Journal of Machine Learning Research 27 (2026) 1-58 Submitted 6/25; Revised 4/26; Published 7/26

Solving Nonlinear PDEs with Sparse Radial Basis Function Networks

Zihan Shao Z6SHAO@UCSD.EDU
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093, United States

Konstantin Pieper PIEPERK@ORNL.GOV
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN 37831, United States

Xiaochuan Tian XCTIAN@UCSD.EDU
Department of Mathematics
University of California, San Diego
La Jolla, CA 92093, United States

Editor: Christophe Giraud

Abstract

We propose a novel framework for solving nonlinear PDEs using sparse radial basis function (RBF) networks. Sparsity-promoting regularization is employed to prevent overparameterization and reduce redundant features. This work is motivated by longstanding challenges in traditional RBF collocation methods, along with the limitations of physics-informed neural networks (PINNs) and Gaussian process (GP) approaches, aiming to blend their respective strengths in a unified framework. The theoretical foundation of our approach lies in the function space of Reproducing Kernel Banach Spaces (RKBS) induced by one-hidden-layer neural networks of possibly infinite width. We prove a representer theorem showing that the sparse optimization problem in the RKBS admits a finite solution and establish error bounds that offer a foundation for generalizing classical numerical analysis. The algorithmic framework is based on a three-phase algorithm to maintain computational efficiency through adaptive feature selection, second-order optimization, and pruning of inactive neurons. Numerical experiments demonstrate the effectiveness of our method and highlight cases where it offers notable advantages over GP approaches. This work opens new directions for adaptive PDE solvers grounded in rigorous analysis with efficient, learning-inspired implementation.

Keywords: Sparse RBF networks; Nonlinear PDEs; Reproducing Kernel Banach Spaces; Representer theorem; Adaptive feature selection; Convergence analysis; Adaptive collocation solver

1. Introduction

This work introduces a sparse neural network approach for solving nonlinear partial differential equations (PDEs). An adaptive training process is introduced for shallow neural networks with sparsity-promoting regularization, where neurons are gradually added to

maintain a compact network structure. The PDEs considered in this paper are defined in a bounded open set $D \subset \mathbb{R}^d$ and subject to "boundary conditions" on ∂D of the following form:

$$\begin{cases} \mathcal{E}[u](x) = 0, & x \in D, \\ \mathcal{B}[u](x) = 0, & x \in \partial D. \end{cases} \tag{1.1}$$

Here, $\mathcal{E}[u](x)$ and $\mathcal{B}[u](x)$ are defined as

$$\mathcal{E}[u](x) := E(x, u(x), \nabla u(x), \nabla^2 u(x)), \quad \mathcal{B}[u](x) := B(x, u(x), \nabla u(x)). \tag{1.2}$$

where E is a real-valued function on $\overline{D} \times \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{d\times d}$ and B is a real-valued function on $\partial D \times \mathbb{R} \times \mathbb{R}^d$ that determine the partial differential equation on D and the boundary condition on ∂D, respectively. We note that here $\mathcal{B}$ is a general boundary operator, which may lead to boundary or initial value problems in the context of PDEs. A typical example falling within this framework is a second-order semilinear elliptic PDE with Dirichlet boundary conditions. Additional assumptions on the PDEs and further examples are provided in Section 1.3.

Classical numerical methods for solving PDEs, such as finite difference methods and finite element methods, have been extensively developed and broadly applied since the mid-20th century. Meshless methods represent a more recent development, with origins tracing back at least to the 1970s through the introduction of smoothed particle hydrodynamics for astrophysical simulations (Gingold and Monaghan, 1977; Lucy, 1977). These methods gained increasing attention in the 1990s, leading to the emergence of various approaches, including the element-free Galerkin method, reproducing kernel particle methods, radial basis function (RBF) methods, and others (Belytschko et al., 1994; Buhmann, 2000; Fasshauer, 1996; Franke and Schaback, 1998; Li and Liu, 2007; Liu et al., 1996; Wendland, 2004). A notable connection exists between meshless methods and machine learning through the use of kernel techniques, which have achieved considerable success in various machine learning applications (Scholkopf and Smola, 2018). The link between kernel-based approaches in machine learning and numerical solutions of PDEs was insightfully highlighted in the survey by Schaback and Wendland (2006). Kernel-based approaches are particularly effective for linear problems, such as regression or numerical solutions of linear PDEs, where the underlying solvers depend on techniques from numerical linear algebra. The recent popularity of neural networks has shifted the focus toward inherently nonlinear techniques, even for linear problems (E and Yu, 2018; Goodfellow et al., 2016; Raissi et al., 2019). As a result, optimization has begun to replace traditional linear algebra as the central computational framework. However, neural network-based approaches, such as physics-informed neural networks (PINNs) (Raissi et al., 2019), present significant challenges in training, due to their sensitivity to hyperparameter tuning and the substantial computational cost associated with large, often over-parameterized architectures that may include many redundant features. A notable recent development is the Gaussian process (GP) approaches (Batlle et al., 2025; Chen et al., 2021, 2025), which extend kernel-based methods to solving nonlinear PDEs. The central idea of GP typically involves solving a linear system as an inner step combined with an outer optimization step. However, since they are reduced to classical kernel-based or RBF methods in linear cases, many of the limitations associated with traditional RBF approaches still apply.

This work is originally motivated by the aim of addressing some of the persistent limitations of traditional RBF collocation methods. While RBF collocation methods are well known for their advantages, such as being inherently meshfree and offering fast or even spectral convergence in certain cases, they also face significant challenges. A central difficulty lies in the choice of the kernel scaling parameter, which plays a critical role in the success of the method but lacks a universal selection criterion. Choosing this parameter typically involves a trade-off between approximation accuracy and the condition number of the resulting system, and there is no clear guideline for achieving an optimal balance. For large linear systems, RBF collocation often leads to extremely ill-conditioned systems. In their survey, Schaback and Wendland (2006) envisioned that one important direction for developments in kernel-based approximation should focus on adaptive algorithms that solve the problems approximately, thereby alleviating the issue of ill-conditioning in large systems. The proposed approach, which is broadly applicable, is an actualization of their concept of adaptivity by selecting only a small number of features through sparsity-promoting regularization. Importantly, the kernel scale parameter is also determined adaptively, eliminating the need for tuning and resolving the long-standing issue of scale selection. This represents a new development, even in the context of kernel-based linear regression or function approximation. Our work is also motivated by the goal of addressing some of the issues of PINNs and GP methods. In this sense, the proposed approach can be viewed as: (1) a PINN framework trained using a shallow and sparse neural network for more efficient optimization, and (2) an extension of GP methods that enables adaptive selection of kernel parameters. We note that the proposed approach can be viewed as a relaxation of kernel collocation methods, as the empirical loss is constructed from pointwise evaluations of PDEs. This leads to an interpretation of the method as an "adaptive collocation solver". The formulation is intentionally simple, allowing us to clearly demonstrate the potential of the idea, both in terms of theoretical foundation and practical performance. Lastly, the adaptivity in our approach lies in the choice of trial functions. Other extensions, such as alternative loss function formulations or adaptivity in the selection of test points, may serve as promising directions for future research.

Structure of this section. We begin in Section 1.1 with a summary of our approach and a highlight of the main contributions. Section 1.2 discusses related work, where we compare our method with existing frameworks. In Section 1.3, we introduce the notation and assumptions used throughout the paper and provide representative examples of the PDE problems considered. Finally, Section 1.4 outlines the structure of the rest of the paper.

1.1 Summary of approach and contributions

A shallow neural network of N neurons is a function defined as

$$u_{c,\omega}(x) = \sum_{n=1}^{N} c_n \varphi(x; \omega_n), \tag{1.3}$$

where φ is the feature function, $c = \{c_n\}_{n=1}^N \subseteq \mathbb{R}$ represents the outer weights, and $\omega = \{\omega_n\}_{n=1}^N \subseteq \Omega$ denotes the inner weights, with Ω being a prescribed parameter space. N

is also referred to as the network width. To illustrate our approach, we consider Gaussian Radial Basis Function (RBF) networks, where the feature function φ is defined by

$$\varphi(x;\ y,\sigma) = \frac{\sigma^s}{\left(\sqrt{2\pi}\sigma\right)^d} \exp\left(-\frac{\|x-y\|_2^2}{2\sigma^2}\right).$$

In this case, the inner weights $\{\omega_n = (y_n, \sigma_n) \in \mathbb{R}^d \times \mathbb{R}_+\}$ represent the centers and shapes of the Gaussian functions. We note that the feature function is the standard Gaussian probability density function multiplied by the scale-dependent weight σ^s, so that $\int_{\mathbb{R}^d} \varphi(x;\ y,\sigma)\,\mathrm{d}x = \sigma^s$. A common approach in the RBF literature is to omit the scaling in front of the exponential entirely, which corresponds to a choice of $s = d$ up to a constant pre-factor $(\sqrt{2\pi})^{-d}$. For a physics informed loss associated to a PDE it is important to consider general s, and set s larger than $d + k$, where k denotes the highest order of the differential operators evaluated in $\mathcal{E}[u]$ and $\mathcal{B}[u]$. Together with the regularization of the outer weights, this ensures stability and generalization properties; see Section 2.

As is customary in neural network-based PDE approximation, we define a measure ν_D on D and $\nu_{\partial D}$ on its boundary and consider the squared residual loss function

$$L(u) = \frac{1}{2}\|\mathcal{E}[u]\|^2_{L^2(\nu_D)} + \frac{\lambda}{2}\|\mathcal{B}[u]\|^2_{L^2(\nu_{\partial D})} \tag{1.4}$$

where λ is an appropriate penalty parameter. In practice, we use the empirical loss function $\hat{L}_{K_1,K_2}$ defined as

$$\hat{L}_{K_1,K_2}(u) = \frac{1}{2}\sum_{k=1}^{K_1} w_{1,k}(\mathcal{E}[u](x_{1,k}))^2 + \frac{\lambda}{2}\sum_{k=1}^{K_2} w_{2,k}(\mathcal{B}[u](x_{2,k}))^2, \tag{1.5}$$

where $\{x_{1,k}\}_{k=1}^{K_1} \subseteq D$ and $\{x_{2,k}\}_{k=1}^{K_2} \subseteq \partial D$ are quadrature or collocation points, and $\{w_{1,k}\}_{k=1}^{K_1} \subseteq \mathbb{R}^+$ and $\{w_{2,k}\}_{k=1}^{K_2} \subseteq \mathbb{R}^+$ are the associated (positive) weights used to approximate the two integrals in (1.4) and the total number of points is $K = K_1 + K_2$. For instance, we can consider a finite number of randomly generated points or (quasi-)uniform grids; the precise assumptions are stated in Assumption 11. We note that the formulation here is general and, in principle, allows nonuniform collocation points, which may be particularly advantageous when solution errors are localized. In the present work, however, all numerical experiments use uniformly spaced collocation points, and we do not investigate adaptive point placement in practice.

The neural network training is based on a sparse minimization problem with an ℓ_1 regularization term:

$$\min_{N,c,\omega} \hat{L}_{K_1,K_2}(u_{c,\omega}) + \alpha\|c\|_1 \tag{P_N^{emp}}$$

Unlike most existing approaches, we do not fix the network width N a priori; instead, it is treated as part of the optimization problem. However, under this formulation, the existence of a minimizer for the empirical problem (P_N^{emp}) is unclear. Conceptually, without a bound on the network width N, we might get a network solution of infinite width. Hence, we generalize the discrete network representation (1.3) to a continuous formulation

by introducing the integral neural network (Bach, 2017; Bengio et al., 2005; Pieper and Petrosyan, 2022; Rosset et al., 2007), where u is defined by

$$u_\mu(x) = \int \varphi(x;\omega)\,\mathrm{d}\mu(\omega), \quad \mu \in M(\Omega).$$

Here $M(\Omega)$ is the space of signed Radon measures endowed with the total variation norm $\|\cdot\|_{M(\Omega)}$. We also note that $u_\mu(x)$ is defined pointwise as a continuous function for $s \geq d$ and an appropriate function space is introduced in the following. Associated to the infinite width networks, we consider the continuous optimization problems

$$\min_{\mu\in M(\Omega)} L(u_\mu) + \alpha\|\mu\|_{M(\Omega)} \tag{P_μ}$$

$$\min_{\mu\in M(\Omega)} \hat{L}_{K_1,K_2}(u_\mu) + \alpha\|\mu\|_{M(\Omega)} \tag{P_μ^{emp}}$$

We note the above characterization degrades to the prior discrete case when μ is a finite linear combination of Dirac-deltas.

We will establish the existence of minimizers for both (P_μ) and (P_μ^{emp}). More significantly, we will show that (P_μ^{emp}) possesses a finite solution, reducing the problem to the discrete formulation (P_N^{emp}). Nevertheless, solving (P_N^{emp}) remains nontrivial since N cannot be optimized through gradient-based methods. We then employ an adaptive search of neurons, where neurons are dynamically added and removed to maintain a compact network structure while minimizing the regularized loss, which we outline below in Section 1.2.5.

We note that functions represented by integral neural networks form a Banach space

$$\mathcal{V}(D) = \left\{ \int_\Omega \varphi(x;\omega)\,\mathrm{d}\mu(\omega) \;\middle|\; \mu \in M(\Omega) \right\}$$

equipped with norm

$$\|u\|_{\mathcal{V}(D)} = \inf\left\{ \|\mu\|_{M(\Omega)} \;\middle|\; \mu \in M(\Omega) \colon \int_\Omega \varphi(\cdot;\omega)\,\mathrm{d}\mu(\omega) = u \text{ on } D \right\}.$$

This is a *reproducing kernel Banach space* (RKBS) by definition in Bartolucci et al. (2023), and $\varphi : D \to C_0(\Omega)$ is the associated feature map. Equivalently, (P_μ) and (P_μ^{emp}) can be reformulated as

$$\min_{u\in\mathcal{V}(D)} L(u) + \alpha\|u\|_{\mathcal{V}} \quad \text{and} \quad \min_{u\in\mathcal{V}(D)} \hat{L}_{K_1,K_2}(u) + \alpha\|u\|_{\mathcal{V}}. \tag{1.6}$$

These formulations identify the natural function space for shallow neural networks, where regularization is imposed through the Banach norm.

The space $\mathcal{V}$ is also referred to as an integral RKBS in the literature (Spek et al., 2025), and is closely related to variation spaces and Barron spaces (E et al., 2022; Parhi and Nowak, 2021; Parhi and Unser, 2025; Siegel and Xu, 2023). For the Gaussian feature function (2.3) with scaling s, the associated RKBS is closely related to the classical Besov space $B^s_{1,1}$; see Section 2.1. A precise relationship between the RKBS, the variation space, and the Besov space for general radial basis kernels is established in Shao et al. (2026). For related studies

on RKBS-based learning, we refer the readers to Kumar et al. (2024); Parhi and Nowak (2022); Parhi and Unser (2025) and the references therein.

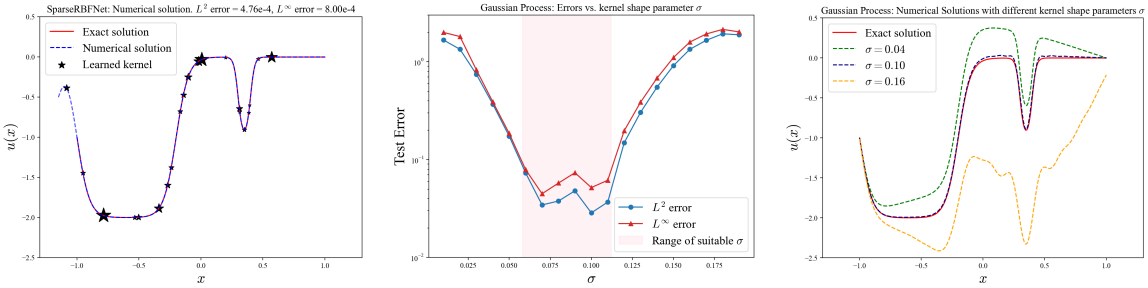
Based on the empirical problem (P_N^{emp}), we propose an efficient numerical framework for solving nonlinear PDEs in the RKBS. The main contributions of our work are summarized as follows.

- *An adaptive extension of kernel collocation methods*: The proposed scheme can be interpreted as an adaptive collocation solver, extending classical RBF-based methods while resolving key issues such as scale selection and large system ill-conditioning.
- *A framework integrating ideas from PINNs and GP methods*: Our approach can be viewed as a PINN-type training scheme using a shallow, sparse neural network for improved optimization and an extension of the GP mechanism with adaptive kernel parameter selection.
- *A representer theorem*: We prove a representer theorem showing that the solution to the continuous empirical problem (P_μ^{emp}) admits a finite linear combination of features; as a result, the problem is practically reduced to the discrete empirical problem (P_N^{emp}).
- *Provable convergence guarantees*: We provide a rigorous theoretical analysis of the convergence of the solutions of the continuous empirical problem (P_μ^{emp}) to the solution of the PDE in the limit $K \to \infty$ and $\alpha \to 0$.
- *An efficient computational framework*: We design a three-phase algorithm that maintains a compact network structure through adaptive neuron insertion (guided by dual variables), efficient optimization via a second-order Gauss-Newton method, and pruning of inactive neurons.
- *Demonstrated numerical advantages*: Through numerical examples and supporting discussions, we illustrate the advantages of our method compared to GP and PINN approaches.
- *Broad applicability*: Although developed in the context of RBF networks, the proposed framework is compatible with general shallow neural networks and other activation functions.

To illustrate the effectiveness of our approach, we present a simple 1D example comparing our results with the results obtained with the GP method from Chen et al. (2021) (using the code in the associated repository; Chen, 2024). Figure 1a demonstrates that our method accurately recovers the solution using a small number of adaptively chosen kernels. Here, the stars indicate the RBF center and shape parameter, with larger stars corresponding to larger RBF bandwidths. In contrast, Figures 1b and 1c show that the GP method is highly sensitive to the kernel shape parameter. For this example, accurate recovery using GP is only possible within a very narrow range of σ and deviations lead to significant errors.

1.2 Related studies

Let us briefly mention some related methods and their conceptual connection to the proposed approach.



(a) (b) (c)

Figure 1: Numerical experiments on a 1D semilinear Poisson equation with Sparse RBFNet (our) and Gaussian Process. The exact solution u is set to be $u(x) = \tanh(10(x + 0.2)) - \tanh(10(x + 1.0)) + 0.5\,(\tanh(30(x - 0.4)) - \tanh(30(x - 0.3)))$. $K = 100$ grid collocation points are used for both methods; L^2 and L^∞ error shown above are estimated at test grid of size 200. (a) Sparse RBFNet solution with ℓ^1-regularization parameter $\alpha = 10^{-4}$, penalty weight $\lambda = 100$; (b) Error curves for GP solutions with different kernel shape parameters; (c) GP solutions with different kernel shape parameters.

1.2.1 The Hilbert space setting

Given a probability measure ρ over Ω, one can define a space

$$\mathcal{H}(D) = \left\{ \int_\Omega \varphi(x;\omega) c(\omega)\, \mathrm{d}\rho(\omega) \mid c \in L^2(\Omega, \mathrm{d}\rho) \right\}$$

equipped with norm

$$\|u\|_{\mathcal{H}} = \inf \left\{ \|c\|_{L^2(\Omega,\mathrm{d}\rho)} \mid c \in L^2(\Omega, \mathrm{d}\rho) \colon \int_\Omega \varphi(\cdot;\omega) c(\omega)\, \mathrm{d}\rho(\omega) = u \text{ on } D \right\}.$$

Given $u \in \mathcal{H}$, the above infimum is attained by a unique minimizer $c_u \in L^2(\Omega, \mathrm{d}\rho)$ using the direct method in the calculus of variations. $\mathcal{H}$ is a Hilbert space equipped with inner product

$$\langle u, v \rangle_{\mathcal{H}} := \langle c_u, c_v \rangle_{L^2(\Omega,\mathrm{d}\rho)}.$$

We also observe that $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) associated with the kernel (Bach, 2017)

$$k(x, x') = \int_\Omega \varphi(x;\omega)\varphi(x';\omega)\, \mathrm{d}\rho(\omega). \tag{1.7}$$

This can be verified easily as follows. For a fixed $x' \in D$, we have $k(\cdot, x') \in \mathcal{H}$. If $\varphi(x', \cdot) \in L^2(\Omega, \mathrm{d}\rho)$ achieves $\|k(\cdot, x')\|_{\mathcal{H}} = \|\varphi(x', \cdot)\|_{L^2(\Omega,\mathrm{d}\rho)}$, then it follows that

$$\langle u, k(\cdot, x) \rangle_{\mathcal{H}} = \langle c_u, \varphi(x, \cdot) \rangle_{L^2(\Omega,d\rho)} = \int_\Omega c_u(\omega)\varphi(x;\omega)\, \mathrm{d}\rho(\omega) = u(x),$$

which verifies the reproducing property of the kernel. In this Hilbert space framework, the minimization problem

$$\min_{c \in L^2(\Omega,\mathrm{d}\rho)} \hat{L}_{K_1,K_2}(u_{c\,\mathrm{d}\rho}) + \frac{\alpha}{2}\|c\|^2_{L^2(\Omega,\mathrm{d}\rho)} \tag{1.8}$$

is equivalent to the minimization problem in the RKHS space:

$$\min_{u\in\mathcal{H}} \hat{L}_{K_1,K_2}(u) + \frac{\alpha}{2}\|u\|_{\mathcal{H}}^2. \tag{1.9}$$

To compare with our approach, we observe that

$$\left(\int_\Omega |c(\omega)|\,\mathrm{d}\rho(w)\right)^2 \le \int_\Omega |c(\omega)|^2\,\mathrm{d}\rho(w),$$

which naturally implies that $\mathcal{H} \hookrightarrow \mathcal{V}$. This embedding is in fact strict, and Spek et al. (2025) show that $\mathcal{V}$ coincides with the union of all such RKHS as ρ ranges over the set of probability measures.

1.2.2 Random feature models

Random feature models offer a simpler alternative to neural network training. Instead of being trained, the internal weights in these models are sampled from a given probability distribution ρ, typically uniform or Gaussian, defined over the parameter space Ω. In a random feature model, one seeks a function expressed as

$$u(x) = \sum_{n=1}^{N} c_n \varphi(x;\omega_n), \quad \{\omega_n\} \sim \rho.$$

Here, the inner weights $\{\omega_n\}$ are sampled from ρ, and only the outer weights $\{c_n\}$ are solved. Consequently, Equation (1.8) becomes

$$\min_{\{c_n\}_{n=1}^N} \hat{L}_{K_1,K_2}(u_{\{c_n,w_n\}}) + \frac{\alpha N}{2}\sum_{n=1}^{N}|c_n|^2.$$

This essentially corresponds to applying the kernel ridge regression method to solve a PDE, where the kernel is given by

$$\hat{k}(x,x') = \frac{1}{N}\sum_{n=1}^{N}\varphi(x;\omega_n)\varphi(x';\omega_n)$$

that approximates k in Equation (1.7). For a linear PDE problem, this leads to solving a linear system, which avoids the neural network training cost. However, the random feature model does not adapt to the solution landscape and may require a large number of features. This can lead to prohibitive computational costs, especially in high-dimensional settings. For related discussions, see Rahimi and Recht (2008); Zhang et al. (2024); Bach (2023); Pieper et al. (2024).

1.2.3 The Gaussian process method

There has been a recent surge in using the Gaussian process (GP) method for solving PDEs (Batlle et al., 2025; Chen et al., 2021, 2025). The GP method proposed by Chen et al. (2021) employs a Gaussian kernel with a fixed shape parameter and randomly chosen

collocation points. Solving (1.9) in the limiting case $\alpha \to 0$ yields a variant of this GP method with the covariance kernel given by (1.7), where the solution can be interpreted as a maximum a posteriori estimate under a GP prior, as shown in Chen et al. (2021).

However, the kernel (1.7) is only indirectly related to the feature function, and also depends on the density $\mathrm{d}\rho$, we cannot directly compare the kernel used for the construction of the feature function and the covariance kernel resulting from this method. However, if we consider a density $\mathrm{d}\rho$ that is concentrated on a single RBF shape parameter, but otherwise uniform over the kernel centers, we have a direct relation: Consider the case where $\omega = y \in B_R \subset \mathbb{R}^d$ is in a ball of large radius $R \gg 0$, $\sigma > 0$ is fixed, and ρ is a uniform probability measure on B_R. Then Equation (1.7) becomes

$$\begin{aligned} k(x, x') &= \frac{\sigma^{2s}}{|B_R|} \int_{B_R} \sigma^{-2d} \hat{\varphi}((x-y)/\sigma)\hat{\varphi}((x'-y)/\sigma)\,\mathrm{d}y \\ &\approx \frac{\sigma^{2s}}{|B_R|} \int_{\mathbb{R}^d} \sigma^{-2d} \hat{\varphi}((x-y)/\sigma)\hat{\varphi}((x'-y)/\sigma)\,\mathrm{d}y \\ &= \frac{\sigma^{2s}}{(\sqrt{2}\sigma)^d |B_R|} \hat{\varphi}\left(\frac{x-x'}{\sqrt{2}\sigma}\right) \end{aligned}$$

where $\hat{\varphi}$ denotes the standard Gaussian function in $\mathbb{R}^d$. This yields a Gaussian kernel with fixed scale parameter $\sqrt{2}\sigma$.

1.2.4 PINNs and other machine learning method for PDEs

Our method is closely related to PINNs (Raissi et al., 2019; Shin et al., 2023; Luo and Yang, 2024; Bonito et al., 2025), which also solve PDEs by minimizing a residual loss over a neural network ansatz. While these works provide convergence results for linear PDEs, they typically rely on large, fixed size neural networks without incorporating sparsity or adaptivity, which are central to our approach. Adaptive approaches are developed by Cai et al. (2022); Liu and Cai (2022) for linear PDEs using residual-based neuron enhancement, while our work differs by focusing on nonlinear PDEs and promoting sparsity through RKBS-based regularization. Radial basis function network based approaches related to PINNs have been developed by Ramabathiran and Ramachandran (2021); Wang et al. (2023); Bai et al. (2023). In particular, similar to our framework, Wang et al. (2023) propose a shallow RBF network trained with a sparsity-promoting ℓ_1 regularized residual loss to obtain compact representation and better interpretability. However, a scaling of the feature functions in (1.3) that allows for stable computation of second derivatives for arbitrarily small length scale $\sigma > 0$ has not been employed. This is a prerequisite for the analysis in terms of RKBS provided in our work, which enables theoretical guarantees in terms of generalization (error estimates), and adaptive training methods that do not require a large initial choice of the network width.

A key difference of our method compared to other approaches using (regularized) RBF networks is the optimization strategy. Instead of updating all parameters using gradient-based method, we employ a greedy, boosting-like approach that adaptively selects neurons while maintaining a simple network structure. To further enhance convergence and promote sparsity, we use a second-order optimization scheme. This helps avoid redundant neurons, which are often seen in first-order methods. Although our theory is developed for Gaussian

RBF, the approach still applies when we replace RBF with other activation functions commonly used in PINNs. Other related approaches, such as variational or weak formulation of loss functions (Yu et al., 2018; Zang et al., 2020), may be explored in the future.

It has been widely observed that the performance of PINNs is highly sensitive to the choice of penalty parameter λ for enforcing boundary conditions (Wang et al., 2021, 2022). As our method employs a similar loss function, it inherits this sensitivity as well. This motivates alternative treatment of boundary conditions, which will be discussed later in Section 3.5 and 4.1.3. Related to this, a linear elliptic problem is studied by Bonito et al. (2025) for a class of collocation methods including PINNs formulations. Using the theory of optimal recovery and regularity theory in Besov scales, an improved PINNs loss is provided that allows for better theoretical guarantees when minimizing the loss over a (neural network) model class that approximates functions in Besov spaces. In our work, we directly regularize in a variation norm, which is related to a Besov norm, see Section 2.1, and we make similar observations on the practical and theoretical deficiency of the standard PINNs loss.

1.2.5 Adaptive methods and greedy algorithms

Our method is closely related to a growing class of approaches that revisit shallow architectures through adaptive basis construction, including boosting-type procedures (Friedman, 2001; Bengio et al., 2005), Frank–Wolfe variants (Bach, 2017; Bredies et al., 2024), and greedy algorithms (Needell and Tropp, 2009; Klusowski and Barron, 2018; Siegel et al., 2023). In these approaches, the network is constructed incrementally by repeatedly solving a nonconvex subproblem over the inner weight set to select and insert new units. Our method distinguishes itself in several important aspects. First, in contrast to pure greedy type methods we include an ℓ_1 sparsity-promoting regularization, which allows for pruning of unneeded features, promotes compact representations and mitigates overfitting; see Section 4. In general, for convex loss functions, classical greedy or Franke–Wolfe style methods converge at a slow sub-linear rate, where the optimization error is proportional to one over the number of insertions. Better resolution of the convex subproblems arising in those strategies results in provably linear convergence (Flinth et al., 2021; Pieper and Walter, 2021; Bredies et al., 2024). For this reason, we incorporate a semismooth Newton weight optimization strategy (Ulbrich, 2011; Ouyang and Milzarek, 2025) with inexact Gauss-Newton Hessian into the algorithm, enabling faster convergence once an appropriate parameter set has been identified. Second, we combine the inner and outer weights in a combined update, similar to over-parametrized gradient based training methods (Chizat and Bach, 2018; Chizat, 2022) to mitigate the weight clustering observed in accelerated Frank–Wolfe methods that do not move the inner weights. However, in contrast to those methods the induced sparsity throughout the iterations enables joint optimization of both inner and outer weights with inexact second order updates. Finally, rather than relying on an exact greedy selection step for the parameter insertion, which typically entails an expensive global optimization, we employ a relaxed randomized and computationally tractable strategy for updating the inner parameters. We outline the proposed algorithm in Section 3, but consider a detailed convergence analysis out of scope for the current work.

1.3 Notation, assumptions and examples

For the PDE problem defined by (1.1) and (1.2), we impose the following assumption throughout this paper.

Assumption 1 *Assume that the functions E and B satisfy the Carathéodory conditions:*

$$\begin{aligned}
(u, g, H) &\mapsto E(x, u, g, H) && \text{is continuous for almost all } x \in D,\\
(u, g) &\mapsto B(x, u, g) && \text{is continuous for almost all } x \in \partial D,\\
x &\mapsto E(x, u, g, H) && \text{is bounded and } \nu_D\text{-measurable for all } (u, g, H) \in \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{d^2},\\
x &\mapsto B(x, u, g) && \text{is bounded and } \nu_{\partial D}\text{-measurable for all } (u, g) \in \mathbb{R} \times \mathbb{R}^d.
\end{aligned}$$

The assumptions stated above are sufficient to ensure the existence of minimizers for the continuous optimization problems, which will be established in Section 2.1. We note, however, that these assumptions do not imply the existence of solutions to the PDE (1.1), which has to be argued for separately. To analyze the convergence of our method, we require the PDEs in consideration to be well-posed. To formalize this, let $\mathcal{U}_o$, $\mathcal{U}$, $\mathcal{F}_o$ $\mathcal{F}$ be Banach spaces in which smooth functions form a dense subset, with continuous embeddings $\mathcal{U}_o \hookrightarrow \mathcal{U}$ and $\mathcal{F}_o \hookrightarrow \mathcal{F}$. In addition, we assume $\mathcal{V} \hookrightarrow \mathcal{U}$. We define the combined PDE and boundary operator $\mathcal{R}[u] := (\mathcal{E}[u], \mathcal{B}[u])$ and regard $\mathcal{R}$ as a mapping from $\mathcal{U}$ to $\mathcal{F}$, with its restriction acting from $\mathcal{U}_o$ to $\mathcal{F}_o$:

$$\mathcal{R} : \mathcal{U} \to \mathcal{F}, \quad \mathcal{R}|_{\mathcal{U}_o} : \mathcal{U}_o \to \mathcal{F}_o.$$

We think of $\mathcal{U}_o$ and $\mathcal{U}$ as Sobolev spaces defined over the domain D where the solution is sought. Typically, $\mathcal{U}$ represents the minimal regularity required for a weak solution, such as $H^1(D)$ for second-order elliptic problems. The more regular space $\mathcal{U}_o$, such as $H^2(D)$, may result from classical regularity theory under more regular data or boundary conditions. The spaces $\mathcal{F}_o$ and $\mathcal{F}$ are given as direct sums of spaces on D and ∂D. For example, $\mathcal{F}$ may equal $H^{-1}(D) \times H^{1/2}(\partial D)$, combining the duals of $H^1(D)$ and its trace space. Similarly, $\mathcal{F}_o$ may be a more regular variant, such as $L^2(D) \times L^2(\partial D)$, into which $\mathcal{R}_{\mathcal{U}_o}$ maps. We now reformulate (1.1) as the problem of finding $u \in \mathcal{U}_o$ such that

$$\mathcal{R}[u] = 0 \quad \text{in } \mathcal{F}_o. \tag{1.10}$$

We assume $\mathcal{F}_o$ is chosen so that the zero element corresponds to functions that vanish almost everywhere on D. We state the assumptions on well-posedness as follows.

Assumption 2 (Well-posedness of the PDE) *We assume the following:*

1. *Existence and uniqueness. There exists a unique solution $u \in \mathcal{U}_o$ to the problem* (1.10)*;*
2. *Continuity and boundedness. The maps $\mathcal{R}$ and $\mathcal{R}_{\mathcal{U}_o}$ are continuous around u and satisfy a growth bound; that is, there exists $C > 0$ such that*

$$\begin{aligned}
&\|\mathcal{R}[v]\|_{\mathcal{F}} = \|\mathcal{R}[u] - \mathcal{R}[v]\|_{\mathcal{F}} \le C\|u - v\|_{\mathcal{U}} \text{ and } \|\mathcal{R}[v]\|_{\mathcal{F}} \le C \max(1, \|v\|_{\mathcal{U}}) \quad \forall v \in \mathcal{U}\\
&\|\mathcal{R}[v]\|_{\mathcal{F}_o} = \|\mathcal{R}[u] - \mathcal{R}[v]\|_{\mathcal{F}_o} \le C\|u - v\|_{\mathcal{U}_o} \text{ and } \|\mathcal{R}[v]\|_{\mathcal{F}_o} \le C \max(1, \|v\|_{\mathcal{U}_o}) \quad \forall v \in \mathcal{U}_o.
\end{aligned}$$

3. Stability of the solution. There exists $C > 0$ such that

$$\|u - v\|_{\mathcal{U}} \le C\|\mathcal{R}[u] - \mathcal{R}[v]\|_{\mathcal{F}} = C\|\mathcal{R}[v]\|_{\mathcal{F}} \quad \forall v \in \mathcal{U}.$$

Such well-posedness properties are well established in the classical theory for several important classes of PDEs; see, for example, Brezis (2011); Caffarelli and Cabré (1995); Dafermos (2005); Evans (2022); Gilbarg and Trudinger (1977). We note that $\mathcal{U}_o$ and $\mathcal{U}$ are Banach spaces in which smooth functions are dense. Together with the characterization of $\mathcal{V}$ in Appendix C (see also Shao et al. (2026)), this implies that $\mathcal{V}$ is dense in both $\mathcal{U}_o$ and $\mathcal{U}$ with respect to their respective norms. Theorem 2 together with the density property allows our method to approximate solutions to (1.1). The corresponding analysis is presented in Section 2.3.

Since many PDEs only depend on a subset of linear partial differential operators, we also introduce two linear operators

$$\begin{aligned}[\mathcal{L}_E u](x) &= [c_i(x)u(x) + b_i(x) \cdot \nabla u(x) + \operatorname{tr}[A_i(x)\nabla^2 u(x)]_{i=1,\dots,N_E} \\ [\mathcal{L}_B u](x) &= [d_i(x)u(x) + e_i(x) \cdot \nabla u(x)]_{i=1,\dots,N_B}\end{aligned} \tag{1.11}$$

with the property that

$$\mathcal{E}[u](x) = \hat{E}(x, [\mathcal{L}_E u](x)), \quad \mathcal{B}[u](x) = \hat{B}(x, [\mathcal{L}_B u](x)) \tag{1.12}$$

for some nonlinear functions $\hat{E}\colon \overline{D} \times \mathbb{R}^{N_E} \to \mathbb{R}$ and $\hat{B}\colon \partial D \times \mathbb{R}^{N_B} \to \mathbb{R}$. This decomposition can be used to simplify the notation, improve the theoretical estimates below, and speed up some algorithms. However, in general, the linear operators could be set to simply evaluate all partial derivatives together with $\hat{E} = E$ and $\hat{B} = B$. To formulate the first order optimality conditions of the optimization problems in Section 2.5, we impose the following assumptions.

Assumption 3 *Let Theorem 1 hold for the residual functions $\hat{E}$ and $\hat{B}$ and let the coefficient functions $\{c_i(x), b_i(x), A_i(x)\}_{i=1,\dots,N_E}, \{d_i(x), e_i(x)\}_{i=1,\dots,N_B}$ be measurable and uniformly bounded. In addition, the functions $(x, l_E) \mapsto \hat{E}(x, l_E)$ and $(x, l_B) \mapsto \hat{B}(x, l_B)$ are differentiable in their second arguments, with gradients uniformly bounded in $x \in \overline{D}$ and $x \in \partial D$, respectively.*

Example Using the notation introduced above, the semilinear Poisson equation with Dirichlet boundary conditions

$$-\Delta u + u^3 = f_{\mathcal{E}} \quad \text{in } D, \qquad u = f_{\mathcal{B}} \quad \text{on } \partial D.$$

can be expressed as

$$\mathcal{E}[u] = -\Delta u + u^3 - f_{\mathcal{E}}, \quad \mathcal{B}[u] = u - f_{\mathcal{B}}, \tag{1.13}$$

where we define $E(x, u, g, H) = -\operatorname{tr} H + u^3 - f_{\mathcal{E}}(x)$ and $B(x, u, g) = u - f_{\mathcal{B}}(x)$. For the second description, we define the linear operators

$$\begin{aligned}\mathcal{L}_E[u] &= [u, \Delta u], \\ \mathcal{L}_B[u] &= [u],\end{aligned}$$

together with $\hat{E}(x, l_1, l_2) = -l_2 + l_1^3 - f_{\mathcal{E}}(x)$ and $\hat{B}(x, l_1) = l_1 - f_{\mathcal{B}}(x)$.

1.3.1 Notation summary

Throughout the paper, $\mathbb{R}$ denotes the set of real numbers, $\mathbb{R}_+$ the set of positive real numbers, $\mathbb{N}$ the set of positive integers, and $\mathbb{N}_0$ the set of non-negative integers. For $m \in \mathbb{N}$, $\mathbb{R}^m$ denotes the m-dimensional Euclidean space. Ω denotes the parameter domain for the integral neural network, while $D \subset \mathbb{R}^d$ is a bounded open domain where the PDE is defined.

We summarize the main notation below.

- Function spaces on a set $A \subset \mathbb{R}^m$
  - $C(A)$: Continuous functions on A.
  - $C(\overline{A})$: Continuous functions on the closure $\overline{A}$, with the supremum norm.
  - $C_0(A)$: Functions in $C(\overline{A})$ that vanish on $\overline{A} \setminus A$.
  - $C^k(\overline{A})$: Functions on $\overline{A}$ with continuous derivatives up to order k.
  - $C^{k,\gamma}(A)$: Hölder space of functions with k continuous derivatives whose kth derivatives are Hölder continuous with exponent $\gamma \in (0, 1]$.
  - $H^s(A) = W^{s,2}(A)$: Sobolev space with smoothness index s and integrability index 2.
  - $B^s_{p,q}(A)$: Besov space with smoothness index s, integrability p, and summability q.
- $M(\Omega)$: Space of signed Radon measures on Ω, equipped with the total variation norm. Unless otherwise stated, $M(\Omega)$ is endowed with the weak-$*$ topology induced by its duality with $C_0(\Omega)$.
- $\mathcal{N}\mu(x) = u_\mu(x)$: Integral neural network defined by $u_\mu(x) = \int_\Omega \varphi(x;\omega)\, d\mu(\omega)$, where $\varphi(x;\omega)$ is a Gaussian RBF with parameter $\omega \in \Omega$.
- For two norms spaces X and Y, $X \hookrightarrow Y$ denote a continuous embedding of X into Y, i.e., $\|x\|_Y \leq C\|x\|_X$ for all $x \in X$.
- $\mathcal{F}_o \hookrightarrow \mathcal{F}, \mathcal{U}_o \hookrightarrow \mathcal{U}$: Banach spaces of admissible input data and PDE solutions, respectively, each containing smooth functions as a dense subset.
- $\mathcal{R}$: The combined PDE and boundary operator $(\mathcal{E}, \mathcal{B}) : \mathcal{U} \to \mathcal{F}$, with restriction $\mathcal{R}|_{\mathcal{U}_o} : \mathcal{U}_o \to \mathcal{F}_o$.
- $\mathcal{V}(D)$: Reproducing Kernel Banach Space (RKBS) of functions represented as $\mathcal{N}\mu$ with $\mu \in M(\Omega)$. $\mathcal{V}(D)$ is a dense subset of $\mathcal{U}$ and is dense in $\mathcal{U}_o$.
- ∂^β: Multi-index partial derivative with $\beta = (\beta_1, \ldots, \beta_d) \in \mathbb{N}_0^d$.

We also introduce some terminology used throughout the paper. By analogy with collocation methods, we refer to the set of quadrature points $\{x_{1,k}\} \cup \{x_{2,k}\}$, which define the empirical loss, as *collocation points*, and denote their total number by K. Note, however, that the PDE is not enforced to be exactly satisfied at these points. The functions $\{\varphi(x;\omega_n)\}_{n=1}^N$ are called *feature functions*, and the associated parameters $\{\omega_n\}_{n=1}^N$ are referred to as *inner weights* or *kernel nodes*, where N is the total number of feature functions.

1.4 Outline of the paper

The rest of the paper is organized as follows. Section 2 develops the theoretical foundation of our approach. We introduce the integral neural network representation and its associated function space, and establish an existence result for the sparse minimization problem. We also provide a convergence analysis for the method applied to solving PDEs, derive a representer theorem ensuring finite representation, and present the optimality conditions and dual variables that underlie the design of our optimization algorithm. Section 3 turns to the algorithmic framework. We introduce a three-phase algorithm, which includes a gradient boosting strategy for inserting kernel nodes, a semi-smooth Gauss–Newton method for optimizing parameters, and a node deletion step to maintain a compact network size. Some additional implementation components are also discussed. Section 4 presents numerical experiments that validate the effectiveness of the method. Finally, Section 5 concludes with a discussion of future directions.

2. Theoretical framework

We begin with a brief overview of the main theoretical results in this section, together with high-level proof sketches.

- **Existence (Theorem 8, Section 2.1).** Under Theorem 1 and the continuity of the feature map established in Theorem 4 and Theorem 7, the continuous sparse minimization problem (P_μ) and its empirical counterpart (P_μ^{emp}) admit global minimizers. The proof follows the direct method in the calculus of variations, using weak-$*$ compactness in $M(\Omega)$ and weak-$*$ lower semicontinuity of the objective.

- **Convergence and error estimates (Theorem 15 and Theorem 18, Section 2.3).** Assuming the quadrature measures converge weakly and remain uniformly bounded (Theorem 11), and the data-space embedding/interpolation condition (Theorem 9), empirical minimizers $u_{K,\alpha}$ (as $K \to \infty$) admit limit points u_α that satisfy an explicit residual-based error estimate in $\mathcal{U}$. We further show how to choose K and α jointly to obtain an explicit approximation bound for $u_{K,\alpha}$ relative to the PDE solution.

- **Finite representation/Representer theorem (Theorem 23, Section 2.4).** For the empirical sparse problem (P_μ^{emp}), there exists an optimal solution supported on finitely many atoms. The argument reduces the nonlinear objective to a constrained minimum-norm problem in $M(\Omega)$ and then applies Carathéodory-type arguments to obtain a finite representation of solutions.

2.1 Integral neural networks and the associated function space

We begin by recalling and formalizing the definition of the integral neural network introduced in the introduction. Specifically, we consider functions represented in the form

$$u_\mu(x) = \mathcal{N}\mu(x) := \int_\Omega \varphi(x;\omega)\,\mathrm{d}\mu(\omega), \tag{2.1}$$

where φ is a Gaussian RBF with variable bandwidth σ

$$\varphi(x;\omega) = \frac{\sigma^s}{\left(\sqrt{2\pi}\sigma\right)^d} \exp\left(-\frac{\|x-y\|_2^2}{2\sigma^2}\right) \quad \text{where } \omega = (y,\sigma) \in \mathbb{R}^{d+1}. \tag{2.2}$$

Introduce the standard Gaussian function on $\mathbb{R}^d$

$$\hat{\varphi}(x') = \frac{1}{\left(\sqrt{2\pi}\right)^d} \exp\left(-\frac{\|x'\|_2^2}{2}\right).$$

Then the scaled Gaussian can be written as

$$\varphi(x;\omega) = \sigma^{s-d}\hat{\varphi}\left(\frac{x-y}{\sigma}\right). \tag{2.3}$$

To define the parameter space Ω, fix a maximum bandwidth $\sigma_{\max} > 0$ and let $D_1 := \overline{D + B_{\sigma_{\max}}(0)}$. We then set

$$\Omega = D_1 \times (0, \sigma_{\max}].$$

We now define the function space associated with the integral neural networks. Specifically, we consider the set of functions on D that can be written as $\mathcal{N}\mu$ for some signed Radon measure μ. This leads to a reproducing kernel Banach space (RKBS) (Bartolucci et al., 2023; Lin et al., 2022; Zhang et al., 2009), defined by

$$\mathcal{V}(D) = \{\mathcal{N}\mu : D \to \mathbb{R} \mid \mu \in M(\Omega)\} \tag{2.4}$$

and equipped with the norm

$$\|u\|_{\mathcal{V}(D)} = \inf\{\|\mu\|_{M(\Omega)} \mid \mu \in M(\Omega) \colon \mathcal{N}\mu = u \text{ on } D\}. \tag{2.5}$$

Here $M(\Omega)$ is the space of signed Radon measures on Ω equipped with the total variation norm

$$\|\mu\|_{M(\Omega)} = \sup\left\{\int_\Omega f \,\mathrm{d}\mu \,:\, f \in C_0(\Omega),\ \|f\|_\infty \le 1\right\}.$$

$M(\Omega)$ is also identified with the the dual space of $C_0(\Omega)$. Unless otherwise stated, $M(\Omega)$ is endowed with the weak-$*$ topology induced by its duality with $C_0(\Omega)$.

Proposition 4 *For $s \ge d+k+\gamma$, with $k \in \mathbb{N}\cup\{0\}$ and $\gamma \in (0,1]$, we have $\partial_x^\beta \varphi(x;\cdot) \in C_0(\Omega)$ uniformly for $x \in \overline{D}$ for and multi-index β with $|\beta| \le k$ and $\varphi(\cdot;\omega) \in C^{k,\gamma}(D_1)$ uniformly for $\omega \in \Omega$.*

Proof Equation (2.3) implies

$$\partial_x^\beta \varphi(x;\omega) = \sigma^{s-d-|\beta|}\partial_x^\beta \hat{\varphi}((x-y)/\sigma).$$

Under the assumptions stated above, with $|\beta| \le k$, this function can be shown to be uniformly Hölder continuous with index $\gamma = \min\{1, s-d-k\}$. Moreover, for $\omega = (y,\sigma)$ with $\sigma \to 0$, it converges to zero at the rate $\sigma^{s-d-|\beta|}$. ■

Remark 5 *1. Concerning the previous result, we note that for the integer cases of $s \in \{d, d+1, d+2, \ldots\}$ we cannot set $k = s - d$, due to the requirement $\gamma > 0$. Indeed, in the case $s = d$ we only obtain that $\varphi(x;\cdot) \in L^\infty(\Omega)$ uniformly for $x \in \overline{D}$ and $\varphi(\cdot;\omega) \in L^\infty(D_1)$ uniformly for $\omega \in \Omega$ but φ cannot be extended continuously to the boundary $\sigma = 0$.*

2. We emphasize that Theorem 4 is indispensable for subsequent analysis in this section, including the existence of the minimizers and associated error estimates. Hence, the scale-dependent weight σ^s with $s \geq d + k + \gamma$, in addition to the standard Gaussian normalization, is fundamental to the theoretical framework of our method.

Proposition 6 *For $s \geq d + k + \gamma$ with $k \in \mathbb{N} \cup \{0\}$ and $\gamma \in (0, 1]$ the linear neural network operator is continuous:*

$$\mathcal{N} \colon M(\Omega) \to C^{k,\gamma}(D_1).$$

In particular, for $s > d + 2$ the required derivatives exist and are bounded in terms of the measure

$$\|\mathcal{N}\mu\|_{C(\overline{D})} + \|\nabla_x \mathcal{N}\mu\|_{C(\overline{D})} + \|\nabla_x^2 \mathcal{N}\mu\|_{C(\overline{D})} \leq C\|\mu\|_{M(\Omega)}.$$

The above proposition implies the continuous embedding $\mathcal{V}(D) \hookrightarrow C^{k,\gamma}(D)$. Moreover, the RKBS $\mathcal{V}$ is closely related to a classical Besov space. In Appendix C, we establish the embedding $B^s_{1,1}(\mathbb{R}^d) \hookrightarrow \mathcal{V}(\mathbb{R}^d)$, where a detailed proof is also provided. In the context of wavelet analysis, Meyer (1992) gives a Besov space characterization of the Gaussian "hump algebra", which is closely related to the RKBS for $s = d$ considered here. In our subsequent work Shao et al. (2026), we establish a precise Besov space characterization for a broad class of integral RKBSs related to scaled RBF kernels including the Gaussian considered here. These refined characterizations together with the embedding properties of Besov spaces also show that the in the case $s = d + k$ for integer k, we still obtain continuously differentiable functions in the RKBS $\mathcal{V}$. However, for the purposes of generalization (in Section 2.3), we need to quantify the degree of continuity, and consider $\gamma > 0$.

We assume $s \geq d + 2 + \gamma$ for $\gamma \in (0, 1]$ for the rest of this paper. Therefore $\mathcal{V}(D) \hookrightarrow C^{2,\gamma}(\overline{D})$. By Theorem 4, we observe that if the sequence of measures μ_k converges to μ in the weak-$*$ sense, then for any $x \in \overline{D}$,

$$\partial^\beta(\mathcal{N}\mu_k)(x) = \int_\Omega \partial_x^\beta \varphi(x;\omega)\,\mathrm{d}\mu_k(\omega) \to \int_\Omega \partial_x^\beta \varphi(x;\omega)\,\mathrm{d}\mu(\omega) = \partial^\beta(\mathcal{N}\mu)(x), \tag{2.6}$$

for a multi-index β with $|\beta| \leq 2$. This result leads to the following lemma, which is useful in establishing the existence theory for the minimization problems (P_μ) and (P_μ^{emp}).

Lemma 7 *Assume $\mu_k \overset{*}{\rightharpoonup} \mu$ in $M(\Omega)$, then we have*

$$\begin{aligned} \mathcal{E}[\mathcal{N}\mu_k](x) &\to \mathcal{E}[\mathcal{N}\mu](x) \quad \forall x \in D, \\ \mathcal{B}[\mathcal{N}\mu_k](x) &\to \mathcal{B}[\mathcal{N}\mu](x) \quad \forall x \in \partial D. \end{aligned}$$

Proof The result follows directly from (2.6) and Theorem 1. ■

Theorem 8 (Existence of minimizers) *Suppose Theorem 1 holds. The sparse minimization problem (P_μ) admits at least one global minimizer. Similarly, (P_μ^{emp}) admits at least one global minimizer.*

Proof We use direct methods in the calculus of variations to establish the results. Specifically, we prove the existence of minimizers for (P_μ), with the result for (P_μ^{emp}) following by a similar argument. Denote $J(\mu) = L(u_\mu) + \alpha\|\mu\|_{M(\Omega)}$. Let $\mu^{(k)} \in M(\Omega)$ be a minimizing sequence, i.e., $J(\mu^{(k)}) \to \inf_{\mu\in M(\Omega)} J(\mu)$. Then since $\|\mu^{(k)}\|_{M(\Omega)}$ is uniformly bounded, there is a subsequence (without relabelling) that converges to $\bar{\mu} \in M(\Omega)$ in the weak-$*$ sense. Now we want to show J is weak-$*$ lower semicontinuous such that $\overline{\mu}$ is a minimizer of the problem. First of all, the lower semicontintuity of $\|\cdot\|_{M(\Omega)}$ under weak-$*$ convergence is obvious. Second, we observe that for any multi-index $\beta \in \mathbb{N}^d$ with $|\beta| \le 2$, $\partial_x^\beta \varphi(x;\cdot) \in C_0(\Omega)$ for a given $x \in \overline{D}$; see Proposition 4. Therefore, by Theorem 7 we have $\mathcal{E}[u_{\mu^{(k)}}](x) \to \mathcal{E}[u_{\bar{\mu}}](x)$ $\forall x \in D$, and $\mathcal{B}[u_{\mu^{(k)}}](x) \to \mathcal{B}[u_{\bar{\mu}}](x)$ $\forall x \in \partial D$. Finally, by Fatou's lemma

$$L(u_{\bar{\mu}}) \le \liminf_k \int_D \frac{1}{2}(\mathcal{E}[u_{\mu^{(k)}}](x))^2 d\nu_D + \frac{\lambda}{2}\int_{\partial D}(\mathcal{B}[u_{\mu^{(k)}}](x)))^2 d\nu_{\partial D} = \liminf_k L(u_{\mu^{(k)}}).$$

Therefore, J is weak-$*$ lower semicontinuous and minimizers exist. ∎

2.2 Functional analytical setup and assumptions

Recall the conditions in Theorem 2 on the well-posedness of the PDEs. We first present an example of a PDE for which these conditions are satisfied. Consider the semilinear Poisson equation in (1.13). For simplicity and without loss of generality, we consider the modified formulation

$$\mathcal{E}[u] = -\Delta u + \phi_M(u) - f_{\mathcal{E}}, \quad \mathcal{B}[u] = u - f_{\mathcal{B}},$$

where $\phi_M \in C^\infty(\mathbb{R})$ is monotone and equals u^3 if $|u^3| < M$ and $\phi_M(u) = \text{sign}(u)\cdot 2M$ if $|u^3| > 2M$. We assume $M > 0$ is a sufficiently large constant. This simplification is justified by the maximum principle, which ensures that any classical solution u to (1.13) satisfies the bound $\sup_D |u| \le 2\sup_D |\bar{u}|$ where $\bar{u}$ is a classical solution that solves the linear problem

$$-\Delta\bar{u} - f_{\mathcal{E}} = 0 \quad \text{in } D, \quad \bar{u} - f_{\mathcal{B}} = 0 \quad \text{on } \partial D.$$

A proof of this estimate can be found, for example, in (Taylor, 1996, Corollary 1.5). For the modified equation, under minimal regularity assumptions, we may take $\mathcal{U} = H_0^1(D) + f_{\mathcal{B}}$ and $\mathcal{F} = H^{-1}(D) \times H^{1/2}(\partial D)$. Indeed, it is not hard to verify that Theorem 2(2) holds for the pair $(\mathcal{U}, \mathcal{F})$ due to the Lipschitz continuity of ϕ_M, and Theorem 2(3) is satisfied by the monotonicity of the operator $-\Delta u + \phi_M(u)$. More precisely, by assuming $f_{\mathcal{E}} \in H^{-1}(D)$ and $f_{\mathcal{B}} \in H^{1/2}(\partial D)$, Theorem 2(2) follows from the estimates

$$\|-\Delta(u-v)\|_{H^{-1}(D)} = \sup_{w\in H_0^1(D)} \frac{\langle -\Delta(u-v), w\rangle}{\|w\|_{H^1}} \le C\|u-v\|_{H^1(D)}$$

$$\|\phi_M(u) - \phi_M(v)\|_{H^{-1}(D)} \le C\|\phi_M(u) - \phi_M(v)\|_{L^2(D)} \le \tilde{C}\|u-v\|_{L^2(D)}$$

along with the trace theorems for functions in $H^1(D)$. Theorem 2(3) is justified by

$$\begin{aligned}&\|-\Delta(u-v)+\phi_M(u)-\phi_M(v)\|_{H^{-1}(D)}\\ \geq&\frac{\langle -\Delta(u-v)+\phi_M(u)-\phi_M(v), u-v\rangle}{\|u-v\|_{H^1(D)}} \geq \frac{\|\nabla(u-v)\|^2_{L^2(D)}}{\|u-v\|_{H^1(D)}} \geq C\|u-v\|_{H^1(D)},\end{aligned}$$

where we have used monotonicity of ϕ_M and Poincaré inequality for $u-v \in H^1_0(D)$. Now the choice of $\mathcal{U}_o$ and $\mathcal{F}_o$ depends on the smoothness of the domain and the data. For example, the classical elliptic regularity for Lipschitz domains allows us to choose $\mathcal{U}_o = H^2(D) \cap \mathcal{U}$ and $\mathcal{F}_o = L^2(D) \times H^{3/2}(\partial D)$. Under sufficient additional smoothness assumptions on the domain D and the data $f_{\mathcal{E}}$ and $f_{\mathcal{B}}$, one may also take $\mathcal{U}_o = H^k(D) \cap \mathcal{U}$ with $\mathcal{F}_o = H^{k-2}(D) \times H^{k-1/2}(\partial D)$ for any integer $k \geq 2$. Recall the embedding $B^s_{1,1} \hookrightarrow \mathcal{V}$ established in Appendix C. By the embedding properties of Besov spaces (see, e.g., Triebel, 2006, Proposition 4.6), for any $k > s$,

$$H^k = B^k_{2,2} \hookrightarrow B^s_{1,1} \hookrightarrow \mathcal{V}.$$

In this case, the PDE solution $u \in \mathcal{U}_o$ also lies in the RKBS $\mathcal{V}$.

To reflect the definition of the loss function (1.4), we introduce the weighted product space $L^2_\lambda = L^2(D, \nu_D) \times L^2(\partial D, \nu_{\partial D})$ with the norm

$$\|e\|^2_{L^2_\lambda} := \frac{1}{2}\|e_E\|^2_{L^2(D,\nu_D)} + \frac{\lambda}{2}\|e_B\|^2_{L^2(\partial D,\nu_{\partial D})}$$

for any $e = (e_E, e_B) \in L^2(D, \nu_D) \times L^2(\partial D, \nu_{\partial D})$. We make the following assumptions on $\mathcal{F}$ and $\mathcal{F}_o$.

Assumption 9 *We assume that one of the following conditions holds:*

1. $\mathcal{F}_o \hookrightarrow L^2_\lambda \hookrightarrow \mathcal{F}$;
2. $\mathcal{F}_o \hookrightarrow \mathcal{F} \hookrightarrow L^2_\lambda$ *and* $\mathcal{F}$ *is an interpolation space in between* L^2_λ *and* $\mathcal{F}_o$, *i.e., there exists* $\theta \in (0,1)$ *and* $C > 0$ *such that*

$$\|e\|_{\mathcal{F}} \leq C\|e\|^\theta_{L^2_\lambda}\|e\|^{1-\theta}_{\mathcal{F}_o}.$$

Remark 10 *We note that the above assumption is made for notational simplicity. Ideally, the function spaces on D and ∂D should be treated separately, and either condition (1) or (2) may hold independently on each component. This is indeed the case for the earlier example with $\mathcal{F} = H^{-1}(D) \times H^{1/2}(\partial D)$.*

Moreover, in certain cases, it is also beneficial to modify the form of the loss function depending on the structure of the underlying PDE. For example, in Dirichlet boundary value problems, incorporating derivatives of the boundary term into the loss function can enhance performance. Such extensions are discussed in the subsequent sections. Related ideas on improved phisics-informed loss functions can be found in (Bonito et al., 2025; Ainsworth and Dong, 2025).

2.3 Convergence analysis for the neural network solution

In this subsection, we establish the convergence of our method in a general setting.

Denote $\nu_D^{K_1} := \sum_{k=1}^{K_1} w_{1,k}\delta_{x_{1,k}}$ and $\nu_{\partial D}^{K_2} := \sum_{k=1}^{K_2} w_{2,k}\delta_{x_{2,k}}$, which are nonnegative measures by construction. The following assumption guarantees that the numerical quadrature used to define the empirical loss function (1.5) provides a consistent approximation of the continuous loss (1.4).

Assumption 11 *We assume that the discrete measures $\nu_D^{K_1}$ and $\nu_{\partial D}^{K_2}$ converge (in the weak-$*$ sense) to ν_D and $\nu_{\partial D}$, respectively, i.e.,*

$$\int_D f \,\mathrm{d}\nu_D^{K_1} \to \int_D f \,\mathrm{d}\nu_D, \quad \int_{\partial D} g \,\mathrm{d}\nu_{\partial D}^{K_2} \to \int_{\partial D} g \,\mathrm{d}\nu_{\partial D}$$

as $K_1 \to \infty$ and $K_2 \to \infty$ for any $f \in C(\overline{D})$ and $g \in C(\partial D)$.

We also note that weak-$*$ convergence in the above implies uniform boundedness of the non-negative measures $\{\nu_D^{K_1}\}$ and $\{\nu_{\partial D}^{K_2}\}$ (by setting $f = 1$ and $g = 1$). Recall the definition of $\mathcal{L}_E$, $\mathcal{L}_B$, $\hat{E}$ and $\hat{B}$ in Equations (1.11) and (1.12). For the rest of this subsection, we further strengthen Assumptions 1 and 3 as follows.

Assumption 12 *Let Theorem 3 hold. In addition, the functions $\{A_i(x), b_i(x), c_i(x)\}_{i=1}^{N_E}$ and $\hat{E}(x, l_E)$ are uniformly γ-Hölder continuous on $\overline{D}$, and $\{d_i(x), e_i(x)\}_{i=1}^{N_B}$ and $\hat{B}(x, l_B)$ are uniformly γ-Hölder continuous on ∂D.*

Remark 13 *Assumption 12 simplifies the subsequent analysis by ensuring the γ-Hölder continuity of the PDE-residual. The arguments below remain valid under weaker assumptions for problems with discontinuous coefficients, such as piecewise continuity for the functions on subdomains. For simplicity of exposition, we omit such discussions.*

The strengthened continuity allows us to show the following useful result.

Lemma 14 *Let $\mathcal{E}[u]$ and $\mathcal{B}[u]$ be defined as in* (1.12)*. Under Theorem 12, if $\mu_k \overset{*}{\rightharpoonup} \mu$ in $M(\Omega)$ and $\|\mu_k\|_{M(\Omega)} \le M < \infty$, then $\mathcal{E}[\mathcal{N}\mu_k](x)$ and $\mathcal{B}[\mathcal{N}\mu_k](x)$ are uniformly γ-Hölder continuous and*

$$\begin{aligned} &\mathcal{E}[\mathcal{N}\mu_k](x) \to \mathcal{E}[\mathcal{N}\mu](x) \quad \textit{uniformly on } D;\\ &\mathcal{B}[\mathcal{N}\mu_k](x) \to \mathcal{B}[\mathcal{N}\mu](x) \quad \textit{uniformly on } \partial D. \end{aligned}$$

Proof First, Theorem 7 gives the pointwise convergence of $\mathcal{E}[\mathcal{N}\mu_k](x) \to \mathcal{E}[\mathcal{N}\mu](x)$ for $x \in D$, and $\mathcal{B}[\mathcal{N}\mu_k](x) \to \mathcal{B}[\mathcal{N}\mu](x)$ for $x \in \partial D$. Further, $\sup_k \|\mu_k\|_{M(\Omega)} \le M < \infty$ and Theorem 6 imply that $\sup_k \|\mathcal{N}\mu_k\|_{C^{2,\gamma}(\overline{D})} < \infty$. By the definition of $\mathcal{E}$ and $\mathcal{B}$ in (1.12), and Theorem 12, we know that $\mathcal{E}[\mathcal{N}\mu_k]$ and $\mathcal{B}[\mathcal{N}\mu_k]$ are uniformly bounded and equicontinuous on D and ∂D, respectively. In fact, a straightforward estimate shows that $\|\mathcal{E}[\mathcal{N}\mu_k]\|_{C^{0,\gamma}(\overline{D})} + \|\mathcal{B}[\mathcal{N}\mu_k]\|_{C^{0,\gamma}(\partial D)} \le C(1 + \|\mathcal{N}\mu_k\|_{C^{2+\gamma}})$. The uniform convergence then follows from the Arzelà–Ascoli theorem. ∎

By Theorem 2(1), we denote the unique solution to (1.1) by $u \in \mathcal{U}_o$. Let $u^* = \mathcal{N}\mu^* \in \mathcal{V} \cap \mathcal{U}_o$ be an approximation of u in $\mathcal{U}_o$. By the continuity of $\mathcal{R}|_{\mathcal{U}_o}$ in Theorem 2(2), it follows that

$$\|\mathcal{R}[u^*]\|_{\mathcal{F}_o} = \|\mathcal{R}[u] - \mathcal{R}[u^*]\|_{\mathcal{F}_o} \leq C_1 \|u - u^*\|_{\mathcal{U}_o}.$$

The right-hand side in the above can be made arbitrarily small by density of $\mathcal{V}$ in $\mathcal{U}_o$. In the following, we derive an error estimate for the neural network solution.

Theorem 15 *Let $u \in \mathcal{U}_o$ denote the unique solution to (1.1) and $u^* = \mathcal{N}\mu^* \in \mathcal{V} \cap \mathcal{U}_o$ be an approximate solution. For a given regularization parameter $\alpha > 0$ and collocation number $K = K_1 + K_2 > 0$, let $u_{K,\alpha} = \mathcal{N}\mu_{K,\alpha}$ be a global minimizer of the empirical problem (P_μ^{emp}). Let Assumptions 9, 11 and 12 hold and denote by*

$$\eta(\mu^*, \alpha) = \left(\|u - u^*\|_{\mathcal{U}_o}^2 + \alpha \|\mu^*\|_{M(\Omega)}\right)^{1/2}$$

the combined approximation and regularization error. As $K_1, K_2 \to \infty$, any limit point μ_α of $\mu_{K,\alpha}$ in $M(\Omega)$ defines a function $u_\alpha = \mathcal{N}\mu_\alpha$ that satisfies

$$\|u_\alpha - u\|_{\mathcal{U}} \leq C\eta(\mu^*, \alpha),$$

if Theorem 9(1) holds, or

$$\|u_\alpha - u\|_{\mathcal{U}} \leq C \max\left(\alpha^{-(1-\theta)} \eta(\mu^*, \alpha)^{2(1-\theta)+\theta}, \eta(\mu^*, \alpha)^\theta\right),$$

if Theorem 9(2) holds, where $C > 0$ is a generic constant independent of α, μ^*, u and u^*. Moreover, the convergence $u_{K,\alpha} \to u_\alpha$ (up to a subsequence) holds in $C^2(\overline{D})$.

Remark 16 *Theorem 15 yields the following implications.*

(1) The above estimate indicates that the approximation error arises from two sources. The first term reflects the error due to approximating the exact solution $u \in \mathcal{U}_o$ using elements from a norm ball in the space $\mathcal{V}$, while the second term characterizes the error due to the regularization.

(2) Using the above estimate, one can select the appropriate α such that u_α converges to the true solution u in $\mathcal{U}$. In the first case, by density of $\mathcal{V}$ in $\mathcal{U}_0$, one can select $u^* = u^\delta \in \mathcal{V} \cap \mathcal{U}_o$ such that $\|u - u^\delta\|_{\mathcal{U}_o}^2 = \delta$. Let $\alpha = \min(\delta/\|u^\delta\|_{\mathcal{V}}, \delta)$, we observe that $\alpha \to 0$ as $\delta \to 0$. Then, by applying the estimate from Theorem 15.

$$\|u_\alpha - u\|_{\mathcal{U}} \leq C(2\delta)^{1/2},$$

which implies $u_\alpha \to u$ in $\mathcal{U}$ as $\alpha \to 0$. The second case is more delicate. Without additional assumptions on the function spaces, selecting α that guarantees convergence is challenging. As a simple illustration, we consider a setting where the space $\mathcal{V}$ contains the solution $u \in \mathcal{U}_o$. This inclusion can be justified in certain cases, such as the example provided in Section 2.2. In this simple case, let $u = \mathcal{N}\mu^*$. Then

$$\|u_\alpha - u\|_{\mathcal{U}} \leq C\alpha^{\theta/2} \max\left(\|\mu^*\|_{M(\Omega)}^{(1-\theta)+\theta/2}, \|\mu^*\|_{M(\Omega)}^{\theta/2}\right),$$

which implies $u_\alpha \to u$ in $\mathcal{U}$ as $\alpha \to 0$.

(3) In the cases where $\mathcal{F}$ does not contain L^2_λ and $\mathcal{V} \cap \mathcal{U}_o$ is strictly smaller than $\mathcal{U}_o$, one possible remedy is to design an appropriate loss function such that the corresponding data space is continuously embedded in $\mathcal{F}$. Revisiting the example with $\mathcal{F} = H^{-1}(D) \times H^{1/2}(\partial D)$, it suffices to define the loss function as

$$L(u) = \frac{1}{2}\|\mathcal{E}[u]\|^2_{L^2(\nu_D)} + \frac{\lambda}{2}\|\mathcal{B}[u]\|^2_{H^1(\nu_{\partial D})},$$

with the corresponding empirical loss function. In this case, we can establish similar error estimates. In Section 4, we present numerical experiments comparing the effects of using an L^2 boundary loss versus an H^1 boundary loss.

Proof We now prove Theorem 15. Denote $\hat{L} = \hat{L}_{K_1,K_2}$ and $\hat{J}(\mu) = \hat{L}(\mathcal{N}\mu) + \alpha\|\mu\|_{M(\Omega)}$. Since $u_{K,\alpha} = \mathcal{N}\mu_{K,\alpha}$ is a global minimizer of (P^{emp}_μ), it satisfies

$$\alpha\|\mu_{K,\alpha}\|_{M(\Omega)} \le \hat{J}(\mu_{K,\alpha}) \le \hat{J}(\mu^*) = \hat{L}(u^*) + \alpha\|\mu^*\|_{M(\Omega)} \quad \text{for all } K > 0,$$

where $\hat{L}(u^*) = \hat{L}_{K_1,K_2}(u^*)$ is given by

$$\hat{L}(u^*) = \frac{1}{2}\int_D (\mathcal{E}[u^*](x))^2 \,\mathrm{d}\nu_D^{K_1} + \frac{\lambda}{2}\int_{\partial D} (\mathcal{B}[u^*](x))^2 \,\mathrm{d}\nu_{\partial D}^{K_2}.$$

This term converges to $L(u^*)$ as $K_1, K_2 \to \infty$, and is therefore uniformly bounded in K. This implies that the sequence $\{\mu_{K,\alpha}\}_K \subset M(\Omega)$ is bounded. By the Banach-Alaoglu theorem, there exists a subsequence (not relabeled) and a $\mu_\alpha \in M(\Omega)$ such that $\mu_{K,\alpha} \overset{*}{\rightharpoonup} \mu_\alpha$ as $K \to \infty$. By Theorem 14 $u_{K,\alpha}$ converges to $u_\alpha = \mathcal{N}\mu_\alpha$ in $C^2(\overline{D})$ as $K \to \infty$.

Now we show the estimate for u_α. Notice that

$$\begin{aligned}\hat{L}(u_{K,\alpha}) - L(u_\alpha) = &\int_D \left((\mathcal{E}[u_{K,\alpha}])^2 - (\mathcal{E}[u_\alpha])^2\right) \mathrm{d}\nu_D^{K_1} + \int_{\partial D} \left((\mathcal{B}[u_{K,\alpha}])^2 - (\mathcal{B}[u_\alpha])^2\right) \mathrm{d}\nu_{\partial D}^{K_2} \\ &+ \int_D (\mathcal{E}[u_\alpha])^2 \left(\mathrm{d}\nu_D^{K_1} - \mathrm{d}\nu_D\right) + \int_{\partial D} (\mathcal{B}[u_\alpha])^2 \left(\mathrm{d}\nu_{\partial D}^{K_2} - \mathrm{d}\nu_{\partial D}\right).\end{aligned}$$

The first two terms in the above converge to zero by the uniform convergence established in Theorem 14 and a uniform bound on the total variation norm of the empirical measures implied by weak-$*$ convergence from Assumption 11 and the last two terms converge to zero. Therefore $\lim_{K\to\infty} \hat{L}(u_{K,\alpha}) = L(u_\alpha)$. Finally, by the lower semicontinuity of the norm $\|\cdot\|_{M(\Omega)}$ under weak-$*$ convergence, we conclude

$$L(u_\alpha) \le J(u_\alpha) \le \liminf_{K\to\infty} \hat{J}(\mu_{K,\alpha}) \le \liminf_{K\to\infty} \hat{J}(\mu^*) = L(u^*) + \alpha\|\mu^*\|_{M(\Omega)}.$$

The estimate above gives us

$$\|\mathcal{R}[u_\alpha]\|^2_{L^2_\lambda} \le \|\mathcal{R}[u^*]\|^2_{L^2_\lambda} + \alpha\|\mu^*\|_{M(\Omega)}. \tag{2.7}$$

We now utilize Equation (2.7) to show the final result. First, we assume Theorem 9(1) holds. Then, combining this with with Theorem 2(3), we have

$$\|u_\alpha - u\|_{\mathcal{U}} \le C_2\|\mathcal{R}[u_\alpha]\|^2_{\mathcal{F}} \le C_2\|\mathcal{R}[u_\alpha]\|^2_{L^2_\lambda}.$$

On the other hand, we use Theorem 9(1) and Theorem 2(2) to obtain

$$\|\mathcal{R}[u^*]\|_{L^2_\lambda}^2 + \alpha\|\mu^*\|_{M(\Omega)} \le C\|\mathcal{R}[u^*]\|_{\mathcal{F}_0}^2 + \alpha\|\mu^*\|_{M(\Omega)} \le CC_1\|u-u^*\|_{\mathcal{U}_0}^2 + \alpha\|\mu^*\|_{M(\Omega)}.$$

Together, these inequalities yield the desired result. Now we assume Theorem 9(2) holds, then the right-hand side of (2.7) is estimated in the same way. For the left-hand side, we use the interpolation equality and Theorem 2(2),

$$\|\mathcal{R}[u_\alpha]\|_{\mathcal{F}}^{2/\theta} \le C\|\mathcal{R}[u_\alpha]\|_{L^2_\lambda}^2\|\mathcal{R}[u_\alpha]\|_{\mathcal{F}_o}^{2(1-\theta)/\theta} \le C\|\mathcal{R}[u_\alpha]\|_{L^2_\lambda}^2 \max(1, \|u_\alpha\|_{\mathcal{U}_o}^{2(1-\theta)/\theta})$$

Note that

$$\|u_\alpha\|_{\mathcal{U}_o} \le \|u_\alpha\|_{\mathcal{V}} = \|\mu_\alpha\|_{M(\Omega)} \le \frac{1}{\alpha}L(u^*) + \|\mu^*\|_{M(\Omega)} = \frac{1}{\alpha}\|\mathcal{R}[u^*]\|_{L^2_\lambda}^2 + \|\mu^*\|_{M(\Omega)}.$$

Together, we have

$$\begin{aligned}
\|u_\alpha - u\|_{\mathcal{U}}^{2/\theta} &\le \|\mathcal{R}[u_\alpha]\|_{\mathcal{F}}^{2/\theta} \\
&\le \max\left(\alpha^{-2(1-\theta)/\theta}\left(\|\mathcal{R}[u^*]\|_{L^2_\lambda}^2 + \alpha\|\mu^*\|_{M(\Omega)}\right)^{2(1-\theta)/\theta+1}, \|\mathcal{R}[u^*]\|_{L^2_\lambda}^2 + \alpha\|\mu^*\|_{M(\Omega)}\right) \\
&\le C\max\left(\alpha^{-2(1-\theta)/\theta}\left(\|u-u^*\|_{\mathcal{U}}^2 + \alpha\|\mu^*\|_{M(\Omega)}\right)^{2(1-\theta)/\theta+1}, \|u-u^*\|_{\mathcal{U}}^2 + \alpha\|\mu^*\|_{M(\Omega)}\right),
\end{aligned}$$

which leads to the desired result. ■

The above theorem describes the limiting behavior of the neural network solution as the number of collocation points K tends to infinity. In practice, however, we are also interested in the behavior of the solution for finite K. More importantly, given a regularization parameter α, we aim to identify a suitable range of K that prevents overfitting. For a neural network solution u_μ, we interpret generalization as the property that the true loss $L(u_\mu)$ is close to the empirical loss $\hat{L}(u_\mu)$ computed from a finite set of collocation points, particularly when the network is trained to achieve a small empirical loss. This can be analyzed as follows. We first define a discrete version of the weighted norm $\|\cdot\|_{L^2_\lambda}$:

$$\|e\|_{2,\lambda,K}^2 := \frac{1}{2}\sum_{k=1}^{K_1} w_{1,k}(e_E(x_{1,k}))^2 + \frac{\lambda}{2}\sum_{k=1}^{K_2} w_{2,k}(e_B(x_{2,k}))^2$$

Now, we leverage that the discrete measures $\nu_D^{K_1}$ and $\nu_{\partial D}^{K_2}$ associated to this norm converge according to Theorem 11. This convergence can be quantified for any residual associated to a trained approximate solution $u_{K,\alpha} = \mathcal{N}(\mu_{K,\alpha})$ by leveraging the norm bound and the improved continuity from Theorem 14.

Lemma 17 *Given $\delta, M > 0$, there exists $K = K(\delta, M) \in \mathbb{N}$, such that for any $v \in \mathcal{V}$ with $\|v\|_{\mathcal{V}} \le M$*

$$\left|\hat{L}(v) - L(v)\right| \le \delta$$

where $\hat{L} = \hat{L}_{K_1,K_2}$ and $K = K_1 + K_2$.

Proof We first claim that

$$\|\nu_D - \nu_D^{K_1}\|_{C^{0,\gamma}(\overline{D})^*} = \sup_{f:\|f\|_{C^{0,\gamma}(\overline{D})}\leq 1} \left| \int_D f \,\mathrm{d}\nu_D - \int_D f \,\mathrm{d}\nu_D^{K_1} \right| \to 0 \tag{2.8}$$

for $K_1 \to \infty$ and similarly for $\nu_{\partial D}$. Given this, define $e = \mathcal{R}[v]$, we have

$$\begin{aligned} \left| \int_D e^2(x) \,\mathrm{d}\nu_D - \int_D e^2(x) \,\mathrm{d}\nu_D^{K_1} \right| &\leq \|e^2\|_{C^{0,\gamma}(\overline{D})} \|\nu_D - \nu_D^{K_1}\|_{C^{0,\gamma}(\overline{D})^*} \\ &\leq C\|e\|^2_{C^{0,\gamma}(\overline{D})} \|\nu_D - \nu_D^{K_1}\|_{C^{0,\gamma}(\overline{D})^*}. \end{aligned}$$

and the same holds for $\nu_{\partial D}$. The above estimate together with the fact that $\|e\|_{C^{0,\gamma}} = \|\mathcal{R}[v]\|_{C^{0,\gamma}} \leq C\|v\|_{\mathcal{V}}$ imply the desired result.

To prove the convergence (2.8), we argue by contradiction. If it fails, then there exists $\epsilon > 0$ and a sequence f_{K_1} in the unit ball of $C^{0,\gamma}$ for all K_1 such that

$$\left| \int_D f_{K_1} \,\mathrm{d}\nu_D - \int_D f_{K_1} \,\mathrm{d}\nu_D^{K_1} \right| \geq \epsilon > 0$$

for all K_1. By Arzelà–Ascoli, up to a subsequence, $f_{K_1} \to \bar{f}$ uniformly on $\overline{D}$. We now test the convergence of $\nu_D^{K_1}$ to ν_D against $\bar{f}$:

$$\int_D \bar{f} \,\mathrm{d}\nu_D^{K_1} - \int_D \bar{f} \,\mathrm{d}\nu_D = \int_D f_{K_1} \,\mathrm{d}\nu_D^{K_1} + \int_D \bar{f} - f_{K_1} \,\mathrm{d}\nu_D^{K_1} - \int_D f_{K_1} \,\mathrm{d}\nu_D - \int_D \bar{f} - f_{K_1} \,\mathrm{d}\nu_D.$$

The second and fourth term converge to zero, but the remaining two terms are always ϵ apart, and thus $\nu_D^{K_1}$ does not converge to ν_D in the weak-$*$ sense, which contradicts Theorem 11. ■

Theorem 18 *Let u and $u_{K,\alpha}$ denote the functions given in Theorem 15. For any $\delta > 0$, there exist $u^\delta \in \mathcal{V}$ and parameters $\alpha = \alpha(\delta) \to 0$ and $K = K(\delta) \to \infty$ as $\delta \to 0$ such that*

$$\|u_{K,\alpha} - u\|_{\mathcal{U}} \leq C\delta^{1/2},$$

if Theorem 9(1) holds, or

$$\|u_{K,\alpha} - u\|_{\mathcal{U}} \leq C\delta^{\theta/2} \max(1, \|u^\delta\|_{\mathcal{V}}^{1-\theta}),$$

if Theorem 9(2) holds, where $C > 0$ is a generic constant.

Remark 19 *As noted in Theorem 16, without further assumptions on the function spaces, the second case guarantees convergence if we know that $\mathcal{V}$ contains the exact solution $u \in \mathcal{U}_o$, which ensures that the $\|u^\delta\|_{\mathcal{V}}$ remains uniformly bounded. Once again, an alternative remedy is to design a loss function that ensures the corresponding data space is continuously embedded in $\mathcal{F}$. Under such construction, convergence can be established in a similar way without additional assumptions.*

Proof First, by density of $\mathcal{V}$ in $\mathcal{U}_o$, for any $\delta > 0$, there exists $u^\delta \in \mathcal{V} \cap \mathcal{U}_o$ such that

$$L(u^\delta) = \|\mathcal{R}[u^\delta]\|_{L^2_\lambda}^2 \leq C\|u - u^\delta\|_{\mathcal{U}_o} = \delta.$$

Using the above lemma, there exists $K = K(\delta, 3\|u^\delta\|_{\mathcal{V}})$ such that

$$\left|\hat{L}(v) - L(v)\right| \leq \delta$$

for all $v \in \mathcal{V}$ satisfying $\|v\|_{\mathcal{V}} \leq 3\|u^\delta\|_{\mathcal{V}}$. Letting $\alpha = \delta/\|u^\delta\|_{\mathcal{V}}$, and noting that $u_{K,\alpha}$ is a global minimizer of (P_μ^{emp}), we obtain

$$\alpha\|u_{K,\alpha}\|_{\mathcal{V}} + \hat{L}(u_{K,\alpha}) \leq \alpha\|u^\delta\|_{\mathcal{V}} + \hat{L}(u^\delta) \leq \alpha\|u^\delta\|_{\mathcal{V}} + L(u^\delta) + \delta = 3\delta.$$

This implies $\|u_{K,\alpha}\|_{\mathcal{V}} \leq 3\|u^\delta\|_{\mathcal{V}}$, and thus

$$L(u_{K,\alpha}) \leq \hat{L}(u_{K,\alpha}) + \delta \leq 4\delta.$$

Under Theorem 9(1), it follows that

$$\|u_{K,\alpha} - u\|_{\mathcal{U}} \leq C\|\mathcal{R}[u_{K,\alpha}]\|_{\mathcal{F}} \leq C\|\mathcal{R}[u_{K,\alpha}]\|_{L^2_\lambda} \leq C\left(L(u_{K,\alpha})\right)^{1/2} \leq C\delta^{1/2}.$$

On the other hand, if Theorem 9(2) holds, then

$$\|\mathcal{R}[u_{K,\alpha}]\|_{\mathcal{F}}^{2/\theta} \leq C\|\mathcal{R}[u_{K,\alpha}]\|_{L^2_\lambda}^2 \|\mathcal{R}[u_{K,\alpha}]\|_{\mathcal{F}_o}^{2(1-\theta)/\theta} \leq CL(u_{K,\alpha}) \max(1, \|u_{K,\alpha}\|_{\mathcal{U}_o}^{2(1-\theta)/\theta})$$

Moreover, since

$$\|u_{K,\alpha}\|_{\mathcal{U}_o} \leq \|u_{K,\alpha}\|_{\mathcal{V}} \leq 3\|u^\delta\|_{\mathcal{V}},$$

we obtain

$$\|u_{K,\alpha} - u\|_{\mathcal{U}}^{2/\theta} \leq C\delta \max(1, \|u^\delta\|_{\mathcal{V}}^{2(1-\theta)/\theta}),$$

which leads to the desired result. ■

Remark 20 *The estimates in Theorem 17 and Theorem 18 can be made more specific when considering a specific quadrature rule. For example, suppose the quadrature points are quasi-uniform, with fill distances h_D and $h_{\partial D}$ for the points on D and ∂D, respectively. The quadrature weights can then be chosen as the measures of the corresponding Voronoi cells. In this setting, one can show that*

$$\left|\|e\|_{L^2_\lambda}^2 - \|e\|_{2,\lambda,K}^2\right| \leq C\left(h_D^\gamma + \lambda\, h_{\partial D}^\gamma\right)\|e\|_{C^{0,\gamma}(\overline{D})}\|e\|_{C(\overline{D})} \leq C\left(h_D^\gamma + \lambda\, h_{\partial D}^\gamma\right)\|e\|_{C^{0,\gamma}(\overline{D})}^2,$$

where C is a generic constant. For $e = \mathcal{R}[v]$ and quasi-uniform quadrature points, this implies

$$\left|L(v) - \hat{L}(v)\right| \leq C\left(K_1^{-\gamma/d} + \lambda\, K_2^{-\gamma/(d-1)}\right)\|v\|_{\mathcal{V}}^2.$$

The constant $K = K(\delta, 3\|u^\delta\|_{\mathcal{V}})$ in the proof of Theorem 18 can also be chosen explicitly in this case. Indeed, by the above estimate, one can take $K_1^{-\gamma/d} \approx \lambda\, K_2^{-\gamma/(d-1)}$ to balance the errors from the interior and the boundary. Then it suffices to choose $K_1 \approx \left(\|u^\delta\|_{\mathcal{V}}^2/\delta\right)^{d/\gamma}$ and $K_2 \approx \left(\|u^\delta\|_{\mathcal{V}}^2/(\lambda\delta)\right)^{(d-1)/\gamma}$.

2.4 Representer theorem

To further simplify the notation, we introduce the combined collocation set for the residual $\{x_{1,k}\} \cup \{x_{2,k}\} \in \overline{D} = D \cup \partial D$, and the combined set of quadrature weights $\{w_{1,k}\} \cup \{\lambda w_{2,k}\}$. Then we introduce a new global index $k = 1, \ldots, K_1 + K_2 = K$, such that $x_k = x_{1,k}$ and $w_k = w_{1,k}$ for $k \leq K_1$, while for $k > K_1$, we set $x_k = x_{2,k-K_1}$ and $w_k = \lambda w_{2,k-K_1}$. With this, we introduce the combined linear operator evaluations as

$$l_k[u_\mu] = \begin{cases} [\mathcal{L}_E u_\mu](x_k) & \text{for } k \leq K_1, \\ [\mathcal{L}_B u_\mu](x_k) & \text{for } k > K_1. \end{cases}$$

Moreover we define the k-th residual function as

$$r_k(l_k) = \begin{cases} \hat{E}(x_k, l_k) & \text{for } k \leq K_1, \\ \hat{B}(x_k, l_k) & \text{for } k > K_1. \end{cases} \tag{2.9}$$

This allows us to rewrite the regularized discrete residual minimization problem (P_μ^{emp}) as

$$\min_{\mu \in M(\Omega)} \frac{1}{2} \sum_{k=1}^{K} w_k (r_k(l_k[u_\mu]))^2 + \alpha \|\mu\|_{M(\Omega)} \tag{2.10}$$

We proceed to prove the representer theorem, which fundamentally relies on variants of the Carathéodory theorem (Dubins, 1962; Klee, 1963). A key ingredient in the argument is the characterization of the extreme points of certain sets arising in the sparse minimization problem. Recall that an extreme point of a convex set is a point that cannot be expressed as a nontrivial convex combination of two distinct elements in the set. To that end, we first summarize some standard properties of closed balls in $M(\Omega)$ in the following lemma.

Lemma 21 *Define $B_t := \{\mu \in M(\Omega) \colon \|\mu\|_{M(\Omega)} \leq t\}$. The following statements hold.*

1. *B_t is compact in the weak-$*$ topology on $M(\Omega)$.*
2. *The extreme points of B_t are given by $\pm t\delta_\omega$ for $\omega \in \Omega$.*

The first statement in the above comes from the Banach-Alaoglu theorem and the lower semicontinuity under the weak-$*$ convergence. The second statement is also well-known, see e.g., Bartolucci et al. (2023); Bredies and Carioni (2020).

Let $\bar{N} = N_E K_1 + N_B K_2$. Using the notation introduced above, we view the collection $l[u_\mu] := (l_k[u_\mu])_{k=1}^{\bar{N}}$ as a vector in $\mathbb{R}^{\bar{N}}$. Given a vector $\vec{l} \in \mathbb{R}^{\bar{N}}$, we consider the following minimization problem:

$$\min_{\mu \in M(\Omega)} \|\mu\|_{M(\Omega)} \quad \text{subject to} \quad l[u_\mu] = \vec{l}. \tag{2.11}$$

Lemma 22 *Assume that the solution set S of (2.11) is nonempty. Then S is a compact convex subset of $M(\Omega)$ equipped with the weak-$*$ topology. As a consequence, S contains at least one extreme point.*

Proof First of all, by the linearity of the constraint, it is straightforward that S is convex. By the lower semicontinuity of the total variation norm and Theorem 7, S is also closed. Finally, since the optimization problem minimizes the total variation norm subject to a constraint, any solution must lie in a norm sublevel set B_t for some $t > 0$. Hence, $S \subset B_t$ is a closed subset of a compact set and is therefore compact. Lastly, by the Krein-Milman theorem (Conway, 1994), S contains at least one extreme point. ■

The representer theorem can now be proved with known arguments (cf., e.g., Boyer et al., 2019, Section 3.3.2).

Theorem 23 (Representer theorem (finite representation)) *The problem* (P_μ^{emp}) *possesses a finite solution*

$$\bar{\mu} = \sum_{n=1}^{N} c_n \delta_{\omega_n}, \quad \mathcal{N}\bar{\mu} = \sum_{n=1}^{N} c_n \varphi(\cdot; \omega_n)$$

where the number of atoms satisfies $N \leq \bar{N} = N_E K_1 + N_B K_2$*, with* N_E *and* N_B *denoting the numbers of components of* $\mathcal{L}_E$ *and* $\mathcal{L}_B$*, respectively, as defined in* (1.11)*.*

Proof According to Theorem 8, a (not necessarily finite) solution exists. However, standard representer theorems derived for convex problems are not immediately applicable (see, e.g., Bartolucci et al., 2023, Theorem 3.9), due to the nonlinearity (2.9). We argue by showing that any solution of (2.10) also solves a simpler convex problem. For that we define the optimal linear operator vector as $\bar{l} = l[u_{\bar{\mu}}] \in \mathbb{R}^{\bar{N}}$ It is easy to see that $\bar{\mu}$ is also a minimizer of problem (2.11) with data $\vec{l} = \bar{l}$ and conversely, any solution to this problem is also a solution of (P_μ^{emp}). This problem is convex and, by applying (Boyer et al., 2019, Theorem 3.1), together with Theorems 21 and 22, we conclude that there exists a solution expressible as a convex combination of at most $\bar{N}$ extreme points of the norm ball B_{t^*}, where t^* is the optimal value of (P_μ^{emp}). This yields the desired expression for $\bar{\mu}$ by Theorem 21(2). ■

We remark that the representer theorem provides an upper bound $\bar{N}$ on the number of atoms, whereas the RKHS formulation in (1.9) typically yields a representation with exactly $\bar{N}$ terms. However, this upper bound is often overly pessimistic and practical solutions usually admit much sparser representations; see Section 4. The main significance of the theorem is therefore that it justifies working directly with the discrete empirical problem rather than the infinite-dimensional continuous problem.

2.5 Optimality conditions and dual variables

First order optimality conditions for this problem can be derived.

Proposition 24 *Any solution* $\bar{u}$ *with* $\bar{u} = u_{\bar{\mu}}$ *to the problem* (2.10) *fulfills the following first order necessary conditions:*

$$\sum_{k=1}^{K} w_k r_k(l_k[\bar{u}])(\nabla_l r_k(l_k[\bar{u}]))^T l_k[\mathcal{N}(\mu - \bar{\mu})] + \alpha(\|\mu\|_{M(\Omega)} - \|\bar{\mu}\|_{M(\Omega)}) \geq 0,$$

for all $\mu \in M(\Omega)$. Equivalently, it fulfills the conditions

$$\sum_{k=1}^{K} w_k r_k(l_k[\bar{u}])(\nabla_l r_k(l_k[\bar{u}]))^T l_k[u-\bar{u}] + \alpha(\|u\|_{\mathcal{V}(D)} - \|\bar{u}\|_{\mathcal{V}(D)}) \geq 0,$$

for all $u \in \mathcal{V}(D)$.

To further interpret these conditions, we use the well-known characterization of the subdifferential

$$\begin{aligned}\partial\|\bar{\mu}\|_{M(\Omega)} &= \{v \in C_0(\Omega) \mid \|\mu\|_{M(\Omega)} - \|\bar{\mu}\|_{M(\Omega)} \geq \langle \mu - \bar{\mu}, v\rangle \text{ for all } \mu \in M(\Omega)\} \\ &= \{v \in C_0(\Omega) \mid \|v\|_{C(\Omega)} \leq 1, v = \operatorname{sgn}\bar{\mu} \text{ on } \operatorname{supp}\bar{\mu}\}\end{aligned}$$

to write the optimality conditions as

$$-\bar{p} \in \alpha\partial\|\bar{\mu}\|_{M(\Omega)}$$

for some appropriate dual variable $\bar{p} \in C_0(\Omega)$, such that

$$\langle \bar{p}, \mu - \bar{\mu}\rangle = \sum_{k=1}^{K} w_k r_k(l_k[\bar{u}])(\nabla_l r_k(l_k[\bar{u}]))^T l_k[\mathcal{N}(\mu - \bar{\mu})]$$

To characterize $\bar{p}$, we require the dual operator of $\mathcal{N}$ and the linear PDE operator in an appropriate sense. However, this can get very technical due to the fact that this dual operator would have to take inputs from the dual space of $C^2(\overline{D})$, which is a space of distributions (cf. Wachsmuth and Walter, 2024, section 5.2). To simplify the presentation, we identify $C^2(\overline{D})$ with a subspace of $C(\overline{D}, \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{d^2})$, using the embedding

$$\iota : C^2(\overline{D}) \to C(\overline{D}, \mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{d^2}), \quad \iota v(x) = (v(x), \nabla v(x), \nabla^2 v(x)) \quad \text{for all } x \in \overline{D}.$$

Combining the neural network and this embedding yields

$$[\iota\mathcal{N}\mu](x) = \int_\Omega (\varphi(x;\omega), \nabla_x\varphi(x,\omega), \nabla_x^2\varphi(x,\omega))\,\mathrm{d}\mu(\omega).$$

Then, we realize that $l_k[\mathcal{N}\mu]$ corresponds to an evaluation of this integral operator together with an inner product in the vector space $\mathbb{R} \times \mathbb{R}^d \times \mathbb{R}^{d^2} = \mathbb{R}^{1+d+d^2}$: For any $x_k \in D$ we have for the i-th entry $i = 1, \ldots, N_E$ of $l_k[\mathcal{N}\mu]$ we have

$$\begin{aligned}l_{k,i}[\mathcal{N}\mu] &= (c_i(x_k), b_i(x_k), A_i(x_k)) : [\iota\mathcal{N}\mu](x_k) \\ &= \int_\Omega c_i(x_k)\varphi(x_k;\omega) + b_i(x_k)\cdot\nabla_x\varphi(x_k,\omega) + \operatorname{tr}(A_i(x_k)\nabla_x^2\varphi(x_k,\omega))\,\mathrm{d}\mu(\omega),\end{aligned}$$

and similarly for $x_k \in \partial D$. Define the function under the integral as

$$\widetilde{\varphi}_{k,i}(\omega) := \begin{cases} c_i(x_k)\varphi(x_k;\omega) + b_i(x_k)\cdot\nabla_x\varphi(x_k,\omega) + \operatorname{tr}(A_i(x_k)\nabla_x^2\varphi(x_k,\omega)) & \text{for } k \leq K_1 \\ d_i(x_k)\varphi(x_k;\omega) + e_i(x_k)\cdot\nabla_x\varphi(x_k,\omega) & \text{for } k > K_1. \end{cases}$$

With this notation, it is easy to see that

$$\bar{p}(\omega) = \sum_{k=1}^{K} w_k r_k(l_k[\bar{u}])(\nabla_l r_k(l_k[\bar{u}]))^T \widetilde{\varphi}_k(\omega) \tag{2.12}$$

defines the dual variable for the optimality conditions.

With this preparation, we can derive the first order necessary conditions in the form of a support condition on the dual variable.

Proposition 25 *The first order necessary conditions derived in Proposition 24 can be expressed with the dual variable* (2.12) *as the subdifferential inclusion* $-\bar{p} \in \alpha \partial \|\bar{\mu}\|_{M(\Omega)}$ *or the support conditions*

$$|\bar{p}(\omega)| \leq \alpha \quad \bar{p} = -\alpha \operatorname{sgn} \bar{\mu} \text{ on } \operatorname{supp} \bar{\mu}.$$

The main practical use of the dual variable is that it can be computed for any given u and provides a way to check for non-optimality. If a node is found that violates the bound on the dual, this provides a descent direction to further decrese the regularized loss.

Proposition 26 (Boosting step) *Define the dual variable* $p[u]$ *associated to the variable* $u = \mathcal{N}\mu$ *as*

$$p[u](\omega) := \sum_{k=1}^{K} w_k r_k(l_k[u])(\nabla_l r_k(l_k[u]))^T \widetilde{\varphi}_k(\omega). \tag{2.13}$$

Let $\hat{\omega}$ *be a coefficient with* $|p[u](\hat{\omega})| > \alpha$*. Then, the Dirac delta function at* $\hat{\omega}$ *with negative sign of* $p[u](\hat{\omega})$ *associated solution perturbation*

$$\hat{u} := \mathcal{N}(-\operatorname{sgn}(|p[u](\hat{\omega})|)\delta_{\hat{\omega}}) = -\operatorname{sgn}(|p[u](\hat{\omega})|)\varphi(\cdot;\hat{\omega})$$

is a descent direction for (2.10) *at* $u = \mathcal{N}\mu$*.*

Remark 27 *By direct calculation, the dual variable* $p[u](\omega)$ *admits the alternative expression*

$$p[u](\omega) = \frac{d}{d\tau}\left[\hat{L}\left(u + \tau\varphi(\cdot;\omega)\right)\right]\Big|_{\tau=0}, \quad \omega \in \Omega. \tag{2.14}$$

This characterizes $p[u](\omega)$ *as the directional derivative of the loss along the candidate feature function* $\varphi(\cdot;\omega)$*, quantifying the potential decrease in the loss if the node* ω *were added. This expression provides an intuitive explanation for Proposition 26: the directional derivative must be large enough (in absolute value) to outweigh the corresponding increase in the regularization penalty. This observation forms the cornerstone of our optimization strategy, where kernel nodes are inserted and deleted dynamically.*

3. Algorithmic framework

We now present the algorithmic framework for solving PDEs by optimizing the empirical problem (P_N^{emp}). Using the notation introduced earlier, we rewrite the optimization problem in the simplified form:

$$\min_{c,\omega} \hat{L}(u_{c,\omega}) + \alpha\|c\|_1 = \min_{c,\omega} \hat{\ell}(c,\omega) + \alpha\|c\|_1 \tag{3.1}$$

where the reduced loss is defined as

$$\hat{\ell}(c,\omega) := \frac{1}{2} R(c,\omega)^T W R(c,\omega).$$

Here the residue vector $R(c,\omega) \in \mathbb{R}^K$ is defined componentwise by $[R(c,\omega)]_k = r_k(l_k(u_{c,\omega}))$. The weight matrix $W = \mathrm{diag}(w_{1,1}, \cdots, w_{1,K_1}, \lambda w_{2,1}, \cdots, \lambda w_{2,K_2}) \in \mathbb{R}^{K\times K}$ is determined by chosen quadrature rule and the penalty parameter λ and is fixed during optimization. To incorporate the network width N into the optimization process, we introduce insertion and deletion steps for kernel nodes, applied respectively before and after optimizing (c,ω) with fixed N. We therefore propose a three-phase algorithm that is executed consecutively and iteratively:

- Phase I inserts kernel nodes based on the gradient boosting strategy;
- Phase II optimizes (c,ω) with fixed N using a semi-smooth Gauss-Newton algorithm;
- Phase III removes kernel nodes whose associated outer weights are zero.

The full procedure is presented in Algorithm 1, with additional implementation details provided in Appendix A.

In the following subsections, we present the details of the kernel node insertion step, the semi-smooth Gauss–Newton algorithm, and other practical components of the proposed framework.

3.1 Kernel nodes insertion via gradient boosting

In this subsection, we describe Phase I in the three-phase algorithm, which inserts new kernel nodes to enhance the approximation capacity of the model. While adding more kernel nodes can improve the model's approximation capability, it also increases the computational cost of the optimization in Phase II. In the worst case, newly inserted nodes may be pruned in Phase III, rendering the added computational cost wasted. Therefore, new nodes should only be introduced when they are expected to yield the steepest descent in the loss function, and when further optimization over existing nodes is no longer effective.

According to Theorem 26 and Theorem 27, the dual variable $p[u](\omega)$ provides a good estimate of the potential reduction in the objective function if a new kernel node is added at location ω. In practice, we uniformly sample candidate locations from Ω and select the one with the largest $|p[u](\omega)|$ that also satisfies $|p[u](\omega)| > \alpha$. We then compare it with the dual variables of existing kernel nodes. The selected node, denoted ω_{N+1}, is inserted into the model $u = \sum_{i=1}^{N} c_n \varphi(\cdot;\omega_n)$ if

$$|p[u](\omega_{N+1})| > \max_{1\le n\le N} |p[u](\omega_n)| \tag{3.2}$$

The newly added node is initialized with coefficient $c_{N+1} = 0$.

We note that this insertion strategy is inherently greedy, as it targets the direction of steepest local descent in the objective. It is therefore reasonable to incorporate heuristic modifications to improve performance of the algorithm in practice. Those are discussed in detail in Appendix A.

Algorithm 1: Main Algorithm

Input: Initial network $u^{(0)}$(defined by $c^{(0)}$, $\omega^{(0)}$) with width $N^{(0)}$
Output: Trained network.
while $t < T$ **do**
  Phase I: *Kernel nodes insertion.*
    Sample (uniformly) $\{\hat{\omega}_m\}_{m=1}^M \subseteq \Omega$, let $\hat{p} = \max_{\hat{\omega}_m} |p(\hat{\omega}_m)[u^{(t)}]|$ and $\omega_{N^{(t)}+1} = \arg\max_{\hat{\omega}_m} |p(\hat{\omega}_m)[u^{(t)}]|$
    Compute $p_0 = \max_{1\le n\le N^{(t)}} |p(\omega_n^{(t)})[u^{(t)}]|$
    if $\hat{p} > \max\{\alpha, p_0\}$ **then**
      $N^{(t+1/2)} = N^{(t)} + 1$, $c^{(t+1/2)} = c^{(t)} \cup \{0\}$, $\omega^{(t+1/2)} = \omega^{(t)} \cup \{\omega_{N^{(t)}+1}\}$
    end
    else
      $N^{(t+1/2)} = N^{(t)}$, $c^{(t+1/2)} = c^{(t)}$, $\omega^{(t+1/2)} = \omega^{(t)}$
    end
  Phase II: *Optimize $c^{(t+1/2)}, \omega^{(t+1/2)}$ simultaneously with a semi-smooth Gauss-Newton algorithm*
    Obtain J_R and then G, DG based on (3.5), (3.6), (3.7).
    Compute search direction $(z_c, z_\omega) = z = -DG^{-1}G$.
    Perform a line search of optimal step size and update $c^{(t+1/2)}$, $\omega^{(t+1/2)}$.
  Phase III: *Kernel nodes deletion.*
    Remove nodes with $c_n^{(t+1/2)} = 0$ (resulting in $c^{(t+1)}, \omega^{(t+1)}$ with width $N^{(t+1)}$)
  $t = t + 1$
end

3.2 Semi-smooth Gauss-Newton

The backbone of the three-phase algorithm is the optimization of $(c, \omega) \in \mathbb{R}^N \times \mathbb{R}^{N(d+1)} = \mathbb{R}^{N(d+2)}$ in Phase II. The first order optimality condition yields

$$-\nabla\hat{\ell}(c,\omega) = -J_R(c,\omega)^T W R(c,\omega) \in \alpha\partial_{c,\omega}\|c\|_1 \tag{3.3}$$

where $J_R(c,\omega) := (\nabla_c R(c,\omega), \nabla_\omega R(c,\omega)) \in \mathbb{R}^{K\times N(d+2)}$ is the Jacobian and $\partial_{c,\omega}\|c\|_1 := \partial_c\|c\|_1 \times \{0_\omega \in \mathbb{R}^{N(d+1)}\}$ is the extended subdifferential. Instead of using a standard (proximal) gradient descent method, we apply a second order semismooth Newton-type method.

To this end, we utilize the proximal operator of $\alpha\|c\|_1$, defined as

$$\begin{aligned}\mathrm{Prox}_{\alpha\|\cdot\|_1}(x) &= \mathrm{argmin}_y \left[\frac{1}{2}\|x-y\|_2^2 + \alpha\|y\|_1\right] \\ &= \left(\mathrm{sign}(x_i)\max\{|x_i| - \alpha, 0\}\right)_{i=1}^n .\end{aligned} \tag{3.4}$$

For notational simplicity, we denote $\mathrm{Prox} := \mathrm{Prox}_{\alpha\|\cdot\|_1}$ in remainder of the text. Using the proximal operator, the subdifferential inclusion (3.3) is equivalent a root-finding problem: find (q, ω) such that $c = \mathrm{Prox}(q)$ and

$$G(q,\omega) = (q - \mathrm{Prox}(q), 0) + \nabla\hat{\ell}(\mathrm{Prox}(q), \omega) = 0. \tag{3.5}$$

G is referred to as the Robinson normal map, introduced by Robinson (1992). To address the non-differentiability introduced by the proximal operator, the generalized derivative of G is calculated in a semismooth context (Pieper, 2015), given by

$$DG(q,\omega) = (\mathrm{Id} - DP(q,\omega)) + \nabla^2\hat{\ell}(\mathrm{Prox}(q),\omega)DP(q,\omega) \tag{3.6}$$

where $DP = (D\,\mathrm{Prox},\ \mathrm{Id})$ is the semismooth derivative of the map $(q,\omega) \to (\mathrm{Prox}(q),\omega)$. The root-finding problem (3.5) is then solved via descent direction $z = -DG^{-1}G$, and a line search strategy is applied to determine the step size. In practice, we approximate the Hessian of the loss by

$$\nabla^2\hat{\ell}(c,\omega) \approx J_R(c,\omega)^T W J_R(c,\omega), \tag{3.7}$$

which is standard in Gauss-Newton type methods (Wright, 2006).

Note that, due to the introduction of Robinson variables, we need to set q_{N+1} for a newly inserted point in the greedy/boosting step. To obtain $c_{N+1} = 0$ and a guarantee that the variable can be immediately activated with a small stepsize, q_{N+1} is set to be $-\alpha\mathrm{sign}(p[u](\omega_{N+1}))$. Moreover, we adopt the convention that $D\mathrm{Prox}(\pm\alpha) = 1$.

We remark that the optimization problem is in general nonconvex, which results both from when r_k is nonlinear in l_k, as is the case with nonlinear PDEs, and also the optimization in the inner weights. Moreover, the second order optimization algorithm exhibits better efficiency than first order methods. We refer readers to Pieper (2015) for a complete discussion of semi-smooth calculus involved here, and additional technical details are discussed in Appendix A.

3.3 Computational cost

The computational cost of our method primarily arises from evaluating the Jacobian J_R as well as dual variables p for sampled weights ω. At each iteration, the computational cost for computing these scales as $\mathcal{O}(NK)$ and $\mathcal{O}(MK)$ respectively. Given the dependencies on number of collocation points K, this cost can be effectively reduced by subsampling a minibatch of collocation points at each iteration, which is widely adopted in the training of data-driven models.

We note that the computational cost is well controlled by our adaptive kernel node insertion and deletion strategy, where the network width N is increased only when necessary. However, N may still become undesirably large when approximating complex solutions (e.g. high-dimensional solutions without localized features or low effective dimensions). This issue becomes more obvious in high-dimensional problems as the number of parameters to optimize also scales with the dimension d. To mitigate this, one may choose, based on their dual variables, only a subset of $\{\omega_n\}_{n=1}^N$ during each iteration to update.

In this work, however, we do not employ such acceleration techniques and instead use the most basic version of the algorithm.

3.4 Choice of hyperparameters: penalty weight λ and regularization parameter α

In practice, the hyperparameters α and λ have a significant impact on solution quality. The parameter α controls the strength of the ℓ_1 regularization and thus influences

the compactness of the learned representation. Larger values of α promote sparser kernel representations, whereas smaller values allow more kernels and may improve accuracy at the risk of overfitting. For this reason, we often adopt a continuation strategy in α: we first train with a moderately large value to obtain a stable sparse representation, and then gradually decrease α while initializing from the previous solution (see Section 4).

The penalty weight λ balances enforcement of boundary conditions against the interior PDE residual. If λ is too small, boundary constraints may be under-enforced, resulting in noticeable boundary errors. Conversely, overly large λ may bias the optimization toward boundary fitting, slowing convergence and potentially degrading overall accuracy. In practice, we choose λ so that the boundary penalty term and the interior residual term remain comparable in magnitude during training. Additional treatments of boundary conditions are discussed in the next subsection.

We also note that both parameters are inherently related to the number of collocation points K. As K increases, the empirical approximations of both the interior residual and boundary penalty terms become more accurate, and the sensitivity to the precise choice of α and λ is typically reduced. While the analysis in Section 2 clarifies the roles of these parameters, we do not claim a universal scaling law with respect to K. Designing automated strategies for tuning α and λ remains an interesting direction for future work. A variety of approaches have been proposed in other settings, including cross-validation, square-root LASSO, information-theoretic criteria, and heuristic methods such as the L-curve. Their applicability to the present PDE-solving setting, however, remains to be established.

3.5 Treatment of boundary conditions

We note that our method does not strictly enforce the boundary condition but includes it as a penalty term in the objective, which is empirical estimation of $\|\mathcal{B}[u_{c,\omega}]\big|_{\partial D}\|^2_{L^2(\partial D)}$. While widely adopted, the choice of L^2-norm is largely heuristic and not always theoretically appropriate. In particular, for problems with Dirichlet boundary conditions, it is more natural to consider a boundary norm consistent with the regularity of the exact solution. For example, if Dirichlet data $u\big|_{\partial D} \in H^{1/2} \subsetneq L^2(\partial D)$, as is the case when $u \in H^1(D)$, merely penalizing boundary misfit via $L^2(\partial D)$ may be insufficient to enforce the boundary condition with the appropriate level of regularity. A stricter penalty term such as $\|u\big|_{\partial D}\|^2_{H^1(\partial D)}$ can better incorporate theoretical requirements of the problem and lead to improved numerical performance.

Even with an appropriate choice of norm, the penalty-based approach remains suboptimal. Since it relaxes the boundary condition rather than enforcing it exactly, tuning the penalty parameter λ becomes as mentioned in the above subsection. Considering this, alternative approaches to enforcing Dirichlet boundary conditions beyond the penalty method are of great interest (Lagaris et al., 1998; Sukumar and Srivastava, 2022). For the homogeneous Dirichlet case, the solution can be parameterized as

$$u(x) \approx \gamma(x) \sum_{n=1}^{N} c_n \varphi(x; \omega_n) \tag{3.8}$$

where $\gamma \in C_0^2(D)$ with $\gamma(x) > 0$ for $x \in D$ is a twice continuously differentiable function that vanishes on the boundary ∂D. This formulation preserves the theoretical guarantees

discussed earlier, provided that $u(x)/\gamma(x)$ satisfies the required regularity. More generally, the method works for any Dirichlet boundary function $f : \partial D \to \mathbb{R}$ under the assumption that f admits an extension $\bar{f} : D \to \mathbb{R}, \bar{f}\big|_{\partial D} = f$ that is twice continuously differentiable. Since such an extension may only exist for smooth domains with f at least twice continuously differentiable, this limits this approach in practice to functions f where such an extension can be easily constructed. We may then set $u(x) = \bar{f}(x) + \gamma(x)\mathcal{N}(x)$. The extension function $\bar{f}$ and the vanishing function γ can often be constructed in low-dimensional domains with simple geometry. However, for complex or high-dimensional geometries, finding such functions becomes significantly more challenging. While there are interesting data-driven approaches, their investigation lies beyond the scope of this work and is left for future research.

3.6 Additional implementation components

We discuss a few additional components for practical convenience here.

3.6.1 Constraining ω

During each Gauss–Newton step, the parameter $\omega = (y, \sigma)$ may fall outside the prescribed parameter space Ω. To ensure that σ remains within $\Omega_\sigma = (0, \sigma_{\max}]$, we introduce a parameterization

$$\sigma(s) = \sigma_{\min} + (\sigma_{\max} - \sigma_{\min})\tau(s), \tag{3.9}$$

where τ is any increasing bijection from $\mathbb{R}$ to $(0, 1)$, typically chosen as the sigmoid function $\tau(s) = 1/(1 + \exp(-s))$. The lower bound $\sigma_{\min}$ is usually set to a small positive value (e.g. 10^{-3}) rather than 0 to safeguard against numerical instability caused by excessively small σ, though in practice σ remains well above this lower bound during training. Occasional out-of-bounds y is not problematic and rarely observed in practice.

3.6.2 Stopping criterion

The iteration in Algorithm 1 may be terminated when neither the optimization of existing kernel nodes nor the insertion of new nodes results in a significant descent of the loss function. Please see Appendix A for more details.

4. Numerical experiments

In this section, we conduct numerical experiments to demonstrate the effectiveness and versatility of the proposed method. We begin with a semilinear Poisson equation (example in Section 1.3) as a simple testbed of the proposed method, including exploratory studies on higher-dimensional PDEs and treatment of boundary constraints. Next, we use our method to solve a regularized Eikonal equation, where we also investigate the convergence behavior of numerical solutions to the unique viscosity solution. Finally, we consider a spatial-temporal viscous Burgers' equation, where our method is coupled with a time discretization scheme.

We begin by outlining some common settings and evaluation metrics used across all experiments.

Table 1: Evaluation Metrics

Metric	Definition
L^2 error	L^2 error between numerical and exact solutions estimated on test grid.
L^∞ error	Maximum absolute error between numerical and exact solutions on test grid.
$\hat{L}$ (Train)	Empirical loss evaluated on training collocation points.
$\hat{L}$ (Test)	Empirical loss evaluated on test grid.
#Kernels	Number of kernels/feature functions used in the numerical solution.

Settings and evaluation metrics

In all numerical experiments, we use the following generic settings:

- We set $s = d + 2 + 0.01$ in the feature function, i.e.,

$$\varphi(x;\omega) = \frac{\sigma^{2.01}}{\left(\sqrt{2\pi}\right)^d} \exp\left(-\frac{\|x-y\|_2^2}{2\sigma^2}\right).$$

  This fulfills the requirement for theory developed in Section 2.

- We set $\tau(s) = 1/(1+\exp(-s))$ and $(\sigma_{\min}, \sigma_{\max}) = (10^{-3}, 1)$ in all experiments in the parameterization of σ given by (3.9).

- $M = 10^4$ candidates feature functions will be sampled from Ω in Phase I of each iteration.

- We use an empty network as the initialization of Algorithm 1 unless otherwise specified.

In addition, all algorithmic components are implemented in Python using JAX[1], executed on a 2023 Macbook Pro with Apple M2 Pro chip (16GB memory) using CPU-only computation. Training times typically range from 30 seconds to several minutes. In all the following experiments, double-precision (float64) arithmetic is used. However, the method is, in general, compatible with single-precision (float32) computations as well.

We evaluate the numerical solution $\hat{u}$ using several metrics (see Table 1). These include the L^2 and L^∞ errors with respect to the true solution u, both estimated on a finer test grid. To assess the impact of the regularization term in preventing overfitting, we also report the empirical loss $\hat{L}$, computed on both the training collocation points and the test grid. In addition, we record the number of kernels used in the solution to indicate the level of sparsity.

The Python codes associated with the examples tested in this section can be accessed from the GitHub repository https://github.com/EddyShao/SparseKernelPDE

1. JAX is used here mainly for auto-differentiation, yet all derivatives can also be directly implemented from their analytical forms.

4.1 Semilinear Poisson equation

We use a semilinear Poisson equation in (1.13) as a simple testbed for our method. We begin with a 2D equation where the exact solution features two steep bumps. We then test our approach in a 4-dimensional problem to explore its capability in handling higher-dimensional equations. Finally, we discuss treatment of boundary conditions, including the mask function technique introduced in the Section 3.5. In all following numerical experiments, we prescribe $D = [-1, 1]^d \subseteq [-2, 2]^d = \Omega$, where d is 2 or 4.

4.1.1 2D equation

We prescribe the exact solution $u : D \subseteq \mathbb{R}^2 \to \mathbb{R}$ as

$$u(x) = v(x; c_1, k_1, R_1) + v(x; c_2, k_2, R_2), \tag{4.1}$$

where

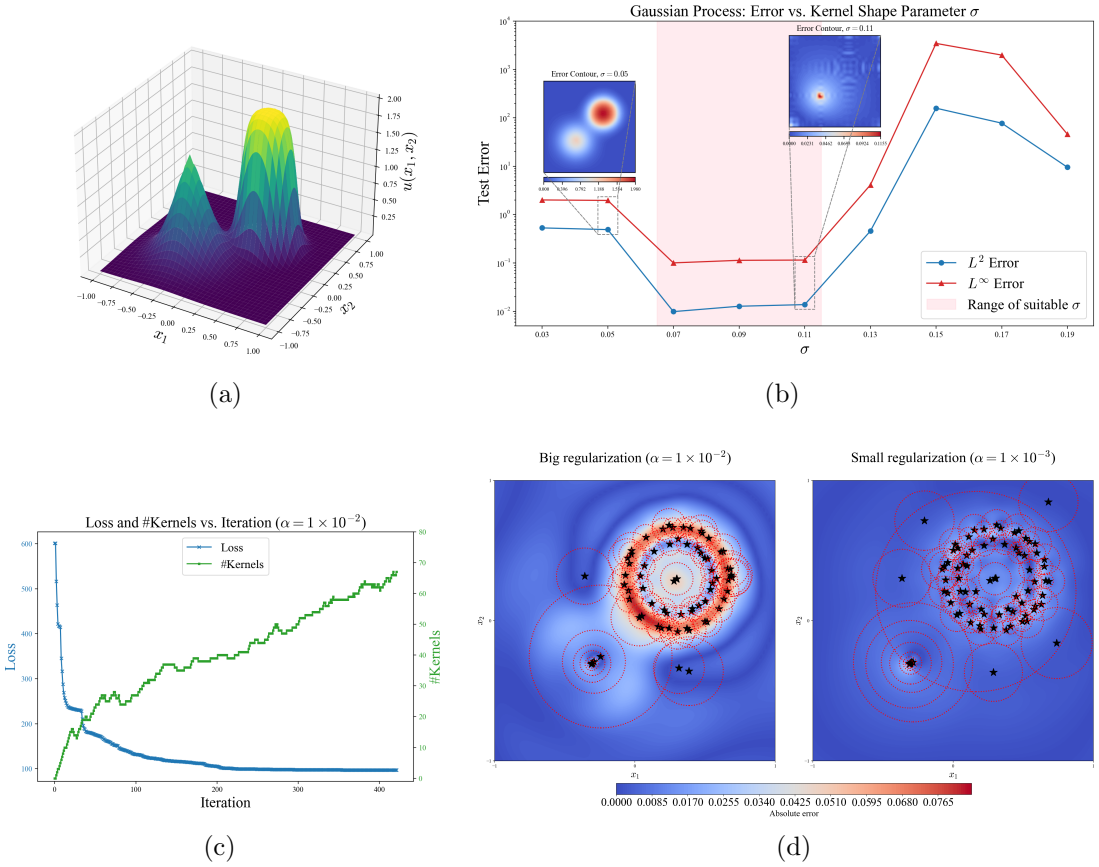
$$v(x; c, k, R) = \tanh\left(k\left(R_0 - |x - c|\right)\right) + 1,$$

$$\begin{cases} c_1 = (0.3,\ 0.3), \\ c_2 = -c_1, \\ (k_1,\ k_2) = (4,\ 12), \\ (R_1,\ R_2) = (0.30,\ 0.15) \end{cases}.$$

The source terms $f_{\mathcal{E}}$ and $f_{\mathcal{B}}$ in the formulation of the example in Section 2 are then computed accordingly. Here v is constructed as a bump with center at $c \in \mathbb{R}^2$ and steepness parameterized jointly by R and k. The exact solution u then possesses two bumps with distinct steepness (see Figure 2a). In particular, the multi-scale steepness poses difficulties for the GP method, where the position of the RBF kernel is prescribed with a fixed shape parameter. To see this, we tested GP method with different shape parameter (σ in $\kappa(x, x') = \exp(-\frac{\|x-x'\|_2^2}{2\sigma^2})$) and observed significant volatility in the results (see Figure 2b). Moreover, different kernel widths might attain maximum absolute error at the center of 2 steepness bumps, respectively, indicating that the single fixed kernel shape parameter σ faces the challenges of handling distinct steepness simultaneously.

We then solve the PDE using proposed method with penalty parameter $\lambda = 10^3$ in the formula of $\hat{L}$ to enforce the boundary condition and regularization parameter $\alpha = 10^{-2}, 10^{-3}$. Convergence of loss and growth of the number of kernels are shown in Figure 2c. In practice, we solve case with $\alpha = 10^{-3}$ using the solution obtained with $\alpha = 10^{-2}$ as initialization. This strategy largely saves computational time and results in better performance occasionally (see Appendix A.4 for more details). The error contours are shown in Figure 2d. We also visualized the positions and shape parameters of learned kernel nodes $\{\omega_n\}$, observing that most of the kernels cluster at the location of the sharp transition. This shows our algorithm can capture localized structure and sharp transitions around the bumps.

We further test the proposed method with a wider range of α and number of collocation grid points K. We compare the results with the GP method with a range of kernel shape



(a) (b) (c) (d)

Figure 2: Numerical results using Sparse RBFNet (our) and Gaussian Process (GP) on solving a semilinear Poisson equation: (a) Exact solution; (b) Error of solutions obtained by the GP method with different kernel shape parameters; (c) Convergence of loss and growth of number of kernels (regularization parameter $\alpha = 10^{-2}$); (d) Error contours of our method. Each $\star$ represents the location parameter y_n of a learned kernel node, and the dotted circle around has radius σ_n (very few kernel nodes outside of D are omitted).

parameter σ, evaluated with metrics listed in Table 1. Results are shown in Table 2. We see that our method effectively finds a sparse and accurate solution without searching optimal σ as necessary in Gaussian Process method. Meanwhile, decreasing α does not always lead to improved solution accuracy, especially when collocation points are relatively sparse in the domain. In such cases, the empirical loss $\hat{L}$ on the test grid is drastically larger than its estimate on the collocation points, suggesting overfitting. This shows the significant role of the regularization term in preventing overfitting, in addition to obtaining a sparse representation of solutions.

We remark that grid collocation points are used in the above experiments. Using random collocation points negatively affect the performance of both methods (See Table 6 in Appendix B). In general, the performance of our method is unfortunately sensitive to the

Table 2: Errors, residual loss, and number of kernels for Sparse RBFNet and GP using grid collocation points. Case with $\alpha = 10^{-3}(10^{-4})$ is solved with solution obtained by $\alpha = 10^{-2}(10^{-3})$ as initialization. Test results are evaluated on a 100×100 grid in D. All results are averaged over 10 runs.

		Sparse RBFNet			**Gaussian Process**		
K (K_1, K_2)	Metric	$\alpha =$ 1e−2	$\alpha =$ 1e−3	$\alpha =$ 1e−4	$\sigma = 0.05$	$\sigma = 0.10$	$\sigma = 0.15$
20 (324, 76)	L^2 error	1.05e−1	8.94e−2	8.42e−2	5.29e−1	3.89e−2	3.62e−2
	L^∞ error	2.52e−1	2.23e−1	2.24e−1	2.02e+0	2.64e−1	2.41e−1
	$\hat{L}$(train)	4.78e+0	7.18e−2	1.69e−3	–	–	–
	$\hat{L}$(test)	6.56e+1	5.89e+1	5.82e+1	–	–	–
	#Kernels	46	65	86	400	400	400
30 (784, 116)	L^2 error	4.99e−2	1.98e−2	1.77e−2	4.89e−1	1.32e−2	nan
	L^∞ error	9.04e−2	5.35e−2	4.87e−2	1.96e+0	1.14e−1	nan
	$\hat{L}$(train)	1.50e+1	2.41e−1	6.00e−3	–	–	–
	$\hat{L}$(test)	1.77e+1	6.81e+0	6.88e+0	–	–	–
	#Kernels	68	83	102	900	900	900
50 (2304, 196)	L^2 error	4.37e−2	1.22e−2	1.08e−2	6.40e−3	nan	6.53e−1
	L^∞ error	8.47e−2	4.51e−2	4.52e−2	7.07e−2	nan	1.81e+0
	$\hat{L}$(train)	1.82e+1	5.57e−1	3.73e−2	–	–	–
	$\hat{L}$(test)	1.75e+1	2.15e+0	1.59e+0	–	–	–
	#Kernels	72	104	125	2500	2500	2500

uniformity of the collocation points. This negative impact is attributed to the increased gap between $\nu_D^{K_1}$, $\nu_{\partial D}^{K_2}$ and ν_D, $\nu_{\partial D}$, which further corrupts the empirical residual loss.

4.1.2 4D equation

We now consider the same equation in 4D as an exploratory example of solving higher-dimensional problems with the proposed method. We set $D = [-1, 1]^4 \subseteq [-2, 2]^4$ and exact solution u be prescribed as

$$u(x) = \prod_{d=1}^{4} \sin(\pi x_d) \tag{4.2}$$

with source term and boundary conditions computed accordingly.

We fix $\lambda = 3000$ use grid collocation points $K = 6^4, 8^4$ with $\alpha = 10^{-3}, 10^{-4}$ respectively. The results are shown in Table 3. While obtaining sparse and accurate solutions, our method effectively prevents overfitting. When there is not a sufficient number of collocation points (e.g., $K = 1296$), decreasing regularization parameter α might lead to severe overfitting ($\hat{L}_{\text{test}} \gg \hat{L}_{\text{train}}$). This results in degraded accuracy even though significantly more kernel nodes are inserted compared to the case with a larger α.

We highlight several key advantages of our method for solving equations in high-dimensional spaces compared with other methods. First, by employing a shallow network

Table 3: Errors and residual loss of numerical solutions to a 4D semilinear Poisson equation using grid collocation points. Test results are evaluated on a grid of 20^4 in $D \subseteq \mathbb{R}^4$. All results are averaged over 10 runs.

K (K_1, K_2)	1296 (256, 1040)		4096 (1296, 2800)	
α	10^{-3}	10^{-4}	10^{-3}	10^{-4}
L^2 error	1.75e−1	2.31e−1	1.45e−1	5.71e−2
L^∞ error	2.39e−1	4.55e−1	1.90e−1	7.69e−2
$\hat{L}$(train)	3.93e+0	3.60e−1	7.06e+0	9.19e−1
$\hat{L}$(test)	1.03e+1	1.78e+1	6.27e+0	1.30e+0
#Kernels	173	239	204	365

with smooth feature functions instead of a multi-layer perception (MLP) as is often used in other PINN-based methods, all required derivatives can be computed analytically. This is especially beneficial when computing higher-order derivatives, as it eliminates the need to construct large computational graphs for automatic differentiation or to rely on Monte Carlo-based estimations, thereby reducing both memory usage and computational cost (Sirignano and Spiliopoulos, 2018). Second, our method offers greater flexibility compared to GP methods. In a GP formulation, the number of trainable/optimizable parameters scales as $\mathcal{O}(K)$, resulting in substantial computational cost for operations such as Cholesky factorization of a matrix of size $\mathcal{O}(K) \times \mathcal{O}(K)$. Since a large number of collocation points K is typically required to learn complex solutions in high-dimensional spaces, this poses a significant computational bottleneck and requires the use of mini-batch techniques (Yang and Owhadi, 2023; Tran-Dinh et al., 2020). In contrast, our method inserts new kernel nodes/feature functions only when necessary, thereby keeping the number of trainable (optimizable) parameters manageable, usually much fewer than collocation points. Lastly, the regularization term in our method effectively prevents overfitting, which is particularly important when collocation points are relatively sparse in the domain, as this is often the case in high-dimensional problems. We emphasize that, however, these advantages are most pronounced where the solution admits additional low-complexity structure (e.g., localized features or low effective dimensionality), so that accurate approximations can be achieved with relatively few active kernels. As with other kernel-based methods, we do not claim to overcome the curse of dimensionality in fully generic unstructured settings.

4.1.3 Boundary conditions treatments

We now offer a numerical experiment on the discussion of the treatment of boundary conditions in Section 3.5. We now consider the same equation in $2D$ but with exact solution prescribed as

$$u(x_1, x_2) = \sin(\pi x_1)\sin(\pi x_2) + 2\sin(2\pi x_1)\sin(2\pi x_2).$$

This function was selected for its nontrivial boundary behavior (see Figure 3a), making it a suitable testbed for various boundary condition treatments. We set $\alpha = 10^{-3}$, $K = 400$ grid points and solve the equation with $\lambda = 10^{-1}, 10^0, 10^1, 10^2, 10^3, 10^4$. We use both

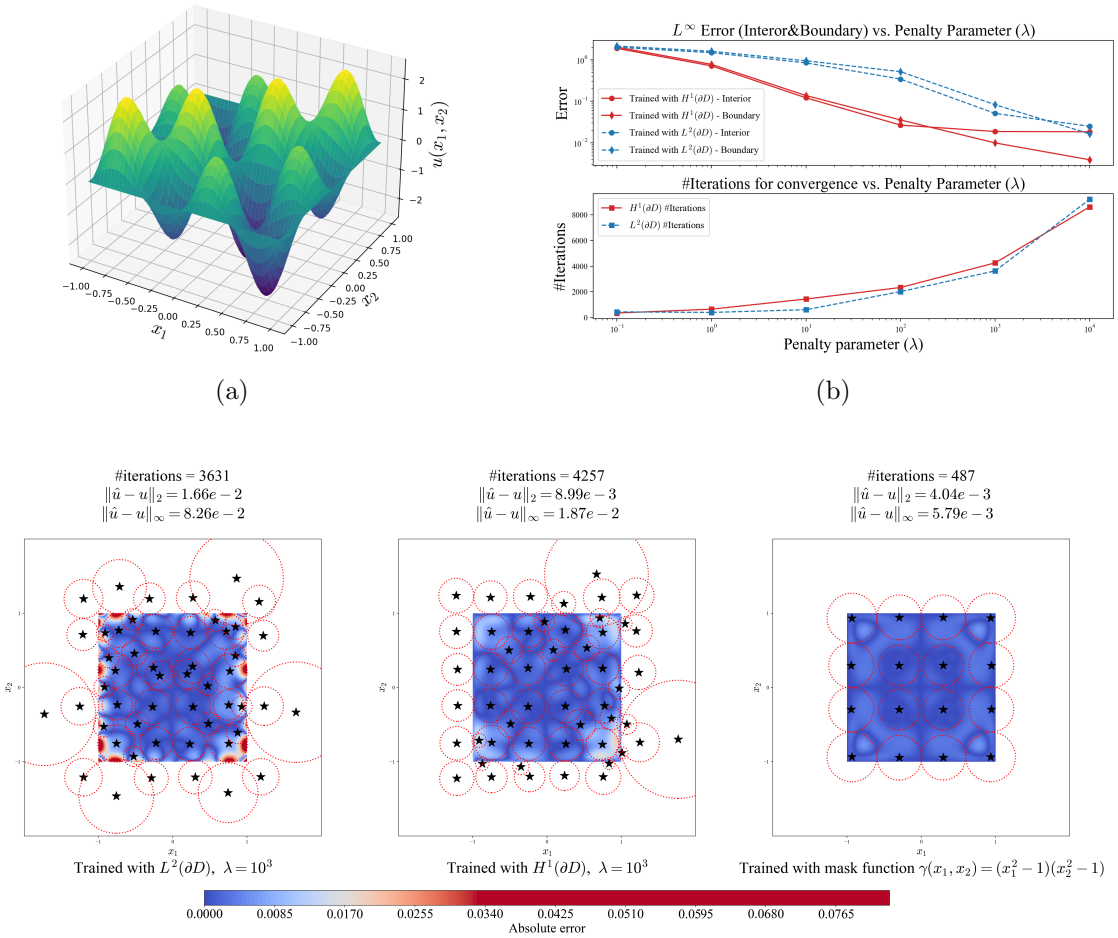


Figure 3: Numerical results of different treatments of boundary conditions. (a) Exact solution $u(x_1, x_2)$; (b) Error and convergence speed – penalty coefficient λ (results are averaged from 10 runs); (c) Error contour of solutions with L^2-type loss, H^1-type loss, and "mask" function for boundary conditions. Each $\star$ represents the location parameter y_n of a learned kernel node, and the dotted circle around has radius σ_n. The contour plots are taken from a single run, while the reported number of iterations, L^2 error and L^∞ error are averaged over 10 runs.

the $L^2(\partial D)$-type loss and $H^1(\partial D)$-type loss mentioned in Section 3.5 to enforce boundary conditions. As is shown in Figure 3b, increasing λ leads to higher-quality solutions. Notably, solutions trained with H^1-type loss achieve comparable accuracy to L^2-type loss with a much smaller λ, which is of practical significance since a larger λ requires substantially more iterations for convergence.

We remark that a larger λ does not always lead to improved solutions in practice, even when sufficient iterations are allowed for convergence. A key contributing factor is the potential overfitting to boundary data, as the regularization effect of the $\alpha\|c\|_1$ term may be

Table 4: Errors and residual loss of numerical solutions to the regularized Eikonal equation ($\epsilon = 0.10$) using grid collocation points with spatial mask $\gamma(x) = (1 - x_1^2)(1 - x_2^2)$. Test results are evaluated on a 100×100 grid in D. All results are averaged over 10 runs.

	$K = 400$ $(324, 76)$			$K = 900$ $(784, 116)$			$K = 2500$ $(2304, 196)$		
Metric	10^{-4}	10^{-6}	10^{-8}	10^{-4}	10^{-6}	10^{-8}	10^{-4}	10^{-6}	10^{-8}
L^2 error	1.94e−2	2.97e−3	3.99e−3	1.94e−2	1.43e−3	8.71e−4	1.95e−2	1.20e−3	1.91e−4
L^∞ error	3.04e−2	6.50e−3	7.43e−3	2.98e−2	5.71e−3	1.90e−3	2.97e−2	5.28e−3	1.28e−3
$\hat{L}$(train)	9.40e−3	2.66e−4	1.30e−6	1.10e−2	5.13e−4	1.35e−5	1.24e−2	7.85e−4	6.01e−5
$\hat{L}$(test)	1.39e−2	3.07e−3	3.91e−3	1.36e−2	1.67e−3	5.70e−4	1.36e−2	1.23e−3	1.94e−4
#Kernels	15	61	81	14	71	89	13	88	115

diminished when λ becomes excessively large. This suggests that the number of boundary collocation points K_2 may need to be increased in proportion to λ to maintain a proper balance between boundary data fitting and regularization. To address this challenge, the "mask" function trick introduced in Section 3.5 can solve this difficulty as it avoids using a penalty method at all. With γ in (3.8) set to be $\gamma(x) = (x_1^2 - 1)(x_2^2 - 1)$, we obtain a more sparse and more accurate solutions within much fewer iterations (see Figure 3c).

4.2 Eikonal equation

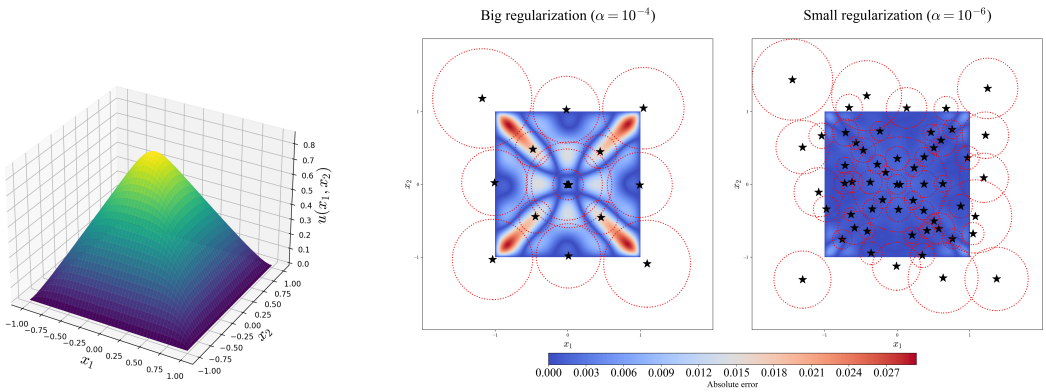
We consider the regularized Eikonal equation with homogeneous Dirichlet boundary condition

$$\begin{aligned} |\nabla u(x)|^2 + \epsilon \Delta u(x) &= f^2(x) \qquad \forall x \in D \\ u(x) &= 0 \qquad \forall x \in \partial D \end{aligned}$$

where $D = [-1, 1]^2 \subseteq [-2, 2]^2 = \Omega$, $f \equiv 1$, and $\epsilon \in \mathbb{R}_+$. In this case, $\mathcal{E}[u] = |\nabla u| + \epsilon \Delta u - f^2$ and $\mathcal{B}[u] = u$. We first set $\epsilon = 0.1$ and use $K = 30^2$ ($K_1 = 784$, $K_2 = 116$) grid collocation points, with $\alpha = 10^{-4}$ and $\alpha = 10^{-6}$ respectively. Resulting error contours[2] are shown in Figure 4b. Our method obtains a highly sparse representation of the solution, with kernel nodes concentrated along the diagonals and the center where the solution undergoes sharp transitions. Results with a wider range of K and α are shown in Table 4.

We then check the convergence towards the unique viscosity solution as $\epsilon \to 0$. The unique viscosity solution in this case is $u_{\text{visc}} = \text{dist}(x, \partial D) = \min(1 - |x_1|, 1 - |x_2|)$. We solve the Eikonal equation with $\epsilon = 0.5, 0.1, 0.05, 0.01$ respectively with $K = 30^2$ grid collocation points with mask function $\gamma(x) = (1 - x_1^2)(1 - x_2^2)$ and $\alpha = 10^{-6}$. We observe that our method shows excellent convergence behavior as ϵ approaches 0 (see Figure 5). As the solution becomes increasingly non-smooth near the diagonals, kernel nodes adaptively cluster in these regions, capturing the sharp transition of the solution.

2. Exact solution is obtained using finite difference method with transformation $u = -\epsilon \log v$ (Same as Chen et al., 2021) with 2000 uniform grids in each dimension of D.



(a) (b)

Figure 4: Numerical results of regularized Eikonal equation ($\epsilon = 0.10$). (a) Exact solution $u(x_1, x_2)$, $\epsilon = 0.10$; (b) Error contour of numerical solutions with regularization parameter $\alpha = 10^{-4}, 10^{-6}$. Each $\star$ represents the location parameters y_n of learned kernel nodes, and the dotted circle around are of radius σ_n.

4.3 Viscous Burgers' equation

We now consider the viscous Burgers' equation

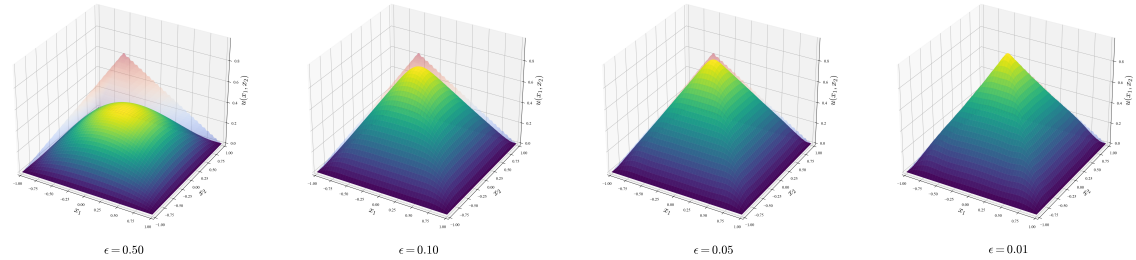
$$\begin{aligned} \partial_t u + u\partial_x u - 0.02\partial_x^2 u = 0, \ \forall (t,x) \in (0,1] \times (-1,1) \\ u(0,x) = -\sin(\pi x) \\ u(t,-1) = u(t,1) = 0 \end{aligned}$$

Instead of using a spatial-temporal formulation as in Chen et al. (2021); Raissi et al. (2019), we employ a simple (fully) implicit backward Euler method for time discretization, which not only reduces the dimension of the problem but also alleviates the difficulty of selecting appropriate penalty parameter λ that simultaneously balances initial and boundary conditions. The resulting algorithm solves an array of one-dimensional equations for $\{u^n\}_{n=1}^N$ ($N = \lceil \frac{1}{\Delta t} \rceil$) sequentially starting from $u^0 = u(0,\cdot)$. To specify, we use Algorithm 1 at each time step n with the residuals

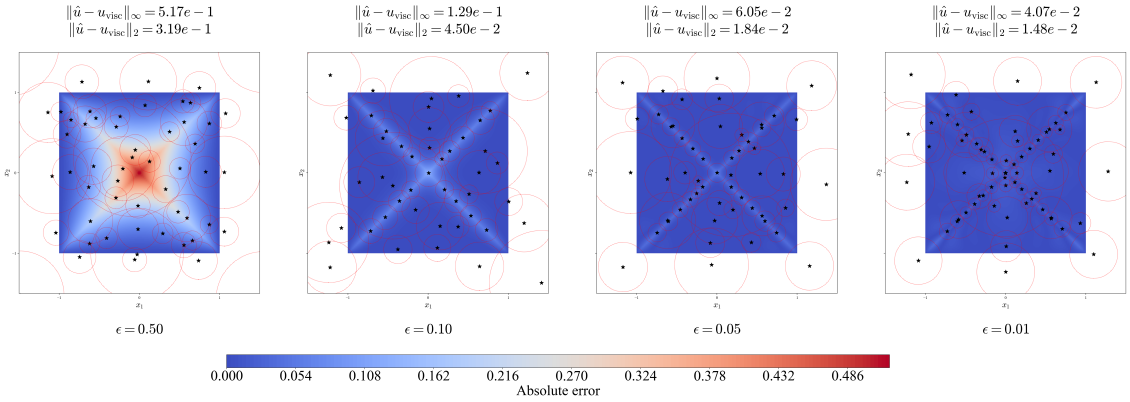
$$\begin{aligned} \mathcal{E}^n[u^n] &:= \frac{u^n - u^{n-1}}{\Delta t} + u^n \partial_x u^n - \nu \partial_x^2 u^n \\ \mathcal{B}^n[u^n] &:= u^n. \end{aligned}$$

We set $\Delta t = 0.01$, $\alpha = 10^{-4}$ and fix $K = 40$ grid collocation points in space domain $D_x = [-1,1] \subseteq [-2,2] = \Omega_x$ for each solve of u^n. The mask function technique introduced in Section 3.5 is used to enforce $u^n(\pm 1) = 0$ (with $\gamma(x) = (x+1)(x-1)$). $\{u^n\}$ is then solved sequentially, with u^n initialized as u^{n-1} for $n \geq 2$ [3]. Convergence plots of u^n for $n = 1, 20, 50, 80$ are shown in Figure 6a. We remark that our sequential solving of u^n benefits tremendously from transfer learning: since the solution is continuous in time, it

3. In practice, we solve scaled equation i.e. $u^n - u^{n-1} + \Delta t(u^n \partial_x u^n - \nu \partial_x^2 u^n) = 0$ for some implementation convenience, in which α needs to be scaled as $\Delta t^2 \alpha$.



(a)



(b)

Figure 5: Convergence of numerical solution to viscosity solution $u_{\text{visc}}(x) = \text{dist}(x, \partial D) = 1 - \max(|x_1|, |x_2|)$ as $\epsilon \to 0$. (a) Solution surfaces (shadow in the background represents viscosity solution); (b) Error contours with shrinking ϵ. Each $\star$ represents the location parameters y_n of learned kernel nodes, and the dotted circle around are of radius σ_n. The contour plots are taken from a single run, while the reported L^2 and L^∞ errors are averaged over 10 runs.

is natural to initialize u^n as u^{n-1} trained in the last iteration, which effectively reduces number of iterations required for convergence. After obtaining $\{u_n\}$, the numerical solution is extended to the entire domain D via linear interpolation. The error contour, together with three slices of exact and numerical solution, is shown in Figure 6b and Figure 6d. We also observe a rapid increase in the number of kernel nodes between $t = 0.2$ and $t = 0.5$, corresponding to the formation of a shock wave. During this period, the network faces increased difficulty in approximating the steepening solution, prompting the insertion of more kernel nodes. This behavior further highlights the adaptivity of our method.

We then test with regularization coefficient $\alpha = 10^{-3}, 10^{-4}$ and time stepsize $\Delta t = 0.1, 0.01$ (see Table 5). As is typical in backward Euler type methods, a smaller regularization parameter α is preferred to obtain a more accurate solve of each u^n, preventing significant error accumulation between time steps.

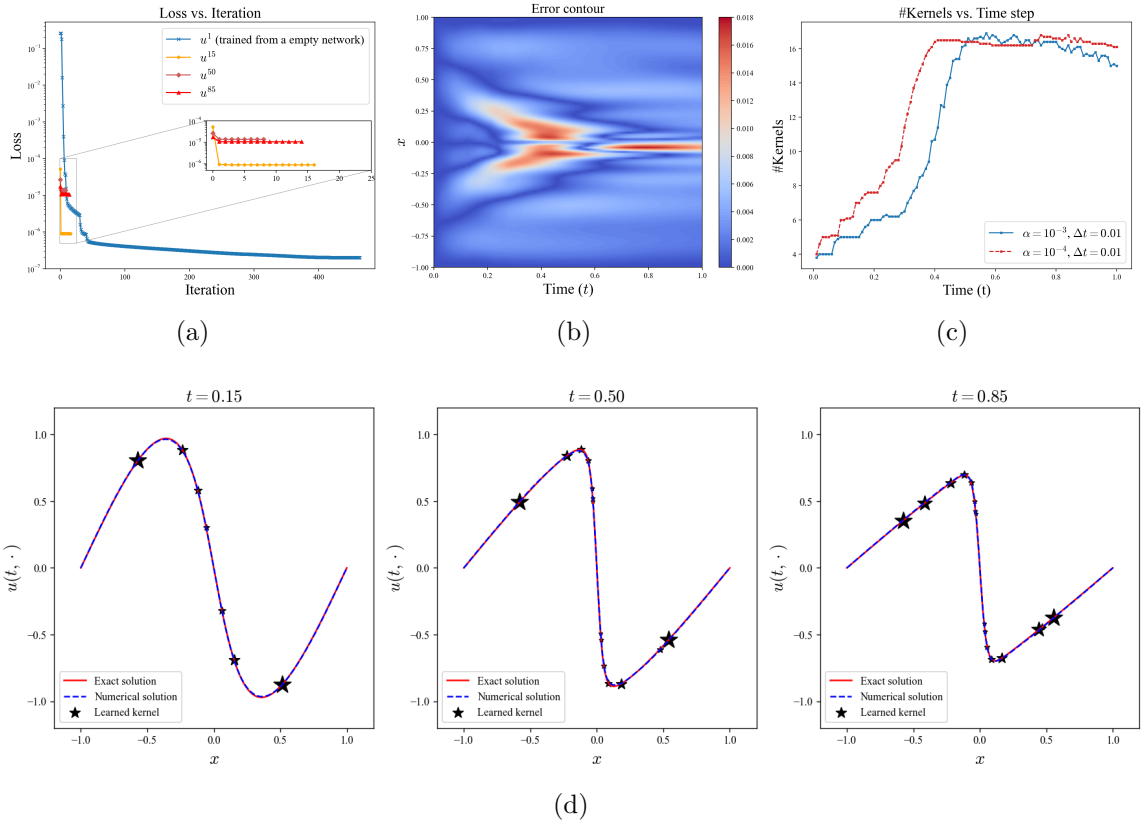


Figure 6: Viscous Burgers equation solved by Sparse RBFNet coupled with fully implicit time discretization. (a) Convergence of loss for selected u^n (regularization parameter $\alpha = 10^{-4}$); (b) Error contour (regularization parameter $\alpha = 10^{-4}$); (c) Growth of number of kernels in the network during time steps ($\Delta t = 0.01$) for regularization parameter $\alpha = 10^{-3}, 10^{-4}$ (results averaged from 10 runs); (d) Time slices of exact and true solution at time $t = 0.15, 0.50, 0.85$ (regularization parameter $\alpha = 10^{-4}$). The size of $\star$ is proportional to the bandwidth of an individual kernel.

We briefly discuss the expected behavior in the small-viscosity regime. As the viscosity (set to 0.02 in the above experiment) decreases, the solution develops a sharper shock wave, which requires more kernels to resolve the steep transition. The increased number of kernels typically leads to a more ill-conditioned Gauss–Newton system and a more challenging optimization, resulting in degraded performance. Moreover, since we employ a classical time discretization, smaller viscosity generally necessitates a finer time step, as is common in convection-dominated problems. In this regime, refined time discretization schemes (e.g., higher-order backward differentiation or adaptive stepping) could also be incorporated to further improve accuracy (Chen et al., 2025). Moreover, we observe that the small-viscosity regime requires a substantially larger number of collocation points (K) to resolve the sharper shock, analogous to the reduced mesh size requirement in classical numerical methods (see Appendix D). These observations suggest that, despite being mesh-free, the method still

Table 5: Errors and number of kernels for the viscous Burgers' equation with different combinations of time step Δt and regularization parameter α. Numerical solution $\{u^n\}$ is extended to the entire D by linear interpolation. Test results are evaluated on a grid of 120×120 in D. The empirical residual loss $\hat{L}$ is not accessible because some test points do not lie on the discretized time steps. The number of kernels shown in the table is first averaged between time steps. All results are averaged over 10 runs.

Δt	0.10		0.01	
α	10^{-3}	10^{-4}	10^{-3}	10^{-4}
L^2 error	9.79e−2	5.42e−2	1.16e−2	6.03e−3
L^∞ error	3.39e−1	1.73e−1	5.55e−2	1.93e−2
#Kernels	9	17	15	16

exhibits a resolution constraint when approximating highly singular solutions. While valid, these points are beyond the scope of the present work and are left for future investigation.

5. Conclusion and discussion

We have developed a sparse kernel/RBF network method for solving PDEs, in which the number of kernels, centers, and the kernel shape parameter are not prescribed but are instead treated as part of the optimization. Sparsity is promoted via a regularization term, added to the customary weighted quadratic L^2 loss on both interior and boundary. Theoretically, we extend the discrete network structure to a continuous integral neural network formulation, which constitutes a Reproducing Kernel Banach Space (RKBS). We establish the existence of a minimizer in the RKBS for the regularized optimization problem. Under additional assumptions, the minimizer has proven convergence properties towards the true PDE solution when the number of collocation points $K \to \infty$ and the regularization parameter $\alpha \to 0$. Notably, we prove a representer theorem that guarantees the existence of a minimizer expressible as a finite combination of feature functions (i.e., Gaussian RBFs) with a bounded number of terms. Computationally, we develop Algorithm 1 which effectively integrates the network width N into the optimization process through an iterative three-phase framework that alternates between optimization $\omega = (y, \sigma)$ via a Gauss-Newton method and nodes insertion/deletion. Here, the insertion is guided by the dual variable derived from the first-order optimality condition. This dual variable, defined for each candidate kernel node ω, serves as an effective estimate of potential reduction in the loss function upon insertion into the network. We validate the proposed algorithm on a semilinear equation, an Eikonal equation, and a viscous Burgers' equation, demonstrating its ability to produce sparse yet accurate solutions across a range of problem settings. Furthermore, the regularization term not only promotes sparsity in the learned solution but also plays a crucial role in preventing overfitting. Overall, our method lays a promising foundation for the development of flexible, scalable, efficient, and theoretically grounded neural network solvers capable of handling a wide range of PDEs with complex solution behaviors.

We note that the analysis of existence and the representer theorem for minimizers is conducted within the RKBS $\mathcal{V}$ associated to the kernel/feature function, and therefore depends critically on its characterization. In the subsequent work Shao et al. (2026), we obtained a precise characterization of $\mathcal{V}$ as Besov space $B^s_{1,1}$. This characterization applies to a broad class of kernels beyond Gaussian RBFs (e.g., Matérn kernels). In addition, anisotropic kernels, where the shape parameter is given by a matrix rather than a scalar, offer more degrees of freedom to capture directional features in PDE solutions, making them a promising choice for achieving sparser representations. Moreover, current convergence analysis is based on the interplay between solution space $\mathcal{U}$, source function space $\mathcal{F}$, and L^2 space induced from the loss/objective function. To better reflect the underlying problem structure and thereby improve error analysis, it would be beneficial to develop PDE-specific loss functions that align more closely with the regularity or other properties of $\mathcal{U}$ and $\mathcal{F}$. Additionally, our theoretical estimate on the number of terms in the representer theorem is likely pessimistic; in practice, the sparse minimization yields significantly fewer active feature functions. Understanding this gap theoretically is an interesting direction.

Further improvement of the computational algorithm can be achieved as well through the usage of kernels and loss functions that better capture features and regularity of both the solution and source function data. For instance, the effectiveness of a well-chosen loss function is exemplified in Section 4.1.3 in which $H^1(\partial D)$ is used in place of the usual $L^2(\partial D)$ for boundary constraints. Additionally, as the computational cost of our method scales with the number of collocation points K, number of kernel functions N, and problem dimension d, it is natural to consider a doubly adaptive sub-sampling framework, where only a subset of the kernel parameters is updated with only a mini-batch of the collocation points at each iteration. While the use of mini-batches of training data is standard in the machine learning literature, incorporating it with partial kernel weights update (usually known as block coordinate descent method (Wright, 2015; Xu and Yin, 2013)) requires further theoretical justification and algorithmic design. Nevertheless, this strategy can significantly improve the computational efficiency of the algorithm, particularly in cases where large K and N are required to capture complex solution geometries, which is common in high-dimensional PDE problems. In Shao et al. (2026), we have also extended the framework to equations involving the fractional Laplacian, providing a preliminary demonstration of its applicability to nonlocal problems (Burkardt et al., 2021; D'Elia et al., 2020; Guo et al., 2022). Several components of that extension, including sampling strategies and accurate implementation of boundary conditions, remain to be further developed. Extensions to inverse problems and to broader classes of nonlocal and fractional equations therefore continue to be promising directions.

Acknowledgments

Z. Shao and X. Tian were supported in part by NSF DMS-2240180 and the Alfred P. Sloan Fellowship. This manuscript has been co-authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or

The authors thank the anonymous referees for their helpful suggestions.

Appendix A. Optimization Algorithm

We provide additional implementation details for Algorithm 1.

A.1 Phase I

As mentioned earlier, we introduce several optional heuristic modifications to the insertion step.

A.1.1 Insertion threshold

While the threshold for insertion, which is the dual variable, only characterizes the descent magnitude by optimizing c, we couple it with $\nabla_{y_n} L$. The threshold of insertion (right-hand side of (3.2)) is then

$$\max_{1\le n\le N}\{|p[u](\omega_n)| + \eta\|\nabla_{y_n} L(u)\|_2\}$$

The additional term is based on the fact that we also gain descent by updating y_n of preexisting nodes. In practice, this largely alleviates the possibility of having a cluster of nodes at a certain location (since the candidate nodes neighboring to preexisting ones are very likely to be inserted since their dual variables are high due to the algorithm). The choice of η is not critical and is typically fixed to be 0.01. For the Burgers' equation experiment in Section , however, we set $\eta = 0.1$ as we observed clustering of nodes otherwise.

A.1.2 Loosening insertion criterion

As mentioned earlier, the scheme of inserting new feature functions is essentially greedy. To accelerate convergence, we may need to relax the criteria for nodes insertion, especially during early iterations in which the learned solution is not close to the true solution. One option is to add a shrinking coefficient to the threshold, but tuning this coefficient is often nontrivial. We adopt a Metropolis-Hasting style criterion, in which a candidate node ω_{N+1} is accepted with probability

$$\Pr(\omega_{N+1}) = \min\{1,\ \exp(-\frac{1}{T\cdot \hat{R}[u]}(\max_{1\le n\le N}|p[u](\omega_n)| - |p[u](\omega_{N+1})|))\} \tag{A.1}$$

where T is a positive constant and $\hat{R}[u]$ is the relative error serving as an annealing term. This scheme admits more plausible points when the current approximation is far from the solution. In practice, we choose $T \propto -\log\alpha$ as a smaller α usually leads to more nodes in the end.

A.2 Phase II

A.2.1 Levenberg–Marquardt type correction term for invertability

To ensure invertability DG, we add the following correction term to approximated Hessian $\nabla^2\hat{\ell}$.

$$\mathrm{Cor} = 0.1\times\|WR\|_1\,\mathrm{diag}\{\underbrace{\epsilon,\dots,\epsilon}_{N},\underbrace{|c_1|,\dots,|c_1|}_{d+1},\underbrace{|c_2|,\dots,|c_2|}_{d+1},\dots,\underbrace{|c_N|,\dots,|c_N|}_{d+1}\}\in\mathbb{R}^{N(d+2)\times N(d+2)}$$

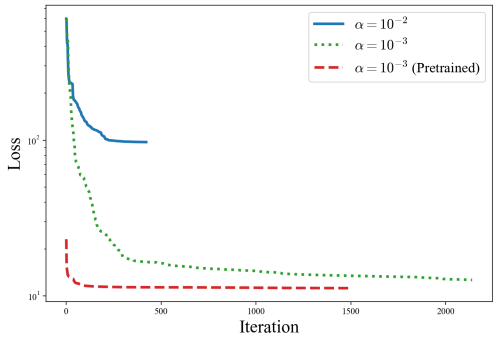


Figure 7: Convergence of residual loss for $\alpha = 10^{-3}$ with/without pretraining.

and

$$\nabla^2 \hat{\ell} \approx J_R W J_R + \text{Cor}$$

A.2.2 Line search

As is customary in Gauss-Newton algorithm, we do a line search of optimal step size. We shrink step size θ, by $2/3$ each time, starting from $\theta = 1$, until the actual descent is within trust-region, which is set to be

$$\text{Actual Descent} \leq h \times \text{Estimated Descent} = h\theta G^T(\text{DP}\, z).$$

where $h \in [1/5, 1/3]$. In all numerical experiments in Section 4, we set $h = 1/5$.

A.3 Stopping Criterion

We stop the iterations if

$$\text{Estimated Descent} < \epsilon \text{ and } |p[u](\omega_{\omega_{N+1}})| - \alpha < \epsilon,$$

which represent both insertion of new nodes and optimization of existing nodes cannot incur obvious descent to the loss function. ϵ is usually taken between $[10^{-5}, 10^{-3}]$.

A.4 Pretraining

As mentioned earlier, we may initialize training using solutions obtained with a larger regularization parameter α. This strategy often reduces computational cost and can occasionally yield higher-quality numerical solutions. For example, when solving the semilinear Poisson equation in Section 4.1 with $K = 300$ and $\alpha = 10^{-3}$, we initialize the network using the solution computed with $\alpha = 10^{-2}$. In this case, such pretraining significantly reduces the overall computational cost (even when accounting for the training effort at $\alpha = 10^{-2}$) and achieves a smaller residual (see Figure 7). This pretraining technique is of considerable practical importance and warrants further theoretical justification and algorithm development.

Table 6: Errors, residual loss, and number of kernels for Sparse RBFNet and GP using **randomly** generated collocation points (in comparison to Table 6). Case with $\alpha = 10^{-3}(10^{-4})$ is solved with solution obtained by $\alpha = 10^{-2}(10^{-3})$ as initialization. Test results are evaluated on a 100×100 grid in D. All results are averaged over 10 runs. For the Gaussian Process method, we retain the data even if some runs diverge due to random sampling.

		Sparse RBFNet			**Gaussian Process**		
K (K_1, K_2)	Metric	$\alpha = 1\text{e}{-2}$	$\alpha = 1\text{e}{-3}$	$\alpha = 1\text{e}{-4}$	$\sigma = 0.05$	$\sigma = 0.10$	$\sigma = 0.15$
400 $(324, 76)$	L^2 error	7.16e−1	7.30e−1	7.30e−1	5.20e−1	4.79e−1	7.62e−1
	L^∞ error	1.51e+0	1.55e+0	1.54e+0	2.03e+0	2.11e+0	3.96e+0
	$\hat{L}$(train)	4.75e+0	9.65e−2	2.80e−3	–	–	–
	$\hat{L}$(test)	2.63e+2	2.56e+2	2.55e+2	–	–	–
	#Kernels	46	75	87	400	400	400
900 $(784, 116)$	L^2 error	2.91e−1	2.90e−1	2.78e−1	5.27e−1	3.63e−1	8.15e−1
	L^∞ error	6.77e−1	6.84e−1	6.61e−1	2.08e+0	1.55e+0	5.42e+0
	$\hat{L}$(train)	1.16e+1	2.64e−1	7.25e−3	–	–	–
	$\hat{L}$(test)	9.65e+1	7.84e+1	7.52e+1	–	–	–
	#Kernels	58	108	135	900	900	900
2500 $(2304, 196)$	L^2 error	8.89e−2	4.73e−2	3.88e−2	4.97e−1	3.54e−1	4.02e−2
	L^∞ error	2.30e−1	1.34e−1	1.17e−1	2.34e+0	4.76e+0	8.73e−1
	$\hat{L}$(train)	1.82e+1	6.82e−1	3.49e−2	–	–	–
	$\hat{L}$(test)	3.48e+1	7.13e+0	3.93e+0	–	–	–
	#Kernels	68	117	155	2500	2500	2500

Appendix B. Choice of collocation points

We observe that when randomly generated collocation points are used, the performance of our method degrades. Here we include the results for the semilinear Poisson equation in Section 4.1, also with comparison with the Gaussian Process method (see Table 6).

There are several possible reasons for this decay in accuracy

1. In this particular case, both our method and the Gaussian Process method have a significant drop in performance. This is likely due to the sampled collocation points failing to adequately capture the sharp transitions in the exact solution.

2. With random points that cluster, overfitting might occur more rapidly at different parts of the domain, which may lead to non-useful kernel points being added in the insertion process.

In practice, it is crucial for a method to perform reliably when using randomly selected collocation points. However, achieving comparable coverage to uniform or quasi-uniform grids typically requires a substantially larger number of random points as observed in various numerical contexts. This underscores an important direction for future research on improving robustness and efficiency under random sampling.

Appendix C. Besov space characterization

In this section, we characterize the RKBS norm in terms of a classical function space norm.

Theorem 28 *Consider the unrestricted dictionary* $\Omega = \mathbb{R}^d \times (0, \sigma_{\max}]$ *with* $\sigma_{\max} = 1$. *Then, for the corresponding space*

$$\mathcal{V} = \mathcal{V}(\mathbb{R}^d) = \left\{ u(x) = \int_\Omega \varphi(x;\omega)\,\mathrm{d}\mu(\omega) \;\middle|\; \mu \in M(\Omega) \right\}$$

there is a continuous embedding of the Besov space $B^s_{1,1}(\mathbb{R}^d)$ *(as defined in, e.g., Sawano, 2018, Section 2.1.1) into* $\mathcal{V}$.

Proof Let $u \in B^s_{1,1}(\mathbb{R}^d)$. Thus, it can be decomposed as

$$u = u_0 + \sum_{j=1}^\infty u_j \text{ with } u_0 = \psi(\mathscr{D})u, u_j = \phi_j(\mathscr{D})u \text{ and } \|f\|_{B^s_{1,1}} = \|u_0\|_{L^1} + \sum_{j=1}^\infty 2^{j\,s}\|u_j\|_{L^1} < \infty;$$

(see Sawano, 2018, Section 2.1.1). Here, $\psi\colon \mathbb{R}^d \to [0,1]$ is a function in $C^\infty_c(\mathbb{R}^d)$ with

$$\chi_{B_1(0)}(\xi) \le \psi(\xi) \le \chi_{B_2(0)}(\xi)$$

and $\phi_j = \psi_j - \psi_{j-1}$, where $\psi_j(\xi) = \psi(2^{-j}\xi)$. Moreover, for any such function ψ (or ϕ_j) the symbol $\psi(\mathscr{D})$ denotes the multiplication in Fourier space $\psi(\mathscr{D})u = \mathcal{F}^{-1}[\psi \cdot \mathcal{F}[u]]$, where $(\psi \cdot \mathcal{F}[u])(\xi) = \psi(\xi)\mathcal{F}[u](\xi)$.

Next, we show that every u_j can be represented as a convolution with a Gaussian $\hat\varphi_j(x) = 2^{jd}\hat\varphi(2^j x)$ with scale $\sigma_j = 2^{-j}$ in the form

$$u_j = \hat\varphi_j * \mu_j \quad \text{with } \mu_j \in L^1(\mathbb{R}^d). \tag{C.1}$$

We argue this by defining the Fourier transform of $\hat\varphi_j$ as $g_j = \mathcal{F}\hat\varphi_j$ with $g_j(\xi) = g_0(2^{-j}\xi)$ and

$$\mu_j = [\psi_{j+1}/g_j](\mathscr{D})u_j.$$

The multiplier ψ_{j+1}/g_j is C^∞ and compactly supported, and thus its inverse Fourier transform $\varphi_j^{-1} = \mathcal{F}^{-1}(\psi_{j+1}/g_j)$ is in $L^1(\mathbb{R}^d)$. Due to $\varphi_j^{-1}(x) = 2^{jd}\varphi_j^{-1}(2^j x)$ the L^1 norm is independent of j and equal to a constant c, which implies

$$\|\mu_j\|_{L^1} = \|[\psi_{j+1}/g_j](\mathscr{D})u_j\|_{L^1} = \|\varphi_j^{-1} * u_j\|_{L^1} \le c\|u_j\|_{L^1}.$$

Moreover, due to $g_j\psi_{j+1}/g_j * \phi_j = \phi_j$ and $g_0\psi_1/g_0 * \psi = \psi$, we have (C.1) as claimed. Now, we define the measure

$$\mathrm{d}\mu(y,\sigma) = \sum_{j=0}^\infty \sigma_j^{-s}\delta_{\sigma_j}(\sigma)\mu_j(y)\,\mathrm{d}y = \sum_{j=0}^\infty 2^{j\,s}\delta_{2^{-j}}(\sigma)\mu_j(y)\,\mathrm{d}y.$$

We can directly verify that $u = \mathcal{N}\mu$ and

$$\|u\|_{\mathcal{V}(\mathbb{R}^d)} \le \|\mu\|_{M(\Omega)} = \sum_{j=0}^\infty 2^{j\,s}\|\mu_j\|_{L^1} \le c\sum_{j=0}^\infty 2^{j\,s}\|u_j\|_{L^1} = c\|u\|_{B^s_{1,1}(\mathbb{R}^d)}.$$

■

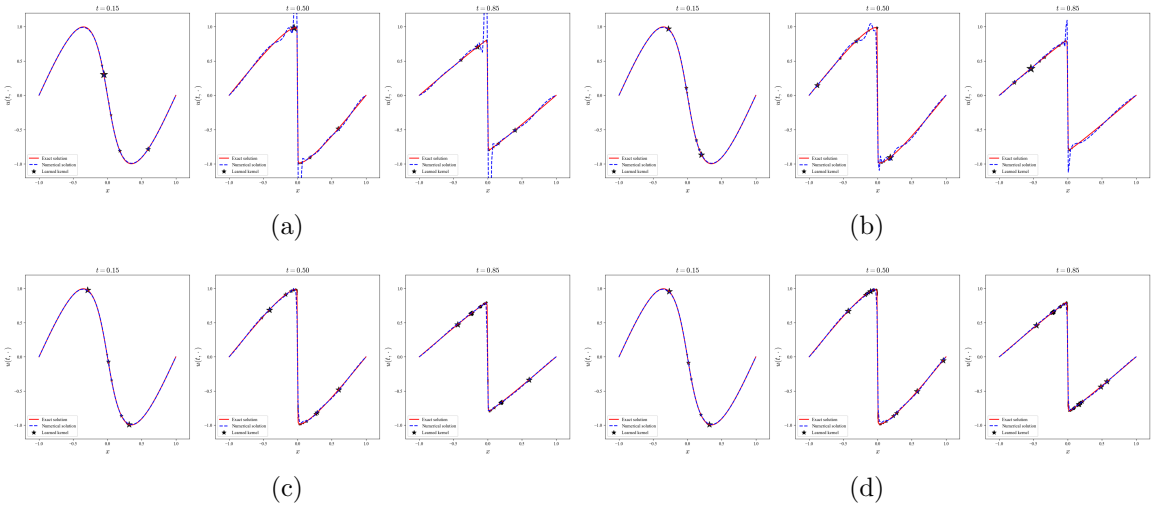


Figure 8: Small-viscosity Burgers equation ($\nu = 0.002$): comparison of solution slices with different numbers of collocation points (K). All results are obtained using time step $\Delta t = 0.01$ and regularization parameter $\alpha = 10^{-4}$. (a) $K = 40$; (b) $K = 80$; (c) $K = 200$; (d) $K = 400$. The size of each $\star$ indicates the bandwidth of the corresponding kernel.

Appendix D. Burgers equation with small viscosity

We briefly discuss the behavior of the proposed method for the viscous Burgers equation in a more challenging small-viscosity regime. We consider the same problem as in Section 4.3 with a general viscosity parameter ν:

$$\begin{aligned} \partial_t u + u\partial_x u - \nu \partial_x^2 u = 0, \ \forall (t,x) \in (0,1] \times (-1,1) \\ u(0,x) = -\sin(\pi x) \\ u(t,-1) = u(t,1) = 0. \end{aligned}$$

In Section 4.3, the viscosity is set to $\nu = 0.02$. As ν decreases, the solution develops a much steeper shock, making the approximation significantly more challenging. We solve the equation with $\nu = 0.002$ using the same time step $\Delta t = 0.01$ and regularization parameter $\alpha = 10^{-4}$. All other settings remain the same as in Section 4.3. Figure 8 compares results with different numbers of collocation points $K = 40, 80, 200, 400$. We observe that, as the viscosity decreases, substantially more collocation points are required to resolve the sharper shock. In particular, the solution improves significantly when increasing K from 40 to 200, while the additional gain from $K = 200$ to $K = 400$ is relatively modest, indicating that a sufficient spatial "resolution" has already been reached.

References

Mark Ainsworth and Justin Dong. Extended galerkin neural network approximation of singular variational problems with error control. *SIAM Journal on Scientific Computing*, 47(3):C738–C768, 2025. doi: 10.1137/24M1658279.

Francis Bach. Breaking the curse of dimensionality with convex neural networks. *J. Mach. Learn. Res.*, 18:53, 2017. ISSN 1532-4435. Id/No 19.

Francis Bach. On the relationship between multivariate splines and infinitely-wide neural networks, 2023. URL `https://doi.org/10.48550/arXiv.2302.03459`.

Jinshuai Bai, Gui-Rong Liu, Ashish Gupta, Laith Alzubaidi, Xi-Qiao Feng, and YuanTong Gu. Physics-informed radial basis network (PIRBN): A local approximating neural network for solving nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 415:116290, 2023.

Francesca Bartolucci, Ernesto De Vito, Lorenzo Rosasco, and Stefano Vigogna. Understanding neural networks with reproducing kernel Banach spaces. *Applied and Computational Harmonic Analysis*, 62:194–236, 2023.

Pau Batlle, Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M Stuart. Error analysis of kernel/GP methods for nonlinear and parametric PDEs. *Journal of Computational Physics*, 520:113488, 2025.

Ted Belytschko, Yun Yun Lu, and Lei Gu. Element-free Galerkin methods. *International journal for numerical methods in engineering*, 37(2):229–256, 1994.

Yoshua Bengio, Nicolas Roux, Pascal Vincent, Olivier Delalleau, and Patrice Marcotte. Convex neural networks. *Advances in neural information processing systems*, 18, 2005.

Andrea Bonito, Ronald DeVore, Guergana Petrova, and Jonathan W. Siegel. Convergence and error control of consistent PINNs for elliptic PDEs, 2025. URL `https://arxiv.org/abs/2406.09217`.

Claire Boyer, Antonin Chambolle, Yohann De Castro, Vincent Duval, Frédéric De Gournay, and Pierre Weiss. On representer theorems and convex regularization. *SIAM Journal on Optimization*, 29(2):1260–1281, 2019.

Kristian Bredies and Marcello Carioni. Sparsity of solutions for variational inverse problems with finite-dimensional data. *Calculus of Variations and Partial Differential Equations*, 59(1):14, 2020.

Kristian Bredies, Marcello Carioni, Silvio Fanzon, and Daniel Walter. Asymptotic linear convergence of fully-corrective generalized conditional gradient methods. *Math. Program.*, 205(1):135–202, 2024. doi: 10.1007/S10107-023-01975-Z.

Haim Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*, volume 2. Springer, 2011.

Martin Dietrich Buhmann. *Radial Basis Functions*, volume 9. Cambridge university press, 2000.

John Burkardt, Yixuan Wu, and Yanzhi Zhang. A unified meshfree pseudospectral method for solving both classical and fractional PDEs. *SIAM Journal on Scientific Computing*, 43(2):A1389–A1411, 2021.

Luis A Caffarelli and Xavier Cabré. *Fully Nonlinear Elliptic Equations*, volume 43. American Mathematical Soc., 1995.

Zhiqiang Cai, Jingshuang Chen, and Min Liu. Self-adaptive deep neural network: Numerical approximation to functions and PDEs. *Journal of Computational Physics*, 455:111021, 2022.

Yifan Chen. NonLinPdes-GPsolver: Gaussian Process Solver for Nonlinear PDEs. Available at: https://github.com/yifanc96/NonLinPDEs-GPsolver, 2024. Accessed: 2025-03-01.

Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M Stuart. Solving and learning nonlinear PDEs with Gaussian processes. *Journal of Computational Physics*, 447:110668, 2021.

Yifan Chen, Houman Owhadi, and Florian Schäfer. Sparse Cholesky factorization for solving nonlinear PDEs via Gaussian processes. *Mathematics of Computation*, 94(353):1235–1280, 2025.

Lenaic Chizat. Sparse optimization on measures with over-parameterized gradient descent. *Mathematical Programming*, 194(1):487–532, 2022.

Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018.

John B Conway. *A course in functional analysis*, volume 96. Springer Science & Business Media, 1994.

Constantine M Dafermos. *Hyperbolic conservation laws in continuum physics*, volume 3. Springer, 2005.

Marta D'Elia, Qiang Du, Christian Glusa, Max Gunzburger, Xiaochuan Tian, and Zhi Zhou. Numerical methods for nonlocal and fractional models. *Acta Numerica*, 29:1–124, 2020.

Lester E Dubins. On extreme points of convex sets. *Journal of Mathematical Analysis and Applications*, 5(2):237–244, 1962.

Weinan E and Bing Yu. The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1): 1–12, 2018.

Weinan E, Chao Ma, and Lei Wu. The Barron space and the flow-induced function spaces for neural network models. *Constructive Approximation*, 55(1):369–406, 2022.

Lawrence C Evans. *Partial Differential Equations*, volume 19. American Mathematical Society, 2022.

Gregory E Fasshauer. Solving partial differential equations by collocation with radial basis functions. In *Proceedings of Chamonix*, volume 1997, pages 1–8. Citeseer, 1996.

Axel Flinth, Frédéric de Gournay, and Pierre Weiss. On the linear convergence rates of exchange and continuous methods for total variation minimization. *Mathematical Programming*, 190:221–257, 2021. doi: 10.1007/s10107-020-01530-0.

Carsten Franke and Robert Schaback. Solving partial differential equations by collocation using radial basis functions. *Applied Mathematics and Computation*, 93(1):73–82, 1998.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

David Gilbarg and Neil S Trudinger. *Elliptic Partial Differential Equations of Second Order*, volume 224. Springer, 1977.

Robert A Gingold and Joseph J Monaghan. Smoothed particle hydrodynamics: Theory and application to non-spherical stars. *Monthly notices of the royal astronomical society*, 181(3):375–389, 1977.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Ling Guo, Hao Wu, Xiaochen Yu, and Tao Zhou. Monte Carlo fPINNs: Deep learning method for forward and inverse problems involving high dimensional fractional partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 400: 115523, 2022.

Victor Klee. On a theorem of Dubins. *Journal of Mathematical Analysis and Applications*, 7(3):425–427, 1963.

Jason M Klusowski and Andrew R Barron. Approximation by combinations of ReLU and squared ReLU ridge functions with ℓ^1 and ℓ^0 controls. *IEEE Transactions on Information Theory*, 64(12):7649–7656, 2018.

Akash Kumar, Mikhail Belkin, and Parthe Pandit. Mirror descent on reproducing kernel banach spaces. *arXiv preprint arXiv:2411.11242*, 2024.

Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE transactions on neural networks*, 9(5):987–1000, 1998.

Shaofan Li and Wing Kam Liu. *Meshfree Particle Methods*. Springer Science & Business Media, 2007.

Rong Rong Lin, Hai Zhang Zhang, and Jun Zhang. On reproducing kernel Banach spaces: Generic definitions and unified framework of constructions. *Acta Mathematica Sinica, English Series*, 38(8):1459–1483, 2022.

Min Liu and Zhiqiang Cai. Adaptive two-layer ReLU neural network: II. Ritz approximation to elliptic PDEs. *Computers & Mathematics with Applications*, 113:103–116, 2022.

Wing Kam Liu, Y Chen, S Jun, JS Chen, T Belytschko, C Pan, RA Uras, and CT Chang. Overview and applications of the reproducing kernel particle methods. *Archives of Computational Methods in Engineering*, 3(1):3–80, 1996.

Leon B Lucy. A numerical approach to the testing of the fission hypothesis. *Astronomical Journal, vol. 82, Dec. 1977, p. 1013-1024.*, 82:1013–1024, 1977.

Tao Luo and Haizhao Yang. Chapter 11 - two-layer neural networks for partial differential equations: optimization and generalization theory. In Siddhartha Mishra and Alex Townsend, editors, *Numerical Analysis Meets Machine Learning*, volume 25 of *Handbook of Numerical Analysis*, pages 515–554. Elsevier, 2024. doi: https://doi.org/10.1016/bs.hna.2024.05.007. URL `https://www.sciencedirect.com/science/article/pii/S1570865924000073`.

Yves Meyer. *Wavelets and operators*. Number 37. Cambridge university press, 1992.

Deanna Needell and Joel A Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and computational harmonic analysis*, 26(3):301–321, 2009.

Wenqing Ouyang and Andre Milzarek. A trust region-type normal map-based semismooth newton method for nonsmooth nonconvex composite optimization. *Mathematical Programming*, 212(1):389–435, 2025.

Rahul Parhi and Robert D Nowak. Banach space representer theorems for neural networks and ridge splines. *Journal of Machine Learning Research*, 22(43):1–40, 2021.

Rahul Parhi and Robert D Nowak. Near-minimax optimal estimation with shallow ReLU neural networks. *IEEE Transactions on Information Theory*, 69(2):1125–1140, 2022.

Rahul Parhi and Michael Unser. Function-space optimality of neural architectures with multivariate nonlinearities. *SIAM Journal on Mathematics of Data Science*, 7(1):110–135, 2025.

Konstantin Pieper. *Finite Element Discretization and Efficient Numerical Solution of Elliptic and Parabolic Sparse Control Problems*. PhD thesis, Technische Universität München, 2015.

Konstantin Pieper and Armenak Petrosyan. Nonconvex regularization for sparse neural networks. *Applied and Computational Harmonic Analysis*, 61:25–56, 2022.

Konstantin Pieper and Daniel Walter. Linear convergence of accelerated conditional gradient algorithms in spaces of measures. *ESAIM: COCV*, 27:38, 2021. doi: 10.1051/cocv/2021042.

Konstantin Pieper, Zezhong Zhang, and Guannan Zhang. Nonuniform random feature models using derivative information. *arXiv preprint arXiv:2410.02132*, 2024.

Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: replacing minimization with randomization in learning. In *Proceedings of the 21st International Conference on Neural Information Processing Systems*, NIPS'08, page 1313–1320, Red Hook, NY, USA, 2008. Curran Associates Inc. ISBN 9781605609492.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Amuthan A Ramabathiran and Prabhu Ramachandran. SPINN: sparse, physics-based, and partially interpretable neural networks for PDEs. *Journal of Computational Physics*, 445: 110600, 2021.

Stephen M Robinson. Normal maps induced by linear transformations. *Mathematics of Operations Research*, 17(3):691–714, 1992.

Saharon Rosset, Grzegorz Swirszcz, Nathan Srebro, and Ji Zhu. ℓ_1 regularization in infinite dimensional feature spaces. In *Learning Theory: 20th Annual Conference on Learning Theory, COLT 2007, San Diego, CA, USA; June 13-15, 2007. Proceedings 20*, pages 544–558. Springer, 2007.

Yoshihiro Sawano. *Theory of Besov spaces*, volume 56. Springer, 2018.

Robert Schaback and Holger Wendland. Kernel techniques: From machine learning to meshless methods. *Acta numerica*, 15:543–639, 2006.

Bernhard Scholkopf and Alexander J Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT press, 2018.

Zihan Shao, Konstantin Pieper, and Xiaochuan Tian. Sparse RBF networks for PDEs and nonlocal equations: function space theory, operator calculus, and training algorithms. *arXiv preprint arXiv:2601.17562*, 2026.

Yeonjong Shin, Zhongqiang Zhang, and George Em Karniadakis. Error estimates of residual minimization using neural networks for linear pdes. *Journal of Machine Learning for Modeling and Computing*, 4(4):73–101, 2023. ISSN 2689-3967.

Jonathan W Siegel and Jinchao Xu. Characterization of the variation spaces corresponding to shallow neural networks. *Constructive Approximation*, 57(3):1109–1132, 2023.

Jonathan W Siegel, Qingguo Hong, Xianlin Jin, Wenrui Hao, and Jinchao Xu. Greedy training algorithms for neural networks and applications to pdes. *Journal of Computational Physics*, 484:112084, 2023.

Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of computational physics*, 375:1339–1364, 2018.

Len Spek, Tjeerd Jan Heeringa, Felix Schwenninger, and Christoph Brune. Duality for neural networks through reproducing kernel Banach spaces. *Applied and Computational Harmonic Analysis*, 78:101765, 2025.

N. Sukumar and Ankit Srivastava. Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks. *Computer Methods in Applied Mechanics and Engineering*, 389:114333, 2022. ISSN 0045-7825. doi: https://

doi.org/10.1016/j.cma.2021.114333. URL https://www.sciencedirect.com/science/article/pii/S0045782521006186.

Michael E Taylor. *Partial differential equations III*, volume 2. Springer, 1996.

Quoc Tran-Dinh, Nhan Pham, and Lam Nguyen. Stochastic Gauss-Newton algorithms for nonconvex compositional optimization. In *International Conference on Machine Learning*, pages 9572–9582. PMLR, 2020.

Hans Triebel. *Theory of Function Spaces III*, volume 100 of *Monographs in Mathematics*. Birkhäuser Basel, 2006. ISBN 978-3-7643-7581-2. doi: 10.1007/3-7643-7582-5.

Michael Ulbrich. *Semismooth Newton Methods for Variational Inequalities and Constrained Optimization Problems in Function Spaces*. MOS-SIAM Series on Optimization. SIAM, 2011. ISBN 9781611970685.

Gerd Wachsmuth and Daniel Walter. No-gap second-order conditions for minimization problems in spaces of measures. *arXiv preprint arXiv:2403.12001*, 2024.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022.

Zhiwen Wang, Minxin Chen, and Jingrun Chen. Solving multiscale elliptic problems by sparse radial basis function neural networks. *Journal of Computational Physics*, 492: 112452, 2023.

Holger Wendland. *Scattered Data Approximation*, volume 17. Cambridge university press, 2004.

Stephen J Wright. Numerical optimization, 2006.

Stephen J Wright. Coordinate descent algorithms. *Mathematical programming*, 151(1):3–34, 2015.

Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on imaging sciences*, 6(3):1758–1789, 2013.

Xianjin Yang and Houman Owhadi. A mini-batch method for solving nonlinear PDEs with Gaussian processes. *arXiv preprint arXiv:2306.00307*, 2023.

Bing Yu et al. The deep ritz method: a deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

Yaohua Zang, Gang Bao, Xiaojing Ye, and Haomin Zhou. Weak adversarial networks for high-dimensional partial differential equations. *Journal of Computational Physics*, 411: 109409, 2020.

Haizhang Zhang, Yuesheng Xu, and Jun Zhang. Reproducing kernel Banach spaces for machine learning. *Journal of Machine Learning Research*, 10(12), 2009.

Zezhong Zhang, Feng Bao, Lili Ju, and Guannan Zhang. Transferable neural networks for partial differential equations. *Journal of Scientific Computing*, 99(1):2, 2024. doi: 10.1007/s10915-024-02463-y. URL https://doi.org/10.1007/s10915-024-02463-y.