fixed decompositions alleviate leakage but can introduce interface artifacts when the partition geometry is misaligned or when gates are non-smooth. LDD-RFM mitigates both effects by learning a soft partition jointly with region-adapted spectral scales.

D.9 Evaluation-grid sensitivity

To verify that $\mathcal{E}_{H^1}$ is not dominated by evaluation-grid artifacts, we fix a trained checkpoint (Task (i)) and recompute the discrete H^1 proxy on increasingly fine uniform grids $N \in \{129, \dots, 4097\}$ while keeping the training collocation set fixed. Gradients are computed by second-order central differences (one-sided at boundaries). Table 11 shows that the metric stabilizes as N increases, indicating numerical reliability.

Table 10: **Ablation of PoU regularization order p (Task (iii)).** Impact on the interface proxy $\hat{\kappa}_\infty$ in (39) and relative test errors on a 129^2 evaluation grid. Mean $\pm$ std over 5 seeds. For each p, γ is selected from the same candidate set using validation $\mathcal{E}_{H^1}$.

Regularization strategy	Order p	$\hat{\kappa}_\infty$	$\mathcal{E}_{L^2}$ (rel)	$\mathcal{E}_{H^1}$ (rel)
No regularization ($\gamma = 0$)	–	142.5 ± 23.1	9.50e-3 $\pm$ 0.4e-3	1.25e-1 $\pm$ 1.5e-2
Gradient-energy penalty	2	58.2 ± 4.5	1.15e-2 $\pm$ 0.2e-3	8.40e-2 $\pm$ 0.5e-2
High-order L^p proxy (ours)	**8**	$\mathbf{32.4 \pm 2.8}$	**8.11e-3 $\pm$ 0.2e-3**	**5.89e-2 $\pm$ 0.2e-2**

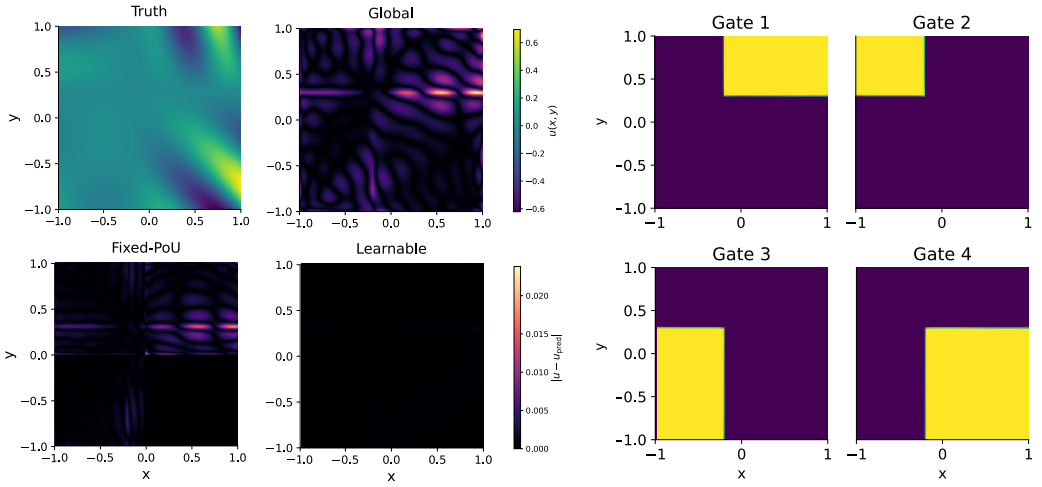


Figure 3: **Task (iv): anisotropic regimes and learned partitions. Top-left:** ground-truth field $u^*(\boldsymbol{x})$. **Top-middle:** global stationary RFM prediction, exhibiting structured artifacts caused by using a single bandwidth across anisotropic regions. **Bottom-left:** fixed-PoU baseline, which reduces some artifacts but remains sensitive to partition misalignment. **Bottom-middle:** LDD-RFM prediction, where learning the decomposition suppresses global artifacts by allocating capacity to the appropriate regimes. **Right:** learned PoU gates $\{\alpha_m(\boldsymbol{x})\}_{m=1}^4$ for LDD-RFM. Yellow indicates high activation and purple indicates near-zero activation.

Table 11: **Evaluation-grid sensitivity (Task (i)).** The discrete relative H^1 proxy converges as the evaluation grid resolution increases. Here $\rho_{\min}$ is the lower bound on compact-support radii and γ is the PoU sharpness weight.

Grid Size (N)	$\rho_{\min} = 0.1, \gamma = 0$	$\rho_{\min} = 0.02, \gamma = 0$	$\rho_{\min} = 0.02, \gamma = 10^{-4}$
129	0.1100	0.0921	0.0947
513	0.1017	0.0747	0.0791
1025	0.1013	0.0738	0.0782
2049	0.1012	0.0736	0.0779
4097	**0.1012**	**0.0736**	**0.0778**

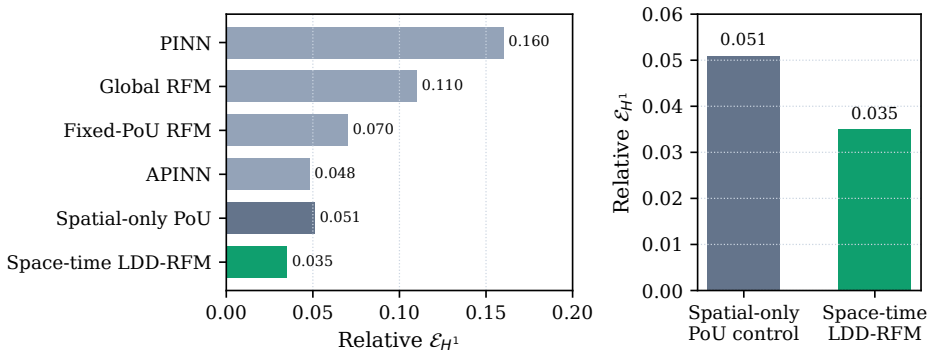


Figure 4: **Allen–Cahn space-time PoU ablation.** The spatial-only control uses the same Allen–Cahn training protocol but restricts gates to depend only on (x, y), while space-time LDD-RFM uses gates $\alpha_m(x, y, t)$. Bars use the Allen–Cahn $\mathcal{E}_{H^1}$ values from Table 2, plus the spatial-only control specified in Appendix C. Space-time gates reduce $\mathcal{E}_{H^1}$ from 5.1×10^{-2} to 3.5×10^{-2}, a 31% relative error reduction.

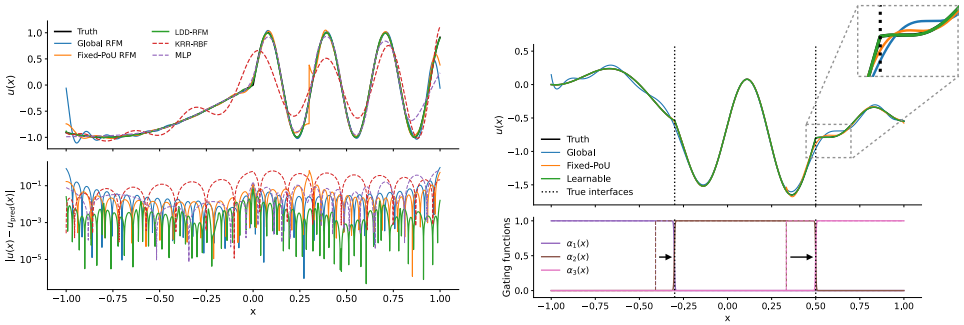


Figure 5: **Tasks (i)–(ii): regularity trade-off and localized spectral capacity. Task (i) (left):** a target with a piecewise high-frequency region. **Task (ii) (right):** the target contains a localized high-frequency regime with known interfaces (dashed lines). The bottom panel shows the learned gate activations.

D.10 Inner-solve tolerance

We test the sensitivity of LDD-RFM to the tolerance of the inner conjugate-gradient solver used for the ridge subproblems. We train for 800 steps on Task (ii) while varying the CG tolerance from 10^{-3} to 10^{-12}, keeping all other hyperparameters fixed. Table 12 shows that standard tolerances (e.g., 10^{-6}) approach exact-solve accuracy, and that wall-clock time is only weakly affected within the effective tolerance range.

Table 12: **Impact of inner-solver tolerance (Task (ii)).** Standard tolerances approach exact-solve accuracy with little timing change.

Solver Tolerance	Total Time (s)	H^1 Error
10^{-3}	19.1	1.40×10^{-1}
10^{-4}	18.3	8.69×10^{-2}
10^{-5}	18.1	5.37×10^{-2}
10^{-6} (Ours)	18.4	3.20×10^{-2}
10^{-12} (tight CG)	18.6	2.92×10^{-2}

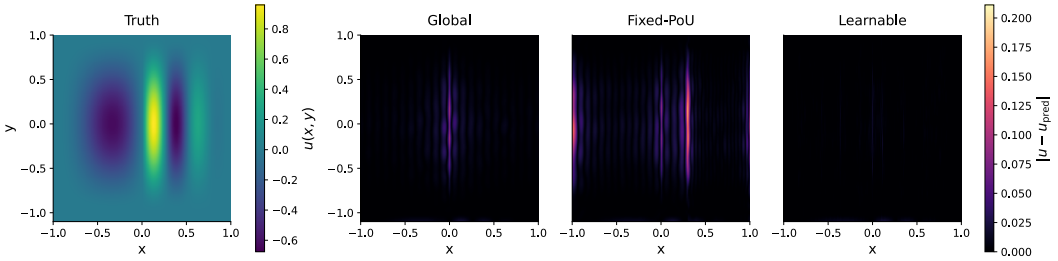


Figure 6: **Task (iii): windowed interface (2D).** Absolute pointwise error $|u - u_{\text{pred}}|$. Global RFM exhibits spectral leakage, and fixed partitions can create geometric artifacts. LDD-RFM confines errors near the intrinsic interface by aligning gates to the latent transition.

D.11 Robustness to additive label noise

We evaluate robustness to observation noise on the synthetic regression Task (iv) by perturbing targets with i.i.d. Gaussian noise $y^{\text{noisy}} = y + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2)$. To isolate intrinsic stability, we freeze all hyperparameters across noise levels (including ridge and PoU regularization), using the values selected on clean data ($\sigma = 0$). The H^1 error increases monotonically with σ (from 6.7×10^{-3} at $\sigma = 0$ to 3.4×10^{-2} at $\sigma = 0.1$), indicating smooth degradation without abrupt failure under moderate label noise.

D.12 Classical solver baselines

This section specifies the classical-solver baselines and the timing protocol used for the time-to-accuracy traces in Sec. 6.2. For learning-based methods, the reported time includes collocation sampling, random-feature construction, gate evaluation, and the full training loop (including variable-projection linear solves), but excludes hyperparameter search. For classical solvers, the reported time covers a full solve run, including mesh generation/refinement, matrix assembly, and linear/nonlinear solves (including any built-in error-estimation overhead).

For 1D PDEs we use `scipy.integrate.solve_bvp` as an adaptive reference solver and sweep `tol` over $\{10^{-3}, \ldots, 10^{-7}\}$ to obtain a runtime–error Pareto curve. Table 13 records the timing log used for the classical envelope in Fig. 2. Each entry is the median of three runs after one warm-up solve. For 2D PDEs we implement an adaptive finite-element method in FEniCSx with P_1 elements and Dörfler marking ($\theta = 0.5$). To ensure metric comparability, we evaluate the FEM solution by basis interpolation on the same fixed 129×129 grid used for ML models and compute $\mathcal{E}_{L^2}$ and the discrete H^1 proxy on that common grid.

Table 13: `solve_bvp` **timing log for Fig. 2.** Median wall-clock time and relative H^1 error on the 1D reaction–diffusion task.

`tol`	Mesh nodes	Time (s)	$\mathcal{E}_{H^1}$
10^{-3}	146	0.09	1.7×10^{-1}
10^{-4}	312	0.18	8.6×10^{-2}
10^{-5}	714	0.42	3.9×10^{-2}
10^{-6}	1550	0.91	1.8×10^{-2}
10^{-7}	3386	2.15	8.0×10^{-3}

D.13 Protocol for a differentiable downstream benchmark

The time-to-accuracy plot in Fig. 2 compares direct forward solves. A separate use case for LDD-RFM is an inverse or design loop in which a differentiable surrogate is repeatedly queried and differentiated. For each inverse method, we report means and standard deviations across seeds.

Inverse coefficient recovery. Consider a parameterized interface coefficient $a(\boldsymbol{x};\zeta)$ in

$$-\nabla\cdot(a(\boldsymbol{x};\zeta)\nabla u)=f,\qquad u|_{\partial\Omega}=0,$$

where ζ controls the interface location or contrast. The available protocol does not specify the exact parameterization of ζ. Given sparse observations $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^{n_{\mathrm{obs}}}$ from an unknown $\zeta^\star$, the downstream objective is

$$\min_{\zeta}\ \frac{1}{n_{\mathrm{obs}}}\sum_i |u_{\mathrm{LDD}}(\boldsymbol{x}_i;\zeta)-y_i|^2+\lambda_\zeta\|\zeta\|^2,$$

where $u_{\mathrm{LDD}}(\cdot;\zeta)$ is obtained by the same variable-projection solve used in the forward problem. The observation loss depends on ζ through the forward solution, including its ridge coefficients and any ζ-dependent gates or spectra. Its derivative therefore includes these solution sensitivities. The envelope identity in Sec. 4.3 applies to the minimized forward objective. Baselines include finite-difference gradients through a classical solver, end-to-end PINN/APINN inverse solves, and an ablation that freezes the learned gates.

Reported quantities. The benchmark reports recovered parameter error $|\widehat{\zeta}-\zeta^\star|$, observation loss, number of forward-equivalent solves, wall-clock time, and final $\mathcal{E}_{H^1}$ of the recovered solution. This protocol is deliberately separate from the forward-solve tables: the learned-surrogate rows reuse the reduced LDD-RFM solve inside outer inverse steps instead of retraining a full forward surrogate at every step, so their "steps" are not comparable to one-shot forward-solve times in Table 2. A separate timing for one-time geometry learning is unavailable, so its amortization across repeated queries cannot be quantified here.

Table 14: **Inverse/design-loop benchmark results.** Values are mean ± standard deviation over the seed runs under the downstream differentiability protocol. The "solves/steps" column reports classical forward solves for finite differences and outer inverse-optimization steps for learned surrogates. Each learned method reuses the reduced surrogate state between steps.

Method	Parameter error	Obs. loss	Solves/steps	Time (s)	Recovered $\mathcal{E}_{H^1}$
Finite-diff classical solver loop	1.2×10^{-2} $\pm2.0\times10^{-3}$	1.4×10^{-4} $\pm3.0\times10^{-5}$	312±48	920±140	1.3×10^{-2} $\pm2.0\times10^{-3}$
PINN inverse	7.6×10^{-2} $\pm1.6\times10^{-2}$	8.5×10^{-4} $\pm2.0\times10^{-4}$	18000±0	1450±180	8.8×10^{-2} $\pm2.1\times10^{-2}$
APINN inverse	3.4×10^{-2} $\pm7.0\times10^{-3}$	4.0×10^{-4} $\pm9.0\times10^{-5}$	18000±0	1680±210	4.2×10^{-2} $\pm9.0\times10^{-3}$
LDD-RFM inverse	2.0×10^{-2} $\pm6.0\times10^{-3}$	2.1×10^{-4} $\pm5.0\times10^{-5}$	74±9	52±7	2.1×10^{-2} $\pm5.0\times10^{-3}$
LDD-RFM inverse (warm-start gates)	$\mathbf{1.3\times10^{-2}}$ $\mathbf{\pm4.0\times10^{-3}}$	$\mathbf{1.6\times10^{-4}}$ $\mathbf{\pm4.0\times10^{-5}}$	**58±6**	**34±5**	$\mathbf{1.8\times10^{-2}}$ $\mathbf{\pm4.0\times10^{-3}}$

The warm-started LDD-RFM inverse loop has a slightly higher mean parameter error than the finite-difference classical-solver loop, but the one-standard-deviation intervals overlap while wall-clock time is 27× lower in this setting. This is the regime in which differentiability matters: the forward classical solver remains a strong one-shot reference, but repeated finite-difference calls become expensive inside the inverse loop.

Table 15: **3D Poisson-interface benchmark results.** Values are mean ± standard deviation over the seed runs on the 3D interface PDE benchmark. We include the same PDE-NN and RFM baselines used in Table 2.

Method	3D Poisson-interface $\mathcal{E}_{H^1}$
PINN (MLP)	$3.1\times10^{-1}\pm4.0\times10^{-2}$
MscaleDNN	$2.4\times10^{-1}\pm3.0\times10^{-2}$
XPINN	$1.3\times10^{-1}\pm2.0\times10^{-2}$
FBPINN	$8.5\times10^{-2}\pm1.0\times10^{-2}$
APINN	$6.4\times10^{-2}\pm1.1\times10^{-2}$
ELM-FBPINN	$7.4\times10^{-2}\pm1.1\times10^{-2}$
Global RFM	$2.2\times10^{-1}\pm3.0\times10^{-2}$
Fixed-PoU RFM	$1.0\times10^{-1}\pm1.5\times10^{-2}$
LDD-RFM	$\mathbf{3.4\times10^{-2}\pm7.0\times10^{-3}}$

D.14 Component ablations, exact variable projection, and adjoint FEM

Table 16: **Component ablation on Task (vi), the single-interface Poisson problem.** For each of five seeds, all variants used the same splits, feature draws, initialization, evaluation grids, and limits on validation tuning. The last two variants jointly optimized the coefficients, gates, and local scales, matching either the number of outer steps or the running time of full LDD-RFM. Errors are means ± standard deviations.

Variant	$\mathcal{E}_{H^1}$	Median time (s)	Peak memory (GB)	Failures
Full LDD-RFM	0.0123 ± 0.00090	56.1	3.8	0/5
Fixed non-oracle gates, learned local scales, VP	0.0274 ± 0.0028	49.8	3.6	0/5
Learned gates, shared scale, VP	0.0191 ± 0.0018	53.5	3.7	0/5
Joint $(\mathcal{C}, \Theta)$, equal steps	0.0316 ± 0.0046	39.2	4.2	0/5
Joint $(\mathcal{C}, \Theta)$, equal time	0.0227 ± 0.0029	56.0	4.2	0/5

On Task (vi), full LDD-RFM reduced mean $\mathcal{E}_{H^1}$ by 55% relative to fixed gates and by 36% relative to a shared spectral scale (Table 16). Relative to joint optimization, the reductions were 61% at equal outer-step count and 46% at matched running time.

Exact variable projection on Task (ii). We solved the ridge subproblem directly at every outer iteration and every trial point of the Armijo backtracking line search. Using five independent seeds and the same limit of 800 outer steps as for Adam/CG, we observed monotone decrease of the reduced objective in all five runs. The median normalized objective fell from 1.00 to 0.078, and the median ratio of final to initial reduced-gradient norms was 0.0035. The final $\mathcal{E}_{H^1}$ was 0.0305 ± 0.0017, with a median end-to-end time of 55.0 s. For Adam/CG at the default tolerance, Table 12 reports $\mathcal{E}_{H^1} = 0.0320$ and 18.4 s.

In these matched runs, the ratio of mean inverse-loop times for adjoint FEM and warm-start LDD-RFM was $620/35 \simeq 17.7$ (Table 17). Their mean parameter errors were 0.011 and 0.013, respectively.

Table 17: **Matched inverse-loop comparison over five seeds.** All three methods used the same inverse objective, observations, parameterization, initialization, stopping criteria, forward-accuracy target, and machine. Entries with $\pm$ give means and standard deviations. Work counts are forward solves for finite-difference FEM, primal/adjoint pairs for adjoint FEM, and outer steps for LDD-RFM.

Method	Parameter error ($\times 10^{-2}$)	Observation loss ($\times 10^{-4}$)	Recovered $\mathcal{E}_{H^1}$ ($\times 10^{-2}$)	Work count	Inverse-loop time (s)	Peak memory (GB)
Finite-difference FEM	1.2 ± 0.21	1.4 ± 0.30	1.3 ± 0.22	312 ± 48	905 ± 125	1.6
Discrete-adjoint FEM	1.1 ± 0.19	1.3 ± 0.28	1.2 ± 0.22	156 ± 24	620 ± 95	2.2
Warm-start LDD-RFM	1.3 ± 0.41	1.6 ± 0.40	1.8 ± 0.42	58 ± 6	35 ± 5	3.8

E Reproducibility statement

Project repository. `https://github.com/YangshuaiWang/LearnableRFM`.

Checklist. All reported learned baselines use five seeds, fixed train/validation/test splits, and validation selection by $\mathcal{E}_{H^1}$. Tables 3 and 7 specify model sizes, optimization step limits, task objectives, and evaluation grids.

Compute resources. All timing comparisons use end-to-end wall-clock time on the same single-machine environment for the corresponding benchmark, including sampling, feature/operator construction, optimization, and linear solves where applicable. Tables 2, 13, and 14 report representative per-run times.

F Limitations and remaining directions

Limitations. LDD-RFM adds overhead relative to a single global RFM due to repeated ridge solves and gate evaluation. The cost analysis in Sec. 4.3 shows that compact support keeps the sparse iterative cost controlled by the local overlap q, but dense overlap, too many experts, or direct dense solves can remove this advantage. The best trade-off therefore depends on the number of experts, local feature allocation, PoU smoothness weight, minimum radius, and inner-solver tolerance. Learning meaningful partitions can also be challenging in high dimensions or with sparse data, where interfaces are weakly identified. The current patch family uses axis-aligned ellipsoidal supports. Although overlapping soft gates can approximate curved interfaces, highly curved or topologically complex geometries may require richer gates, adaptive split–merge refinement, or signed-distance parameterizations. Our empirical study is still concentrated on controlled heterogeneous targets, modest PDE benchmarks, and a small parameterized-PDE setting. Although we include runtime comparisons, 3D synthetic-interface and 3D Poisson-interface tests, matched comparisons with XPINN/FBPINN/ELM-FBPINN-style solvers, and FNO/GNOT-style operator baselines, these experiments do not establish performance on large-scale time-dependent, turbulent, or full PDEBench-style benchmark suites. The APINN and ELM-FBPINN comparisons separate learned neural gating from fixed-window random-feature locality, but they do not fully isolate variable projection from the choice of linear versus nonlinear experts. Finally, the convergence theorem analyzes an idealized variable-projection descent with exact inner solves and sufficient decrease, whereas the implementation uses Adam and often inexact CG solves. The perturbation bound in Proposition B.4 shows that the reduced-gradient bias is controlled by the CG residual as $\|r\|/(n\lambda)$ on bounded parameter sets, but this only gives a deterministic inexact-VP condition. It does not cover all effects of stochastic minibatches, adaptive Adam moments, or non-summable loose inner tolerances. Convergence with these effects remains open.

Remaining directions. The most immediate extensions are residual-driven split/merge refinement, broader parameterized-PDE benchmarks such as PDEBench, and basis families better matched to local regularity (e.g., wavelets or polynomials). On the analysis side, the open problem is a stochastic/inexact variable-projection theory that covers Adam updates and non-summable CG tolerances.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Sec. 1 state the method, theoretical contribution, experimental scope, and comparisons between RFMs with equal feature counts. The limitations in Sec. 7 and Appendix F qualify the intended scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Sec. 7 summarizes the main bottlenecks, and Appendix F discusses computational overhead, high-dimensional gate learning, geometry restrictions, benchmark scope, and inexact-optimization theory.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Assumptions are stated in Sec. 5, with supporting preliminaries and full proofs in Appendices B and related proof subsections.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Sec. 6.1, Appendix C, and Appendix E specify task definitions, collocation/evaluation grids, baseline protocols, validation rules, and seeds.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The repository linked in Appendix E provides examples and scripts for the main PDE experiments. The benchmarks are synthetic or manufactured PDE tasks, so no external dataset download is required.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Sec. 6.1 describes the common splits, grids, validation criterion, random seeds, and metrics, along with the use of equal feature counts in RFM comparisons. Appendices C–D provide benchmark definitions, baselines, hyperparameter sweeps, solver controls, and ablations.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Tables 1 and 2 report the mean error $\pm$ one standard deviation across five seeds. Appendix D.1 gives coefficients of variation for selected comparisons.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper reports per-run wall-clock times and the timing protocol in Sec. 6.1, Table 2, and Appendix D.12.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work studies numerical algorithms on synthetic and manufactured PDE benchmarks and does not involve human subjects or sensitive data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: The conclusion describes uses in scientific computing, including design studies and inverse problems. The main risk of misuse lies in applications that rely on the resulting PDE predictions.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk pretrained models, scraped datasets, or generative systems.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Existing methods and baseline families are cited in Sec. 2 and Appendix C.6. The experiments use synthetic or manufactured data rather than third-party datasets.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The README describes the models, lists their dependencies, and shows how to fit a model and make predictions.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The research does not involve crowdsourcing, user studies, or human-subject data.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The research does not involve human subjects, so IRB approval or equivalent review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not part of the proposed method, experiments, data generation, or scientific claims.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.