

Kernel Methods for Rough PDEs

Ricardo Baptista¹ Edoardo Calvello¹ **Matthieu Darcy**¹ Houman Owhadi¹
Andrew M. Stuart¹ Xianjin Yang¹

¹Department of Computing and Mathematical Sciences
California Institute of Technology

SIAM UQ 2024



The problem

We want to **solve with machine learning rough PDEs** of the form

$$\begin{aligned}\mathcal{P}(u) &= \xi, & x \in \Omega, \\ u &= 0, & x \in \partial\Omega\end{aligned}\tag{RPDE}$$

- $\mathcal{P} : H_0^t(\Omega) \rightarrow H^{-s}$ is (non-linear) differential operator.

Canonical example: $\mathcal{P}(u) = \Delta u + f(u)$.

- $u^* \in H_0^t(\Omega)$ is the **solution** and $\xi \in H^{-s}$ is the **forcing term**.

The problem

We want to **solve with machine learning rough PDEs** of the form

$$\begin{aligned}\mathcal{P}(u) &= \xi, & x \in \Omega, \\ u &= 0, & x \in \partial\Omega\end{aligned}\tag{RPDE}$$

- $\mathcal{P} : H_0^t(\Omega) \rightarrow H^{-s}$ is (non-linear) differential operator.

Canonical example: $\mathcal{P}(u) = \Delta u + f(u)$.

- $u^* \in H_0^t(\Omega)$ is the **solution** and $\xi \in H^{-s}$ is the **forcing term**.

Hierarchy of spaces

$$\underbrace{H_0^t(\Omega)}_{\text{Functions with } t \text{ derivatives}} \subset L^2(\Omega) \subset \underbrace{H^{-s}}_{\text{Dual space of } H_0^s}$$

The problem

We want to **solve with machine learning rough PDEs** of the form

$$\begin{aligned}\mathcal{P}(u) &= \xi, & x \in \Omega, \\ u &= 0, & x \in \partial\Omega\end{aligned}\tag{RPDE}$$

- $\mathcal{P} : H_0^t(\Omega) \rightarrow H^{-s}$ is (non-linear) differential operator.

Canonical example: $\mathcal{P}(u) = \Delta u + f(u)$.

- $u^* \in H_0^t(\Omega)$ is the **solution** and $\xi \in H^{-s}$ is the **forcing term**.

The difficulty

Solving (RPDE) is difficult because the forcing term is rough $\xi \notin L^2$ and, as a result, the solution u^* is also irregular/rough.

The problem

We want to **solve with machine learning rough PDEs** of the form

$$\begin{aligned}\mathcal{P}(u) &= \xi, & x \in \Omega, \\ u &= 0, & x \in \partial\Omega\end{aligned}\tag{RPDE}$$

- $\mathcal{P} : H_0^t(\Omega) \rightarrow H^{-s}$ is (non-linear) differential operator.

Canonical example: $\mathcal{P}(u) = \Delta u + f(u)$.

- $u^* \in H_0^t(\Omega)$ is the **solution** and $\xi \in H^{-s}$ is the **forcing term**.

The solution

We propose a kernel based method for solving rough PDEs using **negative Sobolev norms** and **weak measurements**, with **provable convergence**.

- 1 Introduction and motivation
- 2 Machine learning for smooth PDEs
- 3 Kernel methods for rough PDEs
- 4 Convergence results
- 5 Numerical results

Table of Contents

- 1 Introduction and motivation
- 2 Machine learning for smooth PDEs
- 3 Kernel methods for rough PDEs
- 4 Convergence results
- 5 Numerical results

Motivation: Stochastic Partial Differential Equations

Stochastic Partial Differential Equations (SPDEs) are PDEs that include **random fluctuations**. A general class of interest are semi-linear SPDEs with additive noise:

$$\frac{d}{dt}u(t, \mathbf{x}) = \Delta u(\mathbf{x}, t) + f(u(t, \mathbf{x})) + \xi(t, \mathbf{x}) \quad (\text{SL-SPDE})$$

where:

ξ is **space time white noise**.

Motivation: Stochastic Partial Differential Equations

Stochastic Partial Differential Equations (SPDEs) are PDEs that include **random fluctuations**. A general class of interest are semi-linear SPDEs with additive noise:

$$\frac{d}{dt}u(t, \mathbf{x}) = \Delta u(\mathbf{x}, t) + f(u(t, \mathbf{x})) + \xi(t, \mathbf{x}) \quad (\text{SL-SPDE})$$

where:

ξ is **space time white noise**.

Examples

- Phase field models (Allen-Cahn equation).
- Mathematical biology (Nagumo equation).
- Filtering and sampling.

Motivation: Solving SPDEs

Numerically solving an SPDE of the form

$$\frac{d}{dt}u(t, \mathbf{x}) = \Delta u(\mathbf{x}, t) + f(u(t, \mathbf{x})) + \xi(t, \mathbf{x}) \quad (\text{SL-SPDE})$$

typically involves drawing samples of ξ and solving (SL-SPDE) for that realization.

Motivation: Solving SPDEs

Numerically solving an SPDE of the form

$$\frac{d}{dt}u(t, \mathbf{x}) = \Delta u(\mathbf{x}, t) + f(u(t, \mathbf{x})) + \xi(t, \mathbf{x}) \quad (\text{SL-SPDE})$$

typically involves drawing samples of ξ and solving (SL-SPDE) for that realization.

Difficulty

Solving (SL-SPDE) is difficult because

- 1 The forcing term $\xi \notin L^2$ a.s. and is not pointwise defined.
- 2 The solution u^* is irregular/rough.

This motivates us to develop methods to solve PDEs with very rough forcing terms and/or irregular solutions.

Table of Contents

- 1 Introduction and motivation
- 2 Machine learning for smooth PDEs**
- 3 Kernel methods for rough PDEs
- 4 Convergence results
- 5 Numerical results

Solving PDEs with ML

Machine learning methods are a novel way of solving PDEs in a *data-driven* way

$$u^\dagger := \arg \min_{u \in \mathcal{S}} \underbrace{\gamma_1 \|\mathcal{P}(u) - \xi\|_{L^2(\Omega)}^2}_{\text{PDE data}} + \underbrace{\gamma_2 \|u\|_{L^2(\partial\Omega)}^2}_{\text{Boundary data}} + \underbrace{\gamma_3 \mathcal{R}(u)}_{\text{Regularization}} \quad \text{Infinite data}$$

Solving PDEs with ML

Machine learning methods are a novel way of solving PDEs in a *data-driven* way

$$u^\dagger := \arg \min_{u \in \mathcal{S}} \underbrace{\gamma_1 \|\mathcal{P}(u) - \xi\|_{L^2(\Omega)}^2}_{\text{PDE data}} + \underbrace{\gamma_2 \|u\|_{L^2(\partial\Omega)}^2}_{\text{Boundary data}} + \underbrace{\gamma_3 \mathcal{R}(u)}_{\text{Regularization}} \quad \text{Infinite data}$$

$$\approx \arg \min_{u \in \mathcal{S}} \underbrace{\frac{\gamma_1}{N_\Omega} \sum_{i=1}^{N_\Omega} |\mathcal{P}(u)(x_i) - \xi(x_i)|^2}_{\text{PDE data approximation}} + \underbrace{\frac{\gamma_2}{N_{\partial\Omega}} \sum_{j=1}^{N_{\partial\Omega}} |u(x_j)|^2}_{\text{Boundary approximation}} + \gamma_3 \mathcal{R}(u) \quad \text{Finite data}$$

Solving PDEs with ML

Machine learning methods are a novel way of solving PDEs in a *data-driven* way

$$u^\dagger := \arg \min_{u \in \mathcal{S}} \underbrace{\gamma_1 \|\mathcal{P}(u) - \xi\|_{L^2(\Omega)}^2}_{\text{PDE data}} + \underbrace{\gamma_2 \|u\|_{L^2(\partial\Omega)}^2}_{\text{Boundary data}} + \underbrace{\gamma_3 \mathcal{R}(u)}_{\text{Regularization}} \quad \text{Infinite data}$$

$$\approx \arg \min_{u \in \mathcal{S}} \underbrace{\frac{\gamma_1}{N_\Omega} \sum_{i=1}^{N_\Omega} |\mathcal{P}(u)(x_i) - \xi(x_i)|^2}_{\text{PDE data approximation}} + \underbrace{\frac{\gamma_2}{N_{\partial\Omega}} \sum_{j=1}^{N_{\partial\Omega}} |u(x_j)|^2}_{\text{Boundary approximation}} + \gamma_3 \mathcal{R}(u) \quad \text{Finite data}$$

Three difficulties with Rough PDEs

- 1 The L^2 norm is not appropriate.
- 2 Solves the PDE pointwise.
- 3 Requires that u^* may be well approximated by the class \mathcal{S} .

The **Physics Informed Neural Network** (PINN) approach selects u_θ^\dagger to be a neural network.

The **Kernel/Gaussian process approach**¹ selects u^\dagger from a Reproducing Kernel Hilbert Space (RKHS).

Advantages of the kernel approach

- Provable convergence.
- Uncertainty quantification.

¹Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M. Stuart. “Solving and learning nonlinear PDEs with Gaussian processes”. In: *Journal of Computational Physics* (2021)

Reproducing Kernel Hilbert Space

A kernel/covariance function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ defines a Hilbert space of functions \mathcal{H}_K and corresponding Gaussian Process $\zeta \sim \mathcal{N}(0, K)$.

Kernel Methods and Gaussian Processes

Reproducing Kernel Hilbert Space

A kernel/covariance function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ defines a Hilbert space of functions \mathcal{H}_K and corresponding Gaussian Process $\zeta \sim \mathcal{N}(0, K)$.

GP and kernel regression

Given some noisy observations

$$u^*(x_j) = y_j + \varepsilon_j, \quad j = 1, \dots, N$$

we can recover u^* by **conditioning** or solving a **linear least-squares** problem:

$$u^\dagger = \mathbb{E}[\zeta \mid \zeta(x_j) = y_j + \varepsilon_j] \quad \iff \quad u^\dagger = \arg \min_{u \in \mathcal{H}_K} \|u(X) - Y\|^2 + \gamma \|u\|_{\mathcal{H}_K}^2$$

Gaussian Processes for Solving PDEs

MAP Estimator of a GP²

Solving a **non-linear least squares**

$$u^\dagger = \arg \min_{u \in \mathcal{H}_K} \underbrace{\|\mathcal{P}(u) - \xi\|_{L^2}^2}_{\text{PDE data}} + \underbrace{\gamma \|u\|_{\mathcal{H}_K}^2}_{\text{RKHS regularization}}$$

s.t. $\underbrace{u = 0 \text{ on } \partial\Omega}_{\text{Boundary term}}$

Interpreted as a **MAP estimator** of a GP conditioned on non-linear measurements.

Classical solution

This method solves the PDE in strong form:

$$\mathcal{P}(u)(x_i) = \xi(x_i) \quad i = 1, \dots, N$$

²Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M. Stuart. "Solving and learning nonlinear PDEs with Gaussian processes". In: *Journal of Computational Physics* (2021)

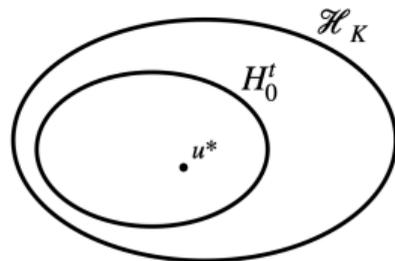
Table of Contents

- 1 Introduction and motivation
- 2 Machine learning for smooth PDEs
- 3 Kernel methods for rough PDEs**
- 4 Convergence results
- 5 Numerical results

Challenges and solutions

Three difficulties with Rough PDEs

- 1 The L^2 norm is not appropriate.
- 2 Solves the PDE pointwise.
- 3 Requires $u^* \in \mathcal{H}_K$ for the convergence theory.



Solutions

Our main contributions are to solve the three main difficulties:

- 1 Using a **negative Sobolev norm** H^{-s} instead of the usual L^2 norm.
- 2 **Efficient approximation** of the H^{-s} norm with weak measurements.
- 3 **Convergence results** of the method without $u^* \in \mathcal{H}_k$.

A Novel Approach to Solving Rough PDEs

New problem

$$u^\gamma = \arg \min_{u \in \mathcal{H}_K} \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \gamma \|u\|_{\mathcal{H}_K}^2 \quad \text{Infinite data}$$

s.t. $u = 0$ on $\partial\Omega$.

$$u^{\gamma, N, M} = \arg \min_{u \in \mathcal{H}_K} \|\mathcal{P}(u) - \xi\|_{V^N}^2 + \gamma \|u\|_{\mathcal{H}_K}^2 \quad \text{Finite data}$$

s.t. $u(x_j) = 0 \quad j = 1, \dots, M$ on $\partial\Omega$.

Negative Sobolev Norm approximation

To go from the infinite data problem to the finite data problem, we need to compute an approximation of the negative Sobolev norm:

$$\|f\|_{H^{-s}} \approx \|f\|_{V^N}$$

³Houman Owhadi and Clint Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization*. Cambridge University Press, Oct. 2019

Negative Sobolev Norm approximation

To go from the infinite data problem to the finite data problem, we need to compute an approximation of the negative Sobolev norm:

$$\|f\|_{H^{-s}} \approx \|f\|_{V^N}$$

Choose Test Space $V^N := \text{span}\{\varphi_i\}_{i=1}^N$ (Ex: Haar basis)

$$[f, \varphi] := \left(\int f\varphi_1, \int f\varphi_2, \dots, \int f\varphi_n \right), \quad A_{i,j} := \int_{\Omega} \varphi_i (-\Delta)^{-s} \varphi_j.$$

Then

$$\|f\|_{V^N} := \sqrt{[f, \varphi]^T A [f, \varphi]}.$$

The entries of A can be efficiently computed with fast solvers for elliptic PDEs³.

³Houman Owhadi and Clint Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization*. Cambridge University Press, Oct. 2019

Finite dimensional problem

Finite dimensional problem

$$u^{\gamma, N, M} = \arg \min_{u \in \mathcal{H}_K} [\mathcal{P}(u) - \xi, \varphi] A[\mathcal{P}(u) - \xi, \varphi] + \gamma \|u\|_{\mathcal{H}_K}^2$$
$$\text{s.t. } u(x_j) = 0 \quad j = 1, \dots, M \text{ on } \partial\Omega.$$

Weak solution

This method can be interpreted as solving the PDE in weak form:

$$[\mathcal{P}(u), \varphi_i] = [\xi, \varphi_i] \quad i = 1, \dots, M.$$

This problem can be solved efficiently with the representer theorem and a non-linear least squares optimization techniques such as a variant of the Gauss-Newton algorithm. 

Table of Contents

- 1 Introduction and motivation
- 2 Machine learning for smooth PDEs
- 3 Kernel methods for rough PDEs
- 4 Convergence results**
- 5 Numerical results

Convergence: Assumptions

Assumptions on the PDE

- The operator $\mathcal{P} : H_0^t \rightarrow H^{-s}$ is continuous.
- The solution operator is (locally) stable

$$\|u - u^*\|_{H_0^t} \leq C \|\mathcal{P}(u) - \xi\|_{H^{-s}}$$

Assumptions on the method

- The space V^N is dense in H^{-s} as $N \rightarrow \infty$.
- The fill distance on the boundary goes to 0 as $M \rightarrow \infty$.
- $\mathcal{H}_K \hookrightarrow H_0^t(\Omega)$ and \mathcal{H}_K is dense in $H_0^t(\Omega)$ (satisfied for Matérn kernels).

Convergence: Main Theorem

Theorem (Convergence to the True Solution)

Let $u^{\gamma, M, N} \in \mathcal{H}_K$ solve the approximate problem, then

$$\lim_{\gamma \rightarrow 0} \lim_{M \rightarrow \infty} \lim_{N \rightarrow \infty} \left\| u^{M, N, \gamma} - u^* \right\|_{H_0^t} = 0.$$

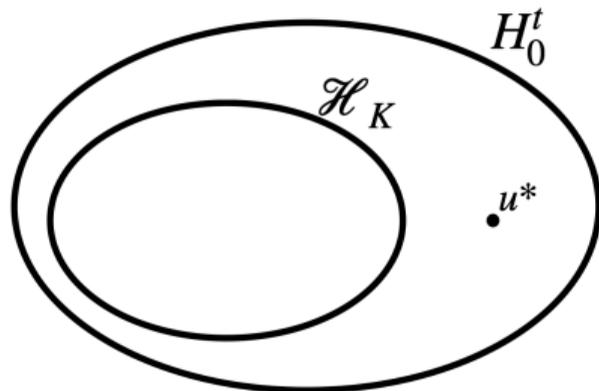


Table of Contents

- 1 Introduction and motivation
- 2 Machine learning for smooth PDEs
- 3 Kernel methods for rough PDEs
- 4 Convergence results
- 5 Numerical results**

Example: Linear Rough PDE

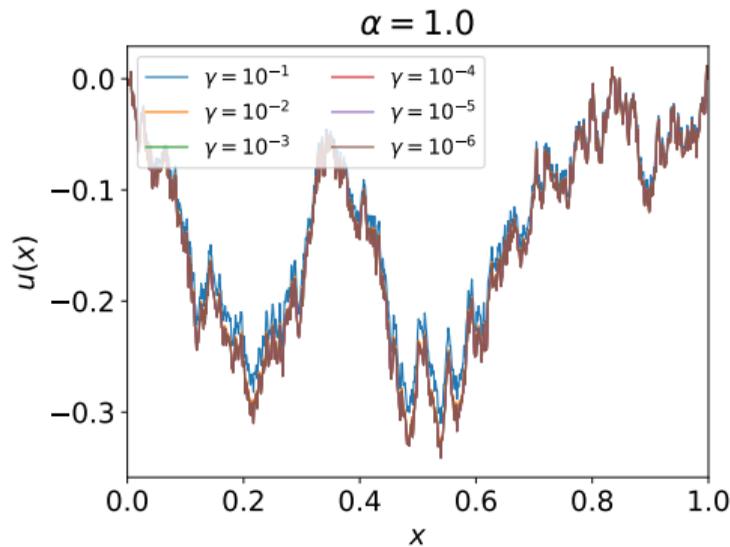
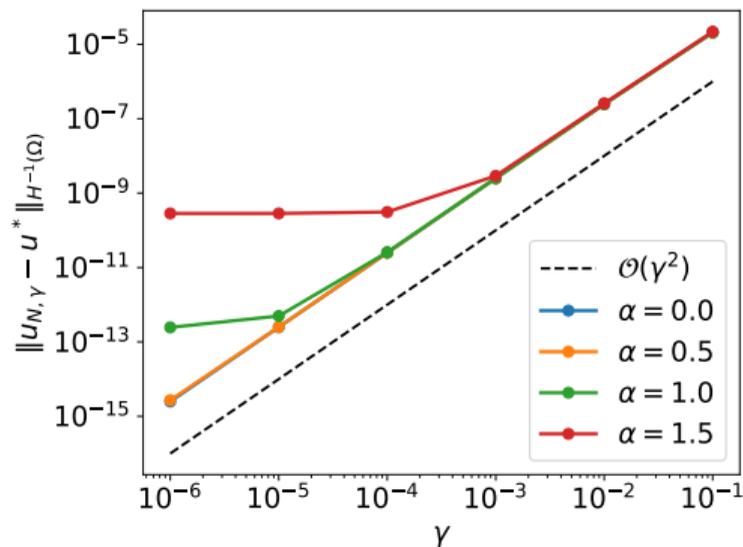


Figure: Linear PDE: $-u_{xx} = f$ with Dirichlet BC for $f(x) = \sum_{k=1}^{\infty} k \xi_k \varphi_k(x)$ and eigenfunction measurements.

Example: Non-linear Rough PDE

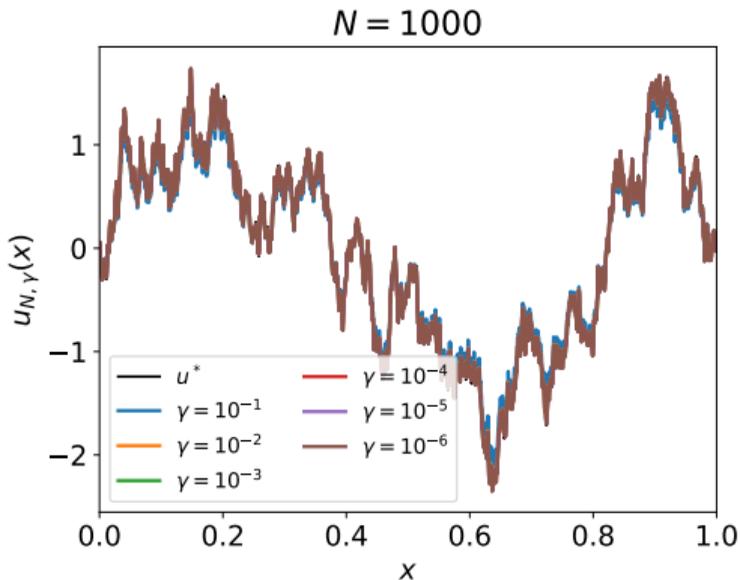
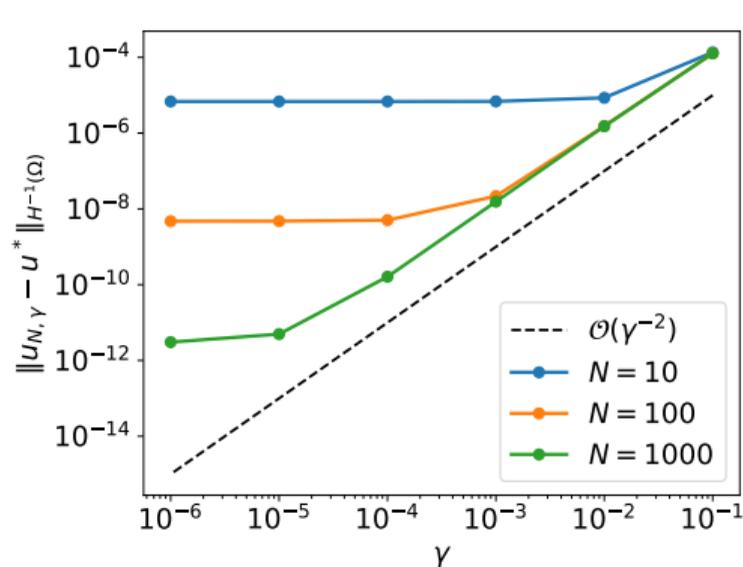


Figure: Nonlinear PDE: $-u_{xx} + \sin(u) = f$ with periodic BC and $f(x) = \sum_{k=1}^{\infty} k^{\alpha} \xi_k \varphi_k(x)$, for $\alpha = 0.49$.

Conclusion

We propose a kernel-based framework for solving PDEs with irregular forcing terms which addresses 3 difficulties:

- ① The L^2 norm is not appropriate \rightarrow Use a weaker Sobolev norm.
- ② Requires pointwise solution \rightarrow Approximate the norm with weak measurements.
- ③ Requires that we approximate u^* \rightarrow Convergence holds even when $u^* \notin \mathcal{H}_K$.

Our numerical experiments show promising initial results and we plan to extend this methodology to solving SPDEs.

mdarcy@caltech.edu