

Gaussian Processes for Scientific Machine Learning: PDE Solvers and Operator Learning

Thesis by
Matthieu Darcy

In Partial Fulfillment of the Requirements for the
Degree of
PhD in Computing and Mathematical Sciences



CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2027
Defended May 5th, 2026

© 2027

Matthieu Darcy

ORCID: 0000-0003-0029-091X

All rights reserved except where otherwise noted

ACKNOWLEDGEMENTS

I would like to thank Houman Owhadi, my advisor, for his mentorship and support throughout these five years, and for giving me the freedom to explore my interests. I would like to thank Andrew Stuart for a wonderful collaboration and for his excellent teaching, from which I learned enormously. Thank you to Amy Braverman for her enthusiasm and encouragement. Thank you to the members of the committee, Franca Hoffmann and Tom Hou, for their time and helpful feedback.

I would like to thank my many collaborators without whom this work would not be possible and who taught me so much throughout the years. A particular thanks to Pau Batlle for many interesting mathematical discussions, as well as Bamdad Hosseini, Adriano Gualandi, Ricardo Baptista, Edoardo Calvello, Xianjin Yang, and Matthew Levine. Thank you to Boumediene Hamzi for his support since my master's at Imperial.

I would like to thank the members of the administrative staff Diana Bohler and Joy Tabuchi for all their help and hard work in making sure my PhD went smoothly.

I am very grateful to all my friends I met while at Caltech, without your support I would not have made it through this PhD. A special thanks to Pau for his incredible friendship since day one, to Eitan for many interesting discussions, to my office mates Edo and Isa for the studious and silly moments, to Théo for occasionally barging into the office, to Roy for reviving my love of movies. Many thanks as well to Sebastian, Siki, Margaret, John, Nina, Matt, Alessandro, Yifan, and Joe. You have made these years truly special.

I would like to thank my family: my parents Milla and Jacques, my siblings Alix and Gaston, my grandparents Elena and Cito, Pennie and David, for their love and support throughout the years.

Finally, I would like to dedicate this thesis to my great-grandfather, Herb Paskett, who has seen the world change so much over 100 years, and yet remains as curious as one can be.

ABSTRACT

We present applications of probabilistic machine learning, with a strong emphasis on Gaussian processes, to several problems arising from differential equations. In particular, we consider three topics: designing solvers for forward and inverse problems, operator learning, and learning and predicting dynamical systems from data. In Chapter 2 we develop a kernel based method to solve partial differential equations with rough forcing and in Chapter 3 we develop a variational inference approach to forward and inverse problems in partial differential equations. In Chapter 4 we develop a Gaussian process approach to operator learning, in Chapter 5 we apply this approach to function renormalization group equations, and in Chapter 6 we develop a new framework for machine precision operator learning. In Chapter 7 we develop a Gaussian process approach to learning stochastic differential equations from one trajectory and in Chapter 8 we study the non-parametric kernel flows approach to learning chaotic systems. In each case, we develop a novel computational methodology, run numerical examples to demonstrate the efficacy of the method, and, where applicable, derive theoretical guarantees for the method.

PUBLISHED CONTENT AND CONTRIBUTIONS

Baptista, Ricardo et al. (2026). “Solving Roughly Forced Nonlinear PDEs via Misspecified Kernel Methods and Neural Networks”. In: *AMS: Mathematics of Computation (to appear)*.

MD was the lead author, developed and implemented the methodology, developed the theoretical results, ran the numerical experiments, and wrote the manuscript.

Darcy, Matthieu and Houman Owhadi (2026). *Sparse Inducing points and Variational Inference I for Forward and Inverse Problems in PDEs*.

MD was the lead author, and developed and implemented the methodology, wrote the theoretical results, and ran the numerical experiments.

Bacho, Aras et al. (2025). *Operator Learning at Machine Precision*. arXiv: 2511.19980. URL: <https://arxiv.org/abs/2511.19980>.

MD ran the benchmarks, and wrote and edited the manuscript.

Darcy, Matthieu, Boumediene Hamzi, Jouni Susiluoto, et al. (2025). “Learning dynamical systems from data: A simple cross-validation perspective, part II: Nonparametric kernel flows”. In: *Physica D: Nonlinear Phenomena* 476, p. 134641. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2025.134641>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278925001204>.

MD developed and implemented the methodology, ran the numerical experiments, and wrote the manuscript.

Yang, Xianjin et al. (2025). *Solving Functional PDEs with Gaussian Processes and Applications to Functional Renormalization Group Equations*. arXiv: 2512.20956 [cs.LG]. URL: <https://arxiv.org/abs/2512.20956>.

MD developed the methodology, implemented early numerical examples, and wrote the manuscript.

Battle, Pau et al. (2024). “Kernel methods are competitive for operator learning”. In: *Journal of Computational Physics* 496, p. 112549. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2023.112549>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999123006447>.

MD developed and implemented the methodology, ran the numerical experiments, and wrote the manuscript.

Darcy, Matthieu, Boumediene Hamzi, Giulia Livieri, et al. (2023). “One-shot learning of stochastic differential equations with data adapted kernels”. In: *Physica D: Nonlinear Phenomena* 444, p. 133583. DOI: <https://doi.org/10.1016/j.physd.2022.133583>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278922002871>.

MD was the lead author, and developed and coded the methodology, ran the numerical experiments, and wrote the manuscript.

TABLE OF CONTENTS

Acknowledgements	iii
Abstract	iv
Published Content and Contributions	v
Table of Contents	v
List of Illustrations	ix
List of Tables	xv
Chapter I: Introduction	1
1.1 Machine learning and numerical approximation	2
1.2 Machine learning for partial differential equations	3
1.3 Two tasks: designing solvers and learning operators.	4
1.4 Learning dynamical systems	6
1.5 Outline of this thesis	7
Chapter II: Solving Roughly Forced Nonlinear PDEs via Misspecified Kernel	
Methods and Neural Networks	8
2.1 Introduction	9
2.2 The Continuum Problem	15
2.3 The Numerical Method	21
2.4 Numerical Experiments	29
2.5 Defining a New Solution Concept for Singular PDEs/SPDEs	39
2.6 Conclusion	41
Chapter III: Sparse Inducing points and Variational Inference for Forward and	
Inverse Problems in PDEs	43
3.1 Introduction: setting and summary of the method	44
3.2 Sparse Variational Inference for PDEs	49
3.3 Numerical experiments	58
3.4 Conclusion	65
Chapter IV: Kernel methods are competitive for operator learning	67
4.1 Introduction	68
4.2 The RKHS/GP framework for operator learning	79
4.3 Convergence and error analysis	84
4.4 Numerics	92
4.5 Conclusions	103
Chapter V: Solving Functional PDEs with Gaussian Processes and applica-	
tions to Functional Renormalization Group Equations	104
5.1 Introduction	105
5.2 Methodology	113
5.3 Numerical results	122
5.4 Conclusion	131
Chapter VI: Operator learning at machine precision	133

6.1	Introduction	134
6.2	Operator Learning	139
6.3	Numerical Experiments	146
6.4	Theoretical results	159
6.5	Conclusion and Outlook	163
6.6	Code and data availability	164
Chapter VII: One-Shot Learning of Stochastic Differential Equations with		
	data adapted kernels	165
7.1	Introduction	166
7.2	Statement of the problem and proposed solution	167
7.3	MAP estimator	169
7.4	Learning the kernels and the hyperparameters	172
7.5	Experimental results	174
7.6	Experimental results: a comparison between perfectly specified and non perfectly specified kernels.	179
7.7	Experimental results: the effect of time discretization	183
7.8	Experimental results: discussion	186
Chapter VIII: Learning dynamical systems from data with non-parametric		
	kernel flows	187
8.1	Introduction	188
8.2	The problem and its proposed cross-validation solutions	191
8.3	Numerical experiments	193
8.4	Conclusion	202
Bibliography		203
Appendix A: Chapter 2		238
A.1	Auxiliary proofs	239
A.2	Neural Network Architecture	247
A.3	Gaussian Measures and White Noise	248
Appendix B: Chapter 3		252
B.1	Sparse variational inference derivation	253
B.2	Gaussian VI proofs	253
B.3	Dirac-SVGP proofs	260
B.4	Model parameters and architecture	262
Appendix C: Chapter 4		263
C.1	Review of operator valued kernels and GPs	264
C.2	An alternative regularization of operator regression	268
C.3	Expressions for the kernels used in experiments	269
Appendix D: Chapter 5		270
D.1	Optimal recovery	271
D.2	Further Experiments	271
D.3	Mathematical presentation of the FRG flow problem	278
D.4	Derivative and Hessian of kernel interpolants	279
D.5	Local Potential Approximation (LPA)	282
D.6	Transfer Matrix Approach	284
D.7	Details of the GP representation for the Wetterich flow	287

D.8 Magnetization and Susceptibility	290
Appendix E: Chapter 6	291
E.1 Gaussian process benchmarks	292
E.2 Nested Kriging	292
E.3 Choice of ordering in the Cholesky Factorization	294
E.4 Discretizations and Cholesky factorization	296
E.5 Choice of the Learning rate α and Relaxation λ	297
E.6 Approximation of the Fréchet derivative	298
E.7 Connection to Attention Mechanism	299
E.8 Proofs	300
Appendix F: Chapter 7	305
F.1 Additional Plots.	306
F.2 Gaussian Process Regression and Extension of Bishop, 2006, Page 306307	
F.3 Log-marginal likelihood for hyper-parameter optimization	308
F.4 Randomized cross-validation for hyper-parameter learning	309
F.5 Extension to the multivariate case	311
Appendix G: Chapter 8	314
G.1 Additional figures	315

LIST OF ILLUSTRATIONS

<i>Number</i>	<i>Page</i>
2.1 Numerical results for the one-dimensional linear elliptic PDE (2.25).	32
2.2 One example solution to the two-dimensional semi-linear elliptic PDE (2.27).	33
2.3 Numerical solutions to the two-dimensional semi-linear elliptic PDE (2.27) using the kernel and NN approximations with either the negative Sobolev loss or the pointwise loss.	34
2.4 Impact of norm selection on the recovered solution.	36
2.5 Numerical solutions for the stochastic heat equation.	38
2.6 Stochastic Allen-Cahn equation.	40
3.1 Darcy coefficient recovery problem.	46
3.2 A comparison between the original GP approach (left), a discrete sparse GP (middle), and our proposed continuous sparse GP (right). The continuous formulation uses the same discrete inducing points for the prior, but leverages the entirety of the continuous domain for the likelihood.	52
3.3 Solutions of two noisy PDEs using SVGP.	60
3.4 Error rate and pointwise error comparison for noiseless forward PDEs.	61
3.5 Darcy coefficient recovery problem.	62
3.6 Arrhenius initial condition recovery problem.	63
3.7 Burger forcing recovery problem.	64
4.1 Commutative diagram of our operator learning setup.	70
4.2 Example of training data and test prediction and pointwise errors for the Darcy flow problem (4.3).	74
4.3 Generalization of fig. 4.1 to the mesh invariant setting where the measurement functionals are different at test time.	85
4.4 Example of training data and test prediction and pointwise errors for the Burger's equation (4.49).	96
4.5 Example of training data and test prediction and pointwise errors for the Advection problem (4.50)-I.	98
4.6 Example of training data and test prediction and pointwise errors for the Advection problem (4.50)-II.	98

4.7	Example of training data and test prediction and pointwise errors for the Helmholtz problem (4.53).	99
4.8	Example of training data and test prediction and pointwise errors for the Structural Mechanics problem (4.54).	99
4.9	Example of training data and test prediction and pointwise errors for the Navier-Stokes problem (4.55).	100
4.10	Accuracy complexity tradeoff achieved in the problems in De Hoop et al., 2022. Data for NNs was obtained from the aforementioned article. Linear model refers to the linear kernel, vanilla GP is our implementation with the nonlinear kernels and minimal preprocessing, and GP+PCA corresponds to preprocessing through PCA both the input and the output to reduce complexity.	102
5.1	Commutative diagram for the functional learning setup.	115
5.2	Numerical results for Gaussian-model solution of the Wilson–Polchinski equation. Subfigures (A)–(C) show results at collocation points, while (D)–(F) present corresponding results at testing points. For visualization, only the first 3 of 1,000 collocation points and the first 3 of 200 testing points are displayed. The relative L^2 error is evaluated for each scale variable κ	125
5.3	Benchmark of the GP reconstruction versus standard LPA, lattice Monte Carlo, and the transfer matrix method. (A) Values of the potential $U_{\kappa_{\text{IR}}}(\phi)$ on $[-5, 5]$ of different methods. (B) Susceptibility $\chi(c)$ versus external field c (log-log). (C) Susceptibility errors (relative to transfer matrix); for most values of c , the GP exhibits smaller errors than LPA, especially at small c . (D) Magnetization $m(c)$ (log-log). (E) Magnetization errors (relative to transfer matrix); in most of the c range the GP is more accurate than LPA.	130
6.1	CHONKNORIS. (a) An initial guess v_0 for the true solution $v = \mathcal{G}(u)$ is iteratively refined by adding a correction term. (b) Each iteration consists of two steps: First, compute the correction term $\delta v = -(\widehat{\mathcal{R}}\widehat{\mathcal{R}}^*[\frac{\delta \mathcal{F}}{\delta v}]^* \mathcal{F})(u, v_k)$, where $\widehat{\mathcal{R}}$ is a learned surrogate for the Cholesky factors of $(\frac{\delta \mathcal{F}}{\delta v} \frac{\delta \mathcal{F}}{\delta v}^T + \lambda I)^{-1}$, $\mathcal{F}$ is the forward map with its Fréchet derivative $\frac{\delta \mathcal{F}}{\delta v}$, v_k is the current approximation of the desired function $v = G(u)$, and u is the input for which we seek the solution. Next, update the current approximation via $v_{k+1} = v_k + \delta v$	145

6.2	Forward problems. (a) Results for the nonlinear elliptic PDE problem. Quantiles of 10% – 90% are shown across test realizations. Our CHONKNORIS method is able to achieve machine precision accuracy in around 10 iterations. (b) Results for Burgers' equation. CHONKNORIS was able to achieve machine precision error in recovering the discretized solution which contained shocks. (c) Results for the Darcy flow PDE: (c1) shows that more challenging realizations require more CHONKNORIS iterations. (c4) shows a single realization of the random coefficient with the corresponding solution in (c2). (c3) shows the fixed forcing term.	151
6.3	Quantiles of predictions of FONKNORIS for 100 realizations of initial conditions, external forces, and conductivities using a mixture of experts consisting of GPs for nonlinear elliptic, nonlinear darcy flow, and Burgers' equation and testing it for withheld Sine–Gordon and Klein–Gordon equations.	153
6.4	Inverse Problems. (a) Results for the seismic imaging problem, showing the iterative solutions and the relative L^2 error across CHONKNORIS iteration. (b) For the seismic imaging problem, evolution of the adaptive regularization term in the Newton–Kantorovich iterations for different resolutions, and comparison of the relative L^2 error between the Newton–Kantorovich method and our CHONKNORIS method. (c) Results for the Calderón problem. (d) Results for the inverse wave scattering problem.	158
7.1	Predicted volatility: well-specified kernel (left), ill specified kernel (middle), and well-specified kernel with learned parameters (right). .	172
7.2	From left to right: drift function, volatility function, sample trajectory, and sample increments of the exponential volatility process. .	177
7.3	Predicted drift and volatility on the testing set for trajectory 2 of the exponential volatility process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel)	177
7.4	From left to right: drift function, volatility function, sample trajectory and sample increments of the trigonometric process (trajectory 2). .	178

7.5	Predicted drift and volatility on the testing set for trajectory 2 of the trigonometric process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel)	178
7.6	Exponential volatility process benchmark prediction. From left to right: drift (trajectory 1), volatility (trajectory 1), drift (trajectory 2), volatility (trajectory 2).	179
7.7	Trigonometric process benchmark prediction. From left to right: drift (trajectory 1), volatility (trajectory 1), drift (trajectory 2), volatility (trajectory 2).	179
7.8	From left to right: drift function, volatility function, sample trajectory, and sample increments of GBM.	180
7.9	Predicted volatility on the testing set for GBM. From left to right: linear kernel, Matern kernel (non-learned parameters), and Matern Kernel (learned parameters).	180
7.10	GBM benchmark prediction. From left to right: volatility (linear kernel) and volatility (Matern kernel).	181
7.11	From left to right: drift function, volatility function, sample trajectory, and sample increments of the OU process.	181
7.12	Prediction on the OU process with the Matern kernel. From left to right: benchmark prediction (drift), benchmark prediction (volatility), CGC prediction (drift), CGC prediction (volatility).	182
7.13	Prediction on the OU process with the linear kernel. From left to right: benchmark prediction (drift), benchmark prediction (volatility), CGC prediction (drift), CGC prediction (volatility).	182
7.14	Exponential decay volatility process: prediction of the volatility for different values of k . From left to right: $k = 1, 2, 5$	184
7.15	Exponential decay volatility process: prediction of the drift for different values of k . From left to right: $k = 1, 2, 5$	184
7.16	Trigonometric process: prediction of the volatility for different values of k . From left to right: $k = 1, 2, 3$	185
7.17	Trigonometric process: prediction of the drift for different values of k . From left to right: $k = 1, 2, 5$	185
8.1	One step ahead times series predictions results for the Bernoulli map R1.	195

8.2	Deformation of input case for different iterations of the flow function ϕ_L (left) and deformed final data (right).	195
8.3	Convergence of the ρ and ρ_{MMD} losses (horizontal axis stands for the number of iterations).	196
8.4	Comparison between the different predictions of the first variables for the baseline, ρ training and ρ_{MMD} training (left, middle, right). The horizontal axis stands for the number of iterations.	200
8.5	One step prediction in the phase space: base estimator (left), kernel flows with ρ_{MMD} (right).	200
8.6	Loss functions through training of the first variable for ρ (left) and ρ_{MMD} (right).	200
8.7	Times series predictions. Top: x , middle: y , bottom: z	201
8.8	Losses. Relative errors ρ for the different variables (blue) and averages (red)- Top: x , middle: y , bottom: z	201
A.1	Space-time white noise generated by two methods.	251
D.1	Numerical results of the Wetterich Gaussian model. Subfigures (A)–(C) show results at collocation points, while (D)–(F) present corresponding results at testing points. For visualization, only the first 3 of 1,000 collocation points and the first 3 of 200 testing points are displayed. The relative L^2 error is evaluated for each scale variable κ	274
D.2	Numerical results for the Wetterich flow with the ϕ^4 bare action. Panels (A)–(B) illustrate three of the $N_{\text{col}} = 1000$ training fields and the corresponding GP predictions, while panels (C)–(D) plot the surrogate and the LPA potential on constant-field inputs in $[0, 4]$. Panel (E) maps the pointwise relative difference across RG scales $\kappa \in [2, 10]$ and field amplitudes $\phi \in [0, 4]$	277
E.1	Options for jointly adaptive relaxation and learning rate.	298
F.1	From left to right: drift function, volatility function, sample trajectory, and sample increments of the exponential volatility process.	306
F.2	Predicted drift and volatility on the testing set for trajectory 1 of the exponential volatility process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel).	306
F.3	From left to right: drift function, volatility function, sample trajectory, and sample increments of the trigonometric process.	306

F.4	Predicted drift and volatility on the testing set for trajectory 1 of the trigonometric process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel).	307
G.1	Prediction results for the Bernoulli map for different initial conditions.	315
G.2	Lorenz system: comparison between the different predictions of the second (y) and third variables (z) (top, bottom) for the baseline, ρ_{MMD} training and ρ training (left, middle, right).	316
G.3	Convergence of the ρ (left) and ρ_{MMD} (right) for higher learning rates on the Bernoulli time series.	316
G.4	Loss functions through training for the second (y) and third variables (z) of the Lorenz times series (top, bottom) for ρ (left) and ρ_{MMD} (right).	317

LIST OF TABLES

<i>Number</i>	<i>Page</i>
2.1 Relative L^2 error of the estimated solutions for the spatial PDEs with $N = 4096$ measurements. The errors are reported for both the kernel and the NN approximation that are trained with either the H^{-1} norm or the pointwise loss (in brackets).	31
2.2 Relative L^2 error for different choices of Sobolev norms as the loss function	35
2.3 Relative $L^2([0, T]; L^2(0, 1))$ error of the kernel-based prediction for the solutions to the time-dependent PDEs.	38
3.1 Comparison of methods across forward PDE problems (error in relative L^2).	60
3.2 Comparison of methods across inverse PDE problems (error in relative L^2).	64
4.1 The L^2 relative test error of the Darcy flow problem in our running example. The kernel approach is compared with variations of DeepONet and FNO. Results of our kernel method are presented below the dashed line with the pertinent choice of the kernel S	74
4.2 Summary of datasets used for benchmarking. The first three examples were considered in L. Lu, X. Meng, et al., 2022, and the last four were taken from De Hoop et al., 2022.	94
4.3 Summary of numerical results: we report the L^2 relative test error of our numerical experiments and compare the kernel approach with variations of DeepONet, FNO, PCA-Net, and PARA-Net. We considered two choices of the kernel S , the rational quadratic and the Matérn, but we observed little difference between the two.	102
4.4 Comparison between Cholesky preconditioning and PCA dimensionality reduction on three examples for our vanilla kernel implementation with the Matérn kernel.	103
5.1 Gaussian-Model Solution of the Wilson–Polchinski Equation: Average relative L^2 errors on collocation and test sets for varying sample sizes N	124
6.1 Summary of median relative L^2 losses over multiple forward and inverse problem benchmarks.	147

6.2	Forward and inverse problem parameters. u distrib. is the distribution of random coefficients. FD grid contains the regular grid sizes for the finite difference scheme. $(R_{\text{train}}, R_{\text{val}})$ are the number of training and validation realizations respectively. n_{warm} denotes the number of Newton–Kantorovich iterations used to generate training data along the flow. λ_{flow} is the relaxation used in the NK method, while λ_{train} is the relaxation shown to the Hessian prediction model. CHONKNORIS training is performed with $\Lambda_{\text{train}} = \{\lambda_{\text{flow}}\}$ in the context of Algorithm 2. GP-kernel denotes a Gaussian process with the given kernel, with the invLaplacian kernel given by $5(-\Delta + 1/100)^{-2}$ where $-\Delta$ denotes the Laplacian. MLP-nonlinearity is a multi-layer perceptron neural network architecture where the tuple specifies layer sizes, including input and output layers, and the given nonlinearity is applied to all hidden layers. Note that for Burgers’ equation, $n_{\text{warm}} = 5$ NK steps were run per time step.	148
6.3	FONKNORIS coefficients.	152
7.1	Results for the exponential volatility process.	177
7.2	Results for the trigonometric process.	178
7.3	Comparison between the linear kernel and the Matern kernel on GBM.	180
7.4	Results for the OU process.	182
7.5	Results for the exponential decay process for different time step Δt observations.	184
7.6	Results for the trigonometric process for different time step Δt observations.	185
8.1	Bernoulli map: 200 training points.	194
8.2	Bernoulli map: 800 training points.	195
8.3	Lorenz system: original data.	198
8.4	Lorenz system: pre-processed data.	198
8.5	Bernoulli map learned with ρ for different learning rates	199
8.6	Bernoulli map learned with ρ_{MMD} for different learning rates	199
A.1	Architecture for the neural networks used for the NeS-PINN approach.	247
B.1	Main parameters of the RFF-based PINN.	262
D.1	The Wetterich Gaussian Model: Average relative L^2 errors on collocation and test sets for varying sample sizes N	273

Chapter 1

INTRODUCTION

This thesis is broadly concerned with the intersection of machine learning and problems from traditional numerical analysis, in particular those related to partial differential equations (PDEs). The overall aim is to develop novel methodologies that are flexible, general, and equipped with theoretical guarantees. We begin with a brief overview of machine learning and its connection to numerical analysis, followed by an overview of applications and problems related to PDEs. Each chapter contains a more detailed introduction specific to its topic.

1.1 Machine learning and numerical approximation

Machine learning is broadly concerned with the design of algorithms that learn patterns and make predictions from data, typically by optimizing an objective over a class of models Hastie, Tibshirani, and Friedman, 2009. *Probabilistic* machine learning, in particular, treats unknown quantities as random variables Murphy, 2025, enabling the systematic quantification and propagation of uncertainty. Machine learning methods generally take a statistical perspective: they rely primarily on data and aim to make relatively few assumptions about the underlying system.

Numerical approximation and scientific computing, by contrast, are powerful computational tools for understanding the natural world when *physical models* are available. In this setting, applied mathematicians have developed highly accurate computational methods capable of accurately approximating these models and making reliable predictions about physical systems.

Recently, there has been growing recognition that machine learning, despite originating from a different community with distinct priorities, provides powerful tools for problems of interest in applied mathematics. This manifests in two main ways. First, machine learning offers a framework for learning and prediction from data in situations where we do not have access to an accurate physical models E, 2021. Integrating these approaches into applied mathematics enables us to tackle new problems where our physical models might be insufficient. Second, applied mathematicians have adapted machine learning frameworks to perform computations relevant to classical problems. In this context, machine learning methods have become important computational tools, with the potential to be more flexible and scalable than traditional approaches Montanelli, n.d. Furthermore, there has been a growing connection between numerical approximation and statistical inference as many numerical algorithms can be cast statistical inference problems Houman Owhadi, 2015a; O’hagan, 1992; Diaconis, 1988, a perspective which has been used

by the probabilistic numerics community Hennig, Osborne, and Kersting, 2022.

In this thesis, we aim to further develop these connections and to introduce new methodologies for problems arising in the study of physical systems, and partial differential equations in particular. We focus primarily on Gaussian processes and kernel methods, which are closely linked through the theory of reproducing kernel Hilbert spaces (RKHS) Martin Hairer, 2023; Houman Owhadi and Scovel, 2019a; Rasmussen and Williams, 2008. This choice has two main advantages. First, it provides a Bayesian interpretation of the algorithms we develop, thereby enabling probabilistic predictions. Second, the theory of RKHS allows us to derive rigorous error estimates and theoretical guarantees for many of the applications we consider.

1.2 Machine learning for partial differential equations

Differential equations are a fundamental tool for describing many phenomena arising in physics, biology, and chemistry, as well as in the social sciences and finance. Despite being an old subject with a rich history, it remains a thriving source of problems and challenges in pure and applied mathematics. Recently, modern machine learning and PDEs have intersected, leading to novel algorithms, applications, problems, and perspectives. Some of the key practical applications that have emerged from this line of research include:

- **Novel solvers.** Machine learning provides new solvers, which have the potential to be more general, more flexible, or faster compared to traditional numerical algorithms.
- **Inverse problems.** Machine learning can provide new approaches to solving inverse problems or integrate within existing pipelines.
- **Uncertainty quantification.** Probabilistic approaches can provide probabilistic predictions that practitioners use to make informed decisions.

We note that many machine learning approaches combine several of these benefits, often in a mutually reinforcing manner. For example, the flexibility of machine learning solvers enables them to be applicable to inverse problems. Likewise, the uncertainty quantification of probabilistic approaches makes them particularly suited for inverse problems. A key question this thesis attempts to answer is how can we provide accurate probabilistic inference in the context of problems arising from PDEs.

1.3 Two tasks: designing solvers and learning operators.

At this stage, we need to distinguish between two important tasks in this context: designing *solvers* (for both forward and inverse problems) versus *operator learning*. Abstractly, a partial differential equation encodes known laws of the system through an operator $\mathcal{F}$ that incorporates differential and boundary information:

$$\mathcal{F}(u, Du, D^2u, \dots; a) = 0. \quad (1.1)$$

Here, u is the solution of the PDE, and a is a coefficient that parametrizes the operator (this coefficient can represent, for example, material properties of the system). We give a brief exposition of the two problems, as well the existing literature at the time of writing of this thesis. A more detailed literature review is available within the relevant chapters.

Machine learning solvers

In the context of machine learning solvers, we parametrize the function we wish to recover (either the solution u or the coefficient a) in a suitable hypothesis space $\mathcal{H}$ and cast this recovery problem as a loss minimization problem. This loss function is formulated to satisfy the constraint eq. (1.1)

$$\min_{u \in \mathcal{H}} \|\mathcal{F}(u, Du, D^2u, \dots; a)\|^2 \quad (1.2)$$

and is referred to as the “physics-informed loss” (because it includes the physics encoded by the operator $\mathcal{F}$). In this setting, machine learning algorithms solve a specific instance of this problem.

This approach was first pioneered by physics informed neural networks (PINN) where the hypothesis space is parametrized by a neural network (Raissi, P. Perdikaris, and G. Karniadakis, 2019c). An alternative approach is to consider the hypothesis space to be a reproducing kernel Hilbert space (Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b). One advantage of this approach is that it provides a probabilistic approach to solving PDEs as it can be interpreted as the maximum a posteriori (MAP) estimation of Gaussian process conditioned on non-linear measurements (Y. Chen, Hosseini, Houman Owhadi, and Andrew M Stuart, 2024a). A further advantage is that it is amenable to second order optimization, such as the Gauss-Newton optimizer Nocedal and Stephen J Wright, 2006a, which was later used for high accuracy PDE solving with PINNs Yongji Wang et al., 2025. Recently, second order optimization methods have been explored for PINNs Jnini, Kiyani, et al., 2026.

Broadly speaking, machine learning solvers provide a generic recipe to solve PDE forward and inverse problems. One must select:

1. An appropriate hypothesis class.
2. An appropriate loss function.

Indeed, the loss eq. (1.2) requires a choice of norm $\|\cdot\|$ which have left deliberately vague. This choice of loss implies assumptions on the smoothness of the problem. Likewise, the choice of hypothesis space $\mathcal{H}$ implies assumptions on the smoothness of the solution u . In chapter 2 we tackle this problem for special class of PDEs driven by a rough forcing. We show that by choosing the right loss (a H^{-s} Sobolev norm), we can solve this class of PDEs even when our hypothesis class is *misspecified*.

Another question that naturally arises in the context of inverse problems is how we can design principled probabilistic approaches that handle non-linear systems and assimilate large amounts of data. In chapter 3, we provide one possible solution through a general variational inference framework for PDEs. A related problem from the field of dynamical systems is how to efficiently learn the evolution of stochastic systems from sparse data, which is covered in chapter 7.

Operator learning

In contrast to the above, where we aim to recover a *function*, in operator learning the aim is to learn an operator associated with the general class of systems eq. (1.1). For example, one might consider the operator that maps the coefficient to the solution

$$\mathcal{G}(a) = u \tag{1.3}$$

or some initial condition to the solution at some future time

$$\mathcal{G}(u_0) = u_t. \tag{1.4}$$

Solving this problem has emerged as a potential solution to two important problems:

- **Surrogate modeling:** the operator $\mathcal{G}$ can be queried, but requires expensive computation (such as an expensive PDE solver). In this case, we are interested in learning a surrogate which makes the evaluation cheaper.
- **Unknown physics:** we might not have access to the PDE eq. (1.1). In this case, we are interested in learning an approximation of $\mathcal{G}$ from data.

Once learned, our approximation for $\mathcal{G}$ can be used to predict new inputs and, in this setting, machine learning algorithms can be used to predict an entire class of systems.

In the basic operator learning problem, the operator is learned in a purely data driven way, from input-output samples $\{u_i, \mathcal{G}(u_i)\}_{i=1}^N$. One major difficulty is to formulate appropriate machine learning algorithm to handle functional inputs and outputs. Existing literature has primarily focused on neural operators, generalizations of neural networks to this setting Zongyi Li, N. Kovachki, et al., 2020b; L. Lu, Jin, and G. E. Karniadakis, 2019; Bhattacharya, Hosseini, Nikola B. Kovachki, et al., 2021b. In chapter 4, we present one of the first probabilistic frameworks for operator learning, based on a simple Gaussian process approach, and show that it is competitive in performance against these neural network approaches. Probabilistic frameworks for operator learning that can provide UQ have become increasingly relevant in areas such as extreme weather forecasting Pathak, Subramanian, et al., 2022; Kossaifi et al., 2026.

The operator learning perspective can also be fruitful in other settings. In chapter 5, the Gaussian process methodology is applied to functional renormalization group (FRG) equations arising in the context of quantum field theory in chapter 5. In this context, we have access to integral-differential equation modeling a *functional*, an operator mapping to the reals Delamotte, 2012. Existing numerical solvers for these problems are typically specialized and limited to specific FRG flows, and a key benefit of this approach is to provide general purpose algorithm which can be applied to a very broad class of problems.

Finally, while operator learning is intimately tied to PDEs and has been successfully applied to many problems in this context, existing approaches typically struggle to match the accuracy of either machine learning or traditional numerical solvers, even when the PDE eq. (1.1) is known Zongyi Li, Zheng, et al., 2023a. In chapter 6, we present a general framework that emulates the optimization algorithms of numerical solvers and can achieve machine precision on a range of problems.

1.4 Learning dynamical systems

Finally, we consider the related problem of learning to predict time series generated by dynamical systems. In this setting, we aim to learn the structure of the underlying equations governing the system, namely the drift and, in stochastic settings, the diffusion terms. We are particularly interested in regimes where only sparse data

are available, for example a single trajectory of a stochastic or chaotic system. In chapter 7, we study the problem of learning a stochastic differential equation from a single trajectory, potentially observed at large time steps. In chapter 8, we apply the nonparametric kernel flows algorithm to the learning of chaotic time series.

1.5 Outline of this thesis

The rest of this thesis is structured in three parts:

1. Chapter 2 and chapter 3 cover applications to solving partial differential equations.
2. Chapter 4, chapter 5, and chapter 6 cover applications to operator learning.
3. Chapter 7 and Chapter 8 covers applications to learning dynamical systems.

Chapter 2

SOLVING ROUGHLY FORCED NONLINEAR PDES VIA
MISSPECIFIED KERNEL METHODS AND NEURAL
NETWORKS

2.1 Introduction

Setting and Literature Review

In this paper, we aim to develop a Gaussian Process (GP)-based kernel method for solving roughly forced partial differential equations (PDE) of the form

$$\begin{aligned}\mathcal{P}(u) &= \xi, & x \in \Omega, \\ u &= g, & x \in \partial\Omega,\end{aligned}\tag{2.1}$$

where $\mathcal{P}$ is a (possibly nonlinear) differential operator¹. In the context of this article, this PDE is said to be roughly forced when ξ is highly irregular. For example, ξ can be less regular than an $L^2(\Omega)$ function or belong to a space where the evaluation functionals δ_x are not continuous, and hence ξ is not well-defined pointwise. As a result, the solution to PDE (2.1) is also irregular and is not a classical solution. Consider as an illustrative example, the following semi-linear elliptic PDE with homogeneous boundary conditions and $\xi \in H^{-1}(\Omega)$:

$$\begin{aligned}-\Delta u + f(u) &= \xi, & x \in \Omega, \\ u &= 0, & x \in \partial\Omega.\end{aligned}$$

Here, f is a pointwise Nemitskii operator that is defined by lifting a nonlinear scalar function $f : \mathbb{R} \rightarrow \mathbb{R}$ to act on functions. This PDE does not have a classical solution because u does not belong to $C^2(\Omega)$.

Roughly forced PDEs arise in the context of stochastic partial differential equations (SPDE) Gabriel J. Lord, Powell, and Shardlow, 2014 where ξ is a sample of a stochastic forcing term such as space-time white noise. SPDEs serve as useful models in many domains of application. These include: mathematical biology, where the Nagumo equation models the voltage in the axon of a neuron Walsh, 1981; Laing and Gabriel J Lord, 2009; physics, where the Allen-Cahn SPDE provides a model of phase transitions with random fluctuations Allen and Cahn, 1979; Garcia-Ojalvo and Sancho, 1999; stochastic filtering, where the Kushner-Stratonovich SPDE governs the evolution of the conditional distribution for the state given related observations Bain and Crisan, 2009; and probability, where Langevin equations provide a mechanism to sample measures on pathspace Andrew M. Stuart, Jochen Voss, and Wilberg, 2004; M. Hairer, A. M. Stuart, J. Voss, and Wiberg, 2005; M. Hairer, A. M. Stuart, and J. Voss, 2007. Computational methods for solving time-dependent SPDEs include finite element Galerkin methods Yan, 2005a, spectral

¹It is possible to generalize the setting to a non-identity boundary operator so that $\mathcal{B}(u) = g$ for $x \in \partial\Omega$; doing so is beyond the scope of this work.

Galerkin methods Grecksch and Kloeden, 1996, wavelet-based methods Gyongy and Millet, 2009, and finite differences Gyongy and Millet, 2009, combined with temporal discretization schemes such as explicit/implicit Euler Printemps, 2001; Yan, 2005b, Milstein Jentzen and Rockner, 2012 or exponential integrators Gabriel J. Lord and Tambue, 2019. In the context of PDEs with random coefficients, methods include stochastic Galerkin methods Roger G. Ghanem and Pol D. Spanos, 1991, Monte Carlo finite elements Barth, Schwab, and Zollinger, 2011; Cliffe et al., 2011, polynomial chaos expansions Xiu and G. E. Karniadakis, 2002a; Zhongqiang Zhang and G. E. Karniadakis, 2017, and dynamic bi-orthogonal methods based on Karhunen-Loève expansions Cheng, Hou, and Zhiwen Zhang, 2013b; Cheng, Hou, and Zhiwen Zhang, 2013a.

Machine learning PDE solvers, which trace back to statistical inference approaches for numerical approximation FM Larkin, 1972b, as reviewed in Houman Owhadi, Scovel, and Schafer, 2019 and Houman Owhadi and Scovel, 2019b, Chap. 20, have become increasingly prevalent due to their ability to solve a wide class of PDEs in a flexible manner. In particular, these approaches are compatible with generic engineering pipelines and have the ability to handle inverse problems and the incorporation of previously collected data, e.g., computed solutions for different initial conditions. There are two main categories of such solvers: methods based on artificial neural networks, such as physics-informed neural networks (PINNs) Raissi, P. Perdikaris, and G. Karniadakis, 2019a; Sifan Wang, Sankaran, et al., 2023, and those based on kernels and Gaussian processes Batlle, Y. Chen, et al., 2025a; Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b. Although GP-based methods have originally been introduced for rough linear PDEs Houman Owhadi, 2015b; Houman Owhadi, 2017; Houman Owhadi and L. Zhang, 2017 and have been studied in the context of probabilistic numerics Cockayne et al., 2017; Pfortner et al., 2022; J. Wang et al., 2021, their convergence guarantees required the solution to belong to the RKHS of the underlying kernel and the generalization to nonlinear PDEs in Batlle, Y. Chen, et al., 2025a; Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b require the solution also to be classically defined. Similarly, the original version of PINNs assumes that the solution is classically defined and while more recent ANN-variants such as the DeepRitz method E and B. Yu, 2018 that is based on a variational approach, and wPINNs De Ryck, Mishra, and Molinaro, 2024 that computes weak solutions of hyperbolic PDEs, have started relaxing those regularity assumptions, an ANN-based framework for roughly forced arbitrary nonlinear PDEs is still lacking.

In this work, we are motivated by the advantages of GP-based methods, such as simple and transparent computational and theoretical guarantees, e.g., their near-linear complexity Y. Chen, Houman Owhadi, and Schafer, 2024b and convergence results on the statistical and approximation error Batlle, Y. Chen, et al., 2025a; Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b. As a result, we place our initial focus on developing a GP-based approach for solving roughly forced arbitrary nonlinear PDEs, such as SPDEs, and subsequently extend the ensuing framework to ANN-based methods, such as PINNs.

Contributions and Outline

In the context of problem (2.1), we consider the operator $\mathcal{P} \times \text{Tr} : H^t(\Omega) \rightarrow H^{-s}(\Omega) \times H^{t-\frac{1}{2}}(\partial\Omega)$ that maps the solution u to the right hand side, where $\mathcal{P}$ is a nonlinear differential operator and Tr is the trace operator. The forcing term ξ belongs to $H^{-s}(\Omega)$ and the boundary term g belongs to $H^{t-\frac{1}{2}}(\partial\Omega)$. In this setting, we assume the existence of a unique solution $u^* \in H^t(\Omega)$. To solve the PDE we first reformulate (2.1) as an infinite-dimensional optimal recovery problem within a reproducing kernel Hilbert space (RKHS). This space will be potentially misspecified in the sense that the true solution u^* is not necessarily contained in it. This reformulation, which we term the *continuum problem*, seeks to identify an optimal approximation to the solution in the RKHS that accounts for the misspecification by incorporating additive singular noise in the source term of (2.1). We then formulate a *discretized problem* by (1) approximating the negative Sobolev norms using test functions and (2) solving the corresponding optimal recovery problem through Gauss-Newton iteration. This process reduces to solving a sequence of weakly defined linearized PDEs with the kernel corresponding to the underlying misspecified RKHS. In this section, we describe the continuum and discretized problem followed by the contributions and outline of this work.

Continuum problem Consider a kernel $K : \overline{\Omega} \times \overline{\Omega} \rightarrow \mathbb{R}$ and its associated RKHS, which we denote by $\mathcal{H}_K$. Given a regularization parameter $\gamma > 0$, we approximate the solution to PDE (2.1) by a misspecified element of $\mathcal{H}_K$ that is obtained from a minimizer of the following *continuum loss* in (2.3). The loss relaxes the PDE constraint using a negative Sobolev H^{-s} -norm rather than the usual (possibly empirical) L^2 norm that is commonly employed in Scientific Machine

Learning. The resulting continuum optimization problem is given by

$$\inf_{u \in \mathcal{H}_K} \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \gamma \|u\|_{\mathcal{H}_K}^2 \quad (2.3a)$$

$$\text{s.t. } u(x) = g(x) \quad x \in \partial\Omega. \quad (2.3b)$$

For ease of presentation, we assume the boundary condition is regular enough to be satisfied by an element of the RKHS². We show under mild assumptions that for any $\gamma > 0$, problem (2.3) has a minimizer $u^\gamma \in \mathcal{H}_K$ and that any minimizing sequence has a convergent subsequence to a minimizer in $\mathcal{H}_K$. Furthermore, u^γ converges to the solution of the PDE (2.1) as the regularization parameter γ goes to 0. The use of a negative Sobolev H^{-s} -loss, which is weaker than the usual L^2 loss, for machine learning solvers was already explored in the context of linear elliptic PDEs in Bonito et al., 2024 and by hybrid methods combining machine learning and traditional numerical schemes Karnakov, Litvinov, and Koumoutsakos, 2024.

Discretized problem Problem eq. (2.3) is an optimization problem over the space $\mathcal{H}_K$ and cannot be directly solved computationally. We introduce a *discretized* approximation of this problem based on a N -dimensional test space $\Phi^N := \text{span}\{\varphi_i\}_{i=1}^N \subset H_0^s(\Omega)$ and a set of M collocation points on the boundary $\partial\Omega$. That is, the discretized problem is given by

$$\inf_{u \in \mathcal{H}_K} |\mathcal{P}(u) - \xi|_{\Phi^N}^2 + \gamma \|u\|_{\mathcal{H}_K}^2, \quad (2.4a)$$

$$\text{s.t. } u(x_j) = g(x_j) \quad \text{for } j = 1, \dots, M. \quad (2.4b)$$

For simplicity, we have assumed g to be continuous on the boundary³. We will show that the seminorm $|\cdot|_{\Phi^N}$ approximates the negative Sobolev norm $\|\cdot\|_{H^{-s}}$, and that the minimizer $u^{\gamma, N, M}$ of eq. (2.4) converges to the true solution of the PDE (2.1) as the regularization $\gamma \rightarrow 0$ and as the number of measurements (given by test functions and collocation points) satisfy $(M, N) \rightarrow \infty$. We then formulate a Gauss-Newton iteration to solve the discretized problem, which leads to a computational method to solve roughly forced PDEs of the form in (2.1). The formal derivation, theoretical analysis and numerical implementation of this algorithm will be the object of this paper. The main contributions are outlined next.

²The ensuing methodology can be extended to the setting where the boundary condition only applies in a trace sense by relaxing the boundary condition as well.

³As for the continuum problem, the proposed approach can naturally be extended to the setting where the boundary condition has to be defined in a weak/trace sense.

Contributions In this work we provide the following theoretical contributions:

- (T1) We propose a loss based on a negative Sobolev norm, which is weaker than the L^2 norm and adapted to the problem⁴. Minimizing this loss can be interpreted as solving a weak form of the PDE and can be applied to general machine-learning solvers, including kernel methods and PINNs.
- (T2) We show that under mild assumptions on the problem, the proposed kernel method is convergent as the regularization parameter γ goes to 0, hence allowing the recovery of solutions that are very irregular. We thus extend the guarantees present in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b beyond the setting of classical solutions. This approach can accommodate both linear and nonlinear PDEs.
- (T3) We introduce a new concept for the solution of PDEs defined by irregular data, allowing regularization of each term in the PDE via a Gaussian process, and resulting in an optimization problem which includes the proposed Sobolev loss as a special case.

From a computational perspective, we provide the following contributions:

- (C1) We propose an efficient numerical discretization of the Sobolev norm, which inherits the complexity of state-of-the-art solvers for dense kernel matrices Houman Owhadi and Scovel, 2019b.
- (C2) We provide an efficient numerical method to minimize the resulting loss based on the Gauss-Newton algorithm.
- (C3) We provide convergence guarantees for the resulting numerical method, extending the convergence results in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b; Battle, Y. Chen, et al., 2025a to the case where the solution does not belong to the RKHS underlying the optimization problem.
- (C4) We demonstrate the competitiveness of the kernel-based framework for roughly forced PDEs on a range of numerical examples.

⁴We note that previous work such as the Deep Ritz method E and B. Yu, 2018 can be interpreted as using a negative Sobolev norm, but these are applied to specific PDEs amenable to a variational formulation. Here we consider generic negative norms and generic non-linear PDEs.

- (C5) We demonstrate the flexibility of the ideas presented by using the loss in the context of PINNs applied to a range of roughly forced PDEs.

Outline The remainder of the paper is structured as follows. In Section 2.1 we introduce notation that will be used throughout. Section 2.2 presents the formal kernel method in the continuum setting for minimization problem (2.3) and investigates the theoretical properties of its minimizer. Hence, this section relates to Contributions (T1) and (T2). In Section 2.3 we derive a discretization of the continuum problem, corresponding to (2.3), resulting in an implementable numerical method with convergence guarantees. Hence, this section relates to Contributions (C1) to (C3). In Section 2.4 we present numerical experiments demonstrating the performance of the proposed kernel method and a PINN variant using the proposed equation-adapted norm on several nonlinear PDEs; we compute empirical convergence rates and compare the methodology to finite element and spectral Galerkin methods, thereby addressing Contributions (C4) and (C5). Our code to reproduce the numerical experiments is available in a public repository ⁵. Finally, using the same kernel-based framework we have developed, Section 2.5 introduces a *generalized continuum problem* involving regularization of nonlinearities and terms under the action of differential operators; this section concerns Contribution (T3).

Notation

We denote the positive integers and non-negative integers respectively by $\mathbb{N} = \{1, 2, \dots\}$ and $\mathbb{Z}^+ = \{0, 1, \dots\}$ and use the notation $\mathbb{R} = (-\infty, \infty)$ for the reals. For a set D , we use $\overline{D}$ to denote the closure of the set, i.e. the union of the set itself and the set of all its limit points. Throughout, $\Omega \subset \mathbb{R}^d$ will be a bounded open domain with Lipschitz boundary $\partial\Omega$. We let $C(\overline{\Omega})$ denote the infinite-dimensional Banach space of continuous functions taking inputs in Ω . The space is endowed with the supremum norm. Similarly $C^r(\overline{\Omega})$ for $r \geq 0$ will denote the space of r -times continuously differentiable functions over Ω . For $p \in [1, \infty)$ we let $L^p(\Omega)$ denote the infinite dimensional space of p -integrable functions over Ω . The space is endowed with the L^p norm. For $p \in [1, \infty)$ and $t > 0$, $W^{t,p}(\Omega)$ denote the (fractional, if $t \notin \mathbb{N}$) Sobolev space of functions. We recall that setting $p = 2$ defines a Hilbert space which we denote by $H^t(\Omega)$. We define the space $H_0^t(\Omega)$ as the closure under the $H^t(\Omega)$ norm of the set of smooth compactly supported functions on Ω . We use the notation $H_g^t(\Omega)$ to denote the space of functions $u \in H^t(\Omega)$ such that

⁵<https://github.com/MatthieuDarcy/Kernel-NeuralNetworks-SPDEs>

applying the trace operator gives $\text{Tr}(u) = g \in H^{t-1/2}(\partial\Omega)$. For ease of presentation, we will keep g fixed and only consider $\mathcal{P} : H_g^t(\Omega) \rightarrow H^{-s}(\Omega)$, keeping the trace operator implicit. Throughout the article, we will use the calligraphic notation $\mathcal{H}_K$ to denote the reproducing kernel Hilbert space (RKHS) associated to the kernel K .

For notational convenience, we will drop Ω for each of the introduced function spaces as this is implied throughout. Furthermore, we let $\|\cdot\|_{\mathcal{U}}$ denote the norm that is endowed with each space $\mathcal{U}$, and similarly let $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ denote the associated inner product to each inner product space $\mathcal{H}$. Convergence of a sequence $(u_n)_{n \in \mathbb{N}} \subset \mathcal{U}$ in the strong topology given by the norm on $\mathcal{U}$ to an element $u \in \mathcal{U}$ will be denoted by $u_n \rightarrow u$, while weak convergence will be denoted by $u_n \rightharpoonup u$. For a normed space $\mathcal{U}$, we will let $B_{\mathcal{U}}(f, \delta) \subset \mathcal{U}$ denote the open ball in $\mathcal{U}$ of radius δ centered at $f \in \mathcal{U}$ i.e.,

$$B_{\mathcal{U}}(f, \delta) := \{g \in \mathcal{U} : \|f - g\|_{\mathcal{U}} < \delta\}.$$

For function spaces $\mathcal{U}$ and $\mathcal{V}$ we write $\mathcal{U} \Subset \mathcal{V}$ to denote $\mathcal{U}$ being compactly embedded in $\mathcal{V}$.

2.2 The Continuum Problem

We begin our discussion by recalling the PDE (2.1) where $\mathcal{P} : H_g^t \rightarrow H^{-s}$ for $t > \frac{1}{2}, s > 0$ is a possibly nonlinear differential operator. We assume that there exists a unique $u^* \in H_g^t$ satisfying the PDE (2.1).

Remark 1. *In general, the boundary condition should be interpreted in a trace sense. If $t > \frac{d}{2}$, however, then the boundary condition can be interpreted pointwise. We will make this assumption, but we note that the methodology can also be extended to the case where the boundary condition is defined in a trace sense by enforcing the condition in a weak sense.*

We recall that $\mathcal{H}_K$ is an RKHS, with associated kernel $K : \overline{\Omega} \times \overline{\Omega} \rightarrow \mathbb{R}$ (see for example Aronszajn, 1950; Bogachev, 1998a; C. Berg, 1984 for an introduction to the theory of RKHS). For $\gamma \in \mathbb{R}^+$, we consider the regularized minimization problem:

$$\inf_{u \in \mathcal{H}_K} \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \gamma \|u\|_{\mathcal{H}_K}^2, \quad (2.5a)$$

$$\text{s.t. } u = g \text{ on } \partial\Omega. \quad (2.5b)$$

We introduce the following assumptions on the differential operator $\mathcal{P}$ and the RKHS $\mathcal{H}_K$.

Assumption (Assumption P). We assume that the differential operator $\mathcal{P}$ satisfies:

(P1) The operator $\mathcal{P} : H_g^t \rightarrow H^{-s}$ is continuous.

(P2) The solution operator associated to $\mathcal{P}$ is locally stable. That is, there exists a $\delta \in \mathbb{R}^+$ and a constant $C(\delta) \in \mathbb{R}^+$ such that for any $u \in H_g^t$ with $\mathcal{P}(u) \in B_{H^{-s}}(\xi, \delta)$, then

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

Assumption (Assumption H). We assume that the kernel $K : \overline{\Omega} \times \overline{\Omega} \rightarrow \mathbb{R}$ is such that the RKHS $\mathcal{H}_K$ satisfies:

(H1) The embeddings $\mathcal{H}_K \Subset C(\overline{\Omega})$ and $\mathcal{H}_K \Subset H^t$ are compact.

(H2) $\mathcal{H}_K$ contains a dense subset of H_g^t ⁶.

The following discussion will focus on identifying the assumptions under which the recovery of u^* can be achieved using a scheme based on the regularized problem in (2.5). Specifically, in Subsection 2.2, we demonstrate that under Assumptions (P1) and (H1), a solution to (2.5) exists for any $\gamma \in \mathbb{R}^+$. In Subsection 2.2 we show, under Section 2.2 and Section 2.2, $(u_n)_{n \in \mathbb{N}} \subset \mathcal{H}_K$ solving (2.5) for $\gamma = \gamma_n \in \mathbb{R}^+$, converges to $u^* \in H_g^t$ solving (2.1) as the sequence $(\gamma_n)_{n \in \mathbb{N}} \subset \mathbb{R}^+$ decreases to zero.

Existence of a Minimizer

In this subsection we prove the existence of a solution to the regularized optimization problem in (2.5) for any $\gamma \in \mathbb{R}^+$. Define the objective

$$J(u; \gamma) := \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \gamma \|u\|_{\mathcal{H}_K}^2, \quad (2.6)$$

and consider the infimization problem

$$\inf_{u \in \mathcal{H}_K} J(u; \gamma) \quad (2.7a)$$

$$\text{s.t. } u = g \text{ on } \partial\Omega. \quad (2.7b)$$

We assume that there is at least one $u \in \mathcal{H}_K$ which satisfies the boundary condition (note that $\mathcal{H}_K \Subset C(\overline{\Omega})$ in Assumption (H1) implies that $\mathcal{H}_K$ is of sufficient regularity to enforce the boundary condition pointwise).

⁶This assumption can be replaced by the more restrictive one that $\mathcal{H}_K$ is such that $C^\infty \subset \mathcal{H}_K$. This assumption is often more easily checked and is the case for example with the Matérn class of kernels Fasshauer and Q. Ye, 2011.

We are now able to state and prove the following existence theorem.

Theorem 1 (Existence of a Minimizer). *Let Assumption (P1) be satisfied and let the kernel K be such that Assumption (H1) is satisfied. Let m be the optimal value of problem (2.7a), i.e.,*

$$m := \inf_{u \in \mathcal{H}_K} J(u; \gamma) \\ \text{s.t. } u = g \text{ on } \partial\Omega.$$

Then it holds that $0 \leq m < \infty$ and that there exists $u_\infty \in \mathcal{H}_K$ such that $m = J(u_\infty; \gamma)$. Moreover, if $(u_n)_{n \in \mathbb{N}}$ is a minimizing sequence, then there is a subsequence which converges strongly to $\bar{u}_\infty$ (possibly distinct from u_∞) in $\mathcal{H}_K$ such that $m = J(\bar{u}_\infty; \gamma)$. Therefore, if the minimizer of J is unique then the entire sequence u_n converges to u_∞ .

Proof. To prove the existence of a minimizer, we will proceed in two steps. In the first, we will consider a minimizing sequence satisfying the boundary conditions and show boundedness in $\mathcal{H}_K$, which will yield a weakly convergent subsequence. In the second, we will show weak lower semi-continuity of the objective on $\mathcal{H}_K$ along any subsequence satisfying the boundary conditions (not necessarily a minimizing subsequence), which when combined with the first step will yield the result. We reformulate our optimization problem as

$$\inf_{u \in \mathcal{H}_K} L(u; \gamma) := \inf_{u \in \mathcal{H}_K} \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \gamma \|u\|_{\mathcal{H}_K}^2 + \chi\left(\sup_{x \in \partial\Omega} |u(x) - g(x)|\right),$$

where

$$\chi(a) = \begin{cases} 0 & \text{if } a = 0, \\ \infty & \text{otherwise.} \end{cases}$$

Step 1: Existence of weak limit. We first observe that $L(u; \gamma) \in [0, \infty)$ for all $u \in \mathcal{H}_K$ satisfying the boundary condition. Therefore $0 \leq m < \infty$. Consider some minimizing sequence $(u_n)_{n \in \mathbb{N}}$ so that

$$L(u_n; \gamma) \rightarrow m \quad \text{as } n \rightarrow \infty.$$

Without loss of generality, we assume that $L(u_n; \gamma) < \infty$ for all n and hence the minimizing sequence satisfies the boundary conditions. By the definition of L , the minimizing sequence is bounded, i.e., there exists $C \in \mathbb{R}^+$ so that

$$\|u_n\|_{\mathcal{H}_K} \leq C,$$

for any $n \in \mathbb{N}$. By the Banach–Alaoglu theorem Alaoglu, 1940, since $\mathcal{H}_K$ is reflexive, there exists $u_\infty \in \mathcal{H}_K$ and a subsequence $(u_{n_k})_{k \in \mathbb{N}}$ such that $u_{n_k} \rightharpoonup u_\infty$ in $\mathcal{H}_K$ as $k \rightarrow \infty$.

Step 2: Weak lower semicontinuity of the objective.. We now show the weak-lower semicontinuity of the objective along any subsequence satisfying the boundary conditions (it is not required that this be a minimizing sequence). By Assumption (H1) and by Brezis, 2010, p.171 we deduce that $u_{n_k} \rightarrow u_\infty$ in H_g^t as $k \rightarrow \infty$. Therefore, by Assumption (P1), we obtain

$$\|\mathcal{P}(u_{n_k}) - \xi\|_{H^{-s}}^2 \rightarrow \|\mathcal{P}(u_\infty) - \xi\|_{H^{-s}}^2 \quad \text{as } k \rightarrow \infty.$$

We note that by the same argument L is also weakly continuous on $\mathcal{H}_K$. Likewise, by Assumption (H1), it follows that $u_{n_k} \rightarrow u_\infty$ in $C(\overline{\Omega})$ as $k \rightarrow \infty$. Since u_{n_k} satisfies the boundary conditions for all k , this implies that $u_\infty = g$ for $x \in \partial\Omega$ and

$$\chi\left(\sup_{x \in \partial\Omega} |u_{n_k}(x) - g(x)|\right) = \chi\left(\sup_{x \in \partial\Omega} |u_\infty(x) - g(x)|\right) = 0.$$

By the same argument this term is also lower-semicontinuous. Since norms are lower-semicontinuous Brezis, 2010, p. 61, it follows that $L : \mathcal{H}_K \rightarrow \mathbb{R}$ is lower-semicontinuous. Therefore

$$u_{n_k} \rightharpoonup u_\infty \Rightarrow \liminf_{n \rightarrow \infty} L(u_{n_k}; \gamma) \geq L(u_\infty; \gamma),$$

and consequently, since $\lim_{n \rightarrow \infty} L(u_n; \gamma) = m$, we have

$$m = L(u_\infty; \gamma).$$

Step 3: Strong convergence of the minimizing sequence. We now turn our attention to showing the existence of a subsequence of $(u_n)_{n \in \mathbb{N}}$ that converges strongly to a minimizer. For ease of exposition, let

$$\Psi(u) := \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \chi\left(\sup_{x \in \partial\Omega} |u(x) - g(x)|\right),$$

so that

$$L(u) = \Psi(u) + \gamma \|u\|_{\mathcal{H}_K}^2.$$

Again, without loss of generality, we assume that the minimizing sequence satisfies the boundary condition so that the penalty corresponding to the boundary condition is 0. Using the polarization equality, we have that

$$\begin{aligned} \frac{\gamma}{4} \|u_l - u_k\|_{\mathcal{H}_K}^2 &= \gamma \left(\frac{1}{2} \|u_l\|_{\mathcal{H}_K}^2 + \frac{1}{2} \|u_k\|_{\mathcal{H}_K}^2 - \left\| \frac{1}{2} (u_l + u_k) \right\|_{\mathcal{H}_K}^2 \right) \\ &= \frac{1}{2} L(u_l) + \frac{1}{2} L(u_k) - L\left(\frac{1}{2} (u_l + u_k)\right) - \frac{1}{2} \Psi(u_l) - \frac{1}{2} \Psi(u_k) + \Psi\left(\frac{1}{2} (u_l + u_k)\right), \end{aligned}$$

for any $l, k \in \mathbb{N}$. Letting $l, k \geq N_1(\delta)$ so that $L(u_k) \leq m + \delta$ and $L(u_l) \leq m + \delta$, we obtain

$$\begin{aligned} \frac{\gamma}{4} \|u_l - u_k\|^2 &\leq m + \delta - m - \frac{1}{2}\Psi(u_l) - \frac{1}{2}\Psi(u_k) + \Psi\left(\frac{1}{2}(u_l + u_k)\right) \\ &= \delta - \frac{1}{2}\Psi(u_l) - \frac{1}{2}\Psi(u_k) + \Psi\left(\frac{1}{2}(u_l + u_k)\right). \end{aligned}$$

Since u_l, u_k and $\frac{1}{2}(u_l + u_k)$ converge strongly to u_∞ in H^t and in $C(\overline{\Omega})$, and satisfy the boundary conditions, we deduce by the continuity of $\mathcal{P}$ from Assumption (P1) that there is integer $N_2(\delta)$ such that for $k, l \geq N_2(\delta)$:

$$\frac{\gamma}{4} \|u_l - u_k\|_{\mathcal{H}_K}^2 \leq \delta + \delta = 2\delta.$$

Therefore, the sequence $(u_n)_{n \in \mathbb{N}}$ is Cauchy in $\mathcal{H}_K$ and as a consequence converges strongly, i.e., $u_n \rightarrow u_\infty$ in $\mathcal{H}_K$ as $n \rightarrow \infty$. $\square$

Convergence as Regularizer is Removed

We now turn our attention to showing that the sequence $(u_n)_{n \in \mathbb{N}} \subset \mathcal{H}_K$ where u_n is the solution to (2.5) for $\gamma = \gamma_n \in \mathbb{R}^+$, converges to $u^* \in H_0^t$ for any sequence $(\gamma_n)_{n \in \mathbb{N}} \subset \mathbb{R}^+$ decreasing to 0. This key result lays the groundwork for the development of an implementable numerical method for the solution of (2.5). To prove the convergence of the numerical method to the true solution, previous work on the use of Gaussian processes to solve PDEs Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b assumes that the true solution belongs to the RKHS, i.e., $u^* \in \mathcal{H}_K$. In this work we treat the case where $u^* \notin \mathcal{H}_K$; extending to this setting is natural for roughly forced PDEs where, due to the lack of regularity of the right-hand side, the solution may live outside the RKHS defined by the kernel.

Theorem 2 (Convergence). *Let Section 2.2 and Section 2.2 hold and let $\xi \in H^{-s}$. For any $n \in \mathbb{N}$, let u_n denote a minimizer of (2.5) with regularizer $\gamma = \gamma_n$. Then, for all sequences $(\gamma_n)_{n \in \mathbb{N}} \subset \mathbb{R}^+$ decreasing to 0, the sequence $(u_n)_{n \in \mathbb{N}} \subset \mathcal{H}_K$ converges to $u^* \in H_g^t$ in H^t and we have*

$$\|\mathcal{P}(u_n) - \xi\|_{H^{-s}}^2 + \gamma_n \|u_n\|_{\mathcal{H}_K}^2 \rightarrow 0.$$

Proof. We will need the following lemma whose proof is straightforward.

Lemma 1 (Density). *Under Assumptions (P1) and (H2), the image of $\mathcal{H}_K$ under $\mathcal{P}$ denoted as $\mathcal{P}(\mathcal{H}_K)$ is dense in $\mathcal{P}(H_g^t)$.*

From theorem 1 we can define u_n as a minimizer of (2.5) with regularizer γ_n . lemma 1 implies that there exists a sequence $(\bar{w}_n)_{n \in \mathbb{N}}$ in $\mathcal{H}_K$ such that $\mathcal{P}(\bar{w}_n) \rightarrow \xi$ in H^{-s} and $\bar{w}_n = g$ on $\partial\Omega$. For $n \geq 0$, let

$$a_n := \max\{m \leq n \mid \|\bar{w}_m\|_{\mathcal{H}_K} \leq \|\bar{w}_0\|_{\mathcal{H}_K} + \gamma_n^{-\frac{1}{3}}\}.$$

Note that $\lim_{n \rightarrow \infty} \gamma_n = 0$ implies that $\lim_{n \rightarrow \infty} a_n = \infty$ and

$$\lim_{n \rightarrow \infty} \gamma_n^{\frac{1}{2}} \|\bar{w}_{a_n}\|_{\mathcal{H}_K} \leq \lim_{n \rightarrow \infty} \gamma_n^{\frac{1}{2}} \|\bar{w}_0\|_{\mathcal{H}_K} + \gamma_n^{\frac{1}{2} - \frac{1}{3}} = 0.$$

Therefore, if we define $w_n := \bar{w}_{a_n}$, we deduce that $(w_n)_{n \in \mathbb{N}}$ is a sequence in $\mathcal{H}_K$ such that $\mathcal{P}(w_n) \rightarrow \xi$ in H^{-s} , $w_n = g$ on $\partial\Omega$ and

$$\|\mathcal{P}(w_n) - \xi\|_{H^{-s}}^2 + \gamma_n \|w_n\|_{\mathcal{H}_K}^2 \rightarrow 0.$$

Now, since

$$\|\mathcal{P}(u_n) - \xi\|_{H^{-s}}^2 + \gamma_n \|u_n\|_{\mathcal{H}_K}^2 \leq \|\mathcal{P}(w_n) - \xi\|_{H^{-s}}^2 + \gamma_n \|w_n\|_{\mathcal{H}_K}^2,$$

we have that

$$\|\mathcal{P}(u_n) - \xi\|_{H^{-s}}^2 \rightarrow 0.$$

Therefore, Assumption (P2) and the fact that u_n satisfies the boundary conditions implies

$$\|u_n - u^*\|_{H_0^t} \rightarrow 0.$$

□

It is of general interest to investigate the rate at which the solution to the continuum problem (2.5) converges to the true solution u^* as the regularizer $\gamma \rightarrow 0$. We provide insight into identifying an explicit convergence rate in the following remark.

Remark 2 (Convergence for the Poisson Equation). *For the purposes of this discussion consider the Poisson equation*

$$-\Delta u = \xi, \quad x \in \Omega \tag{2.10a}$$

$$u = 0, \quad x \in \partial\Omega. \tag{2.10b}$$

Assume the solution u^* of (2.10) to live in H_0^t , and ξ in H^{-s} . We may make the specific choice of RKHS $\mathcal{H}_K = H^q$ (e.g., by letting K be a Matérn kernel), for some $q > t$, so that the solution space is not contained in the RKHS. In Section A.1 we demonstrate the convergence of u^γ to u^* as $\gamma \rightarrow 0$ in the H^α norm for any $\alpha \leq t = 2 - s$. We provide further details in Section A.1.

2.3 The Numerical Method

In this section, we present an algorithm for solving the constrained optimization problem:

$$\inf_{u \in \mathcal{H}_K} \|\mathcal{P}(u) - \xi\|_{H^{-s}}^2 + \gamma \|u\|_{\mathcal{H}_K}^2, \quad (2.11a)$$

$$\text{s.t. } u = g \text{ on } \partial\Omega. \quad (2.11b)$$

The section is structured as follows. First, in Section 2.3, we introduce a discretization of the negative Sobolev norm $\|\cdot\|_{H^{-s}}$, which reduces (2.11) to a nonlinear least-squares problem. This approximation of the norm relies on the choice of a test space. In Section 2.3, we relax the boundary condition to be satisfied at only a finite set of collocation points, and we employ a variant of the Gauss-Newton algorithm to solve the resulting infinite-dimensional minimization problem. The representer theorem allows us to obtain a finite-dimensional representation of the solution for each iteration of Gauss-Newton. Finally, in Section 2.3 we present convergence results for the resulting numerical method. The derivation of this numerical method and its proof of convergence will be based on the following density assumption on the test space and collocation points (Section 2.3) and regularity assumption on the differential operator (Section 2.3).

Assumption (Assumption D). *We assume that the N -dimensional test space $\Phi^N := \text{span}\{\varphi_i\}_{i=1}^N \subset H_0^s$ and set of collocation points on the boundary $\{x_j\}_{j=1}^M \subset \partial\Omega$ satisfy:*

(D1) *It holds that Φ^N becomes dense in H_0^s as $N \rightarrow \infty$, i.e.,*

$$\overline{\text{span}\{\varphi_i | i \in \mathbb{N}\}} = H_0^s.$$

(D2) *The fill-distance on the boundary is such that*

$$\sup_{x \in \partial\Omega} \inf_{x_j \in \{x_i\}_{i=1}^M} |x - x_j|_{\partial\Omega} \rightarrow 0, \quad \text{as } M \rightarrow \infty.$$

Assumption (Assumption R). *Assume that $\xi \in H^{-s+\delta}$ for some $\delta \in \mathbb{R}^+$, and assume $\mathcal{P} : H_g^t \rightarrow H^{-s+\delta}$.*

Intuitively Section 2.3 is necessary to ensure that the measurements sufficiently “fill out” the space without gaps. We note that Assumption (D2) is standard in the kernel

interpolation literature. Section 2.3 will be used to apply a compactness argument to ensure convergence of the numerical method. Throughout this section we consider the assumptions of Section 2.2 to also be satisfied. In particular we consider that Section 2.2 and Section 2.2 are satisfied, that g is of sufficient regularity to enforce the boundary condition pointwise, and that there is at least one $u \in \mathcal{H}_K$ which satisfies the boundary condition.

Approximating The Negative Sobolev Norm

We now derive a method for approximating the H^{-s} norm in problem (2.11). First, recall from Henry, 2006 that one can define a norm (equivalent to the usual $\|\cdot\|_{H^s}$ norm) on the space H_0^s as

$$\|u\|_{H_0^s}^2 = \int u(-\Delta)^s u, \quad (2.12)$$

where Δ is the Laplacian operator with homogeneous Dirichlet boundary conditions. We note that this allows for negative s as well. In this case, the operator is the solution operator (Green's function) of the corresponding Laplace equation with homogeneous Dirichlet boundary conditions. One can additionally consider fractional values of s through the spectral expansion of the negative Laplacian. The negative Sobolev norm $\|\cdot\|_{H^{-s}}$ is then defined as the dual norm to the space H_0^s with respect to the L^2 duality

$$\|f\|_{H^{-s}} := \sup_{u \in H_0^s} \frac{\int f u}{\|u\|_{H_0^s}}. \quad (2.13)$$

This supremum in (2.13) cannot be directly computed as it involves solving an optimization problem over an infinite-dimensional space. Our strategy is to replace the infinite-dimensional space H_0^s by a finite-dimensional test space:

$$\Phi^N := \text{span}\{\varphi_i\}_{i=1}^N \subset H_0^s.$$

Appropriate choices of Φ^N will be discussed later. We further assume that the set $\{\varphi_i\}_{i=1}^N$ is comprised of linearly independent functions. We may now define the seminorm

$$|f|_{\Phi^N} := \sup_{u \in \Phi^N} \frac{\int f u}{\|u\|_{H_0^s}}. \quad (2.14)$$

We make use of the vector notation

$$[f, \varphi] := \left(\int f \varphi_1, \int f \varphi_2, \dots, \int f \varphi_n \right)^\top \in \mathbb{R}^N$$

and define the matrix $A \in \mathbb{R}^{N \times N}$ element-wise as

$$A_{i,j} := \int \varphi_i(-\Delta)^s \varphi_j,$$

for $1 \leq i, j \leq N$. The following result provides an explicit expression for the seminorm in (2.14).

Proposition 1. *For any $f \in H^{-s}$, the seminorm $|\cdot|_{\Phi^N}$ satisfies*

$$|f|_{\Phi^N} = \sqrt{[f, \varphi]^\top A^{-1} [f, \varphi]}. \quad (2.15)$$

Proof. See Section A.1. □

Remark 3. *Another view of our approximation to the norm comes from considering the dual of our test space $\Psi^N := \{(-\Delta)^s \varphi_i\}_{i=1}^N \subset H^{-s}$. One can consider the H^{-s} projection of f onto Ψ^N , denoted as $\Pi^N f$. Then, one can show*

$$|f|_{\Phi^N} = \|\Pi^N f\|_{H^{-s}} \leq \|f\|_{H^{-s}}.$$

Hence, our seminorm can be seen as the norm of the H^{-s} projection of f onto $(-\Delta)^s \Phi^N$.

The next proposition ensures the consistency of our seminorm as the number of test functions $N \rightarrow \infty$. We first make an assumption on our choice of test space.

Proposition 2. *Under Assumption (D1), it holds that for all $f \in H^{-s}$*

$$\lim_{N \rightarrow \infty} |f|_{\Phi^N} \rightarrow \|f\|_{H^{-s}}.$$

Proof. See Section A.1. □

Writing $\Phi^{N,\perp}$ for the orthogonal complement of Φ^N in H^{-s} , we may also define the orthogonal seminorm as

$$|f|_{\Phi^{N,\perp}} := \sup_{u \in \Phi^{N,\perp}} \frac{\int f u}{\|u\|_{H_0^s}}.$$

Since $|f|_{\Phi^{N,\perp}}$ is the H^{-s} -norm of the H^{-s} -projection of f onto $\Phi^{N,\perp}$, we have the orthogonal decomposition

$$\|f\|_{H^{-s}}^2 = |f|_{\Phi^N}^2 + |f|_{\Phi^{N,\perp}}^2. \quad (2.16)$$

Combining this identity with proposition 2 leads to the following corollary.

Corollary 1. For $f \in H^{-s}$, the family of seminorms $|\cdot|_{\Phi^N}$ satisfy

$$\lim_{N \rightarrow \infty} |f|_{\Phi^{N,\perp}} \rightarrow 0.$$

Remark 4. (Choice of test space Φ^N). A standard choice for the basis functions of Φ^N is to use polyharmonic splines Wendland, 2004. With this selection, the Gram matrix A , its inverse, and the projected norms can be computed with near-linear complexity using gamblets Houman Owhadi and Scovel, 2019b. However, this approach is not explored in the present work. Other options include a Fourier or wavelet basis. We expect, based on previous work Houman Owhadi and Scovel, 2019b, Chap. 14, that these different choices yield the same rates of convergence but with different constants. In our numerical experiments, we do observe varying levels of performance but did not systematically investigate this.

Nonlinear Least-squares Problem and the Gauss-Newton Algorithm

We now consider the approximation in (2.15) for the negative Sobolev norm, and relax the boundary condition to be satisfied only at a finite set of collocation points $\{x_j\}_{j=1}^M \subset \partial\Omega$. This leads to the following minimization problem for the regularized PDE solution:

$$\inf_{u \in \mathcal{H}_K} |\mathcal{P}(u) - \xi|_{\Phi^N}^2 + \gamma \|u\|_{\mathcal{H}_K}^2, \quad (2.17a)$$

$$\text{s.t. } u(x_j) = g(x_j) \quad \text{for } j = 1, \dots, M, \quad (2.17b)$$

which by proposition 1, is equivalent to

$$\inf_{u \in \mathcal{H}_K} [\mathcal{P}(u) - \xi, \varphi]^\top A^{-1} [\mathcal{P}(u) - \xi, \varphi] + \gamma \|u\|_{\mathcal{H}_K}^2 \quad (2.18a)$$

$$\text{s.t. } u(x_j) = g(x_j) \quad \text{for } j = 1, \dots, M. \quad (2.18b)$$

Similarly to Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b, we solve (2.18) via a variant of the Gauss-Newton algorithm, which successively linearizes the nonlinear least-squares problem Nocedal and Stephen J. Wright, 2006b. We assume that $\mathcal{P}$ is Fréchet differentiable with bounded derivative. Writing $u_{n+1} = u_n + \delta u$ for the iterations of the Gauss-Newton algorithm, the increment δu is chosen such that

$$\delta u = \arg \inf_{v \in \mathcal{H}_K} [\mathcal{P}(u_n) + \mathcal{P}'(u_n)v - \xi, \varphi]^\top A^{-1} [\mathcal{P}(u_n) + \mathcal{P}'(u_n)v - \xi, \varphi] + \gamma \|u_n + v\|_{\mathcal{H}_K}^2$$

$$\text{s.t. } u_n(x_j) + \delta u(x_j) = g(x_j) \quad \text{for } j = 1, \dots, M.$$

Equivalently, we can identify u_{n+1} directly by solving

$$u_{n+1} = \arg \inf_{v \in \mathcal{H}_K} [\mathcal{P}(u_n) + \mathcal{P}'(u_n)(v - u_n) - \xi, \varphi]^\top A^{-1} [\mathcal{P}(u_n) + \mathcal{P}'(u_n)(v - u_n) - \xi, \varphi] + \gamma \|v\|_{\mathcal{H}_K}^2 \quad (2.20a)$$

$$\text{s.t. } v(x_j) = g(x_j) \quad \text{for } j = 1, \dots, M. \quad (2.20b)$$

The next theorem provides a representer formula for the minimizer of (2.20).

Theorem 3 (Representer Theorem). *For a fixed u , denote χ_i to be the linear functional whose action is defined as*

$$[\chi_i, w] = \int (\mathcal{P}'(u)(x)w(x))\varphi_i(x)dx. \quad (2.21)$$

We use $K(\cdot, \chi_i)$ to denote the function

$$K(\cdot, \chi_i) := \int (\mathcal{P}'(u)(x)K(\cdot, x))\varphi_i(x)dx$$

i.e. the action of χ_i on the the second argument of K . The infimizer

$$\hat{u} := \arg \inf_{v \in \mathcal{H}_K} [\mathcal{P}(u) + \mathcal{P}'(u)(v - u) - \xi, \varphi]^\top A^{-1} [\mathcal{P}(u) + \mathcal{P}'(u)(v - u) - \xi, \varphi] + \gamma \|v\|_{\mathcal{H}_K}^2 \quad (2.22a)$$

$$\text{s.t. } v(x_j) = g(x_j) \quad \text{for } j = 1, \dots, M, \quad (2.22b)$$

admits a representation of the form

$$\hat{u} = \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j),$$

for some scalars $\{\alpha_i\}_{i=1}^N, \{\beta_j\}_{j=1}^M \subset \mathbb{R}$.

Proof. See Section A.1. □

Using theorem 3 it is possible to compute the solution to (2.20). The result is given in the next proposition. To describe this, we introduce the following notation: denote $K(\cdot, \psi_i) = K(\cdot, \chi_i)$ when $1 \leq i \leq N$ and $K(\cdot, \psi_i) = K(\cdot, x_{i-N})$ when $N + 1 \leq i \leq M + N$ (i.e. $\psi = [\chi_1, \chi_2, \dots, \chi_1, \dots, x_m]$). We will also adopt the following matrix notation:

$K(\psi, \psi) \in \mathbb{R}^{(N+M) \times (N+M)}$	with entries $K(\psi, \psi)_{i,j} = K(\psi_i, \psi_j)$
$K(\chi, \psi) \in \mathbb{R}^{N \times (N+M)}$	with entries $K(\chi, \psi)_{i,j} = K(\chi_i, \psi_j)$
$K(X, \psi) \in \mathbb{R}^{M \times (N+M)}$	with entries $K(X, \psi)_{i,j} = K(x_i, \psi_j)$
$g(X) \in \mathbb{R}^M$	with entries $g(X)_i = g(x_i)$

Proposition 3. Let $r_n := \xi - \mathcal{P}(u_n) + \mathcal{P}'(u_n)u_n$. Then, at each step n , the solution to minimization problem (2.20) is given by

$$u_{n+1} = \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j),$$

where $c = (\alpha_1, \dots, \alpha_N, \beta_1, \dots, \beta_M) \in \mathbb{R}^{N+M}$ arises from the solution to the linear system

$$\begin{bmatrix} K(\chi, \psi)^\top A^{-1} K(\chi, \varphi) + \gamma K(\psi, \psi) & K(X, \psi) \\ K(X, \psi) & 0 \end{bmatrix} \begin{bmatrix} c \\ \nu \end{bmatrix} = \begin{bmatrix} K(\chi, \psi)^\top A^{-1} [r_n, \psi] \\ g(X) \end{bmatrix}$$

with $\nu \in \mathbb{R}^M$ are the Lagrange multiplier associated with the boundary condition constraint.

Proof. Equation 2.20 is a standard quadratic program with linear constraints; this is used to facilitate the proof. See Section A.1. $\square$

Remark 5 (Connection to Petrov-Galerkin methods). We note that the solution to the above linear problem is very similar to Petrov-Galerkin methods used in traditional numerical analysis for solving linear PDEs. Typically such methods require an inf-sup type condition to ensure the well-posedness of the discretized problem. Compared to such methods, our approach formulates the problem directly within an infinite dimensional space. At each step, the representer theorem theorem 3 provides a finite dimensional space that is adapted to the linear system to be solved. Finally, the regularization γ ensures that the problem is well-posed and that the solution is unique (which can be seen from the explicit solution proposition 3).

Convergence of the Numerical Method

We now establish the convergence of the minimizer of (2.17) to the true solution. To achieve this, we impose the following assumptions: Section 2.3 ensures the regularity of ξ ; Assumption (D1) guarantees that, as the dimension of the test space increases, the true norm of ξ can be accurately recovered; and Assumption (D2) ensures that the boundary condition is satisfied in the asymptotic limit. The following result demonstrates that, in the limit, the solution of the numerical problem (2.17) converges to the true solution in H_0^l .

Theorem 4 (Convergence to the True Solution). Let Section 2.3 and Section 2.3 be satisfied. Let $u^{\gamma, N, M} \in \mathcal{H}_K$ be the minimizer of (2.17), then

$$\lim_{\gamma \rightarrow 0} \limsup_{N \rightarrow \infty} \limsup_{M \rightarrow \infty} \|u^{\gamma, N, M} - u^*\|_{H^l} = 0.$$

Proof. Since there is at least one $u \in \mathcal{H}_K$ which satisfies the boundary condition, for a fixed N and $\gamma > 0$, (2.17) implies that $\|u^{\gamma,N,M}\|_{\mathcal{H}_K}$ is uniformly bounded in M . By the Banach–Alaoglu theorem Alaoglu, 1940, $u^{\gamma,N,M}$ converges weakly in $\mathcal{H}_K$ along a subsequence (which we denote using the same index M) to some element $u^{\gamma,N,\infty}$. By Assumption (H1), this subsequence converges strongly in $C(\overline{\Omega})$ and in H^l . Using Assumption (D2), since $u^{\gamma,N,M}$ satisfies the boundary condition on an increasingly dense set of collocation points and remains continuous on the boundary uniformly with respect to the number of those points, we deduce Arcangeli, Lopez de Silanes, and Torrens, 2007a that $u^{\gamma,N,\infty}|_{\partial\Omega} := \lim_{M \rightarrow \infty} u^{\gamma,N,M}|_{\partial\Omega} = g$.

We first define

$$\begin{aligned} J^N(u; \gamma) &:= |\mathcal{P}(u) - \xi|_{\Phi_N}^2 + \gamma \|u\|_{\mathcal{H}_K}^2 \\ R^N(u) &:= |\mathcal{P}(u^{\gamma,N}) - \xi|_{\Phi_{N,\perp}}^2. \end{aligned}$$

Recall the definition in equation (2.6) of the functional $J(u; \gamma)$ and observe that by the orthogonal decomposition eq. (2.16), $J(u; \gamma) = J^N(u; \gamma) + R^N(u)$ for any choice of N and γ . By using Assumption (P1) and the lower-semicontinuity of norms in a similar manner as in the proof of theorem 1, we obtain

$$J^N(u^{\gamma,N,\infty}; \gamma) \leq \liminf_{M \rightarrow \infty} J^N(u^{\gamma,N,M}; \gamma).$$

By the definition of $u^{\gamma,N,M}$, for any $u \in \mathcal{H}_K$ such that $u|_{\partial\Omega} = g$, we have

$$J^N(u^{\gamma,N,M}; \gamma) \leq J^N(u; \gamma),$$

from which it follows that

$$\begin{aligned} J^N(u^{\gamma,N,\infty}; \gamma) &\leq \inf_{u \in \mathcal{H}_K} J^N(u; \gamma) \\ &\text{s.t. } u = g \text{ on } \partial\Omega. \end{aligned}$$

Because $u^{\gamma,N,\infty}$ satisfies the boundary conditions, we deduce that $u^{\gamma,N,\infty}$ is a minimizer of the optimization problem eq. (2.17).

For ease of presentation, we will now drop the last superscript and use the notation $u^{\gamma,N} := u^{\gamma,N,\infty}$. Recall the definition in equation (2.6) of the functional $J(u; \gamma)$ and let u^γ be any minimizer of this functional as defined by theorem 1. Observe that

$$J^N(u^{\gamma,N}; \gamma) \leq J^N(u^\gamma; \gamma) \leq J(u^\gamma; \gamma) \leq J(u^{\gamma,N}; \gamma) = J^N(u^{\gamma,N}; \gamma) - R^N(u^{\gamma,N}). \quad (2.24)$$

Now, by (2.24) and the inequality

$$\gamma \|u^{\gamma,N}\|_{\mathcal{H}_K}^2 \leq J(u^\gamma; \gamma),$$

we deduce that the sequence $(u^{\gamma,N})_{N \in \mathbb{N}} \subset \mathcal{H}_K$ is uniformly bounded in N for fixed γ , hence by the assumed continuity of $\mathcal{P}$, the sequence $(\mathcal{P}(u^{\gamma,N}))_{N \in \mathbb{N}}$ is uniformly bounded by some $C(\gamma) \in \mathbb{R}^+$ in the norm of $H^{-s+\delta}$. Then, $H^{-s+\delta} \Subset H^{-s}$ implies that $(\mathcal{P}(u^{\gamma,N}))_{N \in \mathbb{N}}$ converges along a subsequence to some limit in H^{-s} . Therefore, there exists a subsequence (which we denote using the same index N) such that

$$h^N := \mathcal{P}(u^{\gamma,N}) - \xi \rightarrow h^\infty \in H^{-s}.$$

We note that

$$\begin{aligned} |h^N|_{\Phi^{N,\perp}} &\leq |h^N - h^\infty|_{\Phi^{N,\perp}} + |h^\infty|_{\Phi^{N,\perp}} \\ &\leq \|h^N - h^\infty\|_{H^{-s}} + |h^\infty|_{\Phi^{N,\perp}}. \end{aligned}$$

Therefore, the fact that $h^N \rightarrow h^\infty$ in H^{-s} and Corollary 1 imply that

$$|h^N|_{\Phi^{N,\perp}} \rightarrow 0 \quad \text{as } N \rightarrow \infty.$$

Thus, it holds that $|\mathcal{P}(u^{\gamma,N}) - \xi|_{\Phi^{N,\perp}}^2 \rightarrow 0$ as $N \rightarrow \infty$ and we conclude that $R^N(u^{\gamma,N}) \rightarrow 0$ as $N \rightarrow \infty$. Using (2.24), we therefore deduce that

$$\lim_{N \rightarrow \infty} J(u^{\gamma,N}; \gamma) = J(u^\gamma; \gamma).$$

Therefore, applying the result from Theorem 2, we deduce that

$$\lim_{\gamma \rightarrow 0} \lim_{N \rightarrow \infty} J(u^{\gamma,N}; \gamma) = 0.$$

By the local stability assumption on $\mathcal{P}$, given in Assumption (P2), we finally conclude that

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

Therefore, $u^{\gamma,N,M}$ converges along a subsequence to u^* in H^t (which is unique by assumption and independent of the considered subsequence).

We may further conclude that the entire sequence converges to u^* because the previous argument is independent of the considered subsequence. To see this, suppose that $u^{\gamma,N,M}$ does not converge to u^* : there would be a ε and a subsequence u^{γ_k, N_k, M_k} such that $\|u^{\gamma_k, N_k, M_k} - u^*\|_{H^t} \geq \varepsilon$ for all γ_k, N_k, M_k . Then, we can apply

the previous reasoning to the sequence u^{γ_k, N_k, M_k} which would provide a further subsequence that does converge to u^* , thus yielding a contradiction. We therefore conclude that

$$\lim_{\gamma \rightarrow 0} \limsup_{N \rightarrow \infty} \limsup_{M \rightarrow \infty} \|u^{\gamma, N, M} - u^*\|_{H^l} = 0,$$

where the limit in N, M is now taken over the entire sequence. $\square$

2.4 Numerical Experiments

We now investigate our proposed methodology on a variety of spatial and time-dependent PDEs. Section 2.4 contains some overarching details of the computational methodology that apply widely throughout this section. In Section 2.4, we apply our method to PDEs in one and two spatial dimensions, utilizing both a kernel method and an artificial neural network (NN) trained using a physics-informed negative Sobolev loss, which we denote by NeS-PINN. We compare the performance of the methods trained using the negative Sobolev norm against the same methods trained using the usual pointwise loss, which correspond to the GP-PDE Y. Chen, Hosseini, Housman Owhadi, and Andrew M. Stuart, 2021b and PINN Raissi, P. Perdikaris, and G. Karniadakis, 2019a methods⁷. In Section 2.4 we show how to apply our method to solve time-dependent stochastic PDEs.

Details of Computational Methodology

For all numerical experiments, we use the Matérn kernel with smoothness parameter $\nu = \frac{5}{2}$, which is defined as

$$K_{\frac{5}{2}}(x, y) = \left(1 + \frac{\sqrt{5}\|x - y\|}{l} + \frac{5\|x - y\|^2}{3l^2}\right) \exp\left(-\frac{\sqrt{5}\|x - y\|}{l}\right).$$

We find that selecting a length scale l within the range $[0.1, 2.0]$ produces favorable results in our experiments. We note that hierarchical methods to learn the length scale may be needed in general for other applications.

Our standard neural network architecture comprises four layers: a non-trainable random Fourier layer embedding as proposed in Sifan Wang, H. Wang, and Paris Perdikaris, 2021c, two trainable hidden layers, and a trainable outer layer. We use the random embedding layer as our experiments show that it improves accuracy and helps the network capture the high-frequency behavior of the function. Following standard implementations, we use the hyperbolic tangent activation function Sifan

⁷This corresponds to using an L^2 loss, which is approximated using pointwise evaluations of the functions.

Wang, Sankaran, et al., 2023. A detailed table of all architectures can be found in Section A.2. We train the ANNs for 10^5 iterations with the ADAM optimizer Kingma and Ba, 2017 using the learning rate 10^{-3} and an exponential decay schedule.

The integrals against test functions are numerically approximated using either a fast Fourier/sine transform (when using spectral test functions) or a Gauss quadrature rule with either $n = 5$ or $n = 10$ (when using piecewise linear elements as test functions). In all our examples, the entries of the stiffness matrix A can be derived in closed form, but in more general settings, its entries can be also be computed using quadrature methods.

Our code to reproduce the experiments is available in the public repository: <https://github.com/MatthieuDarcy/Kernel-NeuralNetworks-SPDEs>.

Rough Spatial PDEs

In this section, we evaluate our method on two spatial PDEs with rough right-hand sides: a one-dimensional linear PDE in Section 2.4 and a two-dimensional semilinear PDE in Section 2.4. We compare the performance of the kernel method and the PINN when they are trained using the negative Sobolev norm or the standard pointwise loss.

1D Linear Elliptic PDE

We first consider the elliptic equation with homogeneous Dirichlet boundary condition

$$\begin{aligned} -\nu \Delta u + u &= \xi, & x \in \Omega = (0, 1), \\ u &= 0, & x \in \partial\Omega, \end{aligned} \tag{2.25}$$

where $\nu = 10^{-2}$. We sample the right-hand side ξ using the following expansion:

$$\xi(x) = \sum_{j=1}^{\infty} \xi_j \sqrt{2} \sin(\pi j x), \quad \xi_j \sim \mathcal{N}(0, 1) \text{ i.i.d.} \tag{2.26}$$

A fixed realization of ξ belongs to H^{-s} for any $s > \frac{1}{2}$ a.s.. The solution of PDE (2.25) is given in closed form by

$$u^*(x) = \sum_{j=1}^{\infty} \frac{\xi_j}{\nu j^2 \pi^2 + 1} \sqrt{2} \sin(\pi j x),$$

where $u^* \in H^t$ for $t > \frac{3}{2}$ a.s.. In practice, we truncate series (2.26) to $L = 2^{14}$ terms. The number of terms L is set to be much larger than the number of measurements $N + M$ to ensure that the truncation error is negligible.

For the choice of t in the objective norm, we choose the H^{-1} norm and we choose our test space to be spanned by the first $N = 4096$ basis functions $\Phi^N = \text{span}\{\varphi_j\}_{j=1}^N$ with $\varphi_j = \sin(\pi j x)$. Each measurement of ξ or u corresponds to the projection of ξ or u onto the basis function φ_j , with the coefficients given by $\xi_j = \langle \xi, \varphi_j \rangle$ or $u_j = \langle u, \varphi_j \rangle$, respectively. We use the fast discrete sine transform to compute series (2.26) and to compute integrals against the test functions φ_j . In this simple setting, the test functions are eigenfunctions of $-\Delta$. Therefore A is diagonal and its entries (2.3) are given by $A_{ij} = \delta_{ij} \lambda_j^{-1}$. Intuitively, this means that the H^{-1} norm reduces the higher frequencies (corresponding to larger eigenvalues) by the corresponding factor λ_j^{-1} . Choosing s larger would increase this decay, thus further reducing higher frequencies.

We compute the L^2 and pointwise errors of the solution for both the kernel method and the PINN, and compute the L^2 error as a function of the number of measurements N for the kernel method. Our results are summarized in Table 2.1 and illustrated in Figure 2.1. We observe that both the kernel method and the neural network show significant improvement in the errors when trained with the negative Sobolev loss compared to the pointwise loss.

	Kernel (H^{-1})	Kernel (pointwise)	NN (H^{-1})	NN (pointwise)
1D linear	3.95×10^{-5}	1.68	0.0200	1.49
2D semi-linear	0.0232	1.00	0.0203	3.40

Table 2.1: Relative L^2 error of the estimated solutions for the spatial PDEs with $N = 4096$ measurements. The errors are reported for both the kernel and the NN approximation that are trained with either the H^{-1} norm or the pointwise loss (in brackets).

2D Semilinear Elliptic PDE

Next, we consider the semi-linear elliptic PDE with homogeneous Dirichlet boundary condition:

$$\begin{aligned} -\nu \Delta u + u + \sin(\pi u) &= \xi \quad x \in \Omega = (0, 1) \times (0, 1), \\ u &= 0 \quad x \in \partial\Omega. \end{aligned} \tag{2.27}$$

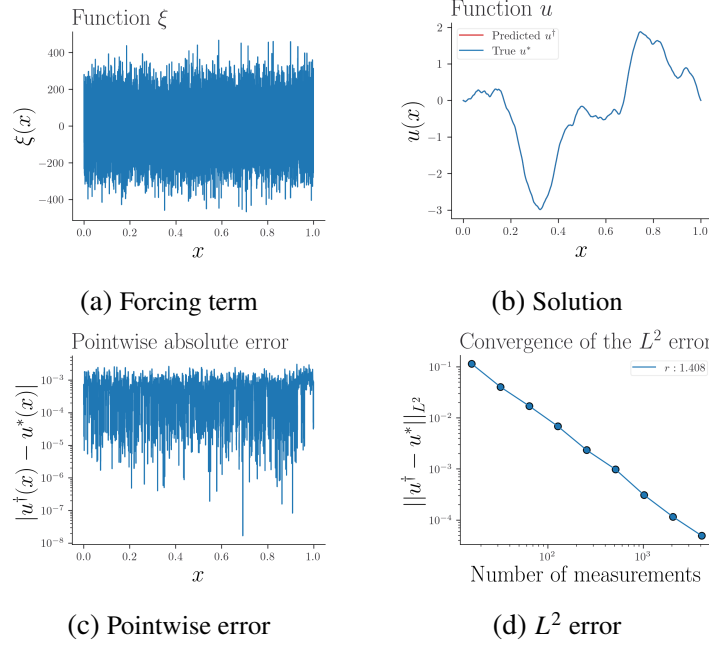


Figure 2.1: Numerical results for the one-dimensional linear elliptic PDE (2.25).

We take $\nu = 10^{-1}$. We consider a manufactured solution that is sampled from the series:

$$u^*(x) = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \frac{u_{ij}}{(i^2 + j^2)^{1+\varepsilon}} 2 \sin(i\pi x) \sin(j\pi y), \quad u_{ij} \sim \mathcal{N}(0, 1) \text{ i.i.d.} \quad (2.28)$$

for some $\varepsilon > 0$. In this case, $u^* \in H_0^t$ for $t < 1 + 2\varepsilon$ a.s. which implies that $\xi \in H^{-s}$ a.s. for $s > 1 - 2\varepsilon$.

Again, we consider a truncated series expansion in (2.28) with $1 \leq i, j \leq L$ and set $L = 2^{10}$ to be much larger than the maximum number of measurements. We generate each realization of the manufactured solution using the discrete sine transform. A sample of the forcing term and the corresponding solution are illustrated in Figure 2.2.

We set $\varepsilon = 0.15$ and, as in the previous section, use the H^{-1} norm as the objective norm. The measurements are chosen as projections onto the basis functions $\varphi_{ij}(x, y) = \sin(\pi i x) \sin(\pi j y)$. We train a kernel method and a NeS-PINN using $1 \leq i, j \leq N$ with $N = 64$ for a total of 4096 measurements. We also train a PINN using the standard pointwise loss on the same grid to assess the impact of using the weak norm compared to the classical pointwise loss. The numerical results are summarized in Table 2.1 and illustrated in Figure 2.3. We again observe significant

improvement for both methods when using the negative Sobolev loss as opposed to the pointwise loss.

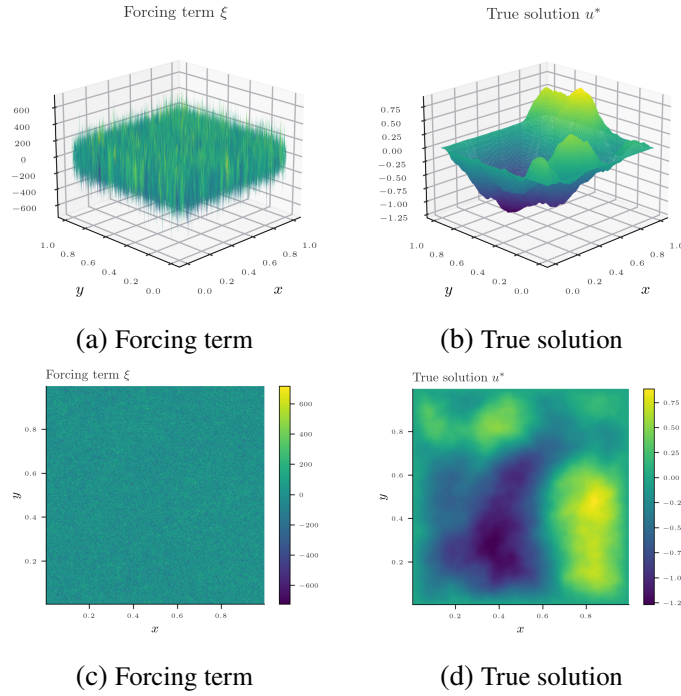


Figure 2.2: One example solution to the two-dimensional semi-linear elliptic PDE (2.27).

The Impact of Norm Selection

We now explore the effect of the selection of the H^{-s} norm, for different choices of s , on the accuracy of the recovered solutions. Our approach differs from the standard physics-informed loss in two ways: it employs measurements as integrals against test functions rather than pointwise evaluations, and it formulates the loss as a negative Sobolev norm based on these measurements. The previous examples demonstrate that using weak measurements yields significantly better performance as compared to pointwise evaluations of the PDE residual.

To demonstrate the effect of the Sobolev norm choice, we consider the two-dimensional PDE in (2.27) with $\varepsilon = 0$. In this case, the solution belongs to H^t for $t < 1$ and the forcing term ξ belongs to H^{-s} for $s > 1$ (note the strict inequality). We recover the true solution using several H^{-s} norms: $s = 0^8, 0.5, 1.0, 1.1, 2.0$, which are all computed with the same test functions. When the norm is misspecified relative to the true solution's regularity (i.e., $s = 0, 0.5, 1.0$), the neural network

⁸The case $s = 0$ corresponds to $H^0 = L^2$.

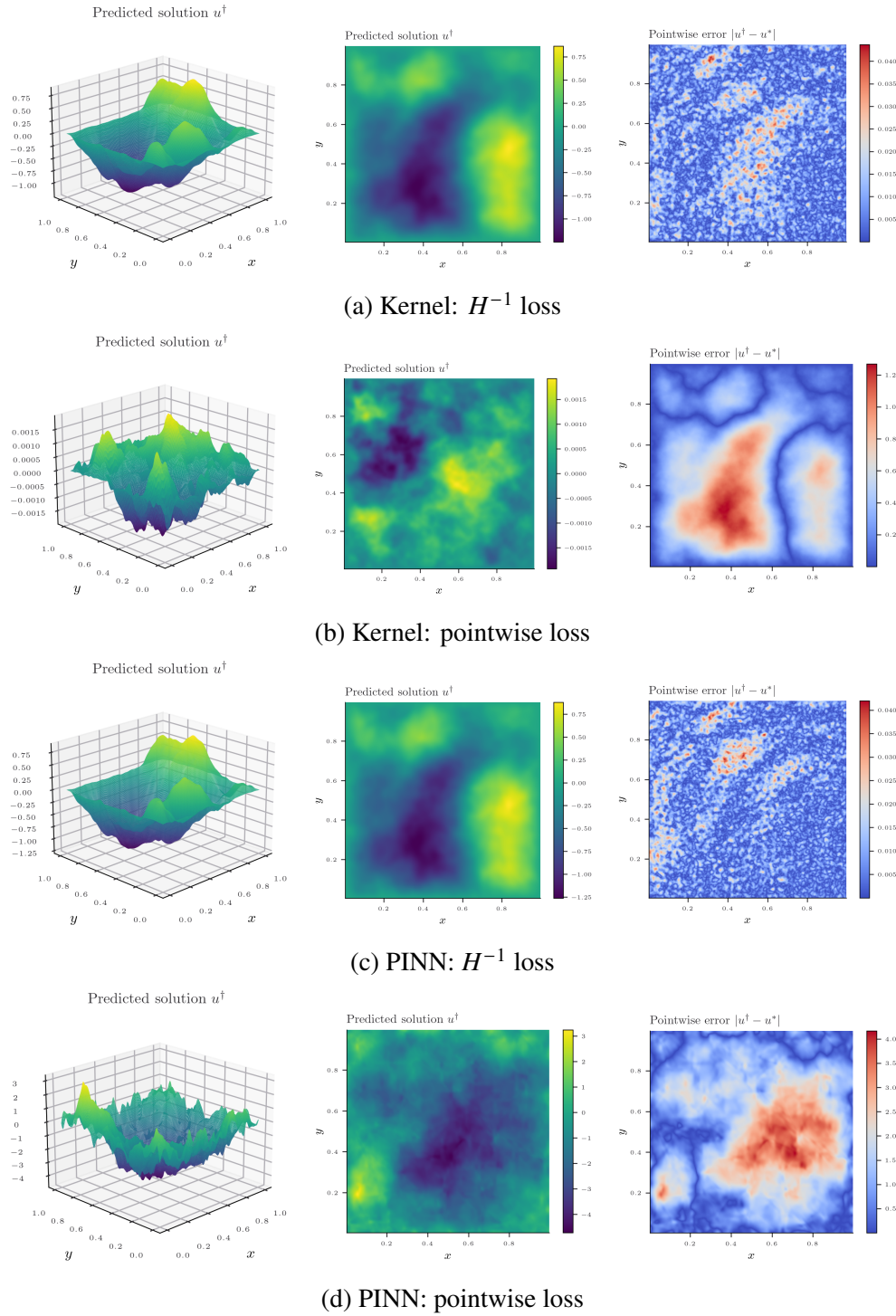


Figure 2.3: Numerical solutions to the two-dimensional semi-linear elliptic PDE (2.27) using the kernel and NN approximations with either the negative Sobolev loss or the pointwise loss.

underperforms as compared to cases where the norm is correctly specified (i.e., $s = 1.1, 2$). Furthermore, we observe a notable difference in performance at the threshold between $s = 1$ and $s = 1.1$, even when truncating the expansion (2.28). When using the L^2 norm, the neural network's training becomes unstable. The best results are achieved when the norm is well-specified with $s = 1.1$. However, when using a weaker norm $s = 2.0$, the error increases again due to over-smoothing effects on the higher frequencies. This effect can also be observed with the kernel method, although its performance is less sensitive to the choice of the norm than the neural network approximation. Based on the results, we deduce that both the use of test functions (as opposed to a pointwise loss) and the choice of loss are crucial to achieving high accuracy on PDEs with rough right-hand sides. These observations are summarized in Table 2.2 and illustrated in Figure 2.4. Moreover, since the right-hand side ξ belongs to the space H^{-s} for any $s > 1$, the choice of $s = 1.1$ is in theory not the only possible one. However, we observe that setting s too close to 1 (such as $s = 1.01$) can also lead to a decrease in performance due to instabilities in training. These observations suggest the following strategy to construct machine learning approximations: at the beginning of training, one may begin the training process with a weaker norm (i.e., large s) to promote an over-smoothed solution, and then gradually decrease s throughout training until convergence has been achieved.

Norm H^{-s}	$s = 0.0$	$s = 0.5$	$s = 1.0$	$s = 1.1$	$s = 2.0$
Kernel method	0.121	0.116	0.046	0.041	0.079
Neural network	0.514	0.434	0.116	0.046	0.082

Table 2.2: Relative L^2 error for different choices of Sobolev norms as the loss function

Time-dependent Stochastic PDEs

In this section, we address the problem of solving time-dependent semi-linear SPDEs of the form:

$$\begin{aligned}
 \partial_t u(t, x) &= Au(t, x) + f(u)(t, x) + \dot{\xi}(t, x) \quad \text{in } \Omega, \\
 u &= 0 \quad \text{on } \partial\Omega \times [0, T], \\
 u &= g \quad \text{on } \Omega \times \{t = 0\},
 \end{aligned} \tag{2.29}$$

where A is a linear operator, $\dot{\xi}$ is a stochastic forcing term and f is a (nonlinear) Nemytskii operator. This is a general form encountered in many applications Gabriel J. Lord, Powell, and Shardlow, 2014.

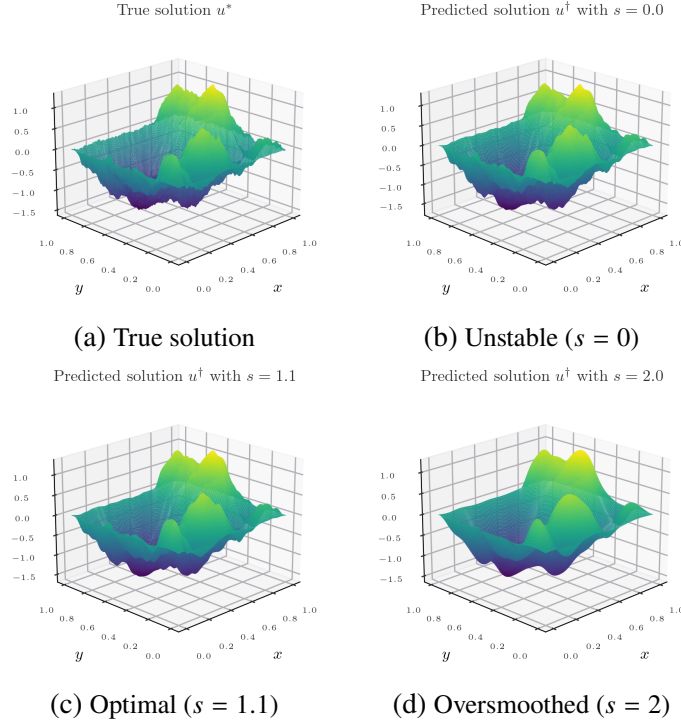


Figure 2.4: Impact of norm selection on the recovered solution.

In Section A.3, we define white noise, which serves as our rough forcing term due to its relevance in practical applications. However, our methodology is equally applicable to more regular forcing terms, such as colored noise, which also appear in areas like stochastic fluid dynamics.

Our method can be applied to temporal problems by first applying a temporal discretization scheme and then using the previously described method at each time-step. This procedure is described in the next section. We begin by describing the semi-implicit Euler time discretization for (2.29) in Section 2.4, and then apply our method to two specific cases: the stochastic heat equation in Section 2.4 and the stochastic Allen-Cahn equation in Section 2.4. A detailed discussion of time-dependent SPDEs and numerical methods for solving them is beyond the scope of this section. For a more comprehensive treatment, we refer the reader to Gabriel J. Lord, Powell, and Shardlow, 2014.

Semi-implicit Euler Method

Given a sample path of $\xi(t)$, the SPDE eq. (2.29) can be discretized in time using a semi-implicit Euler scheme. At the k -th time step, with the solution u_k , the value of u_{k+1} is computed as follows (where the domain and boundary conditions are

included in the definition of the operator A):

$$u_{k+1} = u_k + \delta t f(u_k) + \delta t A u_{k+1} + \delta \xi_k, \quad (2.30)$$

where δt is the step-size in time and $\delta \xi_k$ represents the increment of the stochastic forcing term. Rewriting (2.30), we have

$$(I - \delta t A) u_{k+1} = u_k + \delta t f(u_k) + \delta \xi_k.$$

Thus, given u_k , the solution at the next time step u_{k+1} is obtained by solving a linear PDE with a rough right-hand side, which almost surely does not belong to $L^2(\Omega)$. The linear PDE can be solved using the algorithm described in Section 2.3. Although the fully implicit Euler method could also be used, we do not consider it here.

Stochastic Heat Equation

We now consider the stochastic heat equation over the spatial domain $\Omega = (0, 1)$ and time interval $[0, T]$, which is a form of infinite dimensional Ornstein–Uhlenbeck process Gabriel J. Lord, Powell, and Shardlow, 2014

$$\begin{aligned} \partial_t u &= \nu \Delta u + \sigma \dot{\xi}, \quad \text{in } \Omega, \\ u &= 0, \quad \text{on } \partial\Omega \times [0, T], \\ u &= g, \quad \text{on } \Omega \times \{t = 0\}, \end{aligned}$$

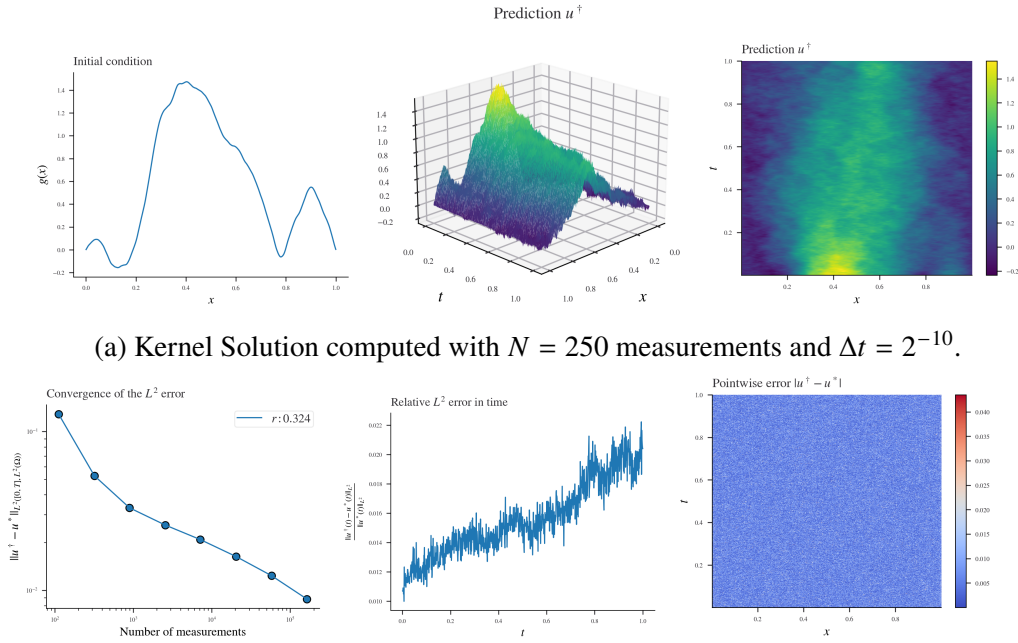
where $\dot{\xi}$ represents space-time white noise, and parameters $\nu = 0.025$ and $\sigma = 0.1$. We first compute a reference solution u^* , using a spectral Galerkin method on a fine mesh, with a time resolution defined by step-size $\Delta t = 2^{-13}$ and spatial resolution $L = 2^{11}$, up to the final time $T = 1.0$. This fine mesh solution is used as the ‘ground truth’ to compute errors and error rates of the kernel solution.

Next, we compute the kernel-based prediction by constructing the measurement space Φ^N using piecewise linear finite element bases, where N represents the number of basis functions. To evaluate the error, we compute the $L^2([0, T]; L^2(0, 1))$ error of the solution in time and space, which is defined as

$$\|u^\dagger - u^*\|_{L^2([0, T]; L^2(0, 1))}^2 = \int_0^T \|u^\dagger(t) - u^*(t)\|_{L^2(0, 1)}^2 dt,$$

where u^* is the fine mesh solution. We employ the semi-implicit integrator in (2.30) to evolve the solution in time with the CFL condition $N^2 \Delta t = 5$. The finest

discretization for the kernel solution is set to $\Delta t = 2^{-11}$ and $N = 250$, which is coarser than the discretization used for the fine mesh solution u^* . The corresponding results and errors are illustrated in Figure 2.5, and the results for the different levels of discretization are recorded in Table 2.3. The results demonstrate that our method offers an accurate and stable approximation of the solution to the stochastic heat equation.



(a) Kernel Solution computed with $N = 250$ measurements and $\Delta t = 2^{-10}$.

(b) Convergence of the L^2 error and the pointwise error for the kernel-based prediction as compared to the fine mesh solution.

Figure 2.5: Numerical solutions for the stochastic heat equation.

Time step Δt	2^{-7}	2^{-8}	2^{-9}	2^{-10}
Heat ($N = 5\Delta t^{-1}$)	2.09×10^{-2}	1.63×10^{-2}	1.24×10^{-2}	8.88×10^{-3}
Allen-Cahn ($N = 3\Delta t^{-1}$)	9.41×10^{-2}	4.69×10^{-2}	3.38×10^{-2}	2.73×10^{-2}

Table 2.3: Relative $L^2([0, T]; L^2(0, 1))$ error of the kernel-based prediction for the solutions to the time-dependent PDEs.

Stochastic Allen-Cahn Equation

Finally, we consider the stochastic Allen-Cahn equation on the spatial domain $\Omega = (0, 1)$ and time interval $[0, T]$:

$$\begin{aligned}\partial_t u &= \nu \Delta u + u - u^3 + \sigma \dot{\xi}, & \text{in } \Omega, \\ u &= 0, & \text{on } \partial\Omega \times [0, T], \\ u &= g, & \text{on } \Omega \times \{t = 0\},\end{aligned}$$

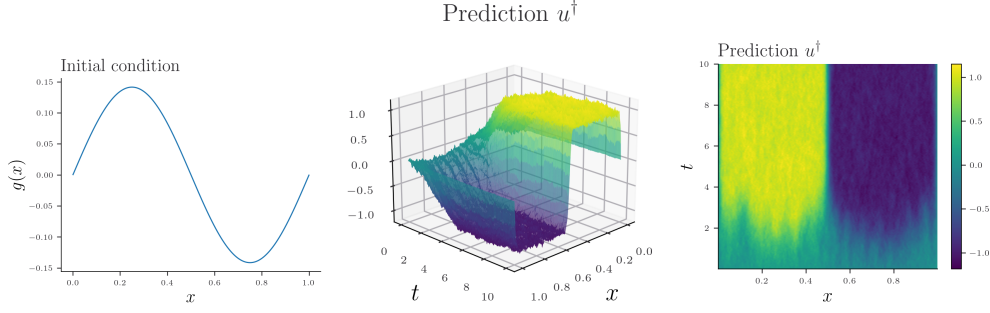
where ξ is space-time white noise, $\nu = 10^{-4}$ and $\sigma = 0.01$. As in the previous case, we first compute a reference solution u^* on a fine mesh using a spectral Galerkin method, integrated with a semi-implicit Euler scheme⁹. The time step is $\Delta t = 2^{-12}$, and the spatial resolution is $L = 2^{11}$, with computations carried out up to the final time $T = 10.0$. This fine mesh solution is used as the ‘ground truth’ to compute errors and error rates of the kernel solution.

Similarly to the previous section, we compute the kernel-based prediction using piecewise linear finite element basis functions and assess the error relative to the reference solution u^* . To satisfy the CFL condition, we set $N^2 \Delta t = 5$, with the finest discretization parameters $\Delta t = 2^{-10}$ and $N = 160$, which is coarser than for the fine mesh solution u^* . The results are illustrated in Figure 2.6, and the quantitative results for the different levels of discretization are presented in Table 2.3.

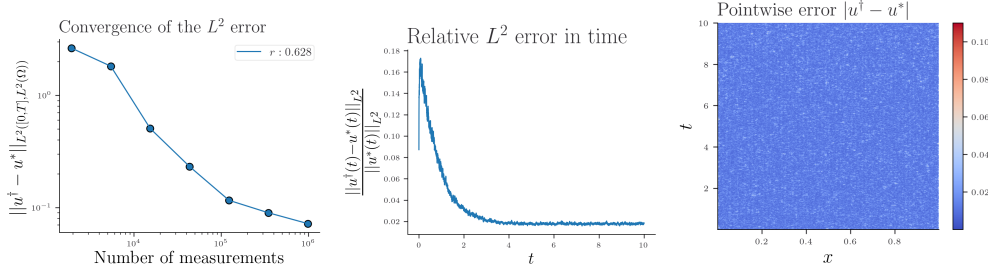
2.5 Defining a New Solution Concept for Singular PDEs/SPDEs

The proposed approach to solving PDEs with singular data can be extended to define a new solution concept for singular PDEs and SPDEs. In this section we explain the idea, leaving detailed development for future work. Some PDEs/SPDEs, such as the stochastic Kardar-Parisi-Zhang (KPZ) equation Kardar, Parisi, and Y.-C. Zhang, 1986 and the Φ_3^4 model Glimm and Jaffe, 2012, do not satisfy our stability assumptions and are ill-posed in the usual sense. An emerging successful methodology for the interpretation of these equations, which formally contain an infinite divergent term, is the theory of regularity structures Martin Hairer, 2012; M. Hairer, 2014. In this section we provide an alternate approach, based on the generalization of (2.5), which is amenable to the computational tools developed in this work. To be concrete, we describe our approach in the context of nonlinear

⁹Here Gabriel J. Lord, Powell, and Shardlow, 2014 note that standard theoretical guarantees for semi-implicit Euler do not apply to this equation, however the standard rates of convergence are observed empirically.



(a) Kernel Solution computed with $n = 160$ and $\delta t = 2^{-10}$.



(b) Error compared to the fine mesh solution.

Figure 2.6: Stochastic Allen-Cahn equation.

PDEs of the form:

$$\begin{aligned} \mathcal{P}(u, Du, D^2u) &= \xi, \quad x \in \Omega, \\ u &= 0, \quad x \in \partial\Omega. \end{aligned} \tag{2.31}$$

In the context of singular stochastic PDEs, the differential operator $\mathcal{P}$ involves nonlinearities that are not well-defined classically. For example, the Kardar-Parisi-Zhang (KPZ) equation includes the nonlinearity $(Du)^2$. When ξ is space-time white noise, Du has to be interpreted in a distributional sense, and the non-linear term $(Du)^2$ does not have an a priori valid interpretation.

To establish a well-defined notion of solutions to (2.31), we introduce continuous regularizers $(v_0, v_1, v_2) \in \mathcal{H}_{K_0} \times \mathcal{H}_{K_1} \times \mathcal{H}_{K_2}$ (a product of three RKHSs) in order to regularize the irregular terms (u, Du, D^2u) . This ensures that the regularization of the operator $\mathcal{P}$ is well-defined in terms of (v_0, v_1, v_2) . The solutions of (2.31) are then defined as the minimizers of the following optimization problem:

$$\inf_{u \in \mathcal{H}_K, v_i \in \mathcal{H}_{K_i}} \left(\|\mathcal{P}(v_0, v_1, v_2) - \xi\|_{H^{-s}}^2 + \sum_{i=0}^2 \|v_i - D^i u\|_{\mathcal{H}_{\Gamma_i}}^2 + \gamma \left(\|u\|_{\mathcal{H}_K}^2 + \sum_{i=0}^2 \|v_i\|_{\mathcal{H}_{K_i}}^2 \right) \right). \tag{2.32}$$

Here, the spaces $\mathcal{H}_{\Gamma_i}$, $i = 0, 1, 2$, are regularization spaces chosen to account for the differences in regularity between the irregular terms (u, Du, D^2u) and their

regularizers (v_0, v_1, v_2) . For each choice of $\gamma > 0$, a solution u^γ exists, as established using a similar approach to the previous sections; see Appendix A.1, theorem 15 for details. The question of interest becomes whether the solution u^γ converges to a well-defined limit (and in which sense) and whether this limit is independent from the approximation procedure, thus yielding a well-defined unique solution. This would yield an equivalent of theorem 2 singular SPDEs. We leave this to a future research direction.

2.6 Conclusion

We have introduced a novel loss for solving partial differential equations with rough forcing terms. This loss is adapted to the irregularity of the right-hand side and allows us to extend the methodology used in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b beyond the setting of classical smooth solutions. We have shown that under mild assumptions on the PDE and the RKHS, the numerical solution provided by this kernel approach converges to the true solution (in the H^t norm). These theoretical results allow us to address the problem of kernel misspecification and show that even when the kernel is not adapted to the solution (i.e., if the solution doesn't belong to the RKHS), we still can still guarantee convergence in the continuum limit of the discretization. Kernel methods can also be viewed from the Bayesian perspective of Gaussian Process Regression Y. Chen, Hosseini, Houman Owhadi, and Andrew M Stuart, 2024a. From this point of view, our approach addresses problems arising from employing misspecified priors, as defined by the choice of kernel. We have not, however, addressed the question of the convergence rates associated with a misspecified kernel and, empirically, the choice of the kernel does have an impact on performance.

We have shown that the corresponding optimization problem for computing a numerical solution can be effectively solved by the Gaussian-Newton method in the functional space. Moreover, we have presented numerical experiments showing that both the kernel method and the neural network approach can effectively recover the solution of partial differential equations with rough forcing terms. Our proposed loss, based on a weak Sobolev norm, significantly improves the performance of both approximation methods as compared to minimizing the classical pointwise empirical L^2 loss.

Additionally, we have explored the impact of loss choice on numerical solutions. We find that misspecified losses lead to high error, while overly weak losses result

in overly smoothed solutions. Our findings underscore the importance of loss selection for enhancing the performance of machine learning solvers, indicating that alternatives to the L^2 loss can yield substantial improvements. The relevance of the choice of loss is also considered in an operator learning context in Zongyi Li, Liu-Schiaffini, et al., 2021 and is integral to the Deep Ritz method E and B. Yu, 2018. Our results demonstrate that both the use of test functions (rather than pointwise evaluations) and selecting the appropriate Sobolev norm are crucial for accurate solution recovery. Finally, we have shown that the the proposed approach can also be employed to identify new solution concepts for singular PDEs and SPDEs.

*Chapter 3***SPARSE INDUCING POINTS AND VARIATIONAL INFERENCE
FOR FORWARD AND INVERSE PROBLEMS IN PDES**

3.1 Introduction: setting and summary of the method

We consider the problem of solving a random (non-linear) PDE of the general strong form

$$\mathcal{F}(u; a)(x) = f(x) \quad x \in \Omega, \quad (3.1a)$$

$$\mathcal{B}(u; a)(x) = g(x) \quad x \in \partial\Omega, \quad (3.1b)$$

where $\mathcal{F}$ is the differential operator and $\mathcal{B}$ is the boundary operator. The function u is the solution of the PDE, while a is a coefficient function. In this setting, we consider three problems.

Problem 1: Solution approximation.

We seek an approximation to the solution u . This is applicable in deterministic settings, where all elements of the PDE are known.

Problem 2: Distribution of the solution.

We seek to obtain a distribution for the solution u . This is applicable in settings where the PDE contains random elements:

- a) The PDE is stochastic: $\mathcal{F}(u; a) = f + \xi$ for some random processes ξ .
- b) We know an element of the PDE, such as the forcing, up to noise $f(\mathbf{x}) + \varepsilon$.

Problem 3: Inverse problems (joint inference of u and a).

We seek a distribution for (u, a) . This is applicable to inverse problems, where a is unknown but we have access to noisy observations of the solution $\mathbf{y} = u(\mathbf{x}_{\text{obs}}) + \varepsilon$.

For ease of exposition, we will first present our method in the context of Problem 2 but there is a natural generalization to both Problem 1 (by considering the small noise limit) and Problem 3 (by applying our method to the pair (u, a)).

Bayesian inversion for PDEs

A natural approach to solving Problem 2 is given by treating this problem as Bayesian inverse problem Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b. In this work, the authors place a Gaussian process prior $u \sim \mathcal{N}(0, \mathcal{K})$ supported on a separable Hilbert space $H \subset L^2(\bar{\Omega})$ with covariance operator $\mathcal{K}$. We then seek to characterize the posterior distribution

$$u | \underbrace{F(u(\mathbf{x}_{\text{int}}), Du(\mathbf{x}_{\text{int}}), \dots)}_{[\phi_{\mathbf{x}_{\text{int}}}, u]} = f(\mathbf{x}_{\text{int}}) + \varepsilon, \underbrace{B(u(\mathbf{x}_{\text{bnd}}), \dots)}_{[\phi_{\mathbf{x}_{\text{bnd}}}, u]} = g(\mathbf{x}_{\text{bnd}}) + \eta \quad (3.2)$$

for $\mathbf{x} = \mathbf{x}_{\text{bnd}} \cup \mathbf{x}_{\text{int}} \subset \Omega \cup \partial\Omega$ is a set of discrete collocation points at which the PDE is imposed. The noise terms $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \sigma^2 I)$ and $\boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2 I)^1$ are only known in distribution, and can represent uncertainty about the true values of f and g . We use the notation $[\phi_{\mathbf{x}}, u]$ to denote the relevant linear functionals (such as pointwise evaluation, derivatives...) at those points. Letting $\mathbf{z} = [\phi_{\mathbf{x}}, u] \in \mathbb{R}^{d_N}$, it can be shown Y. Chen, Hosseini, Houman Owhadi, and Andrew M Stuart, 2024c that $u|\mathbf{z}$ is a Gaussian process with distribution

$$u|\mathbf{z} \sim \mathcal{N}(m, \tilde{K}) \quad (3.3a)$$

$$m = K(\cdot, \boldsymbol{\phi})K(\boldsymbol{\phi}, \boldsymbol{\phi})^{-1}\mathbf{z}_c \quad (3.3b)$$

$$\tilde{K} = K(\cdot, \cdot) - K(\cdot, \boldsymbol{\phi})K(\boldsymbol{\phi}, \boldsymbol{\phi})^{-1}K(\boldsymbol{\phi}, \cdot). \quad (3.3c)$$

Therefore the only remaining unknowns are the values of $\mathbf{z}$, which is what we focus on. The posterior distribution of $\mathbf{z}$ is given by

$$\pi(\mathbf{z}|f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})) = \frac{1}{Z} \ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}) p(\mathbf{z}) \quad (3.4a)$$

$$= \frac{1}{Z} \exp \left(-\frac{1}{2\sigma^2} \|F(\mathbf{z}) - f(\mathbf{x}_{\text{int}})\|^2 - \frac{1}{2\sigma^2} \|B(\mathbf{z}) - g(\mathbf{x}_{\text{bnd}})\|^2 \right) \quad (3.4b)$$

$$\times \exp \left(-\frac{1}{2} \mathbf{z}^\top K(\boldsymbol{\phi}, \boldsymbol{\phi})^{-1} \mathbf{z} \right). \quad (3.4c)$$

There are two main challenges when considering Gaussian process solvers for PDEs:

- C1 The posterior is non-Gaussian when the operators $\mathcal{F}$ and $\mathcal{B}$ are non-linear and challenging to compute efficiently.
- C2 Naively the matrix $K(\boldsymbol{\phi}, \boldsymbol{\phi})$ incurs a $O(N^2)$ storage and $O(N^3)$ inversion cost.

In Section 3.1, we survey previous work which seeks to address these challenges as well as the limitation of current approaches.

One approach to addressing C2 is to limit the number of collocation points $\mathbf{x}$. However, by doing so, one is limited in accuracy: because the PDE is only imposed at the collocation points eq. (3.2) some of the information from the PDE eq. (3.1) is “left on the table”. In this work, we present a variational framework which pairs a small set of inducing points with a *continuous* loss, thus using all the information

¹Other noise models can be used, but will not consider this here.

available

$$J(q_\theta) := \frac{1}{2\sigma^2} \mathbb{E}_{\mathbf{z}_c \sim q_\theta} \mathbb{E}_{u|\mathbf{z}_c} \left[\|\mathcal{F}(u) - f\|_{L^2(\Omega)}^2 + \|\mathcal{B}(u) - g\|_{L^2(\partial\Omega)}^2 \right] + \text{KL}(q_\theta \| p). \quad (3.5)$$

This loss is optimized over a set variational distributions q_θ which approximating the posterior $q_\theta \approx \pi$, thus also addressing C1. This framework can be naturally extended to handling Problem 1 (by considering the zero noise limit $\sigma \rightarrow 0$) and Problem 3 (by incorporating discrete observations). We provide a full derivation in Section 3.2.

A numerical example

Consider the 2d dimensional Darcy PDE:

$$-\nabla(a \cdot \nabla u) = f \quad x \in \Omega = (0, 1)^2 \quad (3.6)$$

$$u = 0 \quad x \in \partial\Omega \quad (3.7)$$

$$\mathbf{y} = u(\mathbf{x}_{\text{obs}}) + \varepsilon, \quad (3.8)$$

The coefficient a is unknown, and we aim to recover the solution u and coefficient a jointly from knowledge of the PDE form and the noisy observations $\mathbf{y}$. In this case $\mathcal{F}(u; a) = -\nabla(a \cdot \nabla u)$ is non-linear in the pair (u, a) and $\mathcal{B}(u; a) = u$. Our variational framework is able to outperform the original GP approach by an order of magnitude and a comparison is presented in Figure 3.1. Detailed numerical experiments are presented in Section 3.3.

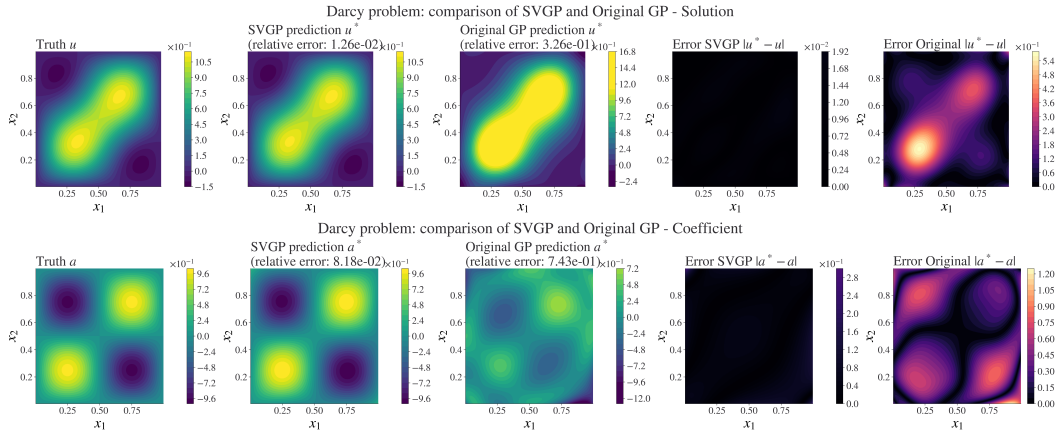


Figure 3.1: Darcy coefficient recovery problem.

Previous work, contributions and outline

Previous work Several approaches have been proposed to address C1 and C2. To address C1 Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b

finds the MAP estimate of the posterior π , while in Y. Chen, Hosseini, Houman Owhadi, and Andrew M Stuart, 2024c, the authors compared a cheaper Laplace approximation of the full posterior to more expensive MCMC approaches. Several approaches have been proposed to address C2, such as those based on sparse Cholesky factors Y. Chen, Houman Owhadi, and Schafer, 2024a, or by selecting a special structure on the prior covariance matrix K that enables fast inversion Fang, Cooley, Long, S. Li, M. Kirby, et al., 2024; A. Sorokin, Robbe, and Hickernell, 2025. Other approaches include inducing points based techniques R. Meng and X. Yang, 2023b and mini-batching X. Yang and Houman Owhadi, 2024. Physics informed neural networks Raissi, P. Perdikaris, and G. Karniadakis, 2019b provide an alternative parametrization of the solution which is obtained by the minimization of a loss similar to eq. (3.5).

Contributions

- C1 We present a general sparse variational approach to solving PDE forward and inverse problems formulated over the *continuum*.
- C2 We show how this framework is amenable to Gaussian variational inference (VI).
- C3 We present an efficient optimization algorithm for the Gaussian VI based on the Gauss-Newton algorithm.
- C4 We show how this framework can be adapted to provide a simple *deterministic* mini-batch framework, which can be interpreted as a generalization of previous approaches based on MAP estimation.
- C5 We provide theoretical justification for our algorithm and characterize its behavior in the small noise limit, which can be used for deterministic PDEs.
- C6 Numerically, we investigate the performance of the approach on a range of forward and inverse problems.

We note that G. Yu and Lan, 2025 has also considered variational inference applied to q-exponential processes (this works is closest to ours). However, our deterministic variation presented in Section 3.2 is new. We justify this approach theoretically in Section 3.2 by considering the zero noise limit, and show that numerically it is much more stable and much more accurate for forward problems in Section 3.3 (as the

posterior covariance tends to be singular with zero noise). Our formulation at the level of the continuum is also new. Our implementation is available in the repository <https://github.com/MatthieuDarcy/SVGP-PDE>.

Outline The remainder of the paper is structured as follows. In Section 3.1, we introduce the notation that will be used throughout as well as some of the basic assumptions. In Section 3.2, we present a derivation of our variational framework. In Section 3.2, this concerns C1. We present two variational strategies in Section 3.2 and 3.2, which concerns C2, C3, and C4. In Section 3.2, we present a theoretical analysis of the zero noise limit, which concerns C5. In Section 3.3, we present detailed numerical experiments for both forward Section 3.3 and inverse Section 3.3 problems of PDEs, which concerns C6.

Notation and assumptions

We use the notation $\mathbb{R} = (-\infty, \infty)$ for the reals. For a set D , we use $\overline{D}$ to denote the closure of the set, i.e. the union of the set itself and the set of all its limit points. Throughout, $\Omega \subset \mathbb{R}^d$ will be a bounded open domain with Lipschitz boundary $\partial\Omega$. We denote $\mathbb{S}_{++}^d$ and $\mathbb{S}_+^d$ as the set of symmetric positive definite and semi-positive definite matrices, respectively.

We use the generic notation $\mathcal{N}(m, \mathcal{K})$ to denote a Gaussian measure over a separable Hilbert space X with mean m and covariance operator $\mathcal{K}$. When $X = \mathbb{R}^d$, this corresponds to the usual finite dimensional Gaussian distribution, and $m \in \mathbb{R}^d$ and we can identify the covariance with a matrix $K \in \mathbb{S}_+^d$. We define the set of all such finite dimensional Gaussian distributions in d dimensions by

$$\mathcal{Q}_d = \{q = \mathcal{N}(m, S) | m \in \mathbb{R}^d, S \in \mathbb{S}_{++}^d\} \quad (3.9)$$

$$\bar{\mathcal{Q}}_d = \{q = \mathcal{N}(m, S) | m \in \mathbb{R}^d, S \in \mathbb{S}_+^d\}. \quad (3.10)$$

Note that $\mathcal{Q}_d$ only allows for positive-definite covariance matrices, whereas $\bar{\mathcal{Q}}_d$ allows for degenerate Gaussian distributions (including the Dirac measures δ_m centered at m). When $X = H \subset L^2(\bar{\Omega})$ an infinite dimensional Hilbert space, we assume that $\mathcal{K}$ is of sufficient regularity to be associated with an integral kernel K so that $\mathcal{N}(m, \mathcal{K}) = \mathcal{GP}(m, K)$ (see Vaart and Zanten, 2008). We denote a generic probability measure on some probability space $(E, \mathcal{E}, \mathbb{P})$ by $\mu \in \mathcal{P}(E)$.

Given some separable Hilbert space $H \subset L^2(\bar{\Omega})$, we use $[\cdot, \cdot]$ to denote the duality pairing with its dual space H^* . We note that we do not necessarily identify H with

H^* using the Riesz representation theorem, thus allowing for a different realization of the dual space. We also assume that the differential and boundary operators *eq.* (3.1) can be written as

$$\mathcal{F}(u)(x) = F([\phi_x, u]) \quad (3.11)$$

for some $F : \mathbb{R}^{d_1} \rightarrow \mathbb{R}^{d_2}$ and some $\phi_x \subset (H^*)^{d_1}$. As an example, the non-linear differential operator $\mathcal{F}(u)(x) = -\Delta u(x) + u(x)^3$ satisfies this with $F(x_1, x_2) = -x_1 + x_2^3$ and $\phi_x = (\delta_x \circ \Delta, \delta_x)$. The same condition applies to the boundary operator $\mathcal{B}$. See Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b for a more detailed discussion on these conditions and examples.

3.2 Sparse Variational Inference for PDEs

In Section 3.2, we provide a derivation of our framework by first considering a finite dimensional formulation, which we then generalize to the continuum. In Section 3.2, we present two variational strategies based on Gaussian variational inference: the first is a standard Gaussian variational inference strategy Section 3.2, the second is a deterministic variational inference approach which we call Dirac SVGP Section 3.2. We also consider the small noise limit of both of these algorithms and show that this limit can be used for deterministic/noiseless PDEs.

Derivation

Discrete formulation Our starting point is the finite dimensional posterior (3.4). However instead of considering discretizing the likelihood and the prior at the same points, we consider a coarse set of inducing points $\mathbf{x}_c = \{x_i\}_{i=1}^{N_{\text{coarse}}}$ and the relevant measurements at those points $\mathbf{z}_c = [\phi_{\mathbf{x}_c}, u] \in \mathbb{R}^d$. We also have a set of finer points $\mathbf{x} = \{x_i\}_{i=1}^{N_{\text{fine}}}$ and the corresponding measurements $\mathbf{z} = [\phi_{\mathbf{x}}, u] \in \mathbb{R}^{d_{\text{fine}}}$, with $N_{\text{coarse}} \ll N_{\text{fine}}$ and $d \ll d_{\text{fine}}$. The coarse and fine measurements are coupled by the standard Gaussian conditioning formulas

$$p(\mathbf{z}|\mathbf{z}_c) = \mathcal{N}(\mathbf{m}, \tilde{\mathbf{K}}) \quad (3.12a)$$

$$\mathbf{m} = K(\phi, \phi_c)K(\phi_c, \phi_c)^{-1}\mathbf{z}_c \quad (3.12b)$$

$$\tilde{\mathbf{K}} = K(\phi, \phi) - K(\phi, \phi_c)K(\phi_c, \phi_c)^{-1}K(\phi_c, \phi).. \quad (3.12c)$$

For the remainder of this article, we now assume that the set of measurements $\phi_{\mathbf{x}}$ are linearly independent so that $K(\phi_c, \phi_c)$ is an invertible matrix. The likelihood then becomes

$$\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}_c) = \int \ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z})p(\mathbf{z}|\mathbf{z}_c)d\mathbf{z}. \quad (3.13)$$

The variational approach seeks an approximation to this posterior $q \approx \pi$ within a parametric family by minimizing the ELBO:

$$J_{\text{ELBO}}(q) = \mathbb{E}_{\mathbf{z}_c \sim q} [-\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}_c))] + \text{KL}(q||p) \quad (3.14)$$

However this objective is difficult to approximate in practice. A drawback of this objective is that it is not separable in $f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})$ (see Section B.1). Hence following Hensman, Fusi, and Neil D. Lawrence, 2013, consider we instead optimize the upper bound (obtained with Jensen's inequality):

$$I(q) = \mathbb{E}_{\mathbf{z}_c \sim q} \left[-\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z})) \right] + \text{KL}(q||p) \quad (3.15)$$

$$= \mathbb{E}_{\mathbf{z}_c \sim q} \left[-\log \left(\int \ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}) p(\mathbf{z}|\mathbf{z}_c) d\mathbf{z} \right) \right] + \text{KL}(q||p) \quad (3.16)$$

$$\leq \mathbb{E}_{\mathbf{z}_c \sim q} \left[\int -\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}) p(\mathbf{z}|\mathbf{z}_c) d\mathbf{z} \right] + \text{KL}(q||p) \quad (3.17)$$

$$= \mathbb{E}_{\mathbf{z}_c \sim q} \mathbb{E}_{\mathbf{z}|\mathbf{z}_c} \left[-\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z})) \right] + \text{KL}(q||p). \quad (3.18)$$

The log likelihood term then admits the expression

$$-\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z})) = \frac{1}{2\sigma^2} \|F(\mathbf{z}) - f(\mathbf{x}_{\text{int}})\|^2 + \frac{1}{2\sigma^2} \|B(\mathbf{z}) - g(\mathbf{x}_{\text{bnd}})\|^2. \quad (3.19)$$

This yields the discrete objective

$$J_{\text{discrete}}(q_\theta) = \frac{1}{2\sigma^2} \mathbb{E}_{\mathbf{z}_c \sim q_\theta} \mathbb{E}_{u|\mathbf{z}_c} \left[\|F(\mathbf{z}) - f(\mathbf{x}_{\text{int}})\|^2 + \|B(\mathbf{z}) - g(\mathbf{x}_{\text{bnd}})\|^2 \right] + \text{KL}(q_\theta||p). \quad (3.20)$$

Continuous formulation At the continuous level, the equivalent to eq. (3.12) is the conditional distribution of random process $u|\mathbf{z}_c = [\boldsymbol{\phi}, u]$. This is a GP with conditional mean and covariance:

$$u|\mathbf{z} \sim \mathcal{GP}(m, \tilde{K}) \quad (3.21a)$$

$$m = K(\cdot, \boldsymbol{\phi}_c) K(\boldsymbol{\phi}_c, \boldsymbol{\phi}_c)^{-1} \mathbf{z}_c \quad (3.21b)$$

$$\tilde{K} = K(\cdot, \cdot) - K(\cdot, \boldsymbol{\phi}_c) K(\boldsymbol{\phi}_c, \boldsymbol{\phi}_c)^{-1} K(\boldsymbol{\phi}_c, \cdot). \quad (3.21c)$$

Formally, taking the continuous limit as $N \rightarrow \infty$ in eq. (3.20) yields the continuous objective

$$J(q_\theta; \sigma) := \frac{1}{2\sigma^2} \mathbb{E}_{\mathbf{z}_c \sim q_\theta} \mathbb{E}_{u|\mathbf{z}_c} \left[\|\mathcal{F}(u) - f\|_{L^2(\Omega)}^2 + \|\mathcal{B}(u) - g\|_{L^2(\partial\Omega)}^2 \right] + \text{KL}(q_\theta||p) \quad (3.22)$$

$$= \frac{1}{\sigma^2} I(q_\theta) + \text{KL}(q_\theta||p), \quad (3.23)$$

where the variational distribution q_θ depends on set of finite dimensional parameters $\theta \in \mathbb{R}^{d_\theta}$. We will sometimes abuse notation by writing $J(\theta) := J(q_\theta)$ (similarly for the data term I).

Remark 6 (Connection to conditioning Gaussian measures). *The posterior measure π eq. (3.2) is obtained by conditioning u at a finite number set of points. One may also consider u at the level of the continuum. When*

$$\mathcal{F}(u) = f + \xi \quad x \in \Omega, \quad (3.24a)$$

$$\mathcal{B}(u) = g + \eta \quad x \in \partial\Omega \quad (3.24b)$$

with $\xi \sim \mathcal{N}(0, \sigma^2 I)$ and $\eta \sim \mathcal{N}(0, \sigma^2 I)$, we may obtain the (formal) probability density

$$\pi(u|f, g) = \frac{1}{Z} \ell(f, g|u) p(u) \quad (3.25a)$$

$$= \frac{1}{Z} \exp \left(-\frac{1}{2\sigma^2} \|\mathcal{F}(u) - f\|_{L^2(\Omega)}^2 - \frac{1}{2\sigma^2} \|\mathcal{B}(u) - g\|_{L^2(\partial\Omega)}^2 \right) \quad (3.25b)$$

$$\times \exp \left(-\frac{1}{2} \|\mathcal{K}^{-\frac{1}{2}} u\|_{L^2(\bar{\Omega})}^2 \right). \quad (3.25c)$$

The variational loss eq. (3.22) can be seen as the equivalent of eq. (3.20) for this probability measure, although this fact would require a careful analysis similar to Y. Chen, Hosseini, Houman Owhadi, and Andrew M Stuart, 2024c that we leave to future work. Note that while the continuous posterior (3.25) is formal, the above objective is well-defined and amenable to sampling techniques. Figure 3.2 presents a schematic comparison between the original GP approach Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b, the discrete sparse GP and our continuous sparse GP.

Remark 7 (Observations data). *It is easy to extend eq. (3.22) to include pointwise data observations. In this case we obtain the combined observation-physics loss*

$$J_{data}(q_\theta; \sigma, \sigma_{obs}) = \frac{1}{\sigma^2} I(q_\theta) + \frac{1}{2\sigma_{obs}^2 N_{obs}} \|\mathbf{y} - u(\mathbf{x})\|^2 + KL(q_\theta||p). \quad (3.26a)$$

We note the scaling by N_{obs} of the data term, which ensures that the different terms of the loss are balanced.

Strategies for variational inference

The variational distribution q can be parametrized in many different ways, such as normalizing flows Kingma, Salimans, et al., 2017; Rezende and Shaker Mohamed,

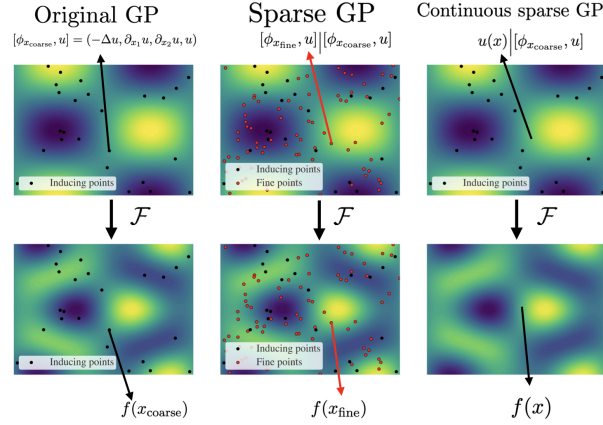


Figure 3.2: A comparison between the original GP approach (left), a discrete sparse GP (middle), and our proposed continuous sparse GP (right). The continuous formulation uses the same discrete inducing points for the prior, but leverages the entirety of the continuous domain for the likelihood.

2016. Here we consider two strategies: one based on Gaussian variational inference Section 3.2, the other based on a deterministic inference strategy we call Dirac-SVGP Section 3.2. We generally observed that these two methods significantly outperformed approaches based on normalizing flows, and we leave the investigation of other strategies to future work.

Sparse variational Gaussian variational inference: SVGP

The objective (3.22) is amenable to several variational strategies. We discuss one such possibility: Gaussian variational inference (VI) Bishop, 2006. In this case the variational distribution is given by a Gaussian $q_\theta \in \mathcal{Q}_d$ and $\theta = (m, S)$, with $m \in \mathbb{R}^d, S \in \mathbb{S}_{++}^d$. In the rest of this section, we drop the dependence on θ , and always assume that q is parametrized in this way. The KL divergence then admits the expression:

$$\text{KL}(q||p) = \frac{1}{2} \left(\log \frac{|K(\phi_c, \phi_c)|}{|S|} - d + m^\top K(\phi_c, \phi_c)^{-1} m + \text{trace}(K(\phi_c, \phi_c)^{-1} S) \right). \quad (3.27)$$

We now aim to show that the objective eq. (3.22) is well-defined and admits minimizers.

Assumption 5 (Assumptions on the data term). *Recall that H is the space associated with the Gaussian measure of u . Assume that*

1. $\mathcal{F} : H \rightarrow L^2(\Omega)$ and $\mathcal{B} : H \rightarrow L^2(\partial\Omega)$ are continuous.

2. The functional

$$\Phi(u) = \|\mathcal{F}(u) - f\|_{L^2(\Omega)}^2 + \|\mathcal{B}(u) - g\|_{L^2(\partial\Omega)}^2 \quad (3.28)$$

has at most polynomial growth in u

$$\Phi(u) \leq C(1 + \|u\|_H^p) \quad (3.29)$$

for some $p \geq 0$.

Lemma 2 (Continuity of the data term). *The data term I is continuous in the topology of weak convergence of Gaussian probability distributions:*

$$q_n \rightharpoonup q \implies I(q_n) \rightarrow I(q), \quad (3.30)$$

where $\rightharpoonup$ denotes weak convergence of probability measures.

Proof. See Section B.2. □

Proposition 4 (Existence of a minimizer). *For any $\sigma > 0$, there is a corresponding $q_\sigma \in \mathcal{Q}_d$ which minimizes the continuous objective eq. (3.22)*

$$q_\sigma \in \arg \min_{q \in \mathcal{Q}_d} J(q; \sigma). \quad (3.31)$$

Proof. See Section B.2. □

A Gauss-Newton approach to Gaussian VI When $\mathcal{F}$ and $\mathcal{B}$ are linear, we may simplify the objective (3.22) and use natural gradients to accelerate convergence Amari, 1999. To exploit this, we employ an iterative linearization procedure. Given a current estimate of u given by $\bar{u}_k$ (typically this would be given by the current posterior mean), consider the linearized objective:

$$J_{\text{lin}}^k(q) = \frac{1}{2\sigma^2} \mathbb{E}_{\mathbf{z}_c \sim q_\theta, u | \mathbf{z}_c} \left[\|\mathcal{F}'(\bar{u}_k)u - r_k\|_{L^2(\Omega)}^2 + \|\mathcal{B}'(\bar{u}_k)u - s_k\|_{L^2(\partial\Omega)}^2 \right] + \text{KL}(q || p) \quad (3.32)$$

where $\mathcal{F}'$ and $\mathcal{B}'$ are Fréchet derivatives (we assume that these exist and are bounded) and

$$r_k := f - \mathcal{F}(\bar{u}_k) + \mathcal{F}'(\bar{u}_k)\bar{u}_k \quad (3.33)$$

$$s_k := g - \mathcal{B}(\bar{u}_k) + \mathcal{B}'(\bar{u}_k)\bar{u}_k. \quad (3.34)$$

We then show in Section B.2 that (3.32) simplifies to

$$J_{\text{lin}}^k(q) = \frac{1}{2\sigma^2} \left(\|r_k - \mathcal{F}'(\bar{u}_k)K(\cdot, \phi_c)K(\phi_c, \phi_c)^{-1}m\|_{L^2(\Omega)}^2 + \frac{1}{2} \int_{\Omega} \mathcal{F}'(\bar{u}_k) \tilde{Q} \mathcal{F}'(\bar{u}_k)^{\top} \right. \quad (3.35)$$

$$\left. + \|s_k - \mathcal{B}'(\bar{u}_k)K(\cdot, \phi_c)K(\phi_c, \phi_c)^{-1}m\|_{L^2(\partial\Omega)}^2 \right) \quad (3.36)$$

$$+ \int_{\partial\Omega} \mathcal{B}'(\bar{u}_k) \tilde{Q} \mathcal{B}'(\bar{u}_k)^{\top} \quad (3.37)$$

$$+ \text{KL}(q\|p). \quad (3.38)$$

The above objective can be estimated using Monte-Carlo sampling or numerical quadrature techniques, and efficiently optimized using natural gradient descent Amari, 1999 algorithm 1. This is an extension of the Gauss-Newton algorithm used in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b and similar to the algorithm used in G. Yu and Lan, 2025.

Algorithm 1 Gauss-Newton algorithm for Gaussian VI.

Require: Interior and boundary data f, g , nonlinear operators $\mathcal{F}, \mathcal{B}$, initial variational distribution $q^0 = \mathcal{N}(m^0, S^0)$.

- 1: **for** $k = 1$ to N_{outer} **do**
 - 2: Compute posterior mean $\bar{u}_k = \mathbb{E}_{\mathbf{z}_c \sim q_{\theta}, u | \mathbf{z}_c} [u]$
 - 3: Interior residual $r_k \leftarrow f - (\mathcal{F}(\bar{u}_k) - \mathcal{F}'(\bar{u}_k)\bar{u}_k)$ ▷ Linearized residual
 - 4: Boundary residual $s_k \leftarrow g - (\mathcal{B}(\bar{u}_k) - \mathcal{B}'(\bar{u}_k)\bar{u}_k)$
 - 5: Define $J^k(q^k)$ by (3.35) ▷ Variational loss
 - 6: Initialize the inner loop $q_0^k = q^k$ ▷ Apply natural gradient descent to optimize q^k
 - 7: **for** $j = 1$ to N_{inner} **do** ▷ One step sufficient in practice
 - 8: Compute a Monte-Carlo approximation of $J^k(q_j^k)$
 - 9: $q_{j+1}^k \leftarrow q_j^k - \eta \tilde{\nabla} J_j^k(q_j^k)$ ▷ Stochastic natural gradient descent
 - 10: **end for**
 - 11: Update the outer loop $q^{k+1} \leftarrow q_{N_{\text{inner}}}^k$
 - 12: **end for** **return** $q^{N_{\text{outer}}}$ ▷ Final optimized variational distribution
-

Sparse deterministic variational inference: Dirac-SVGP

We consider the continuous objective and drop the dependence on S . Formally this corresponds to a degenerate Gaussian distribution $q_{\theta} = \mathcal{N}(m, \delta)$, $\theta = m$ and, in this case, the posterior distribution is deterministic and we aim to only estimate its mean. An advantage of this formulation is that while it yields a deterministic estimate, the

number of optimization parameters is reduced. Furthermore, the loss eq. (3.22) becomes equivalent to :

$$J_{\text{Dirac}}(m; \gamma) = \mathbb{E}_{\mathbf{z}_c \sim q_m} \mathbb{E}_{u|\mathbf{z}_c} \left[\|\mathcal{F}(u) - f\|_{L^2(\Omega)}^2 + \|\mathcal{B}(u) - g\|_{L^2(\partial\Omega)}^2 \right] + \gamma m^\top K(\boldsymbol{\phi}_c, \boldsymbol{\phi}_c)^{-1} m, \quad (3.39)$$

where $\gamma = \sigma^2$. This can be interpreted as a continuous analog of the loss function used in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b, Eq. 2.5, which can be optimized using the stochastic Gauss-newton algorithm. This approach thus yields a simple mini-batch version of the relaxed deterministic framework presented in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b, which can be used for noisy observations.

Zero noise limit

We are interested in characterizing the solutions of eq. (3.22) in the zero noise limit $\sigma \rightarrow 0$, which can also be interpreted as the zero regularization limit. Intuitively, this corresponds to considering only the data term I in the objective eq. (3.22), which is useful when solving PDE forward problems without noise. For a noise standard deviation σ , we consider the optimal Gaussian variational distribution q_σ . The next proposition shows that the zero noise/regularization limit of the objective eq. (3.22) is well-defined and that q_σ has a subsequence that converges to a minimizer of this unregularized objective (although this limit may not be unique). We further show that any limit point of q_σ satisfy a stronger property of being the minimizers of the data term with minimal KL divergence with the prior distribution p .

Assumption 6 (Relative compactness of the data functional I). *For every $c < \infty$, the sublevel set*

$$\{q \in \bar{\mathcal{Q}}_d : I(q) \leq c\}$$

is relatively compact in the weak topology.

Lemma 3. *There is a $q_{\text{opt}} \in \bar{\mathcal{Q}}_d$ which minimizes the data term:*

$$q_{\text{opt}} \in \arg \min_{q \in \bar{\mathcal{Q}}_d} I(q). \quad (3.40)$$

Proof. See Section B.2 □

Proposition 5 (Zero regularization limit). *For any sequence $\sigma_n \rightarrow 0$ and corresponding sequence of optimal $q_n := q_{\sigma_n}$ of the objective eq. (3.22), there is a*

subsequence converging weakly² to some $q_\infty = \mathcal{N}(m_\infty, S_\infty)$ such that

$$q_\infty \in \arg \min_{q \in \tilde{\mathcal{Q}}_d} I(q). \quad (3.41)$$

Moreover, if there is a $q_{opt} \in \arg \min_{q \in \tilde{\mathcal{Q}}_d} I(q)$ such that $KL(q^{opt}||p) < \infty$, then q_∞ is the minimizer with the smallest KL divergence with respect to the prior:

$$q_\infty \in \arg \min_{q \in \tilde{\mathcal{Q}}_d} KL(q||p) \quad (3.42)$$

$$s.t. \ I(q_\infty) = \min_{q \in \tilde{\mathcal{Q}}_d} I(q). \quad (3.43)$$

Conversely, any limit point of q_n in the weak topology of probability distributions will satisfy the above.

Proof. Note that for finite dimensional Gaussian distributions, to show weak convergence, it is sufficient to consider the convergence in the parameters (m_n, S_n) I. Bogachev, 1998. We proceed in three step: first we show that the sequence of optimal parameters (m_n, S_n) has a convergent subsequence. Second we show that any limit point is an optimal solution to the data term (unregularized loss). Third we show that, moreover, any limit point has the minimal divergence property, under the stated assumption. For notational convenience, we slightly abuse notation and let

$$J(q; \sigma) := \frac{1}{\sigma^2} I(q) + KL(q||p). \quad (3.44)$$

Step 1: the sequence q_n has subsequence that converges in weakly to some Gaussian distribution q_∞ . The fact that $J(q_n; \sigma_n) \leq J(q_0; \sigma_0)$ for all n and that the sequence σ_n is bounded implies that

$$I(q_n) \leq I(q_n) + \sigma_n^2 KL(q_n||p) \quad (3.45)$$

$$\leq I(q_0) + \sigma_n^2 KL(q_0||p) \quad (3.46)$$

$$\leq I(q_0) + c KL(q_0||p) \quad (3.47)$$

$$\leq C \quad (3.48)$$

uniformly in n . Therefore by theorem 6, the sequence q_n admits a weakly convergent subsequence, which denote by the same index, $q_n \rightharpoonup q_\infty \in \tilde{\mathcal{Q}}_d$.

²In the sense of probability measures.

Step 2: any limit point q_∞ ³ of the sequence q_n is a minimizer of the data term (unregularized loss). By lemma 2

$$\lim_{n \rightarrow \infty} I(q_n) = I(q_\infty). \quad (3.49)$$

Consider some arbitrary $q \in \mathcal{Q}$. Then by the definition of q_n and positivity of the KL divergence

$$\frac{1}{\sigma_n^2} I(q_n) \leq J(q_n; \sigma_n) \leq J(q; \sigma_n) \quad (3.50)$$

which implies that

$$I(q_n) \leq I(q) + \sigma_n^2 \text{KL}(q||p). \quad (3.51)$$

Hence for a fixed q taking the limit $n \rightarrow \infty$ yields $I(q_\infty) = \lim_{n \rightarrow \infty} I(q_n) \leq I(q)$. Then further infimizing over all $q \in \mathcal{Q}$

$$I(q_\infty) = \inf_{q \in \mathcal{Q}} I(q). \quad (3.52)$$

We can further extend this result to the space $\bar{\mathcal{Q}}_d$. If $q \in \bar{\mathcal{Q}}_d$, there is a sequence $q_m \subset \mathcal{Q}$, $q_m \rightarrow q$ which implies that

$$I(q_\infty) \leq \lim_{m \rightarrow \infty} I(q_m) = I(q) \quad (3.53)$$

by lemma 2.

Step 3: any limit point of q_n further minimizes the KL divergence with the prior distribution among all minimizers of the data term. Consider the element $q_{\text{opt}} \in \arg \min_{q \in \mathcal{Q}} I(q)$ with $\text{KL}(q^{\text{opt}}||p) < \infty$. We have that $I(q_\infty) = I(q^{\text{opt}}) \leq I(q_n)$ and $J(q_n; \sigma_n) \leq J(q^{\text{opt}}; \sigma_n)$. Together these imply that

$$I(q_\infty) + \sigma_n^2 \text{KL}(q_n||p) \leq I(q_n) + \sigma_n^2 \text{KL}(q_n||p) \quad (3.54)$$

$$\leq I(q^{\text{opt}}) + \sigma_n^2 \text{KL}(q^{\text{opt}}||p) \quad (3.55)$$

$$= I(q_\infty) + \sigma_n^2 \text{KL}(q^{\text{opt}}||p) \quad (3.56)$$

and therefore that $\text{KL}(q_n||p) \leq \text{KL}(q^{\text{opt}}||p) < \infty$. Using the weak lower-semicontinuity of the KL divergence Posner, 1975

$$\text{KL}(q_\infty||p) \leq \liminf_{n \rightarrow \infty} \text{KL}(q_n||p) \leq \text{KL}(q^{\text{opt}}||p), \quad (3.57)$$

proving the claim. □

³Note that step 1 shows that there is *at least one* limit point, but step 2 applies to *any* limit point.

Remark 8.

We can further extend this to the Dirac SVGP approach of section. The proof is a simplified version of the previous, under similar assumptions Section B.3.

Proposition 6 (Zero regularization limit for the Dirac SVGP). *For any sequence $\gamma_n \rightarrow 0$ and corresponding sequence of optimal m_n of the objective eq. (3.39), there is a convergent subsequence to some m_∞ with the property that:*

$$m_\infty \in \arg \min_{m \in \mathbb{R}^n} \|m\|_{K(\phi_c, \phi_c)} \quad (3.58)$$

$$s.t. \ I(m_\infty) = \inf_{m \in \mathbb{R}^m} I(m). \quad (3.59)$$

Conversely, any limit point of m_n will satisfy the above.

Proof. See Section B.3. □

Proposition 5 and proposition 6 gives justification for an annealing approach where we alternatively optimize $J(q; \sigma_n)$ using algorithm 1 and reduce $\sigma_n \rightarrow 0$ until convergence. We find that doing this is much more stable than optimizing the data term only. This approach can be interpreted as the continuous version of the constrained loss used in Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b, Eq. 3.9, which can be used for non-noisy settings.

Remark 9. *The inequality $I(m_\infty) \leq I(u)$ for any $u \in \mathbb{R}^d$ immediately implies that $I(m_\infty) \leq \mathbb{E}_{u \sim q}[I(u)]$ for any $q \in \mathcal{P}(\mathbb{R}^d)$. Hence we expect the limiting distribution given by proposition 5 to be precisely the solution given by proposition 6. This justifies using this method when considering noiseless problems. This is justified numerically in Section 3.3 for forward problems where we observe that the zero regularization limit approach outperforms all other considered approaches.*

3.3 Numerical experiments

We present several benchmark PDE problems. In Section 3.3, we present two PDE solving problems, in the noisy and in the noiseless case. In Section 3.3, we present three PDE inverse problems, where some parts of the data are noisy. We test the performance of SVGP Section 3.2 and Dirac-SVGP Section 3.2. When appropriate, we also use the zero noise limit variant of Dirac-SVGP. To benchmark our approach, we compare against the original GP approach Y. Chen, Hosseini, Houman Owhadi,

and Andrew M. Stuart, 2021b (with the same inducing points) and a Physics Informed Neural Network (PINN) Raissi, P. Perdikaris, and G. Karniadakis, 2019b. The parameters and architecture of our models can be found in Section B.4.

Forward problems

We consider two PDEs, a linear elliptic example and a semi-linear elliptic example. In both cases, we consider two settings: a noisy setting where the right-hand side is subject to Gaussian noise and a noiseless setting which we use to test the zero regularization algorithm presented in Section 3.2. Figure 3.3 illustrates an example recovered solution using the SVGP algorithm. table 3.1 summarizes the performance of each method and Figure 3.4 presents error rates in the noiseless setting.

Poisson equation

We consider the linear Poisson equation with homogeneous Dirichlet boundary conditions

$$-\Delta u = f \quad x \in \Omega = [0, 1]^2 \quad (3.60)$$

$$u = 0 \quad x \in \partial\Omega \quad (3.61)$$

with forcing

$$f(x, y) = 32\pi^2 \sin(2\pi x) \sin(2\pi y) - 24\pi^2 \sin(4\pi x) \sin(3\pi y). \quad (3.62)$$

The PDE is solved numerically using a high resolution spectral method. In the noisy case, the noise is sampled from $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ with $\sigma = 14.29$ (representing 10% of the standard deviation of f).

Semi-linear Poisson-Boltzmann equation

We consider the semi-linear Poisson-Boltzmann equation with homogeneous Dirichlet boundary conditions

$$-\Delta u + \frac{1}{2} \sinh(2u) = f \quad x \in \Omega = (0, 1)^2 \quad (3.63)$$

$$u = 0 \quad x \in \partial\Omega \quad (3.64)$$

with a manufactured solution

$$u(x, y) = \sin(\pi x) \sin(\pi y) - \frac{1}{2} \sin(2\pi x) \sin(2\pi y) + \frac{1}{4} \sin(3\pi x) \sin(4\pi y). \quad (3.65)$$

In the noisy case, the noise is sampled from $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ with $\sigma = 3.79$ (representing 10% of the standard deviation of f).

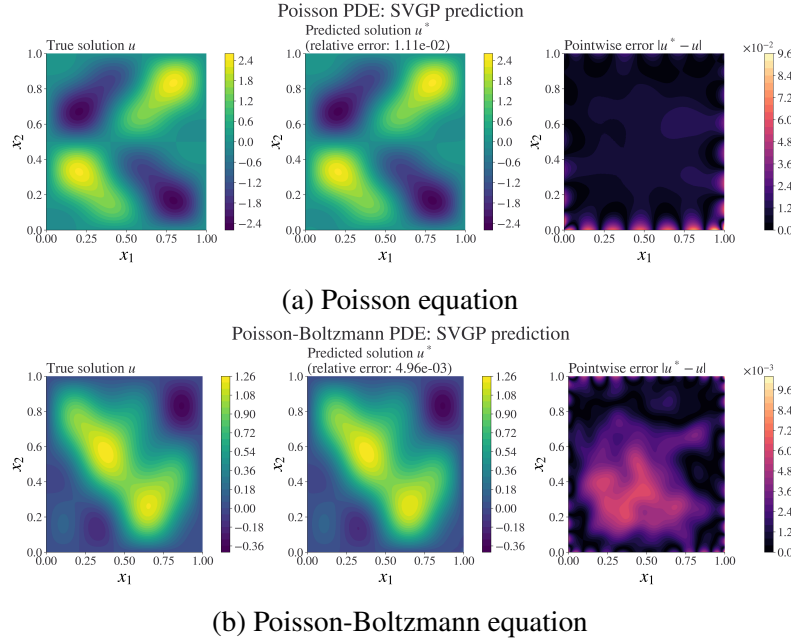


Figure 3.3: Solutions of two noisy PDEs using SVGP.

Method	Poisson (noiseless)	Poisson (noisy)	PB (noiseless)	PB (noisy)
SVGP	3.76×10^{-4}	1.27×10^{-2}	3.14×10^{-3}	4.96×10^{-3}
Dirac	8.54×10^{-5}	9.17×10^{-2}	8.14×10^{-2}	2.11×10^{-2}
Dirac (zero reg limit)	8.53×10^{-5}	—	3.93×10^{-8}	—
Original GP	1.09×10^{-4}	3.26×10^{-2}	4.74×10^{-6}	2.05×10^{-2}
PINN	6.74×10^{-2}	1.43×10^{-1}	1.06×10^{-1}	8.63×10^{-2}

Table 3.1: Comparison of methods across forward PDE problems (error in relative L^2).

Inverse problems

We now consider a general class of inverse problems where an element of the equation (coefficient, forcing term or initial condition) is unknown. We assume that we have access to noisy observations of the solution u locations $\mathbf{x}_{\text{obs}}$. We consider a realistic setup where we have access to many, but clustered observation (we generally observed that in this setting the variational inference approach we propose provides the important gains over the original GP approach). We tune the hyperparameters of all methods using a five-fold cross validation optimized with Optuna Akiba et al., 2019a over 50 iterations. The GP approaches use 100 of the observations locations as inducing points (representing 1-5% of the total observations).

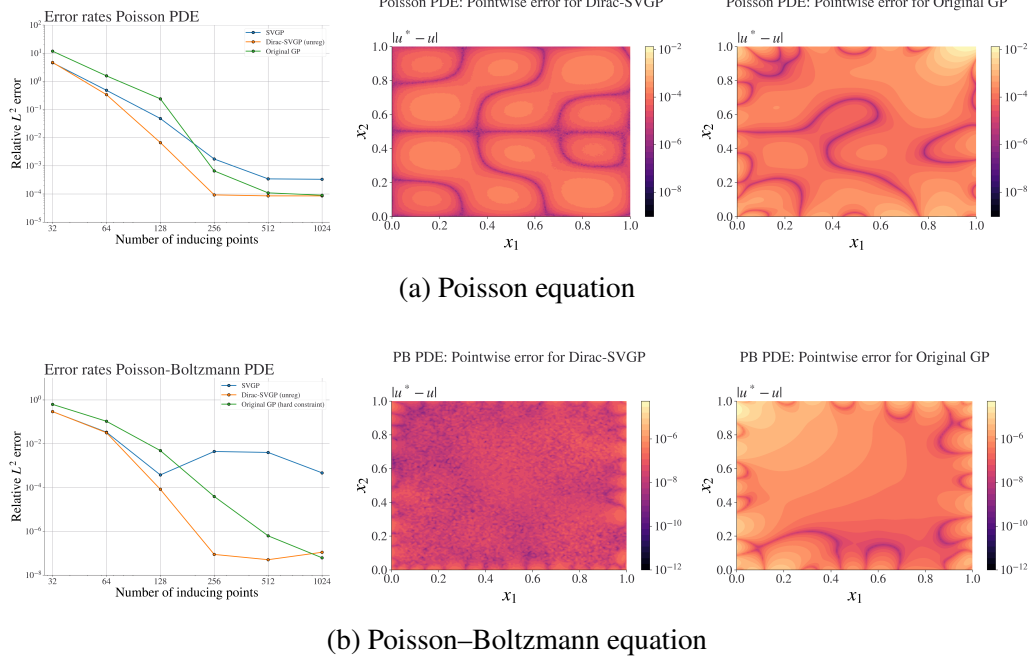


Figure 3.4: Error rate and pointwise error comparison for noiseless forward PDEs.

Coefficient recovery: Darcy problem

We consider the 2D Darcy PDE

$$-\nabla(a \cdot \nabla u) = f \quad x \in \Omega = (0, 1)^2 \quad (3.66)$$

$$u = 0 \quad x \in \partial\Omega \quad (3.67)$$

$$\mathbf{y} = u(\mathbf{x}_{\text{obs}}) + \varepsilon. \quad (3.68)$$

The coefficient a is unknown, and we aim to recover the solution u and coefficient a jointly. We consider a manufactured solution

$$u(x_1, x_2) = \sin(\pi x) \sin(\pi y) + \frac{1}{2} \sin(2\pi x) \sin(2\pi y) \quad (3.69)$$

$$a(x_1, x_2) = \sin(2\pi x) \sin(2\pi y). \quad (3.70)$$

The observations points $\mathbf{x}_{\text{obs}}$ are drawn from two random clusters to mimic a realistic setup where one only has access to localized measurements and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 0.38$ (representing 10% of the standard deviation of u). An example solution is given in Figure 3.5

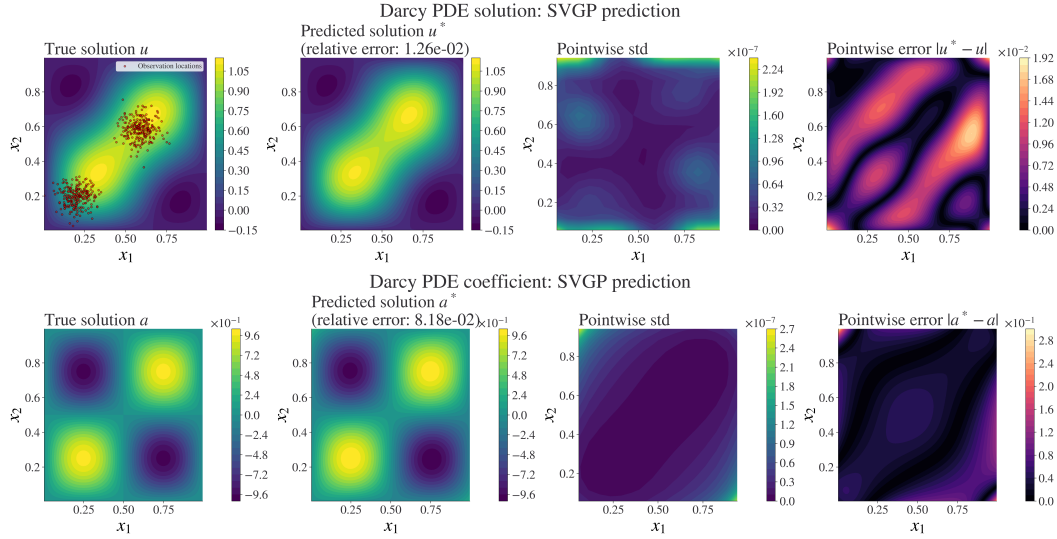


Figure 3.5: Darcy coefficient recovery problem.

Initial condition recovery: Arrhenius equation

We consider the Arrhenius equation with homogeneous Neumann boundary condition but unknown initial condition.

$$\frac{\partial}{\partial t} u = \varepsilon \Delta u + \exp\left(-\frac{\beta}{u}\right) \quad (t, x) \in (0, T) \times (0, 1) \quad (3.71)$$

$$\frac{\partial}{\partial \mathbf{n}} u = 0 \quad (t, x) \in (0, T) \times \{0, 1\} \quad (3.72)$$

$$u(0, x) = u_0(x) \quad x \in [0, 1] \quad (3.73)$$

$$\mathbf{y} = u(\mathbf{t}_{\text{obs}}, \mathbf{x}_{\text{obs}}) + \varepsilon. \quad (3.74)$$

We consider an unknown initial condition

$$u_0(x) = 320 \exp\left(-\frac{1}{2} \frac{(x - 0.35)^2}{0.01^2}\right) x(1 - x) \quad (3.75)$$

and we aim to recover the solution field $u(t, x)$, with particular emphasis on the initial condition $u(0, x) = u_0$. The observations points $(\mathbf{t}_{\text{obs}}, \mathbf{x}_{\text{obs}})$ are located at 10 equispaced spatial points $\{x_i\}_{i=1}^{10}$ but are densely defined in time, to emulate a realistic setup where we have a finite number of spatial sensors measuring continuously in time. We sample the noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$ with $\sigma = 52.95$ (representing 20% of the standard deviation of u). An example solution is given in Figure 3.6.

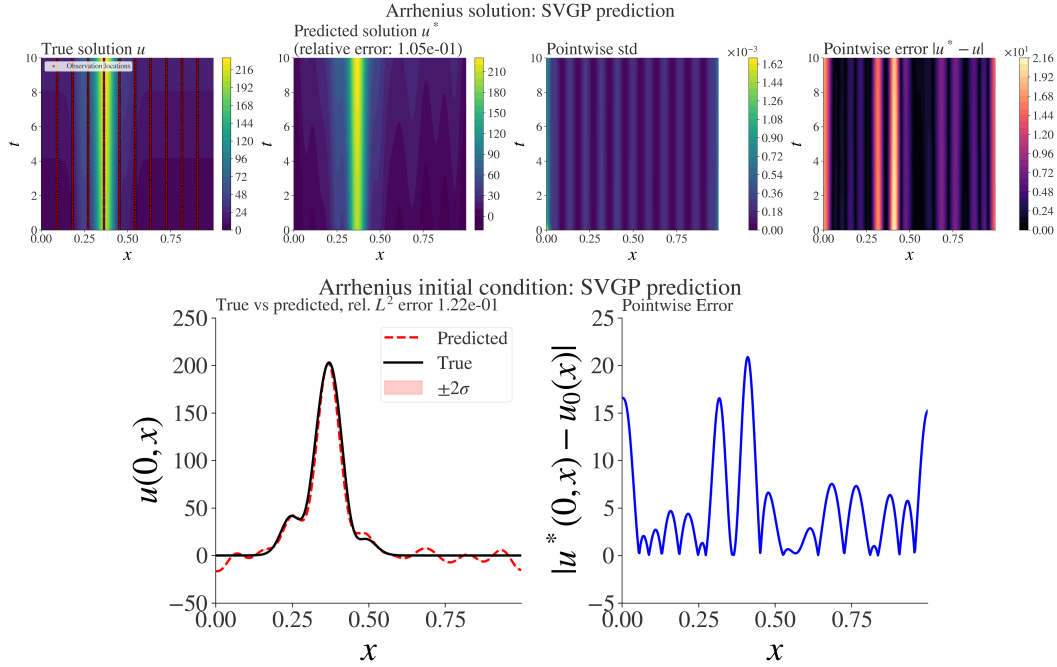


Figure 3.6: Arrhenius initial condition recovery problem.

Forcing term recovery: Burger's equation

We consider the forced viscous Burger equation with homogenous Dirichlet boundary conditions

$$\frac{\partial}{\partial t}u = \varepsilon \Delta u - u \frac{\partial}{\partial x}u + f \quad (t, x) \in (0, T) \times (0, 1) \quad (3.76)$$

$$u = 0 \quad (t, x) \in (0, T) \times \{0, 1\} \quad (3.77)$$

$$u(0, x) = u_0(x) \quad x \in [0, 1] \quad (3.78)$$

$$\mathbf{y} = u(\mathbf{x}) + \varepsilon. \quad (3.79)$$

We consider an unknown forcing term

$$f(x) = 10.0 \sin(2\pi x) \quad (3.80)$$

and we aim to recover both the solution $u(t, x)$ and the forcing term $f(x)$. The observations points $(\mathbf{t}_{\text{obs}}, \mathbf{x}_{\text{obs}})$ are located at 10 equispaced spatial points $\{x_i\}_{i=1}^{10}$ but are densely defined in time, to emulate a realistic setup where we have a finite number of spatial sensors measuring continuously in time. We sample the noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$ with $\sigma = 1.05$ (representing 5% of standard deviation of u). An example solution is given in Figure 3.7.

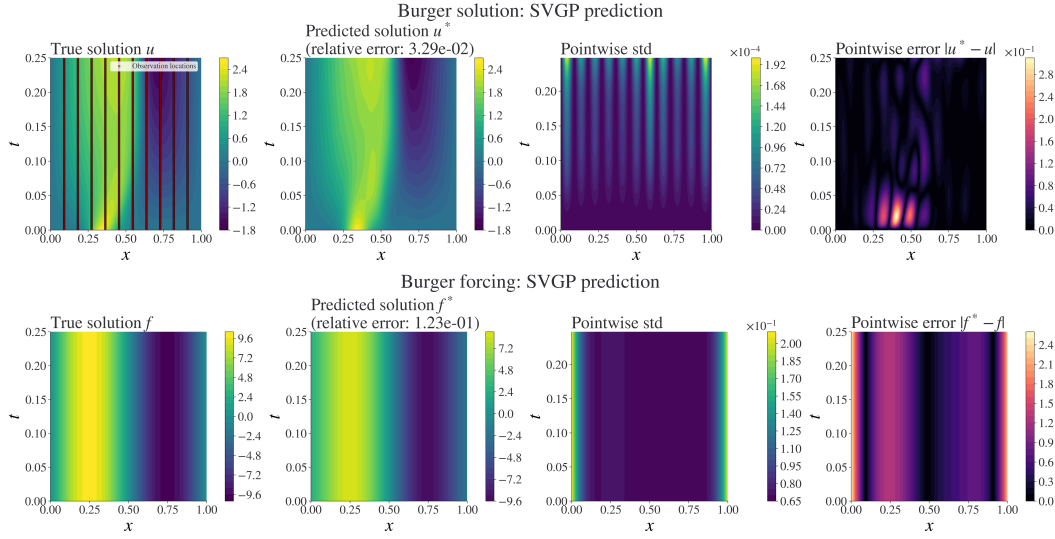


Figure 3.7: Burger forcing recovery problem.

Method	Darcy		Arrhenius		Burger	
	Sol. u	Coef. a	Sol. u	Init. con. u_0	Sol. u	Forc. f
SVGP	1.26×10^{-2}	8.18×10^{-2}	1.05×10^{-1}	1.22×10^{-1}	3.29×10^{-2}	1.23×10^{-1}
Dirac-SVGP	4.58×10^{-3}	4.41×10^{-2}	3.10×10^{-1}	3.66×10^{-1}	4.27×10^{-2}	1.40×10^{-1}
Original GP	3.31×10^{-1}	8.59×10^{-1}	1.48×10^{-1}	1.70×10^{-1}	3.51×10^{-2}	5.40×10^{-1}
PINN	5.93×10^{-2}	5.00×10^{-1}	2.17×10^{-1}	3.22×10^{-1}	1.34×10^{-2}	2.10×10^{-1}

Table 3.2: Comparison of methods across inverse PDE problems (error in relative L^2).

Discussion

Forward problems When the right-hand side is noisy, we consistently observe the best performance with the SVGP approach in Table 3.1. We generally observe that the model requires less tuning of hyperparameters compared to the original GP: we generally find that setting the σ parameter in eq. (3.22) to a small value is sufficient to obtain good performance. The original GP approach appears to require greater tuning and is more sensitive to this parameter. We interpret the continuous loss as improving the model by providing information over the entire domain compared to the original GP which only has access to data at the inducing points. This can be seen as the difference in performance is greatest when using a small number of inducing points.

In the noiseless case the Dirac SVGP in the zero regularization limit performs the best as measured by the error and the error rates, beating other methods by orders of magnitude in some cases. We note however that at high number of inducing

points, the original GP approach performs as well or even slightly better than the Dirac SVGP model. We attribute this to the fact that the optimization becomes less stable without regularization at high number of inducing points. Moreover, when the Dirac SVGP has small (but non-zero) regularization, it performs significantly worse in the noiseless setting. This suggests that there is a qualitative change in the obtained solution when setting the regularization $\gamma = 0$ as opposed to a small value. The SVGP approach (which always uses some form of regularization) also tends to underperform in this setting, and cannot attain errors below 10^{-3} .

Inverse problems For noisy inverse problems, the best solution is provided by the SVGP or the Dirac SVGP table 3.2. In general, we find that both methods perform well compared to the PINN approach, but the original GP can also perform better than the Dirac variant in this setting. The SVGP, however, systematically outperforms the original GP on all problems, supporting the conclusion that a probabilistic approach is beneficial in noisy or ill-posed settings.

We note that in the posterior uncertainty (as measured by the pointwise standard deviation) can be miss calibrated compared to the true error. For example, the uncertainty of the Darcy solution does not capture areas of high error, whereas the uncertainty on the coefficient appears better calibrated Figure 3.5. We observe similar results with the Burger’s solution and forcing term Figure 3.7. This suggests that better calibration is needed to obtain accurate uncertainty quantification.

3.4 Conclusion

We introduced a novel framework for solving PDE forward and inverse problems using Gaussian processes and variational inference. In contrast to previous approaches, we only discretize the prior while keeping the likelihood continuous, which enables a continuous loss for probabilistic solvers, similar to the loss used in PINNs. We showed how to apply this framework to two types of variational inference strategies, one probabilistic using Gaussian variational inference SVGP, one deterministic Dirac-SVGP. We presented several theoretical results to characterize the optimal variational distributions, including their small noise limits. We have not, however, considered convergence rates to the true solution or distribution as our discretization goes to infinity. Empirically, we have tested our framework on several PDE forward and inverse problems, and we have shown that both the SVGP and Dirac SVGP can outperform the original GP approach proposed by Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b and PINNs Raissi, P.

Perdikaris, and G. Karniadakis, 2019b, while providing uncertainty quantification (for SVGP). We note however that the uncertainty quantification provided by the SVGP is not always well calibrated. This suggests that we require proper calibration of this uncertainty Angelopoulos and Bates, 2022; Gneiting, Balabdaoui, and Raftery, 2007. More generally, hyperparameter optimization approaches similar to those considered in Hensman, Fusi, and Neil D. Lawrence, 2013; Nicholas H. Nelsen et al., 2025 could be considered for this problem. Likewise, optimizing the inducing points using methods such as adaptive mesh refinement or gradient based methods has the potential to significantly improve the performance of the method.

Our framework is general and is amenable to several modifications. The first would be to consider a more flexible class of variational distributions beyond the Gaussian VI considered here. A natural future direction is to consider approaches based on normalizing flows Rezende and Shakir Mohamed, 2016. We did consider such approaches, but did not find them to perform as well as the Gaussian VI presented in this paper. Finally, we note that prior others than Gaussian can also be considered: examples include the q-exponential process already considered in G. Yu and Lan, 2025, Besov space priors Saksman and Siltanen, 2009; Lan et al., 2025 or generative priors Mardani et al., 2023; Feng et al., 2023.

Acknowledgments

The authors acknowledge support from the Air Force Office of Scientific Research under MURI award number FA9550-20-1-0358 (Machine Learning and Physics-Based Modeling and Simulation). HO acknowledges support by the Department of Energy under award number DE-SC0023163 (SEA-CROGS: Scalable, Efficient and Accelerated Causal Reasoning Operators, Graphs and Spikes for Earth and Embedded Systems). Additionally, HO acknowledge support from the DoD Vannevar Bush Faculty Fellowship Program.

*Chapter 4***KERNEL METHODS ARE COMPETITIVE FOR OPERATOR
LEARNING**

4.1 Introduction

Operator learning is a well-established field going back at least to the 1970s with the articles Almroth, Stern, and Brogan, 1978; Noor and Peters, 1980 who introduced the reduced basis method as a way speeding up expensive model evaluations. In the most broad sense operator learning arises in the solution of stochastic PDEs Roger G Ghanem and Pol D Spanos, 2003, emulation of computer codes Kennedy and O’Hagan, 2001, reduced order modeling (ROM) Lucia, Beran, and Silva, 2004, and numerical homogenization Houman Owhadi and Scovel, 2019b. In recent years, and with the rise of machine learning, operator learning has become the focus of extensive research with the development of neural net (NN) methods such as Deep Operator Nets L. Lu, Jin, Pang, et al., 2021a and Fourier Neural Nets Zongyi Li, N. Kovachki, et al., 2020a among many others. While these NN methods are often benchmarked against each other L. Lu, X. Meng, et al., 2022, they are rarely compared with the aforementioned classical approaches. Furthermore, the theoretical analysis of NN methods is often limited to density/universal approximation results, showing the existence of a network of a requisite size achieving a certain error rate, without guarantees whether this network is computable in practice (see for example B. Deng et al., 2022; N. Kovachki, Lanthaler, and Mishra, 2021).

In order to alleviate the aforementioned shortcomings we present a mathematical framework for approximation of mappings between Banach spaces using the theory of operator valued reproducing Kernel Hilbert spaces (RKHS) and Gaussian Processes (GPs). Our abstract framework (1) is mathematically simple and interpretable, (2) is convenient to implement, (3) encompasses some of the classical approaches such as linear methods, and (4) comes with a priori error analysis and convergence theory. We further present extensive benchmarking of our kernel method with the DeepONet and FNO approaches and show that the kernel approach either matches or outperforms NN methods in most benchmark examples.

In the remainder of this section we give a summary of our methodology and results: We pose the operator learning problem in Section 4.1 before presenting a running example in Section 4.1 which is used to outline our proposed framework and main theoretical results in Section 4.1 as well as brief numerical results in Section 4.1. Our main contributions are summarized in Section 4.1 followed by a literature review in Section 4.1.

The operator learning problem

Let $\mathcal{U}$ and $\mathcal{V}$ be two (possibly infinite-dimensional) separable Banach spaces and suppose that

$$\mathcal{G}^\dagger : \mathcal{U} \rightarrow \mathcal{V} \quad (4.1)$$

is an arbitrary (possibly nonlinear) operator. Then, broadly speaking, the goal of operator learning is to approximate $\mathcal{G}^\dagger$ from a finite number N of input/output data on $\mathcal{G}^\dagger$. For our framework, we consider the setting where the input/output data are only partially observed through a finite collection of linear measurements which we formalize as follows.

Problem 4. Let $\{u_i, v_i\}_{i=1}^N$ be N elements of $\mathcal{U} \times \mathcal{V}$ such that

$$\mathcal{G}^\dagger(u_i) = v_i, \quad \text{for } i = 1, \dots, N. \quad (4.2)$$

Let $\phi : \mathcal{U} \rightarrow \mathbb{R}^n$ and $\varphi : \mathcal{V} \rightarrow \mathbb{R}^m$ be bounded linear operators. Given the data $\{\phi(u_i), \varphi(v_i)\}_{i=1}^N$ approximate $\mathcal{G}^\dagger$.

Running example

To give context to the above problem and our solution method we briefly outline a running example to which the reader can refer to throughout the rest of this section. Consider the following elliptic PDE, which is of broad interest in geosciences and material science:

$$\begin{cases} -\operatorname{div} e^u \nabla v = w, & \text{in } \Omega, \\ v = 0, & \text{on } \partial\Omega, \end{cases} \quad (4.3)$$

where $\Omega = (0, 1)^2$, $u \in H^3(\Omega)$, $w \in H^1(\Omega)$ and $v \in H^3(\Omega) \cap H_0^1(\Omega)$. For a fixed forcing term w , we wish to approximate the nonlinear operator mapping the diffusion coefficient u to the solution v , i.e., $\mathcal{G}^\dagger : u \mapsto v$. In this case we may take $\mathcal{U} \equiv H^3(\Omega)$ and $\mathcal{V} \equiv H^3(\Omega) \cap H_0^1(\Omega)$. We further assume that a training data set is available in the form of limited observations of input-out pairs. As a canonical example, consider the evaluation bounded and linear operators

$$\phi : u \mapsto (u(X_1), u(X_2), \dots, u(X_n))^T \quad \text{and} \quad \varphi : v \mapsto (v(Y_1), v(Y_2), \dots, v(Y_m))^T, \quad (4.4)$$

where the $\{X_j\}_{j=1}^n$ and $\{Y_j\}_{j=1}^m$ are distinct collocation points in the domain Ω as well as pairs $\{u_i, v_i\}_{i=1}^N$ that satisfy the PDE (4.3). Then our goal is to approximate $\mathcal{G}^\dagger$ from the training data set $\{\phi(u_i), \varphi(v_i)\}_{i=1}^N$ ¹.

¹Choosing ϕ, φ as pointwise evaluation functionals is common to many applications, although our abstract framework readily accommodates other choices such as integral operators and basis projections.

The proposed solution

Our setup naturally gives rise to a commutative diagram depicted in Figure 4.1. Here the map $f^\dagger : \mathbb{R}^n \rightarrow \mathbb{R}^m$, explicitly defined as

$$f^\dagger := \varphi \circ \mathcal{G}^\dagger \circ \psi, \quad (4.5)$$

is a mapping between finite-dimensional Euclidean spaces, and is therefore amenable to numerical approximation. However, in order to approximate $\mathcal{G}^\dagger$ we also need the reconstruction maps $\psi : \mathbb{R}^n \rightarrow \mathcal{U}$ and $\chi : \mathbb{R}^m \rightarrow \mathcal{V}$.

$$\begin{array}{ccc} \mathcal{U} & \xrightarrow{\mathcal{G}^\dagger} & \mathcal{V} \\ \psi \updownarrow \phi & \xrightarrow{f^\dagger} & \chi \updownarrow \varphi \\ \mathbb{R}^n & \xrightarrow{\quad} & \mathbb{R}^m \end{array}$$

Figure 4.1: Commutative diagram of our operator learning setup.

Our proposed solution is to endow $\mathcal{U}$ and $\mathcal{V}$ with an RKHS structure and use kernel/GP regression to identify the maps ψ and χ . As a prototypical example we consider the situation where $\mathcal{U}$ is an RKHS of functions $u : \Omega \rightarrow \mathbb{R}$ defined by a kernel $Q : \Omega \times \Omega \rightarrow \mathbb{R}$ and $\mathcal{V}$ is an RKHS of functions $u : D \rightarrow \mathbb{R}$ defined by a kernel $K : D \times D \rightarrow \mathbb{R}$. For our running example, we have $D = \Omega$, and we can take Q and K to be Matérn like kernels, e.g., the Green's function of elliptic PDEs (possibly on Ω or restricted to Ω) with appropriate regularity. One can also choose Q, K to be smoother kernels such that their RKHSs are embedded in $\mathcal{U}$ and $\mathcal{V}$.

We then define ψ and χ as the following optimal recovery maps²:

$$\begin{aligned} \psi(U) &:= \arg \min_{w \in \mathcal{U}} \|w\|_Q \quad \text{s.t.} \quad \phi(w) = U, \\ \chi(V) &:= \arg \min_{w \in \mathcal{V}} \|w\|_K \quad \text{s.t.} \quad \varphi(w) = V, \end{aligned} \quad (4.6)$$

where $\|\cdot\|_Q$ and $\|\cdot\|_K$ are the RKHS norms arising from their pertinent kernels.

In the case where ϕ and φ are pointwise evaluation maps ($\phi(u) = (u(X_1), \dots, u(X_n))$ and $\varphi(v) = (v(Y_1), \dots, v(Y_m))$ where the X_i and Y_j are pairwise distinct collocation points in Ω and D), our optimal recovery maps can be expressed in closed form

²It is possible to define the optimal recovery maps ψ, χ in the setting where ϕ and ψ are nonlinear, following the general framework of Y. Chen, Hosseini, Houman Owhadi, and Andrew M Stuart, 2021a; Houman Owhadi, 2022b; Houman Owhadi, 2023a. However, in this setting the closed form formulae (4.7) no longer hold.

using standard representer theorems for kernel interpolation Scholkopf, Herbrich, and Smola, 2001a:

$$\psi(U)(x) = Q(x, X)Q(X, X)^{-1}U, \quad \chi(V)(y) = K(y, Y)K(Y, Y)^{-1}V, \quad (4.7)$$

where $Q(X, X)$ and $K(Y, Y)$ are kernel matrices with entries $Q(X, X)_{ij} = Q(X_i, X_j)$ and $K(Y, Y)_{ij} = K(Y_i, Y_j)$ respectively, while $Q(x, X)$ and $K(y, Y)$ denote row-vector fields with entries $Q(x, X)_i = Q(x, X_i)$ and $K(y, Y)_i = K(y, Y_i)$.

We further propose to approximate $f^\dagger$ by optimal recovery in a vector-valued RKHS. Let $\Gamma : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathcal{L}(\mathbb{R}^m)$ be a matrix valued kernel Alvarez, Rosasco, Neil D Lawrence, et al., 2012; here $\mathcal{L}(\mathbb{R}^m)$ is the space of $m \times m$ matrices) with RKHS $\mathcal{H}_\Gamma$ equipped with the norm $\|\cdot\|_\Gamma$ ³ and proceed to approximate $f^\dagger$ by the map $\bar{f}$ defined as

$$\bar{f} := \arg \min_{f \in \mathcal{H}_\Gamma} \|f\|_\Gamma \quad \text{s.t.} \quad f(\phi(u_i)) = \varphi(v_i) \quad \text{for } i = 1, \dots, N.$$

A simple and practical choice for Γ is the diagonal kernel

$$\Gamma(U, U') = S(U, U')I, \quad (4.8)$$

where $S : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is an arbitrary scalar-valued kernel, such as RBF, Laplace, or Matérn, and I is the $m \times m$ identity matrix. More complicated choices, such as sums of kernels or replacing the identity matrix with a fixed positive definite matrix that induces correlations between output coordinates, are also possible. However, these may lead to greater computational cost, and we observe empirically that the simple choice of the identity matrix already provides good performance. Then we can approximate the components of $\bar{f}$ via the independent optimal recovery problems

$$\bar{f}_j := \arg \min_{g \in \mathcal{H}_S} \|g\|_S \quad \text{s.t.} \quad g(\phi(u_i)) = \varphi_j(v_i), \quad \text{for } i = 1, \dots, N. \quad (4.9)$$

for $j = 1, \dots, m$. Here we wrote $\varphi_j(v_i)$ for the entry j of the vector $\varphi(v_i)$ and, as our notation suggests, $\mathcal{H}_S$ is the RKHS of S equipped with the norm $\|\cdot\|_S$. Since (4.9) is a standard optimal recovery problem, each $\bar{f}_j$ can be identified by the usual representer formula:

$$\bar{f}_j(U) = S(U, \mathbf{U})S(\mathbf{U}, \mathbf{U})^{-1}\mathbf{V}_{\cdot,j}, \quad (4.10)$$

³See Section C.1 for a review of operator-valued kernels or the reference Kadri, Duflos, Preux, Stephane Canu, et al., 2016a.

where $\mathbf{U} := (\phi(u_1), \dots, \phi(u_N))$ and $\mathbf{V}_{\cdot,j} := (\varphi_j(v_1), \dots, \varphi_j(v_N))^T$ and $S(U, \mathbf{U})$ is a block-vector and $S(\mathbf{U}, \mathbf{U})$ is a block-matrix defined in an analogous manner to those in (4.7). By combining equations (4.7) and (4.10) we obtain the operator

$$\bar{\mathcal{G}} := \chi \circ \bar{f} \circ \phi \quad (4.11)$$

as an approximation to $\mathcal{G}^\dagger$. We provide further details and generalize the proposed framework in Section 4.2 to the setting where ϕ and φ are obtained from arbitrary linear measurements (e.g., integral operators as in tomography) and $\mathcal{U}$ and $\mathcal{V}$ may not be spaces of continuous functions.

Convergence guarantee

Under suitable regularity assumptions on $\mathcal{G}^\dagger$, our method comes with worst-case convergence guarantees as the number of data points N , i.e., input-output pairs and the number of collocations points n and m go to infinity. We present here a condensed version of this result and defer the proof to Section 4.3. Below we write $B_R(\mathcal{H})$ for the ball of radius $R > 0$ in a normed space $\mathcal{H}$.

Theorem 7 (Condensed version of Thm. 9). *Suppose it holds that:*

- (7.1) *(Regularity of the domains Ω and D) Ω and D are compact sets of finite dimensions d_Ω and d_D and with Lipschitz boundary.*
- (7.2) *(Regularity of the kernels Q and K). Assume that $\mathcal{H}_Q \subset H^s(\Omega)$ and $\mathcal{H}_K \subset H^t(D)$ for some $s > d_\Omega/2$ and some $t > d_D/2$ with inclusions indicating continuous embeddings.*
- (7.3) *(Space filling property of collocation points) The fill distance between the collocation points $\{X_i\}_{i=1}^n \subset \Omega$ and the $\{Y_j\}_{j=1}^m \subset D$ goes to zero as $n \rightarrow \infty$ and $m \rightarrow \infty$.*
- (7.4) *(Regularity of the operator $\mathcal{G}^\dagger$) The operator $\mathcal{G}^\dagger$ is continuous from $H^{s'}(\Omega)$ to $\mathcal{H}_K$ for some $s' \in (0, s)$ as well as from $\mathcal{U}$ to $\mathcal{V}$ and all its Fréchet derivatives are bounded on $B_R(\mathcal{H}_Q)$ for any $R > 0$.*
- (7.5) *(Regularity of the kernels S^n) Assume that for any $n \geq 1$ and any compact subset Υ of $\mathbb{R}^n$, the RKHS of S^n restricted to Υ is contained in $H^r(\Upsilon)$ for some $r > n/2$ and contains $H^{r'}(\Upsilon)$ for some $r' > 0$ that may depend on n .*

(7.6) (*Resolution and space-filling property of the data*) Assume that for n sufficiently large, the data points $(u_i)_{i=1}^N \subset B_R(\mathcal{H}_Q)$ belong to the range of ψ^n and are space filling in the sense that they become dense in $\phi^n(B_R(\mathcal{H}_Q))$ as $N \rightarrow \infty$.

Then, for all $t' \in (0, t)$,

$$\lim_{n,m \rightarrow \infty} \lim_{N \rightarrow \infty} \sup_{u \in B_R(\mathcal{H}_Q)} \|\mathcal{G}^\dagger(u) - \chi^m \circ \bar{f}_N^{m,n} \circ \phi^n(u)\|_{H'(D)} \rightarrow 0, \quad (4.12)$$

where our notation makes the dependence of ψ, ϕ, χ, S and $\bar{f}$ on n, m and N explicit.

We note that Assumptions (7.1)–(7.3) are standard, and concern the accuracy of the optimal recovery maps ϕ^n and χ^m as $n, m \rightarrow \infty$. Assumptions (7.4)–(7.5) are less standard and amount to regularity assumptions on the map $\mathcal{G}^\dagger$ while Assumption (7.6) concerns the acquisition and regularity of the training data set.

In Section 4.3 we also present Theorem 11 as the quantitative analogue of the above result which characterizes how the speed of convergence depends on the regularity of the operator $\mathcal{G}^\dagger$ and the choice of ϕ and φ in the setting of pointwise measurement operators. We also comment on how this analysis could be extended to other linear measurements.

Numerical Framework

Returning to our running example, we implement the proposed framework for learning the non-linear operator mapping u to v in (4.3). We consider 1000 inputs and outputs of u and v . The data is taken from L. Lu, X. Meng, et al., 2022 and the experimental setup is discussed further in Section 4.4. We take φ to be of the form (4.4) with $m = 841$ while we define ϕ through a PCA pre-processing step. More precisely, let $\phi_{\text{pointwise}}$ be of the form (4.4) with $n = 841$. Choose $n_{\text{PCA}} = 202$ (this value captures 95% of the empirical variance of our training data) and define

$$\phi(u) = \Pi_{\text{PCA}} \circ \phi_{\text{pointwise}}(u) \in \mathbb{R}^{202}. \quad (4.13)$$

In other words, we take our ϕ map to be the linear map that computes the first 202 PCA coefficients of the input functions u given on a uniform grid; observe that we do not use PCA pre-processing on the output data here although we do this for some of our other examples in Section 4.4 for better performance.

With ϕ and φ identified (recall Figure 4.1) we proceed to implement our kernel method using the simple choice of a diagonal kernel $S(U, U')I$ where S is a rational

quadratic (RQ) kernel (see Section C.3). This choice transforms the problem into 841 independent kernel regression problems, each corresponding to one component of $f^\dagger$ (i.e., the $f_j^\dagger$'s).

We used the PCA and kernel regression modules of the `scikit-learn` Python library Pedregosa et al., 2011 to implement our algorithm. This implementation automatically selects the best kernel parameters by maximizing the marginal likelihood function Rasmussen and Williams, 2006 jointly for all problems. Our proposed method can therefore be implemented conveniently using off-the-shelf software. Figure 4.2 illustrates examples of the inputs and outputs of our operator learning problem. Despite the simple implementation of our method, we are able to obtain competitive accuracy as shown in Table 4.1 where the relative testing L^2 loss of our method is compared to other popular algorithms. Moreover, our approach is amenable to well-known numerical analysis techniques, such as sparse or low-rank approximation of kernel matrices, to reduce its complexity. For the present example (and those in Section 4.4) we only consider “vanilla” kernel methods which compute (4.10) by computing the full Cholesky factors of the matrix $S(\mathbf{U}, \mathbf{U})$.

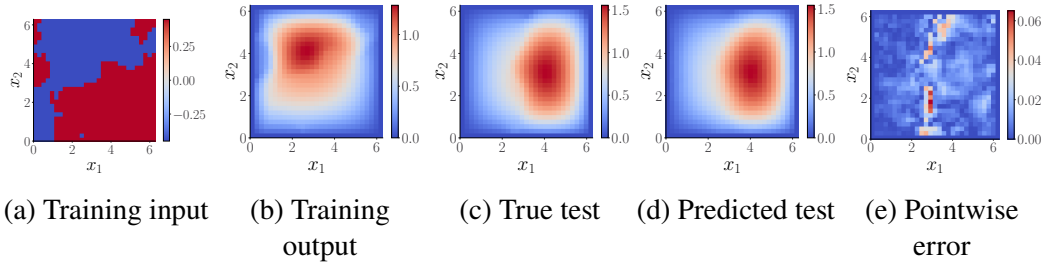


Figure 4.2: Example of training data and test prediction and pointwise errors for the Darcy flow problem (4.3).

Method	Accuracy
DeepONet	2.91 %
FNO	2.41 %
POD-DeepONet	2.32 %
Linear	6.74 %
Rational quadratic	2.87%

Table 4.1: The L^2 relative test error of the Darcy flow problem in our running example. The kernel approach is compared with variations of DeepONet and FNO. Results of our kernel method are presented below the dashed line with the pertinent choice of the kernel S .

Summary of contributions

The main results of the article concern the properties, performance, and error analysis of the map $\tilde{\mathcal{G}}$ defined in (4.11). Our contributions can be summarized under four categories:

1. **An abstract kernel framework for operator learning:** In Section 4.2, we propose a framework for operator learning using kernel methods with several desirable properties. A family of methods of increasing complexity is proposed that includes linear models and diagonal kernels as well as non-diagonal kernels which capture output correlations. These properties make our approach ideal for benchmarking purposes. Furthermore, the methodology is (i) applicable to any choice of the linear functionals φ and ϕ , (ii) minimax optimal with respect to an implicitly defined operator-valued kernel, and (iii) is mesh-invariant. We emphasize in remark 10 that our optimal recovery maps can be applied to *any* operator learning after training to obtain a mesh-invariant pipeline.
2. **Error analysis and convergence rates for $\tilde{\mathcal{G}}$:** In Section 4.3, we develop rigorous worst-case a priori error bounds and convergence guarantees for our method: Theorem 11 provides quantitative error bounds while Theorem 9 (the detailed version of Theorem 11) shows the convergence of $\tilde{\mathcal{G}} \rightarrow \mathcal{G}$ under appropriate conditions.
3. **A simple to use vanilla kernel method:** While our abstract kernel method is quite general, our numerical implementation in Section 4.4 focuses on a simple, easy-to-implement version using diagonal kernels of the form (4.8). Off-the-shelf software, such as the kernel regression modules of `scikit-learn`, can be employed for this task. We empirically observe low training times and robust choice of hyperparameters. These properties further suggest that kernel methods are a good baseline for benchmarking of more complex methods.
4. **Competitive performance.** In Section 4.4 we present a series of numerical experiments on benchmark PDE problems from the literature and observe that our simple implementation of the kernel approach is competitive in terms of complexity-accuracy tradeoffs in comparison to several NN-based methods. Since kernel methods can be interpreted as an infinite-width, one-layer NN, the results raise the question of how much of a role the depth of a deep NN plays in the performance of algorithms for the purposes of operator learning.

Review of relevant literature

In the most broad sense, operator learning is the problem of approximating a mapping between two infinite-dimensional function spaces Bhattacharya, Hosseini, Nikola B. Kovachki, et al., 2021b; De Hoop et al., 2022. In recent years, this problem has become an area of intense research in the scientific machine learning community with a particular focus on parametric or stochastic PDEs. However, the approximation of such parameter to solution maps has been an area of intense research in the computational mathematics and engineering communities, going back at least to the reduced basis method introduced in the 1970s Almroth, Stern, and Brogan, 1978; Noor and Peters, 1980 as a way of speeding up the solution of families of parametric PDEs in applications that require many PDE solves such as design Economon et al., 2016; Martins and Lambe, 2013; Bendsoe and Sigmund, 2003; Boncoraglio and Farhat, 2021, uncertainty quantification (UQ) Sudret, Marelli, and Wiart, 2017; Martin et al., 2012; D. Z. Huang, Schneider, and Andrew M Stuart, 2022, and multi-scale modeling Weinan, 2011; Fish et al., 1997; Feyel and Chaboche, 2000; N. Kovachki, B. Liu, et al., 2022. In what follows we give a brief summary of the various areas and methodologies that overlap with operator learning; we cannot provide an exhaustive list of references due to space, but refer the reader to key contributions and surveys where further references can be found.

Deep learning techniques The use of NNs for operator learning goes back at least to the '90s and the seminal works of Chen and Chen T. Chen and H. Chen, 1995b; T. Chen and H. Chen, 1995a who proved a universal approximation theorem for NN approximations to operators. The use and design of NNs for operator learning has become popular in the last five years as a consequence of growing interest in NNs for scientific computing starting with the article Zhu and Zabaras, 2018 which used autoencoders to build surrogates for UQ of subsurface flow models. Since then many different approaches have been proposed some of which use specific architectures or target particular families of PDEs J. Hesthaven and Ubbiali, 2018; Khoo and Ying, 2019; Zongyi Li, N. Kovachki, et al., 2020a; Khoo, J. Lu, and Ying, 2021; L. Lu, Jin, Pang, et al., 2021a; Gin et al., 2021; Boulle and Townsend, 2022; Kropfl, Maier, and Peterseim, 2022; Kissas et al., 2022. The most relevant of among these methods to our proposed framework are the DeepONet family L. Lu, Jin, Pang, et al., 2021a; Sifan Wang, H. Wang, and Paris Perdikaris, 2021b; L. Lu, X. Meng, et al., 2022; Sifan Wang, H. Wang, and Paris Perdikaris, 2022, FNO Zongyi Li, N. Kovachki, et al., 2020a, and PCA-Net J. Hesthaven and Ubbiali, 2018; Bhattacharya, Hosseini,

Nikola B. Kovachki, et al., 2021b where the main novelty appears to be the use of novel, flexible, and expressive NN architectures that allow the algorithm to learn and adapt the bases that are selected for the input and outputs of the solution map as well as possible nonlinear dependencies between the basis coefficients. Although not part of our comparisons, we note that Fan, Bohorquez, and Ying, 2019; Fan, Feliu-Faba, et al., 2019; Fan, L. Lin, et al., 2019 obtained competitive accuracy by using deep neural networks with architectures inspired by conventional fast solvers.

Classical numerical approximation methods Operator learning has been the subject of intense research in the computational mathematics literature in the context of stochastic Galerkin methods Roger G Ghanem and Pol D Spanos, 2003; Xiu and Shen, 2009, polynomial chaos Xiu, 2010; Xiu and G. E. Karniadakis, 2002b, reduced basis methods Noor and Peters, 1980; Maday, Patera, and Turinici, 2002 and numerical homogenization Houman Owhadi and L. Zhang, 2007a; Houman Owhadi, 2015b; Houman Owhadi and Scovel, 2019b; Altmann, Henning, and Peterseim, 2021. In the setting of stochastic and parametric PDEs, the the goal is often to approximate the solution of a PDE as a function of a random or uncertain parameter. The well-established approach to such problems is to pick or construct appropriate bases for the input parameter and the solution of the PDE and then construct a parametric, high-dimensional map that transforms the input basis coefficients to the output coefficients. Well-established methods such as polynomial chaos, stochastic finite element methods, reduced basis methods Roger G Ghanem and Pol D Spanos, 2003; Xiu, 2010; Cohen and DeVore, 2015; J. S. Hesthaven, Rozza, Stamm, et al., 2016; Lucia, Beran, and Silva, 2004 fall within this category. A vast amount of literature in applied mathematics exists on this subject, and the theoretical analysis of these methods is extensive; see for example Beck et al., 2012; Chkifa, Cohen, DeVore, et al., 2012; Chkifa, Cohen, and Schwab, 2014; Nobile, Raul Tempone, and Webster, 2008a; Nobile, Raul Tempone, and Webster, 2008b; Gunzburger, Webster, and G. Zhang, 2014 and references therein.

Operator compression For solving PDEs, the objectives of operator learning are also similar to those of operator compression Feischl and Peterseim, 2020; Kropfl, Maier, and Peterseim, 2022 as formulated in numerical homogenization Houman Owhadi and Scovel, 2019b; Altmann, Henning, and Peterseim, 2021 and reduced order modeling (ROM) Amsallem and Farhat, 2008; Lucia, Beran, and Silva, 2004, i.e., the approximation of the solution operator from pairs of solutions

and source/forcing terms. While both ROM and numerical homogenization seek operator compression through the identification of reduced basis functions that are as accurate as possible (this translates into low-rank approximations with SVD and its variants Boule and Townsend, 2022), numerical homogenization also requires those functions to be as localized as possible Malqvist and Peterseim, 2014 and in turn leverages both low rank and sparse approximations. These localized reduced basis functions are known as Wannier functions in the physics literature Marzari et al., 2012, and can be interpreted as linear combinations of eigenfunctions that are localized in both frequency space and the physical domain, akin to wavelets. The hierarchical generalization of numerical homogenization Houman Owhadi, 2017 (gamblets) has led to the current state-of-the-art for operator compression of linear elliptic Schaafer, Katzfuss, and Houman Owhadi, 2021; Schafer, Sullivan, and Houman Owhadi, 2021 and parabolic/hyperbolic PDEs Houman Owhadi and L. Zhang, 2017. In particular, for arbitrary (and possibly unknown) elliptic PDEs Schafer and Houman Owhadi, 2021 shows that the solution operator (i.e., the Green’s function) can be approximated in near-linear complexity to accuracy ϵ from only $O(\log^{d+1}(\frac{1}{\epsilon}))$ solutions of the PDE.

GP emulators In the case where the range of the operator of interest is finite dimensional, then operator learning coincides with surrogate modeling techniques that were developed in the UQ literature, such as GP surrogate modeling/emulation Kennedy and O’Hagan, 2001; Bastos and Anthony O’Hagan, 2009. When the kernels of the underlying GPs are also learned from data Houman Owhadi and Gene Ryan Yoo, 2019b; Y. Chen, Houman Owhadi, and A. Stuart, 2021, GP surrogate modeling has been shown to offer a simple, low-cost, and accurate solution to learning dynamical systems Boumediene Hamzi and Houman Owhadi, 2021b, geophysical forecasting Boumediene Hamzi, Maulik, and Houman Owhadi, 2021, and radiative transfer emulation Jouni Susiluoto, Braverman, P. Brodrick, et al., 2021, and the inference of the structure of convective storms from passive microwave observations Prasanth, Z. Haddad, et al., 2021. Indeed, our proposed kernel framework for operator learning can be interpreted as an extension of these well-established GP surrogates to the setting where the range of the operator is a function space.

Outline of the article

The remainder of the article is organized as follows: We present our operator learning framework in Section 4.2 for the generalized setting where ϕ, φ can be

any collection of bounded and linear operators along with an interpretation of our method from the GP perspective. Our convergence analysis and quantitative error bounds are presented in Section 4.3 where we present the full version of Theorem 9. Our numerical experiments, implementation details, and benchmarks against FNO and DeepONet are collected in Section 4.4. We discuss future directions and open problems in Section 4.5. The appendix collects a review of operator valued kernels and GPs along with other auxiliary details.

4.2 The RKHS/GP framework for operator learning

We now present our general kernel framework for operator learning, i.e., the proposed solution to Problem 4. We emphasize that here we do not require the spaces $\mathcal{U}$ and $\mathcal{V}$ to be spaces of continuous functions and in particular, we do not require the maps ϕ and φ to be obtained from pointwise measurements. To describe this, we will introduce the dual spaces of $\mathcal{U}$ and $\mathcal{V}$ to define optimal recovery with respect to kernel operators rather than just kernel functions.

Write $\mathcal{U}^*$ and $\mathcal{V}^*$ for the duals of $\mathcal{U}$ and $\mathcal{V}$, and write $[\cdot, \cdot]$ for the pertinent duality pairings. Assume that $\mathcal{U}$ is endowed with a quadratic norm $\|\cdot\|_Q$, i.e., there exists a linear bijection $Q : \mathcal{U}^* \rightarrow \mathcal{U}$ that is symmetric ($[\phi_a, Q\phi_b] = [\phi_b, Q\phi_a]$), positive ($[\phi_a, Q\phi_a] > 0$ for $\phi_a \neq 0$), and such that $\|u\|_Q^2 = [Q^{-1}u, u]$, $\forall u \in \mathcal{U}$.

As in Houman Owhadi and Scovel, 2019b, Ch. 11, although $\mathcal{U}$, and $\mathcal{U}^*$ are also Hilbert spaces under $\|\cdot\|_Q$ and its dual norm $\|\cdot\|_Q^*$ (with inner products $\langle u, v \rangle_Q = [Q^{-1}u, v]$ and $\langle \phi_a, \phi_b \rangle_Q^* = [\phi_a, Q\phi_b]$), we will keep using the Banach space terminology to emphasize the fact that our dual pairings will not be based on the inner product through the Riesz representation theorem, but on a different realization of the dual space, as this setting is more practical.

If $\mathcal{U}$ is a space of continuous functions on a subset $\Omega \subset \mathbb{R}^{d_\Omega}$ then $\mathcal{U}^*$ contains delta Dirac functions and, to simplify notations, we also write $Q(x, y) := [\delta_x, Q\delta_y]$ for $x, y \in \mathbb{R}^{d_\Omega}$ to denote the kernel induced by the operator Q . Note that in that case, $\mathcal{U}$ is a RKHS with norm $\|\cdot\|_Q$ induced by the kernel Q . Since ϕ is bounded and linear, its entries ϕ_i (write $\phi := (\phi_1, \dots, \phi_n)$) must be elements of $\mathcal{U}^*$. We assume those elements to be linearly independent. Write $\psi : \mathbb{R}^n \rightarrow \mathcal{U}$ for the linear operator defined by

$$\psi(Y) := (Q\phi) Q(\phi, \phi)^{-1} Y \text{ for } Y \in \mathbb{R}^n, \quad (4.14)$$

where we write $Q(\phi, \phi)$ for the $n \times n$ symmetric positive definite (SPD) matrix with

entries $Q(\phi_i, \phi_j) := [\phi_i, Q\phi_j]$ ⁴ and $Q\phi$ for $(Q\phi_1, \dots, Q\phi_n) \in \mathcal{U}^n$. As described in Houman Owahdi and Scovel, 2019b, Chap. 11, for $u \in \mathcal{U}$, given $\phi(u) = Y$, $\psi(Y)$ is the minmax optimal recovery of u when using the relative error in $\|\cdot\|_Q$ -norm as a loss.

Similarly, assume that $\mathcal{V}$ is endowed with a quadratic norm $\|\cdot\|_K$, defined by the symmetric positive linear bijection $K : \mathcal{V}^* \rightarrow \mathcal{V}$. Write $\varphi := (\varphi_1, \dots, \varphi_m)$ and assume the entries of φ to be linearly independent elements of $\mathcal{V}^*$. Using the same notations as in (4.14) write $\chi : \mathbb{R}^m \rightarrow \mathcal{V}$ for the linear operator defined by

$$\chi(Z) := (K\varphi) K(\varphi, \varphi)^{-1} Z \text{ for } Z \in \mathbb{R}^m. \quad (4.15)$$

Then, as above, for $v \in \mathcal{V}$, given $\varphi(v) = Z$, $\chi(Z)$ is the minmax optimal recovery of v when using the relative error in $\|\cdot\|_K$ -norm as a loss.

Write $\mathcal{L}(\mathbb{R}^m)$ for the space of bounded linear operators mapping $\mathbb{R}^m$ to itself, i.e., $m \times m$ matrices. Let $\Gamma : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathcal{L}(\mathbb{R}^m)$ be a matrix-valued kernel Alvarez, Rosasco, Neil D Lawrence, et al., 2012 defining an RKHS $\mathcal{H}_\Gamma$ of continuous functions $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ equipped with an RKHS norm $\|\cdot\|_\Gamma$. For $i \in \{1, \dots, N\}$, write $U_i := \phi(u_i)$ and $V_i := \varphi(v_i)$. Write $\mathbf{U}$ and $\mathbf{V}$ for the block-vectors with entries U_i and V_i . Write $\Gamma(\mathbf{U}, \mathbf{U})$ for the $N \times N$ block-matrix with entries $\Gamma(U_i, U_j)$ and assume $\Gamma(\mathbf{U}, \mathbf{U})$ to be invertible (which is satisfied if Γ is non-degenerate and $U_i \neq U_j$ for $i \neq j$). Let $f^\dagger$ be an element of $\mathcal{H}_\Gamma$ and write $f^\dagger(\mathbf{U})$ for the block vector with entries $f^\dagger(U_i)$. Then given $f^\dagger(\mathbf{U}) = \mathbf{V}$ it follows that

$$\bar{f}(U) := \Gamma(U, \mathbf{U}) \Gamma(\mathbf{U}, \mathbf{U})^{-1} \mathbf{V} \quad (4.16)$$

is the minimax optimal recovery of $f^\dagger$, where $\Gamma(\cdot, \mathbf{U})$ is the block-vector with entries $\Gamma(\cdot, U_i)$.

To this end, we propose to approximate the ground truth operator $\mathcal{G}^\dagger$ with

$$\bar{\mathcal{G}} := \chi \circ \bar{f} \circ \phi, \quad (4.17)$$

also recall Figure 4.1. Combining (4.15) and (4.16) we further infer that $\bar{\mathcal{G}}$ admits the following explicit representer formula:

$$\bar{\mathcal{G}}(u) = (K\varphi) K(\varphi, \varphi)^{-1} \Gamma(\phi(u), \mathbf{U}) \Gamma(\mathbf{U}, \mathbf{U})^{-1} \mathbf{V}. \quad (4.18)$$

⁴For linear measurements involving derivatives the computation of these kernel matrices requires the computation of derivatives of the kernels; see Y. Chen, Hosseini, Houman Owahdi, and Andrew M Stuart, 2021a for practical examples and considerations.

In the remainder of this section we will provide more details and observations regarding our approximate operator $\tilde{\mathcal{G}}$ that is useful later in Section 4.3 and of independent interest.

The kernel and RKHS associated with $\tilde{\mathcal{G}}$

The explicit formula (4.18) suggests that the operator $\tilde{\mathcal{G}}$ is an element of an RKHS defined by an operator-valued kernel, which we now characterize. For $u_1, u_2 \in \mathcal{U}$ and $v \in \mathcal{V}$ write

$$G(u_1, u_2)v := (K\varphi)(K(\varphi, \varphi))^{-1}\Gamma(\phi(u_1), \phi(u_2))(K(\varphi, \varphi))^{-1}\varphi(v). \quad (4.19)$$

It turns out that $G : \mathcal{U} \times \mathcal{U} \rightarrow \mathcal{L}(\mathcal{V})$ is a well-defined operator-valued kernel whose RKHS contains operators of the form $\tilde{\mathcal{G}}$.

Proposition 7. *The kernel G in (4.19) is an operator-valued kernel. Write $\mathcal{H}_G$ for its RKHS and $\|\cdot\|_G$ for the associated norm. Then it holds that $\mathcal{G} \in \mathcal{H}_G$ if and only if $\mathcal{G} = \chi \circ f \circ \phi$ for $f = \varphi \circ \mathcal{G} \circ \psi \in \mathcal{H}_\Gamma$ and $\|\mathcal{G}\|_G = \|f\|_\Gamma$.*

Proof. Since G is Hermitian and positive, we deduce that G is an operator-valued kernel. Indeed for $\tilde{u}_1, \dots, \tilde{u}_m \in \mathcal{U}$ and $\tilde{v}_1, \dots, \tilde{v}_m \in \mathcal{V}$, using $\langle \tilde{v}_i, K\varphi_s \rangle_K = \varphi_s(\tilde{v}_i)$ and the fact that Γ is a matrix-valued kernel we have

$$\begin{aligned} \langle \tilde{v}_i, G(\tilde{u}_i, \tilde{u}_j)\tilde{v}_j \rangle_K &= \varphi(\tilde{v}_i)^T (K\varphi)(K(\varphi, \varphi))^{-1}\Gamma(\phi(\tilde{u}_i), \phi(\tilde{u}_j))(K(\varphi, \varphi))^{-1}\varphi(\tilde{v}_j) \\ &= \langle G(\tilde{u}_j, \tilde{u}_i)\tilde{v}_i, \tilde{v}_j \rangle_K, \end{aligned} \quad (4.20)$$

where we used $\langle \tilde{v}_i, K\varphi_s \rangle_K = \varphi_s(\tilde{v}_i)$ and the fact that Γ is a matrix-valued kernel. Furthermore, summing (4.20), we deduce that $\sum_{i,j=1}^m \langle \tilde{v}_i, G(\tilde{u}_i, \tilde{u}_j)\tilde{v}_j \rangle_K \geq 0$. From (4.19) we infer

$$\sum_{j=1}^m G(u, \tilde{u}_j)\tilde{v}_j = \chi \circ f \circ \phi(u) \quad (4.21)$$

with the function

$$f(U) = \sum_{j=1}^m \Gamma(U, \phi(\tilde{u}_j))(K(\varphi, \varphi))^{-1}\varphi(\tilde{v}_j). \quad (4.22)$$

Furthermore using the reproducing property of G and (4.20) we have $\left\| \sum_{j=1}^m G(u, \tilde{u}_j)\tilde{v}_j \right\|_G^2 = \|f\|_\Gamma^2$. Therefore the closure of the space of operators of the form (4.21) with respect to the RKHS norm induced by G is the space of functions of the form $\chi \circ f \circ \phi$ where f lives in the closure of functions of the form (4.22) with respect to the RKHS

norm induced by Γ . We deduce that $\mathcal{H}_G = \{\chi \circ f \circ \phi \mid f \in \mathcal{H}_\Gamma\}$. The uniqueness of f in the representation $\mathcal{G} = \chi \circ f \circ \phi$ for $f \in \mathcal{H}_G$ follows from $f = \varphi \circ \mathcal{G} \circ \psi$ following the identities $\varphi \circ \chi = I_d$ and $\phi \circ \psi = I_d$. $\square$

Using the above result we can further characterize $\bar{\mathcal{G}}$ and $\bar{f}$ via optimal recovery problems in $\mathcal{H}_G$ and $\mathcal{H}_\Gamma$ respectively. In what follows we will write $\mathbf{u}$ for the N vector whose entries are the u_i , and $\mathcal{G}(\mathbf{u})$ for the N vector whose entries are $\mathcal{G}^\dagger(u_i)$.

Proposition 8. *The operator $\bar{\mathcal{G}}$ is the minimizer of*

$$\begin{cases} \text{Minimize} & \|\mathcal{G}\|_G^2 \\ \text{Over} & \mathcal{G} \in \mathcal{H}_G \text{ such that } \varphi \circ \mathcal{G}(\mathbf{u}) = \varphi \circ \mathcal{G}^\dagger(\mathbf{u}) \end{cases} \quad (4.23)$$

while the map $\bar{f}$ is the minimizer of

$$\begin{cases} \text{Minimize} & \|f\|_\Gamma^2 \\ \text{Over} & f \in \mathcal{H}_\Gamma \text{ such that } f \circ \phi(\mathbf{u}) = \varphi \circ \mathcal{G}^\dagger(\mathbf{u}). \end{cases} \quad (4.24)$$

Proof. By Proposition 7 $\bar{\mathcal{G}}$ is completely identified by $\bar{f}$ and $\|\bar{\mathcal{G}}\|_G = \|\bar{f}\|_\Gamma$. Then solving (4.23) is equivalent to solving (4.24). The statement regarding $\bar{f}$ follows directly from representer formulae for optimal recovery with matrix-valued kernels. $\square$

Regularizing $\bar{\mathcal{G}}$ by operator regression

As is often the case with optimal recovery/kernel regression the estimator for $\bar{f}$ in (4.16) is susceptible to numerical error due to ill-conditioning of the kernel matrix $\Gamma(\mathbf{U}, \mathbf{U})$. To overcome this issue we regularize our estimator by adding a small diagonal perturbation to this matrix. More precisely, let $\gamma > 0$ and write I for the identity matrix. We then define the regularized map

$$\bar{f}_\gamma(U) := \Gamma(U, \mathbf{U})(\Gamma(\mathbf{U}, \mathbf{U}) + \gamma I)^{-1} \mathbf{V}. \quad (4.25)$$

This regularized map gives rise to the regularized approximate operator

$$\bar{\mathcal{G}}_\gamma := \chi \circ \bar{f}_\gamma \circ \phi,$$

which admits the following representer formula:

$$\bar{\mathcal{G}}_\gamma(u) = (K\varphi) K(\varphi, \varphi)^{-1} \Gamma(\phi(u), \mathbf{U})(\Gamma(\mathbf{U}, \mathbf{U}) + \gamma I)^{-1} \mathbf{V}. \quad (4.26)$$

We can further characterize this operator as the solution to an operator regression problem.

Proposition 9. $\bar{\mathcal{G}}_\gamma$ is the solution to

$$\text{Minimize}_{\mathcal{G} \in \mathcal{H}_G} \|\mathcal{G}\|_G^2 + \gamma^{-1} |\varphi \circ \mathcal{G}(\mathbf{u}) - \varphi \circ \mathcal{G}^\dagger(\mathbf{u})|^2. \quad (4.27)$$

Proof. By Proposition 7, $\mathcal{G} = \chi \circ f \circ \phi$ solves (4.27) if and if f solves

$$\text{Minimize}_{f \in \mathcal{H}_\Gamma} \|f\|_\Gamma^2 + \gamma^{-1} |f(\mathbf{U}) - \mathbf{V}|^2. \quad (4.28)$$

It then follows, by standard representer theorems for matrix-valued kernel regression (see Section C.1) that $\bar{f}_\gamma$ is the minimizer of (4.28). $\square$

Interpretation as conditioned operator valued GPs

Our kernel approach to operator learning has a natural GP regression interpretation that is compatible with Bayesian inference and UQ pipelines. We present some facts and observations in this direction.

Write $\xi \sim \mathcal{N}(0, G)$ for the centered operator-valued GP with covariance kernel G ⁵ and $\zeta \sim \mathcal{N}(0, \Gamma)$ for a centered vector valued GP with covariance kernel Γ . Then it is straightforward to show that the law of ξ is equivalent to that of $\chi \circ \zeta \circ \phi$. Let $Z = (Z_1, \dots, Z_N)$ be a random block-vector, independent from ξ , with i.i.d. entries $Z_j \sim \mathcal{N}(0, \gamma I_m)$ for $j = 1, \dots, N$; here $\gamma \geq 0$ and I_m is the $m \times m$ identity matrix.

Then ξ conditioned on $\varphi \circ \xi(\mathbf{u}) = \varphi(\mathbf{v}) + Z$ is an operator-valued GP with mean $\bar{\mathcal{G}}_\gamma$, as in (4.26), and conditional covariance kernel

$$\begin{aligned} G^\perp(u, u')v &= (K\varphi)(K(\varphi, \varphi))^{-1}\Gamma(\phi(u_1), \phi(u_2)) \\ &\quad (\Gamma(\phi(u), \phi(u')) - \Gamma(\phi(u), \mathbf{U})(\Gamma(\mathbf{U}, \mathbf{U}) + \gamma I)^{-1}\Gamma(\mathbf{U}, \phi(u')))(K(\varphi, \varphi))^{-1}\varphi(v) \end{aligned}$$

Furthermore, the law of ξ conditioned on $\varphi \circ \xi(\mathbf{u}) = \varphi(\mathbf{v}) + Z$ is equivalent to that of $\chi \circ \zeta^\perp \circ \phi$ where $\zeta^\perp \sim \mathcal{N}(\bar{f}_\gamma, \Gamma^\perp)$ is the GP ζ conditioned on $\zeta(\mathbf{U}) = \mathbf{V} + Z'$, whose mean is $\bar{f}_\gamma$ as in (4.25) and conditional covariance kernel is

$$\Gamma^\perp(U, U') = \Gamma(U, U') - \Gamma(U, \mathbf{U})(\Gamma(\mathbf{U}, \mathbf{U}) + \gamma I)^{-1}\Gamma(\mathbf{U}, U').$$

We also use the GP approach to derive an alternative regularization of (4.27) in Section C.2.

⁵See Section C.1 for a review of operator valued GPs.

Measurement and mesh invariance

As argued in Zongyi Li, N. Kovachki, et al., 2020a, mesh invariance is a key property for operator learning methods, i.e, the learned operator should be generalizable at test time beyond the specific discretization that was used during training. In our framework, this translates to being able to predict the output of a test input function $\tilde{u}$ given only a linear measurement $\tilde{\phi}(\tilde{u})$, where $\tilde{\phi}$ was unknown at training time. For example $\tilde{\phi}$ could be of the same form as ϕ (say (4.4)) but on a finer or coarser grid. Similarly, we may choose to output with an operator $\tilde{\varphi}$ which is a coarse/fine version of φ . Our proposed framework can easily provide mesh invariance using additional optimal recovery and measurement operators at the input and outputs of the operator $\tilde{\mathcal{G}}$ as depicted in Figure 4.3. In fact, we can not only accommodate modification of the grid but completely different measurement operators at testing time. For example, while ϕ, φ may be of the form (4.4) we may take $\tilde{\phi}$ and $\tilde{\varphi}$ to be integral operators such as Fourier or Radon transforms.

Let us describe our approach to mesh invariance in detail. Given bounded and linear operators $\tilde{\phi} : \mathcal{U} \rightarrow \mathbb{R}^{\tilde{n}}$ and $\tilde{\varphi} : \mathcal{V} \rightarrow \mathbb{R}^{\tilde{m}}$ we can approximate $\tilde{\varphi}(\tilde{\mathcal{G}}^\dagger(\tilde{u}))$ using the map $\tilde{f}$ obtained from (4.16) defined in terms of our training. To achieve mesh invariance we simply need a consistent approach to interpolate/extend the testing measurement operators to those used for training and we achieve this using the optimal recovery map $\tilde{\psi}$ that is defined from $\tilde{\phi}$ analogously to ψ in (4.14). This setup gives rise to a natural approximation of $\mathcal{G}^\dagger$ in terms of the function $h^\dagger : \mathbb{R}^{\tilde{n}} \rightarrow \mathbb{R}^{\tilde{m}}$ depicted in Figure 4.3 which in turn can be approximated with $\bar{h} := \tilde{\varphi} \circ \chi \circ \tilde{f} \circ \phi \circ \tilde{\psi} \equiv \tilde{\varphi} \circ \tilde{\mathcal{G}} \circ \tilde{\psi}$. This expression further gives rise to another approximation to $\mathcal{G}^\dagger$ given by the operator $\tilde{\mathcal{G}} = \tilde{\chi} \circ \bar{h} \circ \tilde{\phi}$.

Remark 10. *Observe that the definition of $\bar{h}$ (and consequently $\tilde{\mathcal{G}}$) is independent of the fact that $\tilde{f}$ is constructed using the kernel approach. Thus, the optimal recovery maps χ and $\tilde{\psi}$ can be used to retrofit any fixed-mesh operator learning algorithm, to become mesh-invariant and able to use arbitrary linear measurements of the function $\tilde{u}$ at test time.*

4.3 Convergence and error analysis

In this section, we present convergence guarantees and rigorous a priori error bounds for our proposed kernel method for operator learning and give a detailed statement and proof of Theorem 11. We assume that $\mathcal{H}_Q$ is a space of continuous functions from $\Omega \subset \mathbb{R}^{d_\Omega}$ and that $\mathcal{H}_K$ is a space of continuous functions from $D \subset \mathbb{R}^{d_D}$.

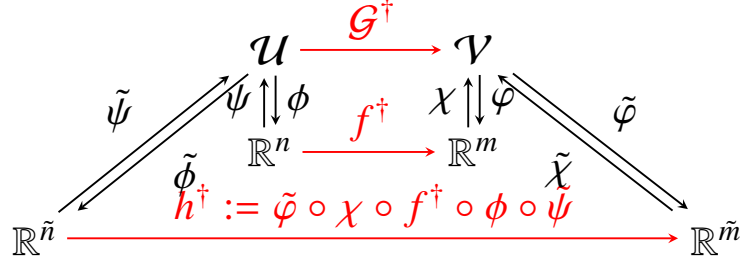


Figure 4.3: Generalization of fig. 4.1 to the mesh invariant setting where the measurement functionals are different at test time.

Abusing notations we write $Q : \Omega \times \Omega \rightarrow \mathbb{R}^{d_\Omega}$ and $K : D \times D \rightarrow \mathbb{R}^{d_D}$ for the kernels induced by the operators Q and K . Let $X = (X_1, \dots, X_n) \subset \Omega$ and $Y = (Y_1, \dots, Y_m) \subset D$ be distinct collections of points and define their fill-distances

$$h_X := \max_{x' \in \Omega} \min_{x \in X} |x - x'|, \quad h_Y := \max_{y' \in D} \min_{y \in Y} |y - y'|.$$

This section focuses on operators ϕ and φ that are linear combinations of pointwise measurements in X and Y . The presented results can be extended by using analogs of the sampling inequalities for other linear measurements; see Houman Owjadi and Scovel, 2019b, Theorem 4.11, Lemma 14.34 for a general framework that allows one to obtain such inequalities.

Let L_Q and L_K be invertible $n \times n$ and $m \times m$ matrices. For $u \in \mathcal{H}_Q$ write $u(X)$ for the n -vector with entries $u(X_i)$ and let $\phi : \mathcal{H}_Q \rightarrow \mathbb{R}^n$ be the bounded linear map defined by

$$\phi(u) = L_Q u(X). \quad (4.29)$$

For $v \in \mathcal{H}_K$ write $v(Y)$ for the m -vector with entries $v(Y_j)$ and let $\varphi : \mathcal{H}_K \rightarrow \mathbb{R}^m$ be the bounded linear map defined by

$$\varphi(v) = L_K v(Y). \quad (4.30)$$

Write $\|\phi\| := \sup_{u \in \mathcal{H}_Q} |\phi(u)| / \|u\|_Q$ and $\|\psi\| := \sup_{U' \in \mathbb{R}^n} \|\psi(U')\|_Q / |U'|$, and similarly $\|\varphi\| := \sup_{v \in \mathcal{H}_K} |\varphi(v)| / \|v\|_K$ and $\|\chi\| := \sup_{V' \in \mathbb{R}^m} \|\chi(V')\|_K / |V'|$. We will also assume the following regularity conditions on the domains Ω, D , the kernels Q, K , and the operator $\mathcal{G}^\dagger$.

Condition 1. Assume that the following conditions hold.

(1.1) Ω and D are compact sets with Lipschitz boundary.

(1.2) There exist indices $s > d_\Omega/2$ and $t > d_D/2$ so that $\mathcal{H}_Q \subset H^s(\Omega)$ and $\mathcal{H}_K \subset H^t(D)$, with inclusions indicating continuous embeddings.

(1.3) $\mathcal{G}^\dagger$ is a (possibly) nonlinear operator from $H^{s'}(\Omega)$ to $\mathcal{H}_K$ with $s' < s$ that satisfies,

$$\|\mathcal{G}^\dagger(u) - \mathcal{G}^\dagger(v)\|_K \leq \omega\left(\|u - v\|_{H^{s'}(\Omega)}\right) \quad (4.31)$$

where $\omega : \mathbb{R} \rightarrow \mathbb{R}_+$ is the modulus of continuity of $\mathcal{G}^\dagger$.

Note that conditions (1.2) and (1.3) imply

$$\|\mathcal{G}^\dagger\|_{B_R(\mathcal{H}_Q) \rightarrow \mathcal{H}_K} := \sup_{u \in B_R(\mathcal{H}_Q)} \|\mathcal{G}^\dagger(u)\|_K < +\infty.$$

Proposition 10. Suppose that Condition 1 holds. Let $0 < t' < t$. Then there exist constants $h_\Omega, h_D, C_\Omega, C_D > 0$ such that if $h_X < h_\Omega$ and $h_Y < h_D$, then

$$\|\mathcal{G}^\dagger(u) - \chi \circ f^\dagger \circ \phi(u)\|_{H^{t'}(D)} \leq C_D \omega\left(C_\Omega h_X^{s-s'} R\right) + C_D h_Y^{t-t'} (\|\mathcal{G}^\dagger(0)\|_K + \omega(C_\Omega R)),$$

for any $u \in B_R(\mathcal{H}_Q)$, where $f^\dagger$ is defined as in (4.5).

Proof. By the definition of $f^\dagger$ and the triangle inequality we have

$$\begin{aligned} \|\mathcal{G}^\dagger(u) - \chi \circ \varphi \circ \mathcal{G}^\dagger \circ \psi^\dagger \circ \phi(u)\|_{H^{t'}(\Gamma)} &\leq \|\mathcal{G}^\dagger(u) - \mathcal{G}^\dagger \circ \psi \circ \phi(u)\|_{H^{t'}(\Gamma)} \\ &\quad + \|\mathcal{G}^\dagger \circ \psi \circ \phi(u) - \chi \circ \varphi \circ \mathcal{G}^\dagger \circ \psi \circ \phi(u)\|_{H^{t'}(\Gamma)} \\ &=: T_1 + T_2. \end{aligned}$$

Let us first bound T_1 : By conditions (1.2) and (1.3), we have

$$T_1 \leq C_D \|\mathcal{G}^\dagger(u) - \mathcal{G}^\dagger \circ \psi \circ \phi(u)\|_K \leq C_D \omega\left(\|u - \psi \circ \phi(u)\|_{H^{s'}(\Omega)}\right).$$

At the same time, since $(u - \psi \circ \phi(u))(X) = 0$, condition (1.1) and the sampling inequality for interpolation in Sobolev spaces Arcangeli, Lopez de Silanes, and Torrens, 2007b, Thm. 4.1, and condition (1.2) imply that there exists a constant $h_\Omega > 0$ so that if $h_X < h_\Omega$ then

$$\|u - \psi \circ \phi(u)\|_{H^{s'}(\Omega)} \leq C'_\Omega h_X^{s-s'} \|u - \psi \circ \phi(u)\|_{H^s(\Omega)} \leq C_\Omega h_X^{s-s'} \|u - \psi \circ \phi(u)\|_Q, \quad (4.32)$$

where $C'_\Omega, C_\Omega > 0$ are constants that are independent of u . Using $\|u - \psi \circ \phi(u)\|_Q \leq \|u\|_Q$ Houman Owhadi and Scovel, 2019b, Thm. 12.3 we deduce the desired bound

$$T_1 \leq C_D \omega\left(C_\Omega h_\Omega^{s-s'} \|u\|_Q\right). \quad (4.33)$$

Let us now bound T_2 : Once again, by the continuous embedding of condition (1.2) and the sampling inequality for interpolation in Sobolev spaces, we have that there exists $h_D > 0$ so that if $h_Y < h_D$, then for any $v \in H^t(D)$ it holds that

$$\|v - \chi \circ \varphi(v)\|_{H^{t'}(D)} \leq C'_D h_Y^{t-t'} \|v - \chi \circ \varphi(v)\|_{H^t(D)} \leq C_D h_Y^{t-t'} \|v - \chi \circ \varphi(v)\|_K \leq C_D h_Y^{t-t'} \|v\|_K.$$

Taking $v \equiv \mathcal{G}^\dagger \circ \psi \circ \phi(u)$, we deduce that

$$\begin{aligned} T_2 &\leq C_D h_Y^{t-t'} \|\mathcal{G}^\dagger \circ \psi \circ \phi(u)\|_K, \\ &\leq C_D h_Y^{t-t'} (\|\mathcal{G}^\dagger(0)\|_K + \omega(\|\psi \circ \phi(u)\|_{H^{s'}(\Omega)})). \end{aligned}$$

Using $\|\psi \circ \phi(u)\|_{H^{s'}(\Omega)} \leq C_\Omega \|\psi \circ \phi(u)\|_Q \leq C_\Omega \|u\|_Q$ concludes the proof. $\square$

While Proposition 10 gives an error bound for the distance between the maps $\mathcal{G}^\dagger$ and $\varphi \circ f^\dagger \circ \phi$, we can never compute this map when $N < \infty$ and so we have to approximate this map as well. Given the kernel Γ , our optimal recovery approximant for the map $f^\dagger$ is $\bar{f}$ as in (4.16), which we recall is the minimizer of (4.24).

To proceed, we need to consider another intermediary problem that defines an approximation $\hat{f}$ to the map $f^\dagger$:

$$\hat{f} := \begin{cases} \text{Minimize} & \|f\|_\Gamma^2 \\ \text{Over} & f \in \mathcal{H}_\Gamma \text{ such that } f \circ \phi(\mathbf{u}) = f^\dagger \circ \phi(\mathbf{u}). \end{cases} \quad (4.34)$$

We emphasize that the difference between the problems (4.24) and (4.34) is simply in the training data that is injected in the equality constraints, and this difference is quite subtle.

In practical applications, observations may be taken from $\mathcal{G}^\dagger(u_i)$, which is different from $f^\dagger \circ \phi(u_i) \equiv \varphi \circ \mathcal{G}^\dagger \circ \psi \circ \phi(u_i)$. To make our analysis simple, henceforth we assume the following condition on our input data.

Condition 2. *The input data points u_i satisfy*

$$u_i = \psi \circ \phi(u_i) \text{ for } i = 1, \dots, N.$$

We observe that this condition implies $\mathcal{G}^\dagger(u_i) = f^\dagger \circ \phi(u_i)$ and $\bar{f} = \hat{f}$. Removing this assumption requires bounding some norm of the error $f^\dagger - \bar{f}$, and we postpone that analysis to a sequel paper as this step can become very technical.

The next step in our convergence analysis is then to control the error between the maps $\hat{f}$ and $f^\dagger$ which we will achieve using similar arguments as in the proof of

Proposition 10. For our analysis, we take Γ to be a diagonal, matrix-valued kernel, of the form (4.8) which we recall for reference

$$\Gamma(U, U') = S(U, U')I, \quad (4.35)$$

where I is the $m \times m$ identity matrix and $S : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is a real valued kernel.

Proposition 11. *Suppose that Condition 2 holds. Let $\Upsilon \subset \mathbb{R}^n$ be a compact set with Lipschitz boundary and consider $U = (U_1, \dots, U_N) \in \Upsilon$ with fill distance*

$$h_\Upsilon := \max_{U' \in \Upsilon} \min_{1 \leq i \leq N} |U_i - U'|.$$

Let Γ be of the form (4.35), with S restricted to the set Υ , and suppose $\mathcal{H}_S \subset H^r(\Upsilon)$ for $r > n/2$ and that $f_j^\dagger \in \mathcal{H}_S$ for $j = 1, \dots, m$. Then there exist constants $h'_\Upsilon, C_\Upsilon > 0$ so that whenever $h_\Upsilon < h'_\Upsilon$ then for any $r' < r$ it holds that

$$\|f_j^\dagger - \hat{f}_j\|_{H^{r'}(\Upsilon)} \leq C_\Upsilon h_\Upsilon^{r-r'} \|f_j^\dagger\|_S.$$

Proof. The proof is a direct consequence of the fact that the components of $\hat{f}$ are given by the optimal recovery problems (4.34) and the sampling inequality for interpolation in Sobolev spaces Arcangeli, Lopez de Silanes, and Torrens, 2007b, Thm. 4.1 following the same arguments used in the proof of Theorem 10. $\square$

We can now combine the above results to obtain the following theorem.

Theorem 8. *Suppose that Conditions 1 and 2 hold in addition to those of Proposition 11 with a set of inputs $(u_i)_{i=1}^N \subset B_R(\mathcal{H}_Q)$, the set $\Upsilon = \phi(B_R(\mathcal{H}_Q))$, and index $n/2 < r' < r$. Then for any $u \in B_R(\mathcal{H}_Q)$, it holds that*

$$\begin{aligned} \|\mathcal{G}^\dagger(u) - \chi \circ \bar{f} \circ \phi(u)\|_{H^{r'}(D)} &\leq C_D \omega\left(C_\Omega h_X^{s-s'} R\right) + C_D h_Y^{t-t'} (\|\mathcal{G}^\dagger(0)\|_K + \omega(C_\Omega R)) \\ &\quad + \sqrt{m} C_D C_\Upsilon \|\chi\| h_Y^{(r-r')} \max_{1 \leq j \leq m} \|f_j^\dagger\|_S \end{aligned} \quad (4.36)$$

Proof. An application of the triangle inequality yields

$$\begin{aligned} \|\mathcal{G}^\dagger(u) - \chi \circ \bar{f} \circ \phi(u)\|_{H^{r'}(D)} &\leq \|\mathcal{G}^\dagger(u) - \chi \circ f^\dagger \circ \phi(u)\|_{H^{r'}(D)} \\ &\quad + \|\chi \circ f^\dagger \circ \phi(u) - \chi \circ \hat{f} \circ \phi(u)\|_{H^{r'}(D)} \\ &\quad + \|\chi \circ \hat{f} \circ \phi(u) - \chi \circ \bar{f} \circ \phi(u)\|_{H^{r'}(D)} =: I_1 + I_2 + I_3. \end{aligned}$$

We can bound I_1 immediately using Proposition 10. Furthermore, by Condition 2 we have that $I_3 = 0$. So it remains for us to bound I_2 : by the continuous embedding of $\mathcal{H}_K$ into $H'(\mathcal{D})$ we can write

$$\begin{aligned} I_2 &\leq C_D \|\chi \circ f^\dagger \circ \phi(u) - \chi \circ \hat{f} \circ \phi(u)\|_K \leq C_D \|\chi\| \|f^\dagger \circ \phi(u) - \hat{f} \circ \phi(u)\| \\ &\leq C_D \|\chi\| \sqrt{\sum_{j=1}^m \|f_j^\dagger - \hat{f}_j\|_{H^{r'}(\mathcal{Y})}^2}, \end{aligned}$$

where the last line follows from the Sobolev embedding theorem and the assumption that $r' > n/2$. Then an application of Proposition 11 yields

$$I_2 \leq \sqrt{m} C_D C_Y \|\chi\| h_Y^{(r-r')} \max_{1 \leq j \leq m} \|f_j^\dagger\|_S.$$

□

Convergence theorem

Our next step will be to consider the limits $N, n, m \rightarrow \infty$ and show the convergence of $\bar{\mathcal{G}}$ to $\mathcal{G}^\dagger$. To obtain this result we first need to make assumptions on the regularity of the true operator $\mathcal{G}^\dagger$.

For $k \geq 1$ write $D^k \mathcal{G}^\dagger$ for the functional derivative of $\mathcal{G}^\dagger$ of order k . Recall that for $u \in \mathcal{H}_Q$, $D^k \mathcal{G}^\dagger(u)$ is a multilinear operator mapping $\otimes_{i=1}^k \mathcal{H}_Q$ to $\mathcal{H}_K$. For $w_1, \dots, w_k \in \mathcal{H}_Q$ write $[D^k \mathcal{G}^\dagger(u), \otimes_{i=1}^k w_i]$ for the (multilinear) action of $D^k \mathcal{G}^\dagger(u)$ on $\otimes_{i=1}^k w_i$ and write $\|D^k \mathcal{G}^\dagger(u)\|$ for the smallest constant such that for $w_1, \dots, w_k \in \mathcal{H}_Q$,

$$\|[D^k \mathcal{G}^\dagger(u), \otimes_{i=1}^k w_i]\|_{\mathcal{H}_K} \leq \|D^k \mathcal{G}^\dagger(u)\| \prod_{i=1}^k \|w_i\|_{\mathcal{H}_Q}. \quad (4.37)$$

Similarly, for $k \geq 1$ write $D^k f^\dagger$ for the derivation tensor of $f^\dagger$ of order k (the gradient for $k = 1$ and the Hessian for $k = 2$, etc). Recall that for $U \in \mathbb{R}^n$, $D^k f^\dagger(U)$ is a multilinear operator mapping $\otimes_{i=1}^k \mathbb{R}^n$ to $\mathbb{R}^m$. For $W_1, \dots, W_k \in \mathbb{R}^n$ write $[D^k f^\dagger(U), \otimes_{i=1}^k W_i]$ for the (multilinear) action of $D^k f^\dagger(U)$ on $\otimes_{i=1}^k W_i$ and write $\|D^k f^\dagger(U)\|$ for the smallest constant such that for $W_1, \dots, W_k \in \mathbb{R}^n$,

$$|[D^k f^\dagger(U), \otimes_{i=1}^k W_i]| \leq \|D^k f^\dagger(U)\| \prod_{i=1}^k |W_i|, \quad (4.38)$$

where $|\cdot|$ is the Euclidean norm.

Lemma 4. *It holds true that $\|D^k f^\dagger(U)\| \leq \|\varphi\| \|\psi\|^k \|D^k \mathcal{G}^\dagger \circ \psi(U)\|$, $\forall U \in \mathbb{R}^n$.*

Proof. The chain rule and the linearity of φ and ψ imply that

$$[D^k f^\dagger(U), \otimes_{i=1}^k W_i] = \varphi[D^k \mathcal{G}^\dagger \circ \psi(U), \otimes_{i=1}^k \psi(W_i)].$$

We then conclude the proof by writing

$$\begin{aligned} |[D^k f^\dagger(U), \otimes_{i=1}^k W_i]| &\leq \|\varphi\| \|D^k \mathcal{G}^\dagger \circ \psi(U)\| \prod_{i=1}^k \|\psi(W_i)\|_{\mathcal{H}_Q} \\ &\leq \|\varphi\| \|\psi\|^k \|D^k \mathcal{G}^\dagger \circ \psi(U)\| \prod_{i=1}^k |W_i|. \end{aligned}$$

□

Let us now consider an infinite and dense sequence of points $X_1, X_2, X_3, \dots$ of Ω , such that the closure of $\cup_{i=1}^\infty \{X_i\}$ is the closure of Ω . Write X^n for the n -vector formed by the first n points, i.e.,

$$X^n := (X_1, \dots, X_n) \quad (4.39)$$

and let L_Q^n be an arbitrary invertible $n \times n$ matrix. Further let $\phi^n : \mathcal{H}_Q \rightarrow \mathbb{R}^n$ be defined by

$$\phi^n(u) = L_Q^n u(X^n). \quad (4.40)$$

Write ψ^n for the corresponding optimal recovery ψ -map. Similarly, we assume that we are given an infinite and dense sequence of points $Y_1, Y_2, Y_3, \dots$ of D , such that the closure of $\cup_{i=1}^\infty \{Y_i\}$ is the closure of D . Write Y^m for the m -vector formed by the first m points, i.e.,

$$Y^m := (Y_1, \dots, Y_m). \quad (4.41)$$

Let L_K^m be an arbitrary invertible $m \times m$ matrix and let $\varphi^m : \mathcal{H}_K \rightarrow \mathbb{R}^m$ be defined by

$$\varphi^m(v) = L_K^m v(Y^m). \quad (4.42)$$

Write χ^m for the corresponding optimal recovery χ -map. We also assume that we are given a sequence of diagonal matrix-valued kernels $\Gamma^{m,n} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathcal{L}(\mathbb{R}^m)$ with scalar-valued kernels $S^n : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ as diagonal entries. Write $\bar{f}_N^{m,n}$ for the corresponding minimizer of (4.24) (also identified by the formula (4.16)) for the above setup.

Theorem 9. Let m, n be the dimensionality of the input and output observations $\phi : \mathcal{U} \rightarrow \mathbb{R}^n$ and $\varphi : \mathcal{V} \rightarrow \mathbb{R}^m$. Suppose that the closure of $\lim_{n \uparrow \infty} \cup_{i=1}^n \{X_i\}$ is equal to the closure of Ω and that the closure of $\lim_{m \uparrow \infty} \cup_{i=1}^m \{Y_i\}$ is equal to the closure of D . Suppose Condition 1 is satisfied and that

$$\sup_{u \in B_R(\mathcal{H}_Q)} \| [D^k \mathcal{G}^\dagger(u)] \| < \infty \text{ for all } k \geq 1, \quad (4.43)$$

for an arbitrary $R > 0$. Assume that for any $n \geq 1$ and any compact set Υ of $\mathbb{R}^n$, the RKHS of S^n restricted to Υ (which we write $\mathcal{H}_{S^n}(\Upsilon)$) is contained in $H^r(\Upsilon)$ for some $r > n/2$ and contains $H^{r'}(\Upsilon)$ for some $r' > 0$ that may depend on n . Let $(u_i)_{i=1}^N$ be a sequence of inputs in $B_R(\mathcal{H}_Q)$. Assume that there exists an integer n_0 such that for $n \geq n_0$, the data points $(u_i)_{i=1}^N$ satisfy Condition 2, i.e., they satisfy $u_i = \psi^n \circ \phi^n(u_i)$ for all $i \geq 1$. Further assume that the $(\phi^n(u_i))_{1 \leq i \leq N}$ are space filling in the sense that for any $n \geq n_0$ we have

$$\lim_{N \rightarrow \infty} \sup_{u \in B_R(\mathcal{H}_Q)} \min_{1 \leq i \leq N} |u_i(X^n) - u(X^n)| = 0. \quad (4.44)$$

Then for any $t' \in (0, t)$, it holds that

$$\lim_{n, m \rightarrow \infty} \lim_{N \rightarrow \infty} \sup_{u \in B_R(\mathcal{H}_Q)} \|\mathcal{G}^\dagger(u) - \chi^m \circ \tilde{f}_N^{m, n} \circ \phi^n(u)\|_{H^{t'}(D)} = 0. \quad (4.45)$$

Proof. Following Houman Owhadi and Scovel, 2019b, Chap. 12.1 define the projection $P_n^{\mathcal{U}} = \psi^n \circ \phi^n$ onto the range of ψ^n . Since the points X_i and Y_j are dense in Ω and D we have $h_{X^n} \downarrow 0$ as $n \rightarrow \infty$ and $h_{Y^m} \downarrow 0$ as $m \rightarrow \infty$. Given n , take $\Upsilon = \phi^n(B_R(\mathcal{H}_Q))$. Then Lemma 4 and (4.43) imply that $\tilde{f}_j^{m, n} \in H^{r'}(\Upsilon)$ for all $r' \geq 0$. Therefore $\tilde{f}_j^{m, n} \in \mathcal{H}_{S^n}(\Upsilon)$. Now (4.44) implies that for any n , the fill distance, in $\phi^n(B_R(\mathcal{H}_Q))$, between the points $(\phi^n(u_i))_{1 \leq i \leq N}$ goes to zero as $N \rightarrow \infty$. Since the conditions of Proposition 11 are satisfied, we conclude by taking the limit $N \rightarrow \infty$ in (4.36) before taking the limit $m, n \rightarrow \infty$. $\square$

The effect of the L_Q and L_K preconditioners

We conclude this section and our discussion of convergence results, by highlighting the importance of the choice of the matrices L_Q^n and L_K^m in (4.40) and (4.42). It is clear from the bounds (4.36) and (4.38) that our error estimates depend on the norms of the linear operators φ^m, ψ^n and χ^m . To ensure that those norms do not blow up as $n, m \rightarrow \infty$ we can select the matrices L_Q^n and L_K^m to be the Cholesky factors of the precision matrices obtained from pointwise measurements of the kernels Q and

K , i.e.,

$$L_Q^n (L_Q^n)^T = Q(X^n, X^n)^{-1} \quad \text{and} \quad L_K^m (L_K^m)^T = K(Y^m, Y^m)^{-1}. \quad (4.46)$$

We now obtain the following proposition.

Proposition 12. *If ϕ^n is as in (4.40) and L_Q^n as in (4.46), then $\|\phi^n\| = 1$ and $\|\psi^n\| = 1$. If φ^m is as in (4.30) and L_K^m as in (4.46), then $\|\varphi^m\| = 1$ and $\|\chi^m\| = 1$.*

Proof. For $u \in \mathcal{H}_Q$, $|\phi^n(u)|^2 = u(X^n)^T Q(X^n, X^n)^{-1} u(X^n) = \|\psi^n \circ \phi^n(u)\|_Q^2$. Since $\psi^n \circ \phi^n$ is a projection Houman Owhadi and Scovel, 2019b, Chap. 12.1 we deduce that $\|\phi^n\| = 1$. Using $\psi^n(U') = Q(\cdot, X^n) L_Q^n U'$ leads to $\|\psi^n(U')\|_Q^2 = |U'|^2$ and $\|\psi^n\| = 1$. The proof of $\|\varphi^n\| = 1$ and $\|\chi^n\| = 1$ is similar. $\square$

We note that although useful for obtaining tighter approximation errors, this particular choice for the matrices L_Q^n and L_K^m is not required for convergence if one first takes the limit $N \rightarrow \infty$ as in Theorem 9, which does not put any requirements on the matrices L_Q^n and L_K^m beyond invertibility.

4.4 Numerics

In this section, we present numerical experiments and benchmarks that compare a straightforward implementation of our kernel operator learning framework to state-of-the-art NN-based techniques. We discuss some implementation details of our method in Section 4.4 followed by the setup of experiments and test problems in Sections 4.4 and 5.3. A detailed discussion of our findings is presented in Section 7.8.

Implementation considerations

Below we summarize some of the key details in the implementation of our kernel approach for operator learning for benchmark examples. Our code to reproduce the experiments can be found in a public repository⁶.

Choice of the kernel Γ

Following our theoretical discussions in Sections 4.2 and 4.3, we primarily take Γ to be a diagonal kernel of the form (4.35). This implies that our estimation of $\tilde{f}$ can be split into independent problems for each of its components $\tilde{f}_j$ in the RKHS of the scalar kernel S . In our experiments, we investigate different choices of S belonging

⁶<https://github.com/MatthieuDarcy/KernelsOperatorLearning/>

to the families of the linear kernel, rational quadratic, and Matérn; see Section C.3 for detailed expressions of these kernels. The rational quadratic kernel has two parameters: the lengthscale l and the exponent α . We tuned these parameters using standard cross validation or log marginal likelihood maximization over the training data (see Rasmussen and Williams, 2006, p.112 for a detailed description). The Matérn kernel is parameterized by two positive parameters: a smoothness parameter ν and the length scale l . The smoothness parameter ν controls the regularity of the RKHS and we considered $\nu \in \{\frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \frac{7}{2}, \infty\}$. In practice we found that $\nu = \frac{5}{2}$ almost always had the best performance. For a fixed choice of ν we tuned the length scale l similarly to the rational quadratic kernel. We implemented the kernel regressions of the $\tilde{f}_j$ and parameter tuning algorithms in scikit-learn for low-dimensional examples and manually in JAX for high-dimensional examples.

Preconditioning and dimensionality reduction

Following (4.29) and (4.30) and the discussion in Section 4.3, we consider two preconditioning strategies for our pointwise measurements, i.e., choices of the matrices L_Q and L_K : (1) we consider the Cholesky factors of the underlying covariance matrices as in (4.46); (2) we use PCA projection matrices of the input and output functions computed from the training data. We truncated the PCA expansions to preserve (0.90, 0.95, 0.99) of the variance. The use of PCA in learning mappings between infinite dimensional spaces was proposed in Krischer et al., 1993 and recently revisited in Bhattacharya, Hosseini, Nikola B. Kovachki, et al., 2021b; J. Hesthaven and Ubbiali, 2018.

Experimental setup

We compare the test performance of our method with different choices of the kernel S of increasing complexity using the examples in De Hoop et al., 2022 and L. Lu, X. Meng, et al., 2022 and their reported test relative L^2 loss (see eq. (4.48) below). We use the data provided by these papers for the training set and the test set⁷. Both articles provide performance comparisons between different variants of Neural Operators (most notably FNO and DeepONet) on a variety of PDE operator learning tasks, where the data is sampled independently from a distribution $(Id, \mathcal{G}^\dagger)^\# \mu$ supported on $\mathcal{U} \times \mathcal{V}$, where μ is a specified (input) distribution on $\mathcal{U}$. The example problems are outlined in detail in Section 5.3; a summary of the

⁷See <https://github.com/Zhengyu-Huang/Operator-Learning> and <https://github.com/lu-group/deeponet-fno>, respectively, for the data

specific PDEs, problem type, and distribution μ for each test is given in table 4.2. In some instances the train-test split of the data was not clear from the available online repositories in which case we re-sampled them from the assumed distribution μ . The datasets from L. Lu, X. Meng, et al., 2022 contain 1,000 training data-points per problem (which we will refer to as the “low-data regime”), whereas the datasets from De Hoop et al., 2022 contain 20,000 training data-points (which we will refer to as the “high-data” regime). We make this distinction because the complexity of kernel methods, unlike that of neural networks, may depend on the number of data-points.

Following the suggestion of De Hoop et al., 2022 we not only compare test errors and training complexity but also the complexity of operator learning at the inference/evaluation stage in Section 4.4. For the examples in De Hoop et al., 2022, we investigate the accuracy-complexity trade-off of our method against the reported values of that article.

Equation	Input	Output	Input Distribution μ
Burger’s	Initial condition	Solution at time T	Gaussian field (GF)
Darcy problem	Coefficient	Solution	Binary function of GF
Advection I	Initial condition	Solution at time T	Random square waves
Advection II	Initial condition	Solution at time T	Binary function of GF
Helmholtz	Coefficient	Solution	Function of Gaussian field
Structural mechanics	Initial force	Stress field	Gaussian field
Navier Stokes	Forcing term	Solution at time T	Gaussian field

Table 4.2: Summary of datasets used for benchmarking. The first three examples were considered in L. Lu, X. Meng, et al., 2022, and the last four were taken from De Hoop et al., 2022.

Measures of accuracy

As our first performance metric we measured the accuracy of models by a relative loss on the output space $\mathcal{V}$:

$$\mathcal{R}(\mathcal{G}) = \mathbb{E}_{u \sim \mu} \left[\frac{\|\mathcal{G}^\dagger(u) - \mathcal{G}(u)\|_{\mathcal{V}}}{\|\mathcal{G}^\dagger(u)\|_{\mathcal{V}}} \right], \quad (4.47)$$

where $\mathcal{G}^\dagger$ is true operator and $\mathcal{G}$ is a candidate operator. Following previous works, we often took $\|u\|_{\mathcal{V}} = \|u\|_{L^2} := (\int u(x)^2 dx)^{\frac{1}{2}}$, which in turn is discretized using the trapezoidal rule. In practice, we do not have the access to the underlying probability

measure μ and we compute the empirical loss on a withheld test set:

$$\mathcal{R}_N(\mathcal{G}) = \frac{1}{N} \sum_{n=1}^N \left[\frac{\|\mathcal{G}^\dagger(u^n) - \mathcal{G}(u^n)\|_{\mathcal{V}}}{\|\mathcal{G}^\dagger(u^n)\|_{\mathcal{V}}} \right], \quad u^i \sim \mu. \quad (4.48)$$

Measures of complexity

For our second performance metric we considered the complexity of operator learning algorithms at the inference stage (i.e., evaluating the learned operator). Complexity at inference time is the main metric used in De Hoop et al., 2022 to compare numerical methods for operator learning. The motivation is that training of the methods can be performed in an offline fashion, and therefore the cost per test example dominates in the limit of many test queries. In particular, they compare the online evaluation costs of the neural networks by computing the requisite floating point operations (FLOPs) per test example. We adopt this metric as well for the methods not based on neural networks that we develop in this work, and we compare, when available, the cost-accuracy tradeoff with the numbers reported in De Hoop et al., 2022. We computed the FLOPs with the same assumptions as in the original work: a matrix-vector product where the input vector is in $\mathbb{R}^n$ and the output vector is in $\mathbb{R}^m$ amounts to $m(2 - 1)$ flops, and non-linear functions with n -dimensional inputs (activation functions for neural networks, and kernel computations for kernel methods) are assumed to have cost $\mathcal{O}(n)$.

Remark 11 (Training complexity). *While the inference complexity of a model eventually dominates the cost of training during applications, the training cost cannot be ignored since the allocated computational resources during this stage may still be limited and the resulting errors will have a profound impact on the quality and performance of the learned operators. Therefore numerical methods in which the offline data assimilation step is cheaper, faster, and more robust will always be preferred. Computing the exact number of FLOPs at training time is difficult to estimate for NN methods, as it depends on the optimization algorithms used, the hyperparameters, and the optimization over such hyperparameters, among many other factors. Therefore in this work we limit the training complexity evaluation to the qualitative observation that kernel methods provided in this work are significantly simpler at training time, as they have no NN weights, they do not require the use of stochastic gradient descent, and have few or no hyperparameters which can be tuned using standard methods such as grid search or gradient descent in a low-dimensional space.*

Test problems and qualitative results

Below we outline the setup of each of our benchmark problems. In all cases, $\mathcal{U}$ and $\mathcal{V}$ are spaces of real-valued functions with input domains $\Omega, D \subset \mathbb{R}^k$ for $k = 1$ or 2 . Whenever $\Omega = D$, we simply write $\mathcal{D}$ for both.

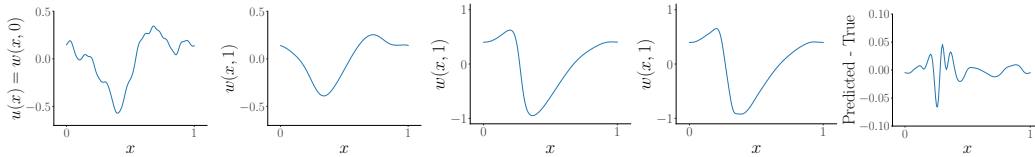
Burger's equation

Consider the one-dimensional Burger's equation:

$$\begin{aligned} \frac{\partial w}{\partial t} + w \frac{\partial w}{\partial x} &= \nu \frac{\partial^2 w}{\partial x^2}, \quad (x, t) \in (0, 1) \times (0, 1], \\ w(x, 0) &= u(x), \quad x \in (0, 1) \end{aligned} \quad (4.49)$$

with $\mathcal{D} = (0, 1)$, and periodic boundary conditions. The viscosity parameter ν is set to 0.1. We learn the operator mapping the initial condition u to $v = w(\cdot, 1)$, the solution at time $t = 1$, i.e., $\mathcal{G}^\dagger : w(\cdot, 0) \mapsto w(\cdot, 1)$.

The training data is generated by sampling the initial condition u from a GP with a Riesz kernel, denoted by $\mu = \mathcal{GP}(0, 625(-\Delta + 25I)^{-2})$. As in L. Lu, X. Meng, et al., 2022, we used a spatial resolution with 128 grid points to represent the input and output functions, and used 1,000 instances for training and 200 instances for testing. Figure 4.4 shows an example of training input and output pairs as well as a test example along with its pointwise error.



(a) Training input (b) Training output (c) True test (d) Predicted test (e) Pointwise error

Figure 4.4: Example of training data and test prediction and pointwise errors for the Burger's equation (4.49).

Darcy flow

Consider the two-dimensional Darcy flow problem (4.3). Recall that in this example, we are interested in learning the mapping from the permeability field u to the solution v and the source term w is assumed to be fixed, hence $\mathcal{D} \equiv \Omega = (0, 1)^2$ and $\mathcal{G}^\dagger : u \mapsto v$. The coefficient u is sampled by setting $u \sim \log \circ h_\# \mu$ where $\mu = \mathcal{GP}(0, (-\Delta + 9I)^{-2})$ is a GP and h is binary function mapping positive inputs to 12 and negative inputs to 3. The resulting permeability/diffusion coefficient e^u is

therefore piecewise constant. As in L. Lu, X. Meng, et al., 2022, we use a discretized grid of resolution 29×29 , with the data generated by the MATLAB PDE Toolbox. We use 1,000 points for training and 200 points for testing. Figure 4.2 shows an example of training input and output of the map $\mathcal{G}^\dagger$, and an example of predictions along with pointwise error at the test stage.

Advection equations (I and II)

Consider the one-dimensional advection equation:

$$\begin{aligned} \frac{\partial w}{\partial t} + \frac{\partial w}{\partial x} &= 0 & x \in (0, 1), t \in (0, 1] \\ w(x, 0) &= u(x) & x \in (0, 1) \end{aligned} \quad (4.50)$$

with $\mathcal{D} = (0, 1)$ and periodic boundary conditions. Similar to the example for Burgers' equation, we learn the mapping from the initial condition u to $v = w(\cdot, 0.5)$, the solution at $t = 0.5$, i.e., $\mathcal{G}^\dagger : w(\cdot, 0) \mapsto w(\cdot, 0.5)$.

This problem was considered in L. Lu, X. Meng, et al., 2022; De Hoop et al., 2022 with different distributions μ for the initial condition. We will show in the following section how these different distributions lead to different performances. In L. Lu, X. Meng, et al., 2022, henceforth referred to as Advection I, the initial condition is a square wave centered at $x = c$ of width b and height h :

$$u(x) = h \mathbf{1}_{\{c-\frac{b}{2}, c+\frac{b}{2}\}}, \quad (4.51)$$

where the parameters $(c, b, h) \sim \mathcal{U}([0.3, 0.7] \times [0.3, 0.6] \times [1, 2])$. In De Hoop et al., 2022, henceforth referred to as Advection II, the initial condition is

$$u = -1 + 2 \mathbf{1}_{\{\tilde{u}_0 \geq 0\}}, \quad (4.52)$$

where $\tilde{u}_0 \sim \mathcal{GP}(0, (-\Delta + 3^2 I)^{-2})$.

For Advection I, the spatial grid was of resolution 40, and we used 1000 instances for training and 200 instances for testing. For Advection II, the resolution was of 200 and we used 20,000 training and test instances, following De Hoop et al., 2022.

Figures 4.5 and 4.6 show an example of training input and output for Advection the I and II problems, respectively. Observe that the functional samples from the distribution in Advection I will have exactly two discontinuities almost surely, but the samples for Advection II can have many more jumps. We observe that prediction is challenging around discontinuities, and hence Advection II is a significantly harder

problem (across all benchmarked methods) than Advection I. Figures 4.5 and 4.6 also show an instance of a test sample, along with a prediction and the pointwise errors.

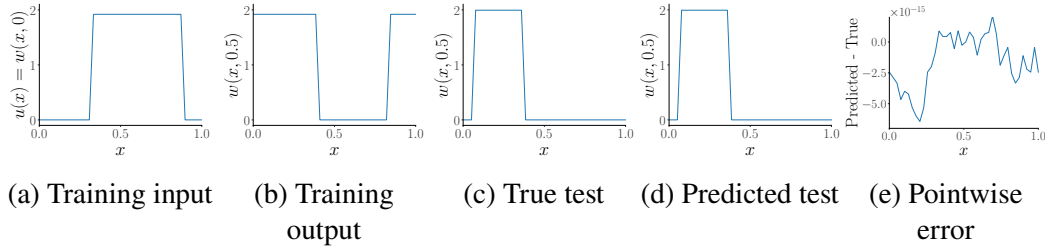


Figure 4.5: Example of training data and test prediction and pointwise errors for the Advection problem (4.50)-I.

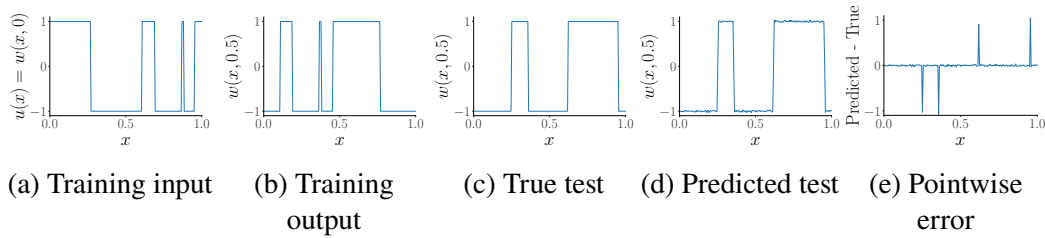


Figure 4.6: Example of training data and test prediction and pointwise errors for the Advection problem (4.50)-II.

Helmholtz's equation

For a given frequency ω and wavespeed field $u : \mathcal{D} \rightarrow \mathbb{R}$, with $\mathcal{D} = (0, 1)^2$, the excitation field $v : \mathcal{D} \rightarrow \mathbb{R}$ solves

$$\left(-\Delta - \frac{\omega^2}{u^2(x)}\right)v = 0, \quad x \in (0, 1)^2$$

$$\frac{\partial v}{\partial n} = 0, \quad x \in \{0, 1\} \times [0, 1] \cup [0, 1] \times \{0\} \quad \text{and} \quad \frac{\partial v}{\partial n} = v_N, \quad x \in [0, 1] \times \{1\}. \quad (4.53)$$

In the results that follow, we take $\omega = 10^3$, $v_N = \mathbf{1}_{\{0.35 \leq x \leq 0.65\}}$, and we aim to learn the map $\mathcal{G} : u \mapsto v$, i.e., the mapping from the wavespeed field to the excitation field. The distribution μ is specified as the law of $u(x) = 20 + \tanh(\tilde{u}(x))$, where $\tilde{u}$ is drawn from the GP, $\mathcal{GP}(0, (-\Delta + 3^2 I)^{-2})$. The training and test data were generated by solving (4.53) with a Finite Element Method on a discretization of size 100×100 of the unit square. Figure 4.7 shows an example of training input and output, a test prediction, and pointwise errors.

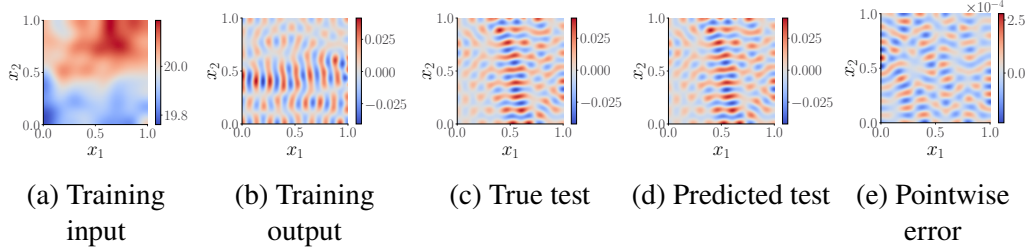


Figure 4.7: Example of training data and test prediction and pointwise errors for the Helmholtz problem (4.53).

Structural mechanics

We let $\Omega = [0, 1]^2$, $D = [0, 1]^2$, the equation that governs the displacement vector w in an elastic solid undergoing infinitesimal deformations is

$$\nabla \cdot \sigma = 0 \quad \text{in } (0, 1)^2, \quad w = \bar{w}, \quad \text{on } \Gamma_w, \quad \nabla \cdot n = u \quad \text{on } \Gamma_u, \quad (4.54)$$

where the boundary ∂D is split in $[0, 1] \times 1 = \Gamma_t$ (the part of the boundary subject to stress) and its complement Γ_u .

The goal is to learn the operator that maps the one-dimensional load u on Γ_u to the two-dimensional von Mises stress field v on Ω , i.e., $\mathcal{G} : u \mapsto v$. Here the distribution μ is $\mathcal{GP}(100, 400^2(-\Delta + 3^2 I)^{-1})$, with Δ being the Laplacian subject to homogeneous Neumann boundary conditions on the space of zero-mean functions. The function v was obtained by a finite element code; see De Hoop et al., 2022 for implementation details and the constitutive model used. Figure 4.8 shows an example of training input and outputs, a test prediction, and pointwise errors.

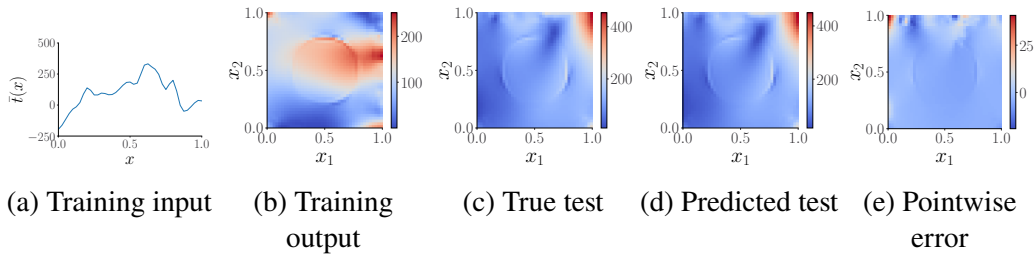


Figure 4.8: Example of training data and test prediction and pointwise errors for the Structural Mechanics problem (4.54).

Navier-Stokes equations

Consider the vorticity-stream (ω, ψ) formulation of the incompressible Navier-Stokes equations:

$$\frac{\partial \omega}{\partial t} + (c \cdot \nabla) \omega - \nu \Delta \omega = u, \quad \omega = -\Delta \psi, \quad \int_D \psi = 0, \quad c = \left(\frac{\partial \psi}{\partial x_2}, -\frac{\partial \psi}{\partial x_1} \right), \quad (4.55)$$

where $\mathcal{D} = [0, 2\pi]^2$, periodic boundary conditions are considered and the initial condition $w(\cdot, 0)$ is fixed. Here we are interested in the mapping from the forcing term u to $v = \omega(\cdot, T)$, the vorticity field at a given time $t = T$, i.e., $\mathcal{G}^\dagger : u \mapsto \omega(\cdot, T)$.

The distribution μ is $\mathcal{GP}(0, (-\Delta + 3^2 I)^{-4})$. The viscosity ν is fixed and equal to 0.025, and the equation is solved on a 64×64 grid with a pseudo-spectral method and Crank-Nicholson time integration; see De Hoop et al., 2022 for further implementation details. Figure 4.9 shows an example of input and output in the test set, along with an example of test prediction and pointwise errors.

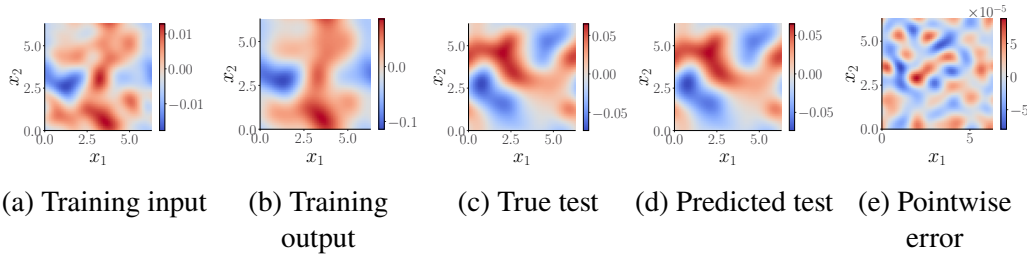


Figure 4.9: Example of training data and test prediction and pointwise errors for the Navier-Stokes problem (4.55).

Results and discussion

Below we discuss our main findings in benchmarking our kernel method against state-of-the-art NN based techniques

Performance against NNs

Table 4.3 summarizes the L^2 relative test error of our vanilla implementation of the kernel method along with those of DeepONet, FNO, PCA-Net, and PARA-Net. We observed that our vanilla kernel method was reliable in terms of accuracy across all examples. In particular, observe that between the Matérn or rational quadratic kernel, we always managed to get close to the other methods (see for example the results for the Burgers' equation or Darcy problem) and even outperform them in several examples such as Navier-Stokes and Helmholtz. Overall we observed that

the performance of the kernel method is stable across all examples suggesting that our method is reliable and provides a good baseline for a large class of problems. Moreover, we did not observe a significant difference in performance in terms of the choice of the particular kernel family once the hyper-parameters were tuned. This indicates that a large class of kernels are effective for these problems. Furthermore, we found the hyper-parameter tuning to be robust, i.e., results were consistent in a reasonable range of parameters such as length scales.

In the high data regime, we found the vanilla kernel method to be the most accurate, although this comes with a greater cost, as seen in Figure 4.10. However, the kernel method appears to provide the highest accuracy for its level of complexity as the accuracy of NNs typically stagnates or even decreases after a certain level of complexity; see the Navier-Stokes and Helmholtz panels of Figure 4.10 where most of the NN methods seem to plateau after a certain complexity level.

We also observed that the linear model did not provide the best accuracy as it quickly saturated in performance. Nonetheless, it provided surprisingly good accuracy at low levels of complexity: for example, in the case of Navier-Stokes, the linear kernel provided the best accuracy below 10^6 FLOPS of complexity. This indicates that while simple, the linear model can be a valuable low-complexity model. Another notable example is the Advection equation (both I and II), where the operator $\mathcal{G}^\dagger$ is linear. In this case, the linear kernel had the best accuracy and the best complexity-accuracy tradeoff. We note, however, that while the linear model was close to machine precision on Advection I (error on the order $10^{-13}\%$), its performance was significantly worse on Advection II (error on the order of 10%). Moreover, the gap between the linear kernel and all other models was significantly smaller for Advection I; we conjecture this difference in performance is likely due to the setup of these problems.

Finally, we note that the most challenging problem for our kernel method was the Structural Mechanics example. In this case, the vanilla kernel method has higher complexity but did not beat the NNs. In fact, the NNs seem to be able to reduce complexity without loss of accuracy compared to our method.

Effect of preconditioners

Table 4.4 compares the performance of our method with the Matérn kernel family using various preconditioning steps. Overall we observed that both PCA and Cholesky preconditioning improved the performance of our vanilla kernel method.

	Low-data regime			High-data regime			
	Burger's	Darcy problem	Advection I	Advection II	Helmholtz	Structural Mechanics	Navier Stokes
DeepONet	2.15%	2.91%	0.66%	15.24%	5.88%	5.20%	3.63%
POD-DeepONet	1.94%	2.32%	0.04%	n/a	n/a	n/a	n/a
FNO	1.93%	2.41%	0.22%	13.49%	1.86%	4.76%	0.26%
PCA-Net	n/a	n/a	n/a	12.53%	2.13%	4.67%	2.65%
PARA-Net	n/a	n/a	n/a	16.64%	12.54%	4.55%	4.09%
Linear	36.24%	6.74%	$2.15 \times 10^{-13}\%$	11.28%	10.59%	27.11%	5.41%
Best of Mat�rn/RQ	2.15%	2.75%	$2.75 \times 10^{-3}\%$	11.44%	1.00%	5.18%	0.12%

Table 4.3: Summary of numerical results: we report the L^2 relative test error of our numerical experiments and compare the kernel approach with variations of DeepONet, FNO, PCA-Net, and PARA-Net. We considered two choices of the kernel S , the rational quadratic and the Mat rn, but we observed little difference between the two.

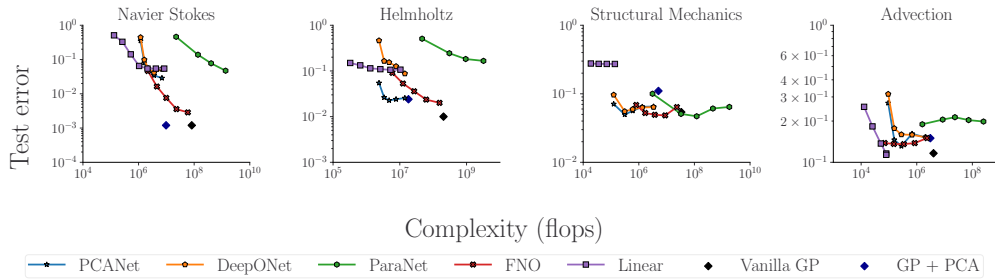


Figure 4.10: Accuracy complexity tradeoff achieved in the problems in De Hoop et al., 2022. Data for NNs was obtained from the aforementioned article. Linear model refers to the linear kernel, vanilla GP is our implementation with the nonlinear kernels and minimal preprocessing, and GP+PCA corresponds to preprocessing through PCA both the input and the output to reduce complexity.

The Cholesky preconditioning generally offers the greatest improvement. However, we observed that getting the best results from the Cholesky approach required careful tuning of the parameters of the kernels K and Q which we did using cross-validation. While tuning the parameters does not increase the inference complexity, it does increase the training complexity.

On the other hand, the PCA approach was more robust to changes in hyperparameters, i.e., the number of PCA components following Section 4.4. We observed that applying PCA on the input and output reduces complexity and has varying levels of effectiveness in providing a better cost-accuracy tradeoff. For example, for Navier-Stokes, it greatly reduced the complexity without affecting accuracy. But for the Helmholtz and Advection equations, PCA reduced the accuracy while remaining competitive with NN models. For structural mechanics, however, PCA significantly reduced accuracy and was worse than other models. We hypothesize that the loss in accuracy can be related to the decay of the eigenvalues of the PCA matrix in that example.

	Advection II	Burger's	Darcy problem
No preprocessing	14.37%	3.04%	4.47%
PCA	14.50%	2.41%	2.89%
Cholesky	11.44%	2.15%	2.75%

Table 4.4: Comparison between Cholesky preconditioning and PCA dimensionality reduction on three examples for our vanilla kernel implementation with the Matérn kernel.

4.5 Conclusions

In this work we presented a kernel/GP framework for the learning of operators between function spaces. We presented an abstract formulation of our kernel framework along with convergence proofs and error bounds in certain asymptotic limits. Numerical experiments and benchmarking against popular NN based algorithms revealed that our vanilla implementation of the kernel approach is competitive and either matches the performance of NN methods or beats them in several benchmarks. Due to simplicity of implementation, flexibility, and the empirical results, we suggest that the proposed kernel methods are a good benchmark for future, perhaps more sophisticated, algorithms. Furthermore, these methods can be used to guide practitioners in the design of new and challenging benchmarks (e.g, identify problems where vanilla kernel methods do not perform well). Numerous directions of future research exist. In the theoretical direction it is interesting to remove the stringent Condition 2 and we anticipate this to require a particular selection of the kernel employed to obtain the map $\tilde{f}$. Moreover, obtaining error bounds for more general measurement functionals beyond pointwise evaluations would be interesting. One could also adapt our framework to non-vanilla kernel methods such as random features or inducing point methods to provide a low-complexity alternative to NNs in the large-data regime. Finally, since the proposed approach is essentially a generalization of GP Regression to the infinite-dimensional setting, we anticipate that some of the hierarchical techniques of Houman Owhadi, 2017; Schaafer, Katzfuss, and Houman Owhadi, 2021; Schafer, Sullivan, and Houman Owhadi, 2021 could be extended to this setting and provide a better cost-accuracy trade-off than current methods.

Chapter 5

SOLVING FUNCTIONAL PDES WITH GAUSSIAN PROCESSES
AND APPLICATIONS TO FUNCTIONAL RENORMALIZATION
GROUP EQUATIONS

5.1 Introduction

Functional renormalization group (FRG) equations provide a powerful non-perturbative framework for studying quantum and statistical field theories, enabling interpolation between microscopic and macroscopic descriptions of physical systems Delamotte, 2012; Dupuis et al., 2021. Mathematically, they take the form of exact functional integro-differential equations for field-theoretic generating functionals, but solving these analytically is generally intractable, requiring numerical approximations. Traditional approaches rely on the truncation of expansions, such as the derivative expansion or loop expansion, restricting their validity to specific applications where those approximations are accurate. Moreover, numerical schemes are often tailored to particular models, requiring detailed knowledge of the underlying physics. While there have been efforts to propose general and flexible approaches to solving FRG equations Fischbacher and Synatschke-Czerwonka, 2013, including machine learning methods Yokota, 2024a, existing frameworks are either tied to a specific discretization of the underlying continuous functional PDE, restricted to specific fields, or rely upon a prescribed ansatz.

In this work, we introduce an operator learning perspective for solving FRG equations. Our framework, based on Gaussian processes (GPs), directly treats FRG flows in function space, sidestepping discretization dependence and enabling application across a wide range of FRG equations. This represents a step toward a general and flexible methodology for functional differential equations in theoretical physics. The remainder of this introduction reviews operator learning, provides an overview of FRG equations, surveys existing numerical methods, and outlines how our approach links the two.

Functional data and operator learning

The functional regression problem, where we aim to learn a mapping with functional inputs and outputs, has emerged as an important problem within the scientific machine learning literature where many applications naturally have functions as data points. A major challenge in this context is how to effectively adapt existing learning methods to this setting. The earliest kernel-based approach to functional regression appears in Kadri, Duflos, Preux, Stephane Canu, et al., 2016b, which generalizes vector-valued function learning to the setting of operator-valued kernels (see also Houman Owhadi, 2023b, which extends these concepts to compositions of operator-valued Gaussian processes). More recently, there has also been renewed interest in the field of operator learning, especially in the context of accelerating the

numerical computation of partial differential equations (PDEs). This problem can be traced back to the use of reduced basis methods Almroth, Stern, and Brogan, 1978; Lucia, Beran, and Silva, 2004; Noor and Peters, 1980, and in more recent years, the development of neural network methods such as DeepONet L. Lu, Jin, and G. E. Karniadakis, 2019, the Fourier neural operator Zongyi Li, N. Kovachki, et al., 2020b, the convolutional neural operators Raonic et al., 2023, or PCA-Net Bhattacharya, Hosseini, Nikola B Kovachki, et al., 2021a, have extended neural networks to functional inputs and outputs. However, the theoretical analysis of these methods is often limited to proving an existence theorem without guarantees about whether these networks are computable in practice. To address this, a general kernel-based operator-learning framework is introduced in Batlle, Matthieu Darcy, Hosseini, et al., 2024b, together with theoretical guarantees and convergence rates. A notable feature of such frameworks is their discretization-agnostic character Zongyi Li, N. Kovachki, et al., 2020b: they are formulated independently of any specific discretization and can therefore be applied to data given on different grids at prediction time.

Although the functional data in these problems often arise from PDE models, most operator-learning approaches treat the task purely as a regression problem and do not exploit the PDE structure that generates the data. In parallel, there has been extensive development of machine learning methods for approximating solutions of PDEs without relying on paired input-output data: the governing equations are enforced directly through a physics-informed loss function. Representative examples include physics-informed neural networks (PINNs) Raissi, P. Perdikaris, and G. Karniadakis, 2019a; Sifan Wang, Sankaran, et al., 2023 and kernel- or GP-based schemes Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b; Mou, X. Yang, and Zhou, 2022; R. Meng and X. Yang, 2023a. Kernel methods in this setting also enjoy stronger theoretical guarantees Baptista et al., 2025a; Batlle, Y. Chen, et al., 2025b.

While there has been a growing interest in combining elements of both approaches, for example by combining operator learning with a physics-informed loss Zongyi Li, Zheng, et al., 2023a, there has been little research into the applications of machine learning methods to *partial differential equations defined over functionals*, i.e., mappings from functions to a real number. Solving such equations requires elements from both of the outlined problems above: the ability to effectively deal with functions as inputs to the algorithm and the ability to impose the PDE constraints.

Such a problem naturally arises in the context of FRG equations which define integro-differential equations over functionals. We briefly describe these equations in the next section. Recently, deep learning methods have been applied to problems related to the FRG flow. For example, Chung and Kao, 2021; Di Sante, Medvidovic, et al., 2022a learn the flow directly from data, and Rancon et al., 2025 uses a simple one layer convolutional neural network to learn the inverse of flow, from coarse scale to fine scale. A physics-informed neural network (PINN) Raissi, P. Perdikaris, and G. Karniadakis, 2019a has recently been proposed to solve FRG equations on a lattice Yokota, 2024a, thereby reducing the problem to solving a PDE in a high-dimensional lattice space. The approach presented in the latter paper is analogous to the *discretize then optimize* paradigm where the infinite-dimensional problem is first discretized into a finite dimensional problem and solved through existing methods. In this paper, we take an *optimize-then-discretize* perspective: we formulate a solution of the flow using the governing equations directly in function space before discretizing. To the best of our knowledge, there is no pre-existing FRG solution framework formulated directly on function space. This formulation has the potential to enable rigorous a priori error analysis taking into account the various sources of error, similar to the one presented in Batlle, Matthieu Darcy, Hosseini, et al., 2024b.

Functional renormalization group equations

The nonperturbative FRG provides a framework for interpolating between microscopic and macroscopic descriptions of many systems, from condensed matter to statistical field theory. A central object in this framework is a family of scale-dependent functionals, such as the effective average action. These functionals evolve with the renormalization scale κ and encode how fluctuations at momenta above κ are progressively integrated out. Two widely used FRG flow equations are the Wetterich–Morris equation

$$\partial_\kappa \Gamma_\kappa[\phi] = \frac{1}{2} \int_{x,y} \partial_\kappa R_\kappa(x, y) \left(\Gamma_\kappa^{(2)}[\phi] + R_\kappa \right)^{-1}(x, y) dx dy, \quad (5.1)$$

which governs the evolution of the effective average action Γ_κ , and the Wilson–Polchinski equation

$$\partial_\kappa W_\kappa[J] = \frac{1}{2} \int_{x,y} \partial_\kappa R_\kappa(x, y) \left(\frac{\delta^2 W_\kappa[J]}{\delta J_x \delta J_y} + \frac{\delta W_\kappa[J]}{\delta J_x} \frac{\delta W_\kappa[J]}{\delta J_y} \right) dx dy, \quad (5.2)$$

which describes the corresponding flow of the cumulant-generating functional W_κ , related to the effective-average action by a generalized Legendre transform duality

relation

$$\Gamma_\kappa[\phi] + W_\kappa[J] = \int_x J_x \phi_x - \frac{1}{2} \int_{x,y} R_\kappa(x,y) \phi_x \phi_y. \quad (5.3)$$

Here, ϕ and J are fields (functions), R_κ is a scale-dependent regulator kernel that suppresses long-wavelength modes below the running scale κ , and $\Gamma_\kappa^{(2)}[\phi]$ denotes the operator defined by the second functional derivative of Γ_κ with respect to ϕ . For clarity of exposition, in what follows we primarily formulate the discussion in terms of the Wetterich equation and its solution, the effective average action Γ_κ . The corresponding formulations for the Wilson–Polchinski flow W_κ are entirely analogous, and in Section 5.3 we include experiments for both formulations.

In typical applications one prescribes an initial condition for (5.1) at a large *ultra-violet* (UV) scale κ_{UV} . For such large values of κ , the regulator strongly suppresses fluctuations at all resolved scales, so that the description remains close to the underlying microscopic model. At this scale one sets $\Gamma_{\kappa_{\text{UV}}} = S_{\text{bare}}$, where S_{bare} , called the *bare action*, is a prescribed action functional specifying the microscopic model (for example, the local interactions and parameters before fluctuations are integrated out). The flow equation is then integrated in κ towards smaller *infrared* (IR) scales $\kappa_{\text{IR}} > 0$, corresponding to lower momenta and larger distances. Along this flow, Γ_κ evolves from a microscopic description to an effective macroscopic one. Analytic solutions are available only in very simple models, so practical computations typically rely on truncations or modeling assumptions tailored to the problem at hand.

From a mathematical perspective, equations such as (5.1) and (5.2) are examples of functional PDEs. Abstractly, one may view Γ_κ as a functional that assigns a real number to each field configuration ϕ , and $\kappa \mapsto \Gamma_\kappa$ as an evolution in the scale variable on some interval $I \subset \mathbb{R}$. We write this in the generic form

$$\partial_\kappa \Gamma_\kappa[\phi] = F\left(\kappa, \phi, \Gamma_\kappa[\phi], \frac{\delta \Gamma_\kappa}{\delta \phi}[\phi], \frac{\delta^2 \Gamma_\kappa}{\delta \phi^2}[\phi]\right), \quad \kappa \in I, \quad (5.4)$$

with an initial condition prescribed at some reference scale $\kappa_0 \in I$,

$$\Gamma_{\kappa_0}[\phi] = \Gamma_{\kappa_0}^{\text{init}}[\phi]. \quad (5.5)$$

Here $\Gamma_{\kappa_0}^{\text{init}}$ is a prescribed functional that encodes the initial data of the problem (for instance, the bare action in an FRG flow, or a given energy functional in other applications). The map F is a (typically nonlocal) functional that may involve spatial integrals and depends on Γ_κ and its functional derivatives. The Wetterich–Morris

equation arises from a particular choice of F , in which the right-hand side is written as an integral involving the inverse operator of $\Gamma_\kappa^{(2)}[\phi] + R_\kappa$, and $\Gamma_{\kappa_0}^{\text{init}}$ is given by the bare action S_{bare} at the ultraviolet scale $\kappa_0 = \kappa_{\text{UV}}$. In FRG applications one usually takes $I = [\kappa_{\text{IR}}, \kappa_{\text{UV}}]$ and integrates the flow from κ_{UV} down to κ_{IR} , but the abstract formulation (5.4)–(5.5) equally covers forward evolutions in κ .

In this work we develop a GP method for solving such functional PDEs. The basic idea is to approximate, at each scale κ , the unknown functional $\Gamma_\kappa[\cdot]$ by a GP surrogate $\Gamma_\kappa^\dagger[\cdot]$. We first fix a finite set of collocation fields $\Phi = \{\phi^{(i)}\}_{i=1}^N$. For a given κ , the values $\{\Gamma_\kappa[\phi^{(i)}]\}_{i=1}^N$ are regarded as observations of a GP prior on Γ_κ . Concretely, we place a GP prior on the map $\phi \mapsto \Gamma_\kappa[\phi]$ with a covariance kernel between fields, and define the surrogate $\Gamma_\kappa^\dagger$ as the corresponding GP posterior mean. In other words, for any input field ϕ ,

$$\Gamma_\kappa^\dagger[\phi] := \mathbb{E}[\Gamma_\kappa[\phi] \mid \Gamma_\kappa[\phi^{(1)}], \dots, \Gamma_\kappa[\phi^{(N)}]], \quad (5.6)$$

so $\Gamma_\kappa^\dagger$ is uniquely determined by the chosen kernel and by the functional values on the collocation set. The functional PDE is then enforced only on the collocation set. At a fixed scale κ and for each $\phi^{(i)} \in \Phi$, we insert $\Gamma_\kappa^\dagger$ into the right-hand side of the evolution equation, and compute the required functional derivatives (for example, functional gradients and Hessians) at $\phi^{(i)}$ through $\Gamma_\kappa^\dagger$. Matching this right-hand side with the κ -derivative of $\Gamma_\kappa^\dagger[\phi^{(i)}]$ yields, for $i = 1, \dots, N$, a closed system of ordinary differential equations (ODEs) in κ for the unknown values $\Gamma_\kappa[\phi^{(i)}]$ on the collocation set. Solving this finite-dimensional ODE system over the κ -interval of interest determines $\Gamma_\kappa[\phi^{(i)}]$ for all κ and all collocation fields $\phi^{(i)}$. At any given scale, the GP surrogate $\Gamma_\kappa^\dagger[\cdot]$ given by (5.6) can then be evaluated on arbitrary input fields, providing a global approximation to the solution of the functional PDE.

For clarity of exposition, in Section 5.2 we first develop the methodology in detail for a general class of functional PDEs of the form (5.4). We then illustrate the framework, and provide the corresponding implementation details, for the Wetterich–Morris equation (5.1).

Numerical methods for FRG and functional PDEs

The FRG leads to exact flow equations for scale-dependent functionals such as the effective average action Γ_κ or the cumulant-generating functional W_κ . Prototypical examples are the Wetterich–Morris equation (5.1) and the Wilson–Polchinski equation (5.2). Except in special Gaussian or mean-field limits, these functional PDEs are not analytically solvable, and practical FRG computations rely on truncations

in functional space followed by numerical solution of the resulting PDEs or ODE systems; see, e.g., the reviews Berges, Tetradis, and Wetterich, 2002; Delamotte, 2012; Jan M Pawłowski, 2007; Metzner et al., 2012; Dupuis et al., 2021.

A large class of methods begins by projecting the flow onto a finite number of field-dependent functions or couplings. The simplest and most widely used example is the local potential approximation (LPA), where the flow is restricted to a scale-dependent local potential evaluated on homogeneous backgrounds. Derivative expansions and vertex expansions extend this ansatz by including higher-derivative operators or n -point vertex functions, but they still reduce the functional PDE to a coupled system of equations in a few field or momentum variables. Once such a truncation is fixed, standard grid-based methods (finite differences or finite volumes) are applied to the remaining variables. In particular, finite-volume and discontinuous Galerkin schemes have been designed to handle non-analytic features in the flow without spurious oscillations and with good stability properties Grossi and Wink, 2023.

Higher-order Galerkin and finite-element methods have recently been developed to improve accuracy and adaptivity. In these approaches, the scale-dependent quantities (e.g., effective potentials or vertices) are expanded in local basis functions, and the FRG flow is enforced in a weak sense. This allows for mesh refinement near sharp transitions or non-analytic regions while keeping a coarser resolution elsewhere. The DiFfRG framework is a representative example, providing a finite-element toolbox for a broad class of bosonic and fermionic FRG flows, with parallel and GPU implementations for demanding applications Franz R Sattler and Jan M Pawłowski, 2024a. Discontinuous Galerkin variants combine the flexibility of finite elements with robust shock-capturing and have been used, for instance, to resolve first and second order phase transitions in scalar models with high accuracy Grossi and Wink, 2023.

Spectral and pseudo-spectral methods take a complementary, global viewpoint. In these approaches the scale-dependent functions (for instance effective potentials or fixed-point solutions) are expanded in global bases such as Chebyshev polynomials, and the FRG flow equations are enforced at appropriate collocation points. The articles Borchardt and Knorr, 2015; Borchardt and Knorr, 2016 demonstrate that such pseudo-spectral schemes can produce exponentially convergent, high-precision solutions for global fixed points and critical exponents in scalar and Yukawa models, capturing the behavior at both small and large fields. Spectral methods are particularly effective when the flow remains sufficiently smooth, while non-analyticities

typically require domain decompositions or hybrid strategies.

A common feature of these FRG approaches is that they start from a model-specific truncation (such as the local potential approximation, derivative expansions, or vertex expansions) and then approximate the resulting flow equations in terms of a small number of field or momentum variables; see, for example, the reviews Berges, Tetradis, and Wetterich, 2002; Delamotte, 2012. While the functional dependence $\phi \mapsto \Gamma_\kappa[\phi]$ is approximated, one works ultimately with a finite family of scalar or tensor functions defined on physical or momentum space.

More recently, machine-learning-based FRG solvers have been proposed. In particular, physics-informed neural networks (PINNs) have been used to encode FRG flow equations in the loss functional and to approximate solutions without an explicit spatial grid, for both scalar and fermionic models Yokota, 2024a. Such approaches can in principle handle higher-dimensional truncations, although issues of training cost, robustness, and a posteriori error control remain active topics of research.

By contrast, we represent the functional with a GP and impose the FRG flow equation as a structural constraint, yielding a GP-based numerical scheme for functional PDEs.

Summary of contributions

The main contributions of this work can be summarized as follows.

1. **GP framework for functional PDEs.** We develop a GP method for solving functional evolution equations of the form (5.4). The method represents the unknown functional directly in function space, so it is independent of any particular spatial or temporal discretization. Different physical representations (real or complex Fourier modes, pointwise samples, lattice fields) enter only through the choice of feature map and kernel, while the core algorithm remains unchanged.
2. **Application to functional renormalization group flows.** We specialize this framework to the FRG, with a detailed formulation for the Wetterich–Morris equation (5.1), and numerical experiments for both the Wetterich–Morris and the Wilson–Polchinski equations (5.2).
3. **Treatment of non-homogeneous fields.** In contrast to standard truncation schemes such as the local potential approximation, which are formulated on

constant or homogeneous backgrounds, our GP method can be trained on nonconstant collocation fields when feasible, and it can then evaluate the functional on arbitrary inputs.

4. **Physics-informed GP priors.** We show how known structure from standard FRG approximations can be encoded in the GP prior, for example through local-potential-type ansatzes. This allows the surrogate to incorporate key physical insights while remaining fully nonparametric in function space.
5. **Numerical validation on FRG benchmarks.** We test the method on several FRG flows, including Gaussian models with closed-form solutions and interacting ϕ^4 field-theory models where no explicit solution is available. All applications here consider fields defined on a one-dimensional input domain. In the Gaussian cases, the GP surrogate reproduces the exact flow to high accuracy both on and off the collocation set. For the interacting models, we consider a lattice formulation. In fact, among the spatial dimensions considered in Zorbach et al., 2025, the approximation error of the LPA is largest in the one-dimensional case. Building on the benchmarks of Zorbach et al., 2025, we find that our method yields solutions with smaller pointwise errors than a standard LPA flow, over most of the parameter range, for standard physical observables.

Notation.

We write $\mathbb{R} = (-\infty, \infty)$ for the real numbers. We are interested in approximating functionals (i.e., functions of functions) that arise as solutions of integro-differential equations. Throughout we use the following conventions:

- Capital letters W and Γ are reserved for *functionals*, i.e. maps from the function space $\mathcal{U}$ to the real line $\mathbb{R}$.
- For a functional Γ , we write $\delta\Gamma/\delta\phi$ for its first functional derivative and $\delta^2\Gamma/\delta\phi\delta\phi$ for its second functional derivative. The latter may be taken as formal kernel defining an integral operator, which, following standard practice, we denote as $\Gamma^{(2)}[\phi]$.
- The evolution parameter is the RG scale and is denoted by $\kappa \in [\kappa_{\text{IR}}, \kappa_{\text{UV}}]$.
- Lower-case Greek letters such as ϕ , φ , and ψ are used for real-valued functions (fields) in the context of the Wetterich equation and, more generally, functional

PDEs. Following the conventional notation of the Wilson–Polchinski equation in the field theory literature, we keep J as the field variable for the Wilson–Polchinski equation.

- Elements of the function space, $\phi \in \mathcal{U}$, are referred to as *function-space collocation functions or fields*.

We use the following vector broadcasting convention: given a function $f: \mathbb{R} \rightarrow \mathbb{R}$ and a vector $\boldsymbol{\phi} = (\phi_i)_{i=1}^n$, the notation $f(\boldsymbol{\phi})$ denotes elementwise application, $f(\boldsymbol{\phi}) := (f(\phi_1), \dots, f(\phi_n))$.

Outline of the rest of the paper

The rest of the paper is organized as follows. In Section 5.2, we present our methodology for solving general functional PDEs with GPs and its application to the illustrative example of the Wetterich–Morris equation. Section 5.3 presents our main numerical results for two FRG flows of interest, the Wilson–Polchinski equation and the Wetterich–Morris equation. Several appendices fill in background information. Section D.1 reviews the connection between optimal recovery and kernel-based interpolants. We report additional numerical experiments in Section D.2. In Section D.3, we provide a self-contained mathematical presentation of the FRG flow problem. In Section D.4, we discuss functional derivatives and Hessians of kernel interpolants. In Section D.5, we give a brief introduction to the local potential approximation (LPA), a standard method for obtaining approximate solutions of the Wetterich equation on constant fields. In Section D.6, we review the transfer-matrix method for computing the limit solution of the Wetterich equation in one dimension. Section D.7 details the implementation of the Wetterich equation on the lattice with ϕ^4 bare action used in the numerical experiment of Section 5.3. Finally, Section D.8 briefly reviews the definitions of two physical observables, magnetization and susceptibility, which can be computed from the infrared local potential and used as numerical benchmarks for the Wetterich equation.

5.2 Methodology

We now outline the methodology used in this work and its application to FRG flows. Our starting point is the kernel-based operator learning framework of Batlle, Matthieu Darcy, Hosseini, et al., 2024b, which is designed to learn maps between function spaces from data. In this work we adapt this framework to represent functionals, that is, maps from function spaces to scalars. The key distinction from

data-driven operator learning is that, in our setting, the functional values are not directly observed. Instead, we incorporate the FRG flow equation as an additional constraint on the surrogate, so that the reconstructed functional is informed by the underlying functional PDE rather than by paired input-output data. We begin in Section 5.2 by reviewing the relationship between GP regression and kernel interpolation. Section 5.2 then discusses kernel-based approximation of functionals. In Section 5.2, we develop a general framework for solving functional PDEs. Finally, in Section 5.2 we specialize this framework to the Wetterich–Morris equation (5.1) as an illustrative example.

Gaussian processes and kernel interpolation

In this subsection, we briefly review GPs and their connection with kernel interpolation. A real-valued GP is a collection of random variables $\{f(x)\}_{x \in \mathcal{X}}$ indexed by points of an input space $\mathcal{X}$, such that every finite vector $(f(x_1), \dots, f(x_n))$ is jointly Gaussian. A GP is characterised by its mean function $m(x) = \mathbb{E}[f(x)]$ and its covariance kernel $k(x, x') = \text{Cov}(f(x), f(x'))$. For a zero-mean prior, $m(x) \equiv 0$, the law of the random field f is completely determined by the positive-definite kernel k .

The same kernel k defines a reproducing kernel Hilbert space (RKHS) $\mathcal{H}_k$. Given exact observations $y_i = f(x_i)$ at points $\{x_i\}_{i=1}^n$, the RKHS kernel interpolant $f^\dagger \in \mathcal{H}_k$ is the unique minimum-norm function satisfying

$$f^\dagger(x_i) = y_i, \quad i = 1, \dots, n,$$

and it admits the explicit representation

$$f^\dagger(x) = k(x, X) K(X, X)^{-1} y,$$

where $X = (x_1, \dots, x_n)$, $y = (y_1, \dots, y_n)^\top$, and $K(X, X)_{ij} = k(x_i, x_j)$.

If we place a zero-mean GP prior with covariance k on the random field f and condition on the same noiseless data, the posterior GP has mean

$$\mathbb{E}[f(x) \mid y] = k(x, X) K(X, X)^{-1} y.$$

Thus the GP posterior mean coincides exactly with the RKHS kernel interpolant $f^\dagger$.

Kernel approximation of functionals

Let $\mathcal{U}$ be a Banach space of functions and further let $\Gamma : \mathcal{U} \rightarrow \mathbb{R}$ be an unknown functional that we wish to approximate. The basic idea is to encode each field

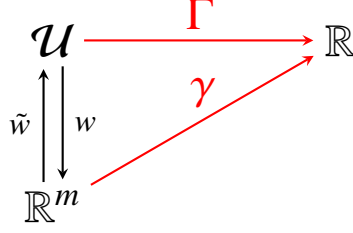


Figure 5.1: Commutative diagram for the functional learning setup.

$\phi \in \mathcal{U}$ by a finite vector of features $w(\phi) \in \mathbb{R}^m$, learn a scalar function $\gamma : \mathbb{R}^m \rightarrow \mathbb{R}$ on this finite-dimensional feature space, and then approximate the functional $\Gamma(\phi)$ by the composition $\gamma(w(\phi))$. A commutative diagram illustrating the relationships among operators, function spaces, and functionals is shown in Figure 5.1, where the reconstruction operator $\tilde{w}$ which “inverts” w is discussed later in Remark 12. Concretely, when $\mathcal{U}$ is infinite-dimensional, one does not compute directly with $\phi \in \mathcal{U}$, but with the finite-dimensional projections $w(\phi) \in \mathbb{R}^m$ as in Houman Owahdi and Scovel, 2019c. In the applications we mainly use two concrete choices. The first choice is based on pointwise measurements in space (or time):

$$w(\phi) = (\phi(x_j))_{j=1}^m,$$

where $\{x_j\}_{j=1}^m$ are prescribed collocation points. This requires the evaluation functionals δ_{x_j} to lie in the dual space of $\mathcal{U}$. A second choice is to use spectral coefficients in an orthonormal basis for the case that $\mathcal{U}$ is a Hilbert space. For example, if $\mathcal{U} \subset L^2([0, 1])$, we may expand

$$\phi(x) = \sum_{j=-\infty}^{\infty} c_j e^{i2\pi jx},$$

and define the feature vector as a truncated set of Fourier coefficients,

$$w(\phi) = (c_j)_{j=-m}^m.$$

This construction only requires that the fields belong to $L^2([0, 1])$; other orthonormal bases (for instance, real Fourier bases or wavelets) can be used in the same way.

Once a projection w is fixed, our aim is to approximate Γ by a functional that depends on ϕ only through the finite-dimensional feature vector $w(\phi)$. Concretely, we seek a scalar function $\gamma : \mathbb{R}^m \rightarrow \mathbb{R}$ such that

$$\Gamma(\phi) \approx \gamma(w(\phi)) \quad \text{for } \phi \in \mathcal{U}.$$

Thus the problem of approximating the functional Γ is reduced to approximating the finite-dimensional function γ . To construct γ we use optimal recovery and kernel interpolation Houman Owhadi and Scovel, 2019c. Suppose we are given training fields $\Phi = \{\phi^{(1)}, \dots, \phi^{(N)}\} \subset \mathcal{U}$ and their functional values

$$Y := (\Gamma(\phi^{(1)}), \dots, \Gamma(\phi^{(N)}))^T \in \mathbb{R}^N.$$

Let $z^{(i)} := w(\phi^{(i)}) \in \mathbb{R}^m$ denote the corresponding feature vectors, and let $\mathcal{H}_K$ be the RKHS associated with the kernel K . We seek γ as the minimum-norm interpolant

$$\gamma^\dagger = \arg \min_{\gamma \in \mathcal{H}_K} \|\gamma\|_{\mathcal{H}_K}^2 \quad (5.7)$$

$$\text{subject to } \gamma(z^{(i)}) = \Gamma(\phi^{(i)}), \quad i = 1, \dots, N. \quad (5.8)$$

The representer theorem (see Houman Owhadi and Scovel, 2019c, Sec. 17.8) implies that $\gamma^\dagger$ has the explicit form

$$\gamma^\dagger(z) = K(z, Z)^T K(Z, Z)^{-1} Y, \quad (5.9)$$

where $Z = (z^{(1)}, \dots, z^{(N)})$, $K(z, Z) \in \mathbb{R}^N$ has entries $K(z, Z)_i = K(z, z^{(i)})$, and $K(Z, Z) \in \mathbb{R}^{N \times N}$ is the Gram matrix with entries $K(Z, Z)_{ij} = K(z^{(i)}, z^{(j)})$. Our kernel surrogate for the functional is then

$$\Gamma^\dagger(\phi) := \gamma^\dagger(w(\phi)) = K(w(\phi), Z)^T K(Z, Z)^{-1} Y, \quad (5.10)$$

which we will use in the sequel as an approximation of Γ .

The framework of Battle, Matthieu Darcy, Hosseini, et al., 2024b uses this projection-reconstruction machinery to address data-driven operator learning, where one is given evaluations of Γ at many input functions (typically from an existing numerical solver) and no physics is imposed. In our setting we use the same kernel representation (5.10) for Γ , but the functional values are not directly available. Instead, we enforce that the GP surrogate $\Gamma^\dagger$ satisfies the FRG flow equations, so that the surrogate is informed by the underlying functional PDE rather than by paired input-output data. Compared to neural network approaches to approximating functionals L. Lu, Jin, and G. E. Karniadakis, 2019; Zongyi Li, N. Kovachki, et al., 2020b; Houman Owhadi, 2023b, the proposed approach benefits from the solid theoretical foundations of GP methods Houman Owhadi and Scovel, 2019c, and can be employed to solve integro-differential equations in functional spaces, as shown in the following section.

Remark 12. *An important theoretical advantage of this formulation is that it may be rendered discretization-agnostic by means of a reconstruction operator $\tilde{w} : \mathbb{R}^m \rightarrow \mathcal{U}$ that maps measurement vectors back to elements of $\mathcal{U}$. See Figure 5.1. This reconstruction operator can be defined using the framework of optimal recovery (see Section D.1), which enables rigorous a priori error estimates for the recovery of $\phi \in \mathcal{U}$. Combined with the error rates associated with kernel interpolation, this can yield rigorous convergence guarantees and error rates for the recovery of Γ_κ . We leave this to future work.*

A GP method for solving functional PDEs

Let $\mathcal{U}$ be a (typically infinite-dimensional) function space and let $\Gamma_\kappa : \mathcal{U} \rightarrow \mathbb{R}$, $\kappa \in I \subset \mathbb{R}$, be a one-parameter family of functionals. We consider the functional evolution problem (5.4)–(5.5), where the right-hand side F is built from ϕ , $\Gamma_\kappa[\phi]$, and a finite number of functional derivatives of Γ_κ . The variable κ plays the role of an evolution parameter and may be advanced either forward or backward. In FRG applications κ is interpreted as a momentum scale and the flow is typically integrated backward from a UV scale κ_{UV} to an IR scale κ_{IR} , but the construction below is independent of this choice of direction.

The main difficulty is that Γ_κ lives on the infinite-dimensional space $\mathcal{U}$. Our strategy is to represent the evolution entirely through the values of Γ_κ on a finite set of collocation fields and to use the GP surrogate from Section 5.2 to extend these values to all $\phi \in \mathcal{U}$. In this way the functional PDE is reduced to a closed ODE system in finitely many unknowns.

We recall the kernel approximation of functionals from Section 5.2. We fix once and for all a feature map $w : \mathcal{U} \rightarrow \mathbb{R}^m$, a positive-definite kernel K on $\mathbb{R}^m$ with reproducing kernel Hilbert space $\mathcal{H}_K$, and a set of functional collocation fields

$$\Phi = \{\phi^{(1)}, \dots, \phi^{(N)}\} \subset \mathcal{U}.$$

For instance, one may take $w_k(\phi) = \langle e_k, \phi \rangle$, where $e_k \in \mathcal{U}$ are prescribed basis functions and $\langle \cdot, \cdot \rangle$ denotes the inner product in the underlying Hilbert space. Other projections tailored to the problem can also be used. The associated feature vectors are

$$z^{(i)} := w(\phi^{(i)}) \in \mathbb{R}^m, \quad Z := (z^{(1)}, \dots, z^{(N)}),$$

and we write $K(Z, Z) \in \mathbb{R}^{N \times N}$ for the Gram matrix introduced in Section 5.2. Given a vector $Y \in \mathbb{R}^N$ of functional values on Φ , the kernel construction (5.10)

defines a GP surrogate $\Gamma^\dagger$. In numerical computations we replace $K(Z, Z)^{-1}$ by $(K(Z, Z) + \varepsilon I)^{-1}$ for a small $\varepsilon > 0$ for stability, but the representer form itself is unchanged.

In the evolutionary setting the role of Y is played by the values of the scale-dependent functional on the collocation set. For each $\kappa \in I$ we introduce the vector

$$Y_\kappa := (\Gamma_\kappa[\phi^{(1)}], \dots, \Gamma_\kappa[\phi^{(N)}])^\top \in \mathbb{R}^N, \quad (5.11)$$

which is the primary unknown. At the initial scale $\kappa_0 \in I$ these values are prescribed by the initial functional $\Gamma_{\kappa_0}^{\text{init}}$,

$$Y_{\kappa_0} = \Gamma_{\kappa_0}^{\text{init}}(\Phi) = (\Gamma_{\kappa_0}^{\text{init}}[\phi^{(1)}], \dots, \Gamma_{\kappa_0}^{\text{init}}[\phi^{(N)}])^\top,$$

which will serve as the initial condition for the finite-dimensional evolution.

Given Y_κ , we approximate $\Gamma_\kappa[\cdot]$ at scale κ by the GP surrogate defined, as in (5.10), with $Y = Y_\kappa$:

$$\Gamma_\kappa^\dagger[\phi] := K(w(\phi), Z) (K(Z, Z) + \varepsilon I)^{-1} Y_\kappa, \quad \phi \in \mathcal{U}. \quad (5.12)$$

Thus $\Gamma_\kappa^\dagger$ is defined for all input fields and depends on κ through the finite-dimensional vector Y_κ .

To obtain a closed evolution for Y_κ , we enforce the functional PDE (5.4) only on the collocation set Φ . For a fixed $\kappa \in I$ and each $\phi^{(i)} \in \Phi$, we approximate the right-hand side of (5.4) by inserting the surrogate (5.12):

$$\partial_\kappa \Gamma_\kappa[\phi^{(i)}] \approx F\left(\kappa, \phi^{(i)}, \Gamma_\kappa^\dagger[\phi^{(i)}], \frac{\delta \Gamma_\kappa^\dagger}{\delta \phi}\Big|_{\phi^{(i)}}, \frac{\delta^2 \Gamma_\kappa^\dagger}{\delta \phi^2}\Big|_{\phi^{(i)}}\right), \quad i = 1, \dots, N, \quad (5.13)$$

where the functional derivatives of $\Gamma_\kappa^\dagger$ are computed analytically from the kernel representation; see Section D.4 for details. By definition $Y_\kappa^{(i)} = \Gamma_\kappa[\phi^{(i)}]$, so the left-hand side of (5.13) is exactly

$$\partial_\kappa \Gamma_\kappa[\phi^{(i)}] = \frac{d}{d\kappa} Y_\kappa^{(i)}.$$

Hence (5.13) naturally induces a closed system of ODEs for the vector Y_κ :

$$\frac{d}{d\kappa} Y_\kappa^{(i)} = F\left(\kappa, \phi^{(i)}, \Gamma_\kappa^\dagger[\phi^{(i)}], \frac{\delta \Gamma_\kappa^\dagger}{\delta \phi}\Big|_{\phi^{(i)}}, \frac{\delta^2 \Gamma_\kappa^\dagger}{\delta \phi^2}\Big|_{\phi^{(i)}}\right), \quad i = 1, \dots, N, \quad (5.14)$$

with initial condition

$$Y_{\kappa_0} = \Gamma_{\kappa_0}^{\text{init}}(\Phi).$$

Equation (5.14) is a standard ODE system on $\mathbb{R}^N$. In FRG applications one typically takes $I = [\kappa_{\text{IR}}, \kappa_{\text{UV}}]$ and integrates this ODE backward from κ_{UV} to κ_{IR} , but the same formulation applies equally to forward evolutions in κ . Once Y_κ has been computed on the interval of interest, the surrogate (5.12) provides an approximation of $\Gamma_\kappa[\phi]$ for arbitrary fields $\phi \in \mathcal{U}$.

A functional RG example: the Wetterich–Morris equation

Section 5.2 provides a framework for solving general functional PDEs. To illustrate how this framework applies to FRG flows, we now specialize it to the Wetterich–Morris equation (5.1), for fields that are either periodic on the one-dimensional torus $\mathbb{T}$, identified with the interval $[0, 1]$, or defined on a lattice. Recall that the scale parameter κ plays the role of a sliding momentum cutoff. At a large ultraviolet scale κ_{UV} the functional $\Gamma_{\kappa_{\text{UV}}}$ is prescribed by a microscopic action S_{bare} that encodes the model at short distances. We integrate (5.1) backward in κ from κ_{UV} down to κ_{IR} . As κ is lowered towards an infrared scale $\kappa_{\text{IR}} > 0$, more and more high-wavenumber modes are integrated out and Γ_κ evolves towards the full effective action.

Finite-dimensional representation of fields. In the GP method of Section 5.2 the first step is to fix a feature map $w : \mathcal{U} \rightarrow \mathbb{R}^m$ that encodes each field by a finite vector. We now specify this map for the Wetterich–Morris equation.

We work with a function space $\mathcal{U}$ of fields and view $\mathcal{U}$ as continuously embedded in a Hilbert space $\mathcal{X}$ that carries the natural inner product of the problem. For continuum examples in this work, $\mathcal{X}$ is the periodic L^2 -space on the one-dimensional torus $\mathbb{T}$ or the corresponding Sobolev space $H^1(\mathbb{T})$. For lattice examples, $\mathcal{X}$ is the discrete L^2 -space of grid functions.

It is convenient to represent fields in an orthonormal basis of Laplacian eigenfunctions, so that the Laplacian and standard translation-invariant regulators act diagonally in this basis. We therefore fix a real orthonormal basis $\{e_\alpha\}_{\alpha \in \mathcal{A}}$ of $\mathcal{X}$ consisting of eigenfunctions of the continuum Laplacian or the discrete Laplacian on the lattice. On the torus $\mathbb{T}$, a standard choice is the complex Fourier basis $\{e^{2\pi i p x}\}_{p \in \mathbb{Z}}$, or, equivalently, the associated real Fourier basis

$$e_0(x) = 1, \quad e_{2p-1}(x) = \sqrt{2} \cos(2\pi p x), \quad e_{2p}(x) = \sqrt{2} \sin(2\pi p x), \quad p \geq 1.$$

On a periodic lattice we use the corresponding discrete Fourier modes.

For numerical purposes we truncate to a finite index set $\mathcal{A}_P \subset \mathcal{A}$ of active modes and work on the finite-dimensional subspace $\mathcal{X}_P := \text{span}\{e_\alpha : \alpha \in \mathcal{A}_P\}$. Here P is

a truncation parameter that controls how many Fourier modes (or, more generally, eigenmodes) are retained: increasing P increases both the dimension of $\mathcal{X}_P$ and the highest resolved frequency. The truncation level P is chosen so that $\{e_\alpha\}_{\alpha \in \mathcal{A}_P}$ captures all physically relevant modes while keeping the associated matrices of moderate size. Such wavenumber truncation is standard in physics practice as well, e.g. in the case of a spatial lattice model with spacing a naturally $P = \pi/a$.

Given this basis, we define the feature map w by the truncated expansion coefficients

$$c_\alpha[\phi] := \langle \phi, e_\alpha \rangle_{\mathcal{X}}, \quad \alpha \in \mathcal{A}_P,$$

and set

$$w(\phi) := (c_\alpha[\phi])_{\alpha \in \mathcal{A}_P} \in \mathbb{R}^m, \quad m := |\mathcal{A}_P|.$$

On a lattice the inner product $\langle \cdot, \cdot \rangle_{\mathcal{X}}$ is understood as the corresponding discrete L^2 inner product. With this choice, all subsequent GP surrogates $\Gamma_\kappa^\dagger$ are constructed from the finite-dimensional feature vectors $w(\phi)$ as in Section 5.2.

GP formulation of the Wetterich–Morris flow. We now specialise the general GP framework of Section 5.2 to the Wetterich–Morris equation (5.1).

On the truncated mode set $\mathcal{A}_P$ we represent each field $\phi \in \mathcal{U}$ by its coefficients in the eigenbasis discussed above, and use these coefficients as features:

$$w(\phi) := (c_\alpha[\phi])_{\alpha \in \mathcal{A}_P}, \quad c_\alpha[\phi] = \langle \phi, e_\alpha \rangle_{\mathcal{X}}.$$

We then fix a finite set of collocation fields

$$\Phi = \{\phi^{(1)}, \dots, \phi^{(N)}\} \subset \mathcal{U}, \quad z^{(i)} := w(\phi^{(i)}) \in \mathbb{R}^m,$$

and collect the feature vectors into $Z = (z^{(1)}, \dots, z^{(N)})$. A covariance kernel K on feature space is fixed once and for all, and its Gram matrix $K(Z, Z)$ enters the GP surrogate exactly as in (5.12).

For each scale κ , the unknown values on the collocation set are collected into the vector Y_κ defined in (5.11), which in turn determines the GP surrogate $\Gamma_\kappa^\dagger[\cdot]$ through (5.12). At the ultraviolet scale κ_{UV} the initial condition is given by the bare action,

$$Y_{\kappa_{UV}} = \Gamma_{\kappa_{UV}}(\Phi) = (S_{\text{bare}}[\phi^{(1)}], \dots, S_{\text{bare}}[\phi^{(N)}])^\top,$$

which plays the role of $\Gamma_{\kappa_0}^{\text{init}}(\Phi)$ in (5.14). The evolution of Y_κ is governed by the finite-dimensional ODE system (5.14), whose right-hand side is obtained by

evaluating the Wetterich–Morris flow with the surrogate $\Gamma_\kappa^\dagger$. In this setting, we obtain

$$\frac{d}{d\kappa} Y_\kappa^{(i)} = \frac{1}{2} \int_{x,y} \partial_\kappa R_\kappa(x, y) \left(\Gamma_\kappa^{\dagger(2)}[\phi^{(i)}] + R_\kappa \right)^{-1}(x, y) \, dx \, dy, \quad i = 1, \dots, N, \quad (5.15)$$

where the second functional derivative operator $\Gamma_\kappa^{\dagger(2)}$ is computed from the surrogate $\Gamma_\kappa^\dagger$.

A computable ODE system for the Wetterich–Morris equation. Next, our goal is to rewrite the right-hand side of (5.15) in an explicit, numerically tractable form.

In our GP framework all information about fields enters through the feature map w , which only retains their projection onto the truncated subspace $\mathcal{X}_P = \text{span}\{e_\alpha : \alpha \in \mathcal{A}_P\}$. It is therefore natural to evaluate the integral in (5.15) on the same finite-dimensional subspace. Concretely, we replace $\Gamma_\kappa^{\dagger(2)}[\phi]$ and R_κ by their restrictions to $\mathcal{X}_P$, represented in the basis $\{e_\alpha\}_{\alpha \in \mathcal{A}_P}$. This approximation is consistent with the truncation implicit in w and reduces the integration to a finite sum over the modes in $\mathcal{A}_P$.

For a fixed field $\phi \in \mathcal{U}$ and scale κ , the second functional derivative defines a linear operator $\Gamma_\kappa^{\dagger(2)}[\phi]$ on $\mathcal{X}$. Restricted to the truncated subspace $\mathcal{X}_P$, we represent the operator $\Gamma_\kappa^{\dagger(2)}[\phi]$ by the matrix $H_\kappa(\phi)$ with entries

$$H_\kappa(\phi)_{\alpha\beta} := \langle e_\alpha, \Gamma_\kappa^{\dagger(2)}[\phi] e_\beta \rangle_{\mathcal{X}}, \quad \alpha, \beta \in \mathcal{A}_P.$$

The matrix elements $H_\kappa(\phi)_{\alpha\beta}$ are computed from the surrogate $\Gamma_\kappa^\dagger$ by taking directional second derivatives along e_α and e_β ; see Section D.4 for the computations.

In line with standard FRG practice for field theory and critical phenomena, we restrict attention to translation-invariant regulators (see, e.g., Berges, Tetradis, and Wetterich, 2002; Delamotte, 2012). In the Laplacian eigenbasis $\{e_\alpha\}_{\alpha \in \mathcal{A}}$, with $-\Delta e_\alpha = \lambda_\alpha e_\alpha$, such regulators are diagonal and admit the spectral representation

$$R_\kappa(x, y) \approx \sum_{\alpha \in \mathcal{A}_P} r_\kappa(\lambda_\alpha) e_\alpha(x) e_\alpha(y),$$

for some scalar profile $\lambda \mapsto r_\kappa(\lambda)$, where the sum is restricted to the truncated mode set $\mathcal{A}_P$. Its scale derivative has the analogous expansion

$$\partial_\kappa R_\kappa(x, y) \approx \sum_{\alpha \in \mathcal{A}_P} \partial_\kappa r_\kappa(\lambda_\alpha) e_\alpha(x) e_\alpha(y).$$

On the truncated subspace $\mathcal{X}_P$ we represent $\Gamma_k^{\dagger(2)}[\phi] + R_k$ by the matrix $A_k(\phi)$ with entries

$$A_k(\phi)_{\alpha\beta} := H_k(\phi)_{\alpha\beta} + r_k(\lambda_\alpha) \delta_{\alpha\beta}, \quad \alpha, \beta \in \mathcal{A}_P,$$

and we denote its inverse by

$$B_k(\phi) := A_k(\phi)^{-1}.$$

In this basis, the inverse kernel

$$\mathcal{G}_k[\phi](x, y) := (\Gamma_k^{\dagger(2)}[\phi] + R_k)^{-1}(x, y)$$

is then approximated by

$$\mathcal{G}_k[\phi](x, y) \approx \sum_{\beta, \ell \in \mathcal{A}_P} B_k(\phi)_{\beta\ell} e_\beta(x) e_\ell(y).$$

Using orthonormality in $\mathcal{X}$, $\int e_\alpha(x) e_\beta(x) dx = \delta_{\alpha\beta}$ (and similarly in y), the approximation of (5.15) on $\mathcal{X}_P$ takes the form

$$\frac{d}{d\kappa} Y_k^{(i)} = \frac{1}{2} \sum_{\alpha \in \mathcal{A}_P} \partial_\kappa r_k(\lambda_\alpha) B_k(\phi^{(i)})_{\alpha\alpha}, \quad i = 1, \dots, N. \quad (5.16)$$

This ODE system is the numerical approximation of the flow (5.15) used in our computations. Together with the initial condition $Y_{\kappa_{UV}} = \Gamma_{\kappa_{UV}}(\Phi)$, this defines a closed ODE system for the collocated values Y_k . We integrate (5.16) backward in κ , from κ_{UV} down to κ_{IR} , using a standard ODE solver. At any intermediate scale, the surrogate $\Gamma_k^{\dagger}[\cdot]$ for arbitrary $\phi \in \mathcal{U}$ is recovered by inserting the current Y_k into (5.12).

5.3 Numerical results

We evaluate the proposed GP solver on FRG flows. The first example is an explicitly solvable benchmark: the exact quadratic solution of the Wilson–Polchinski equation for the Gaussian model (Subsection 5.3). This case is used to validate correctness under controlled conditions. Here, we vary the number and type of collocation functions, report errors both at the collocation set and on independently drawn test functions from a different distribution, and study convergence as the number of collocation functions increases.

The second experiment is carried out for the Wetterich equation on a periodic lattice. As documented in Zorbach et al., 2025, the results produced by the LPA when restricted to constant fields can differ from those obtained by lattice Monte Carlo (MC)

Zorbach et al., 2025 for certain choices of bare parameters. In one spatial dimension, both lattice Monte Carlo (MC) Zorbach et al., 2025 and transfer matrix (TM) Baxter, 2016 methods provide highly accurate reference solutions for some physical observables of the system. We therefore benchmark our method against LPA, lattice MC, and TM. Our primary accuracy metrics are three physical observables: the effective potential, the magnetization, and the susceptibility (see Appendix D.8). The magnetization is the field value that minimizes the infrared effective potential once a homogeneous source has tilted it by a linear term in the field, and it captures the system’s equilibrium response to that source. The susceptibility measures how rapidly this equilibrium magnetization changes as the source varies, serving as the local slope of the magnetization curve and characterizing the strength of the system’s linear response. In Subsection 5.3, we apply our GP method to solve the Wetterich equation. Here the GP prior differs from the one in the first case: it is derived from the LPA ansatz, which consists of a kinetic term and a local potential term. We model the local potential as a GP, and by the linearity of the ansatz in this potential, the full functional inherits a GP prior. This construction readily extends to other settings where a physical ansatz is available and can be encoded into the structure of the GP prior. Across the tests, our method produces magnetization and susceptibility curves that agree with the TM and MC references more closely than LPA over most of the sampled range, demonstrating systematic improvements.

Throughout the experiments, we study FRG flows on fixed finite spatial domains. For a discussion of the differences between finite-volume and infinite-volume formulations of the FRG, we refer the reader to Zorbach et al., 2025.

All experiments are implemented in Python with JAX Bradbury et al., 2018a. The resulting ODE systems are integrated using the LSODA algorithm to automatically handle both nonstiff and stiff regimes. Detailed solver settings for each problem, together with scripts to reproduce all experiments, are available in the public repository^{1,2}.

The Wilson–Polchinski equation: Gaussian-model

We consider the Wilson–Polchinski equation

$$\partial_k W_k[J] = \frac{1}{2} \int_0^1 \int_0^1 \partial_k R_k(x, y) \left(\frac{\delta^2 W_k[J]}{\delta J_x \delta J_y} + \frac{\delta W_k[J]}{\delta J_x} \frac{\delta W_k[J]}{\delta J_y} \right) dx dy, \quad (5.17)$$

¹<https://github.com/yangx0e/KernelsRenormalizationGroups>

²All calculations are performed on a MacBook Pro equipped with an Apple M1 Max processor and 64 GB of memory.

which admits the following explicit solution for a Gaussian free-field theory;

$$W_\kappa[J] = -\frac{1}{2} \int_0^1 \int_0^1 J(x) [\Gamma_2 + R_\kappa]^{-1}(x, y) J(y) dx dy - \frac{1}{2} \ln \frac{\det(\Gamma_2 + R_\kappa)}{\det \Gamma_2}, \quad (5.18)$$

with $\Gamma_2 = -\Delta + 1$. Here, the initial data for (5.17) at the UV scale κ_{UV} is prescribed by the explicit expression (5.18) evaluated at $\kappa = \kappa_{UV}$. We work on the periodic Sobolev space

$$H_{\text{per}}^1([0, 1]; \mathbb{R}) = \{J \in H^1([0, 1]; \mathbb{R}) : J(0) = J(1)\},$$

and choose the Litim regulator Litim, 2001

$$R_\kappa(p) = (\kappa^2 - p^2) \Theta(\kappa^2 - p^2),$$

which adds a mass-like term $\kappa^2 - p^2$ for Fourier modes with $p^2 < \kappa^2$ and vanishes otherwise. Here, we recall that Θ denotes the Heaviside step function, equal to 1 for positive arguments and 0 for nonpositive arguments.

For computations we truncate to the $(2P + 1)$ -dimensional Fourier subspace

$$V_P = \left\{ J(x) = \sum_{p=-P}^P c_p e^{2\pi i p x} : c_p \in \mathbb{R}, c_p = c_{-p} \right\} \subset H_{\text{per}}^1([0, 1]; \mathbb{R}).$$

Numerical setup

Table 5.1: Gaussian-Model Solution of the Wilson–Polchinski Equation: Average relative L^2 errors on collocation and test sets for varying sample sizes N .

	$N = 50$	$N = 100$	$N = 200$	$N = 500$	$N = 1000$
Collocation set	1.91×10^{-8}	2.51×10^{-9}	1.28×10^{-9}	8.29×10^{-10}	8.29×10^{-10}
Test set	5.10×10^{-8}	3.99×10^{-9}	1.46×10^{-9}	7.48×10^{-10}	6.59×10^{-10}

In this benchmark we fix $P = 20$. We compute the reference solution W_κ using the explicit formula (5.18). To study how the solver behaves on a broad collection of fields, and to check its ability to generalize beyond the inputs used to drive the flow, we work with a train-test split in function space. We first construct a random positive-definite covariance on Fourier coefficients by drawing a matrix $A \in \mathbb{R}^{(P+1) \times (P+1)}$ with i.i.d. standard normal entries and setting $\Sigma = A^\top A$. To generate the collocation functions, we draw $N = 1,000$ coefficient vectors $\{\hat{c}^{(i)}\}_{i=1}^N \subset \mathbb{R}^{2P+1}$ from $\mathcal{N}(0, \Sigma)$.

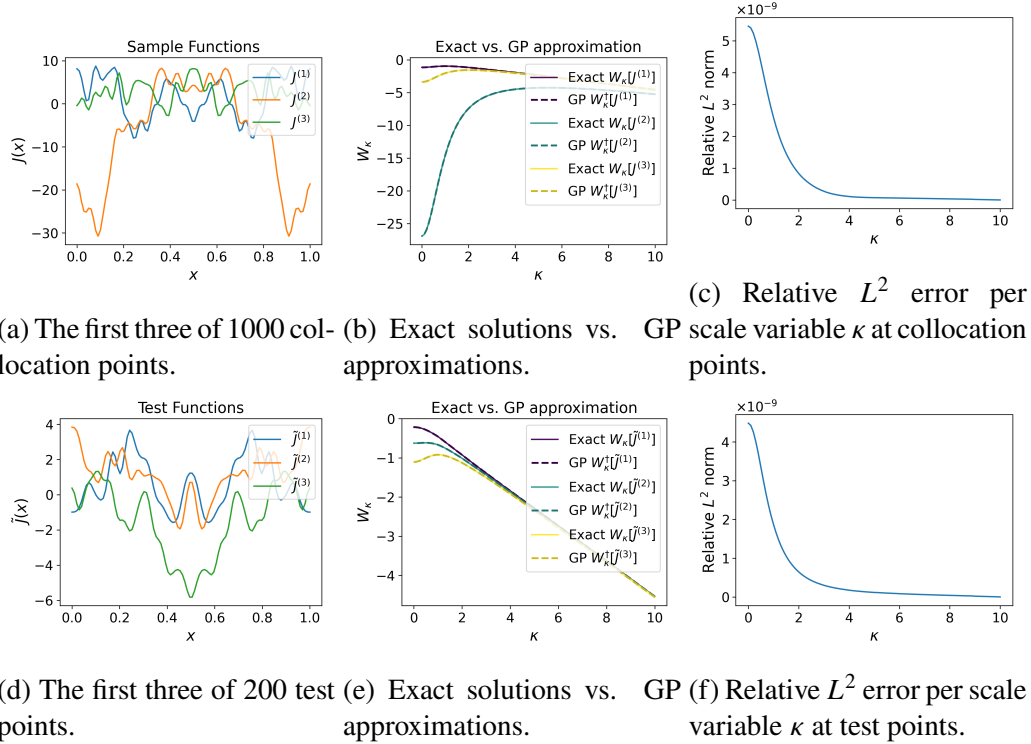


Figure 5.2: Numerical results for Gaussian-model solution of the Wilson–Polchinski equation. Subfigures (A)–(C) show results at collocation points, while (D)–(F) present corresponding results at testing points. For visualization, only the first 3 of 1,000 collocation points and the first 3 of 200 testing points are displayed. The relative L^2 error is evaluated for each scale variable κ .

We enforce reality by mirroring $\hat{c}_p \mapsto \hat{c}_{-p}$ and impose a mild high-frequency decay

$$c_p \leftarrow \begin{cases} \hat{c}_0, & p = 0, \\ \hat{c}_p/|p|, & p \neq 0, \end{cases}$$

so that higher Fourier modes are present but slightly damped. Each resulting vector defines a real field $J^{(i)}(x) = \sum_{p=-P}^P c_p^{(i)} e^{2\pi i p x} \in V_P$, yielding the collocation set $\mathbf{J} = \{J^{(1)}, \dots, J^{(N)}\} \subset V_P$.

To evaluate out-of-sample performance, we construct an independent test set of size $N_{\text{test}} = 200$ using the same symmetry and decay, but with coefficients $(\hat{c}_{-P}^{(j)}, \dots, \hat{c}_P^{(j)}) \sim \mathcal{N}(0, I)$ instead of the correlated covariance Σ . We denote these fields by $\tilde{\mathbf{J}} = \{\tilde{J}^{(j)}\}_{j=1}^{N_{\text{test}}}$. This independent ensemble plays the role of a hold-out set in function space and is used only for testing the generalization of the learned surrogate.

Since the Gaussian model solution depends quadratically on the Fourier coefficients,

we equip the field space on V_P with the quadratic feature-space kernel

$$k(J, J') = \sum_{p=-P}^P (c_p[J] c_p[J'] + 1)^2, \quad (5.19)$$

where $c_p[J]$ is the p th Fourier coefficient of J . For the GP method described in Section 5.2, a small nugget $\varepsilon = 10^{-12}$ is added to the Gram matrix for numerical stability. We denote the GP solution at scale κ by $W_\kappa^\dagger$.

The initial condition of the ODE system is computed from the explicit solution in (5.18) at the ultraviolet scale $\kappa_{UV} = 10$. We then integrate the associated ODE system in κ from κ_{UV} down to $\kappa_{IR} = 10^{-10}$. The trajectories are sampled at $N_T = 100$ uniformly spaced scales

$$\kappa_\ell = \kappa_{UV} + \ell \Delta\kappa, \quad \Delta\kappa = \frac{\kappa_{IR} - \kappa_{UV}}{N_T}, \quad \ell = 0, \dots, N_T.$$

At each κ_ℓ we evaluate the exact solution and the GP surrogate on both J and $\tilde{J}$. To measure accuracy we use relative discrete L^2 errors

$$\varepsilon_W^{\text{coll}}(\kappa_\ell) := \frac{\|W_{\kappa_\ell}(J) - W_{\kappa_\ell}^\dagger(J)\|_2}{\|W_{\kappa_\ell}(J)\|_2}, \quad \varepsilon_W^{\text{test}}(\kappa_\ell) := \frac{\|W_{\kappa_\ell}(\tilde{J}) - W_{\kappa_\ell}^\dagger(\tilde{J})\|_2}{\|W_{\kappa_\ell}(\tilde{J})\|_2},$$

and their averages over the scale grid,

$$\bar{\varepsilon}_W^{\text{coll}} = \frac{1}{N_T + 1} \sum_{\ell=0}^{N_T} \varepsilon_W^{\text{coll}}(\kappa_\ell), \quad \bar{\varepsilon}_W^{\text{test}} = \frac{1}{N_T + 1} \sum_{\ell=0}^{N_T} \varepsilon_W^{\text{test}}(\kappa_\ell).$$

Numerical results and discussion

Figure 5.2 summarizes the performance of the GP solver on this quadratic benchmark. Panel (A) shows three representative collocation fields generated by the sampling procedure above. Panel (B) compares, for the same three fields, the exact trajectories obtained from (5.18) with the GP surrogates. The curves are visually indistinguishable across the whole range, indicating that the GP surrogate $W^\dagger$ accurately captures the functional dependence on the RG scale.

Panel (C) reports the collocation relative errors $\varepsilon_W^{\text{coll}}(\kappa)$ as a function of κ . Panels (D)–(F) repeat the same diagnostics for three randomly chosen test fields and for the full test ensemble $\tilde{\Phi}$. Again, the pointwise trajectories of W_κ and $W_\kappa^\dagger$ coincide to plotting accuracy, and the test errors $\varepsilon_W^{\text{test}}(\kappa)$ match the collocation errors in magnitude, confirming out-of-sample generalization in function space.

Table 5.1 aggregates these results by listing $\bar{\varepsilon}_W^{\text{coll}}$ and $\bar{\varepsilon}_W^{\text{test}}$ for increasing numbers of collocation fields N . As N increases, the errors decay rapidly, providing numerical evidence for convergence. These results demonstrate that the proposed GP surrogate achieves accurate interpolation and generalization across the entire RG flow.

Wetterich–Morris equation: lattice ϕ^4 -model

We work on a one-dimensional periodic lattice with $N_x \in \mathbb{N}$ sites, lattice spacing equal to 1, and index set $\mathbb{T}_{N_x} := \mathbb{Z}/N_x\mathbb{Z}$ (indices taken modulo N_x). A lattice field is $\phi = (\phi_i)_{i \in \mathbb{T}_{N_x}} \in \mathbb{R}^{N_x}$. The forward difference and second-order discrete Laplacian are

$$(\nabla\phi)_i = \phi_{i+1} - \phi_i, \quad (-\Delta\phi)_i = 2\phi_i - \phi_{i+1} - \phi_{i-1}.$$

We use the discrete inner product $\langle u, v \rangle := \sum_{i \in \mathbb{T}_{N_x}} u_i v_i$ and the standard matrix trace $\text{Tr}(\cdot)$ on $\mathbb{R}^{N_x \times N_x}$.

The effective average action $\Gamma_\kappa[\phi]$ depends on the RG scale κ that flows backward from a large ultraviolet value $\kappa_{\text{UV}} \geq \pi$ to an infrared value $0 \leq \kappa_{\text{IR}} < \pi/N_x$. For each κ , a symmetric translation-invariant regulator $R_\kappa \in \mathbb{R}^{N_x \times N_x}$ is given and $\partial_\kappa R_\kappa$ denotes its scale derivative. The lattice Wetterich equation is

$$\partial_\kappa \Gamma_\kappa[\phi] = \frac{1}{2} \text{Tr} \left[(\Gamma_\kappa^{(2)}[\phi] + R_\kappa)^{-1} \partial_\kappa R_\kappa \right], \quad \Gamma_\kappa^{(2)}[\phi] := \frac{\delta^2 \Gamma_\kappa[\phi]}{\delta \phi \delta \phi} \in \mathbb{R}^{N_x \times N_x}. \quad (5.20)$$

At κ_{UV} , we prescribe the bare lattice ϕ^4 action

$$\Gamma_{\kappa_{\text{UV}}}[\phi] = \sum_{i \in \mathbb{T}_{N_x}} \left(\frac{1}{2} |(\nabla\phi)_i|^2 + \frac{m^2}{2} \phi_i^2 + \frac{\lambda}{24} \phi_i^4 \right), \quad m^2 \in \mathbb{R}, \lambda > 0. \quad (5.21)$$

To solve (5.20), we place a GP prior on Γ_κ , induced by the local ansatz

$$\Gamma_\kappa[\phi] = \sum_{i \in \mathbb{T}_{N_x}} \left(\frac{1}{2} |(\nabla\phi)_i|^2 + U_\kappa(\phi_i) \right), \quad (5.22)$$

where the scalar potential U_κ is modeled as a GP on a fixed field window $[-\phi_{\text{max}}, \phi_{\text{max}}]$. Hence, by linearity, Γ_κ inherits a GP prior induced by that on U_κ . Concretely, we decompose

$$U_\kappa(\phi) = m_0(\phi) + v_\kappa(\phi) + g_\kappa(\phi), \quad \phi \in [-L, L], \quad (5.23)$$

where

$$m_0(\phi) = \frac{1}{2} m^2 \phi^2 + \frac{\lambda}{24} \phi^4$$

is the bare ϕ^4 potential. We track the running quadratic and quartic couplings through a low-dimensional parametric term

$$v_\kappa(\phi) = \text{softplus}(\theta_{2,\kappa}) \phi^2/2 + \text{softplus}(\theta_{4,\kappa}) \phi^4/12, \quad \text{softplus}(\theta) = \ln(1 + e^\theta),$$

with latent parameters $\theta_\kappa = (\theta_{2,\kappa}, \theta_{4,\kappa})^\top \in \mathbb{R}^2$. Finally,

$$g_\kappa \sim \mathcal{GP}(0, K_U)$$

is a zero-mean GP remainder with fixed covariance kernel K_U on $[-\phi_{\max}, \phi_{\max}]$, capturing the residual nonpolynomial structure.

Jointly determining v_κ and g_κ from the flow at each scale would amount to solving a fully nonlinear constrained regression problem in function space at every κ , which is prohibitively expensive over long RG trajectories. Instead, we adopt a predictor-projector scheme that interleaves ODE integration with constrained reconstruction. At each scale step, we first form a predictor by replacing the nonlinear parametrization of v_κ with an unconstrained linear-Gaussian surrogate, yielding a closed, linearly parameterized curvature map U''_κ as a function of the sampled potential values. We then apply a projection (correction) step: we fit a constrained polynomial parametrization together with a GP remainder to the predicted potential and replace the predicted state by the reconstructed potential. The corrected potential is used as the initial condition for the next predictor step. This produces an interleaved algorithm that enforces the constraints along the RG evolution while keeping the per-step cost manageable. Implementation details of the predictor closure, the projection objective, and the numerical integrator are given in Appendix D.7. Here we only summarize the numerical setup and results.

Numerical setup.

We discretize the periodic lattice with $N_x = 32$ sites and take the field window $[-\phi_{\max}, \phi_{\max}]$ with $\phi_{\max} = 5.0$ on a uniform set of $N_\phi = 800$ constant-field collocation points, which is denoted by X . The remainder g_κ is modeled by a GP on $[-\phi_{\max}, \phi_{\max}]$ with the kernel K_U given explicitly by

$$K_U(\phi, \phi') = \sum_{q=0}^{Q_{\max}} \sigma^2 (\eta^2 + (\pi q / \phi_{\max})^2)^{-\beta} \cos\left(\frac{\pi q}{\phi_{\max}} \phi\right) \cos\left(\frac{\pi q}{\phi_{\max}} \phi'\right), \quad (5.24)$$

with hyperparameters $\sigma^2 = 1.0$, $\eta = 0.98$, $\beta = 2.0$, and spectral cutoff $Q_{\max} = \min\{N_\phi - 1, 1024\}$. The kernel Gram matrix is stabilized with a nugget $\epsilon = 10^{-11}$.

The kernel choice in (5.24) reflects that the effective potential U_κ should vary smoothly with the field, and that the model is even in the field, i.e., $U_\kappa(\phi) = U_\kappa(-\phi)$.

The flow is integrated backward in κ from $\kappa_{\text{UV}} = 100$ to $\kappa_{\text{IR}} = 10^{-10}$, and we record the solution at 1001 uniformly spaced output scales $\{\kappa_s\}_{s=0}^S$ with $S = 1000$. We initialize the constrained parameters at the UV scale by $\theta_{\kappa_{\text{UV}},2} = \theta_{\kappa_{\text{UV}},4} = -8.0$, so that $\text{softplus}(\theta) \approx 0$, and warm-start each subsequent projection at κ_{s+1} using the previous iterate. At each scale, the projected parameters are obtained by minimizing the projection objective (Appendix D.7) with Tikhonov weight $\gamma = 10^{-6}$ via a damped Gauss–Newton method (at most six iterations, unit damping). At the infrared scale κ_{IR} , we set $\theta_{\text{IR}} := \theta_{\kappa_S}$ and denote by g_{IR} the GP posterior mean from the final projection. Together these define the reconstructed infrared potential

$$U_{\text{IR}}(\phi) = m_0(\phi) + \text{softplus}(\theta_{\text{IR},2}) \phi^2/2 + \text{softplus}(\theta_{\text{IR},4}) \phi^4/12 + g_{\text{IR}}(\phi).$$

The reconstructed infrared potential $U_{\text{IR}}(\phi)$ is then used to extract observables (see Appendix D.8). In particular, the magnetization $m(c)$ is obtained by solving $U'_{\text{IR}}(m(c)) = c$ with the Newton method on $c \in [10^{-4}, 10^{0.4}]$ sampled at 18 logarithmic points, and the longitudinal susceptibility is given by $\chi(c) = 1/U''_{\text{IR}}(m(c))$.

Numerical results and discussion.

Figure 5.3 reports a five-panel comparison of our GP-based reconstruction against a standard local potential flow (LPA) (see Appendix D.5), lattice Monte Carlo (MC) Zorbach et al., 2025, and a transfer-matrix (TM) reference (see Appendix D.6). For the one-dimensional lattice ϕ^4 model with the parameters considered here, TM and high-precision lattice MC provide essentially state-of-the-art benchmarks for the magnetization $m(c)$ and the susceptibility $\chi(c)$ as functions of the external field c , see, e.g., Zorbach et al., 2025 and Appendix D.6. Panel (A) shows the infrared effective potential $U_{\kappa_{\text{IR}}}(\phi)$ on $[-5, 5]$. Panel (B) plots the susceptibility $\chi(c)$ on log-log axes, and panel (C) displays the corresponding pointwise absolute errors $|\chi(c) - \chi_{\text{TM}}(c)|$ relative to the TM reference. Panel (D) shows the magnetization $m(c)$ on log-log axes, while panel (E) reports the pointwise absolute errors $|m(c) - m_{\text{TM}}(c)|$. In panels (C) and (E), smaller values indicate closer agreement with the TM benchmark.

The study Zorbach et al., 2025 demonstrates that, for this model, the standard LPA flow already captures the qualitative behavior of $U_{\kappa_{\text{IR}}}(\phi)$, $m(c)$, and $\chi(c)$, as well

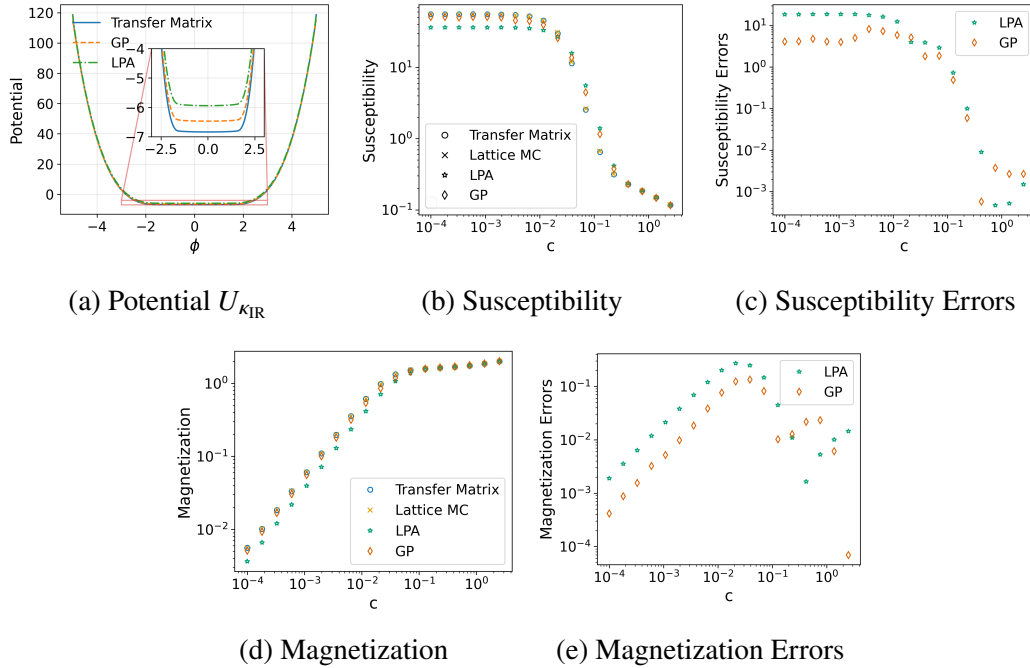


Figure 5.3: Benchmark of the GP reconstruction versus standard LPA, lattice Monte Carlo, and the transfer matrix method. (A) Values of the potential $U_{\kappa_{\text{IR}}}(\phi)$ on $[-5, 5]$ of different methods. (B) Susceptibility $\chi(c)$ versus external field c (log-log). (C) Susceptibility errors (relative to transfer matrix); for most values of c , the GP exhibits smaller errors than LPA, especially at small c . (D) Magnetization $m(c)$ (log-log). (E) Magnetization errors (relative to transfer matrix); in most of the c range the GP is more accurate than LPA.

known, but exhibits systematic quantitative deviations when compared to lattice MC. These discrepancies originate from the fact that the truncation keeps only a local potential $U_\kappa(\phi)$ and neglects momentum-dependent corrections in $\Gamma_\kappa^{(2)}$, including wave-function renormalization and higher-derivative operators. As a result, the local flow misrepresents how fluctuations redistribute curvature in field space, and this manifests as a visible gap between the LPA and the lattice MC/TM results.

Our GP reconstruction is built on the same local ansatz for Γ_κ that underlies the standard flow, but it replaces a finite low-order polynomial parameterization of U_κ by a physics-informed GP prior. Concretely, we encode the bare ϕ^4 potential in the mean function m_0 , represent the running quadratic and quartic couplings in a low-dimensional parametric component v_κ , and model the remaining nonpolynomial structure by a zero-mean GP remainder g_κ with a covariance kernel K_U derived from a spectral decomposition of the lattice kinetic operator along constant fields. In this way, the GP kernel is not a generic smoothness prior but a kernel engineered

from the RG ansatz.

Using TM as the reference, the pointwise errors of $m(c)$ and $\chi(c)$ obtained from $U_{\kappa_{\text{IR}}}$ are reduced for the GP reconstruction compared to the purely LPA flow on most values of c . For LPA, the computation requires 27.90 seconds, whereas for the GP method it requires 59.95 seconds. Conceptually, this illustrates how physics-driven kernel design can be used to refine the effective description of Γ_κ along selected observables: an ansatz or additional physical information (symmetries or invariances, etc.) for Γ_κ can be incorporated into the construction of GP priors.

5.4 Conclusion

We have introduced a general framework for solving functional renormalization group (FRG) equations directly in function space, leveraging tools from operator learning. The method is both general and flexible, and can be applied to a broad class of FRG problems. We have demonstrated that it accurately reproduces several flows of interest, while also revealing multiple directions for further development.

On the methodological side, we have worked with a fixed kernel when solving the FRG equations. Kernel-learning techniques such as cross-validation, maximum likelihood estimation (MLE), or maximum a posteriori (MAP) estimation Nicholas H Nelsen, Houman Owhadi, et al., 2025; Y. Chen, Houman Owhadi, and Andrew M. Stuart, 2021; Hastie, Tibshirani, and Friedman, 2009; Rasmussen and Williams, 2006 could be used to tune kernel hyperparameters and improve accuracy. In addition, approaches based on programmable kernels Houman Owhadi, Scovel, and Gene Ryan Yoo, 2021b; Th'eo Bourdais et al., 2024 may help uncover structural properties of the solution. Once additional physical insight into the underlying FRG solution is available, this knowledge can be encoded directly into the GP kernel.

On the theoretical side, kernel-based methods open the door to rigorous error analysis. Although we do not pursue this here, error estimates analogous to those in Batlle, Matthieu Darcy, Hosseini, et al., 2024b could be derived in our setting. The encoder-decoder structure of diagram in Figure 5.1 and the optimal-recovery viewpoint of Remark 12 were first used in the operator-learning analyses of Batlle, Matthieu Darcy, Hosseini, et al., 2024b to provide convergence guarantees and rigorous a priori error estimates. We do not pursue such convergence analysis here, but note that it could be used in conjunction with an error analysis of time-stepping schemes to provide detailed error analysis of the recovery of Γ , and instead leave it for future work. In contrast to purely data-driven settings as in Batlle, Matthieu

Darcy, Hosseini, et al., 2024b where evaluations of Γ are given on many input functions, our surrogate $\Gamma^\dagger$ is constrained to satisfy the FRG flow equations. As a result, the representation is informed by the underlying functional PDE rather than by paired input-output data. Moreover, the Gaussian process viewpoint naturally provides uncertainty quantification and a Bayesian interpretation, which could be exploited in inverse problems associated with functional differential equations.

In terms of applications, the proposed framework is particularly promising in situations where traditional FRG techniques are difficult to use. Potential targets include large deviation problems for dynamical systems and phenomena such as spontaneous stochasticity. Our experiments also show that the GP method can handle inhomogeneous fields, suggesting extensions to more complex field theories, for instance the study of kink solutions in the ϕ^4 model.

*Chapter 6***OPERATOR LEARNING AT MACHINE PRECISION**

6.1 Introduction

Operator learning Boulle and Townsend, 2024; Nikola B Kovachki, Lanthaler, and Andrew M Stuart, 2024; Batlle, Matthieu Darcy, Hosseini, et al., 2024c is the problem of approximating, from limited data, an infinite-dimensional mapping $\mathcal{G} : \mathcal{U} \rightarrow \mathcal{V}$ between Banach spaces $\mathcal{U}$ and $\mathcal{V}$. Often, $\mathcal{G}$ may be implicitly defined by another operator $\mathcal{F}$ such that $\mathcal{F}(u, \mathcal{G}(u)) = 0$. Operator learning has attracted significant attention in scientific computing and has become one of the core problems of physics-informed machine learning (PIML) G. E. Karniadakis, Ioannis G Kevrekidis, et al., 2021a for learning equations Steven L Brunton, Joshua L Proctor, and J Nathan Kutz, 2016a; Rudy et al., 2019; Jalalian et al., 2025, and solving Partial Differential Equations (PDEs) and inverse problems Cuomo et al., 2022; Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b; Francesco Bartolucci et al., 2024; Molinaro et al., 2023. Among the most well-known operator learning models are Deep Operator Nets (DeepONet) L. Lu, Jin, Pang, et al., 2021b and Fourier Neural Operators (FNO) Zongyi Li, N. Kovachki, et al., 2020b. While differing in their parameterizations, operator learning methods are typically designed to learn discretized approximations of operators between function spaces from empirical data.

We consider two main categories of operator learning problems:

- **Category 1:** The target operator is available only through input-output pairs, and the governing equations or boundary conditions are unknown or incomplete. This often necessitates a purely data-driven approach.
- **Category 2:** The target operator is known to satisfy a set of constraints, typically in the form of a PDE with appropriate boundary conditions.

In the context of **Category 2**, physics-informed operator learning methods Goswami et al., 2023 are often used to encode the constraints into an accelerated emulator for the PDE solution. While these approaches can provide substantial acceleration, to the best of our knowledge existing methods cannot match the accuracy of the numerical solvers they emulate.

Summary of contributions

In this article we focus on operator learning problems under **Category 2** and make four key contributions:

- **Introduction of CHONKNORIS:** We propose a novel neural operator learning method inspired by the Newton–Kantorovich method Polyak, 2006. The CHONKNORIS model explicitly learns the dependence of the Cholesky factor $\mathcal{R}$ of the Gauss-Newton Hessian matrix $(\lambda I + \mathcal{F}'(u, v)^* \mathcal{F}(u, v))^{-1}$ on the approximate PDE solution v and the random coefficients u . Integrating CHONKNORIS Cholesky factor predictions $\hat{\mathcal{R}}$ into a quasi-Newton iterative algorithm enables us to emulate PDE solvers to machine precision error.
- **Generalization via FONKNORIS:** Building upon CHONKNORIS, we introduce FONKNORIS, a foundation modeling framework which learns the dependence of the Cholesky factors $\mathcal{R}$ on the approximate PDE solution v and a set of coefficients functions u common to a large class of PDEs. The enhanced generalization of FONKNORIS enables machine precision recovery of PDEs not seen during training.
- **Enhanced Accuracy and Interpretability:** By explicitly embedding the underlying physics into a kernel interpolation model or a neural network architecture resembling ResNet / Transformer layers, CHONKNORIS significantly surpasses the accuracy limitations of existing operator learning frameworks while providing improved interpretability through the physically meaningful structure of each iteration.
- **Theoretical Guarantees.** We establish an inexact Newton–Kantorovich analysis for the *learned* Tikhonov inverse approximate Hessian, yielding rigorous Kantorovich-style convergence guarantees. The theoretical result is instantiated on a nonlinear elliptic PDE, where we explicitly compute the design error for a kernel-based approximation of the learned Cholesky factors.
- **Comprehensive Experimental Validation:** Extensive numerical experiments demonstrate that CHONKNORIS and FONKNORIS are capable of machine precision recovery across a wide variety of forward and inverse nonlinear PDE problems. These include a nonlinear elliptic equation, the Burgers' equation, a nonlinear Darcy flow equation, Calderón's inverse problem, an inverse wave scattering problem, a seismic imaging full waveform inversion problem, and a FONKNORIS generalization to the Klein–Gordon and Sine–Gordon equations.
- **Benchmarking against operator learners.** Compared to strong baselines (kernel methods, Fourier Neural Operators, and Transformer-based Neural

Operators), our approach lowers typical relative errors from around 10^{-3} or 10^{-2} to near 10^{-16} (machine precision) on benchmark forward and inverse problems.

Finally we highlight the practical and scientific use cases of CHONKNORIS and FONKNORIS algorithms:

- **Known physics.** When the residual map $\mathcal{F}(u, v) = 0$ is specified (forward or inverse settings) and high accuracy is required, the solver-emulating updates attain machine precision accuracy in practice, substantially exceeding typical operator-learning baselines.
- **Reliability.** CHONKNORIS is equipped with theoretical guarantees (e.g., convergence and stability under standard assumptions), yielding reproducible and dependable results.
- **Controllable accuracy and cost.** Accuracy is governed by the iteration budget; per-iteration complexity is dominated by Jacobian actions in the form of two triangular matrix–vector products.
- **Cross-PDE transfer.** Our foundation model, FONKNORIS, learns a single mapping from operator coefficients to Cholesky factors, enabling transfer across PDE families and generalization to previously unseen equations without retraining.

Brief review of the relevant literature

Methods for operator learning can be broadly categorized into three main groups: (1) Artificial neural network (ANN)-based approaches L. Lu, Jin, Pang, et al., 2021b; Zongyi Li, N. Kovachki, et al., 2020b; N. Kovachki, Zongyi Li, et al., 2023, (2) kernel-based methods Kadri, Duflos, Preux, Stephane Canu, et al., 2016a; Nicholas H Nelsen and Andrew M Stuart, 2021; Batlle, Matthieu Darcy, Hosseini, et al., 2024c, and (3) hybrid approaches Houman Owhadi and Gene Ryan Yoo, 2019a; Houman Owhadi, 2023a; Mora et al., 2025. Kernel-based methods are well-established, offering strong theoretical foundations and convergence guarantees. Theoretical support for ANN-based methods has been developed more recently Nikola B Kovachki, Lanthaler, and Andrew M Stuart, 2024; De Ryck and Mishra, 2024; Mishra and Molinaro, 2022; Reinhardt, Sven Wang, and Zech, 2024. ANN

methods have become popular as they can benefit from advanced hardware and software ecosystems, enabling efficient scaling to large datasets. When $\mathcal{G}$ corresponds to the solution operator of a PDE, learning $\mathcal{G}$ through evaluations of $\mathcal{F}$ is referred to as physics-informed operator learning Sifan Wang, H. Wang, and Paris Perdikaris, 2021b; Goswami et al., 2023; Zongyi Li, Zheng, et al., 2024b. These methods typically incorporate a loss term to enforce the consistency of the learned operator with the underlying PDE. As mentioned earlier, a common issue with operator learning methods is their limitations in terms of accuracy. For example, in tasks such as mapping diffusion coefficients to solutions in second order elliptic PDEs a relative L^2 error of around 0.1% is achieved on benchmark data sets, even in the physics-informed setting Kissas et al., 2022; He, X. Liu, and Xu, 2024; Zongyi Li, Zheng, et al., 2024b. Adjacent to the above works, the recent papers Long et al., 2024; Jalalian et al., 2025 explored operator learning via equation learning where $\mathcal{F}$ is learned from data and then inverted numerically to estimate $\mathcal{G}$ and achieved significant improvements in accuracy and data efficiency. However, these methods are not true emulators since every evaluation of the learned operator requires the numerical solution of a nonlinear PDE. In parallel, there is a growing interest in *foundation models* for operator learning, where large neural operators are pre-trained across families of PDEs and subsequently adapted to solve new unseen PDEs or tasks: see e.g. Bodnar et al., 2025; Herde et al., 2024; Z. Ye et al., 2024.

These results raise a compelling question:

Can operator learning for PDE problems achieve machine precision or at least approach it, if we incorporate explicit knowledge that the underlying map $\mathcal{G}$ is fully determined by known constraints?

The recent works He, X. Liu, and Xu, 2024; W. Hao, X. Liu, and Y. Yang, 2024; Fabiani et al., 2025; Georgiou, Siettos, and Yannacopoulos, 2025a; Georgiou, Siettos, and Yannacopoulos, 2025b introduced operator learning techniques based on traditional numerical algorithms. In He, X. Liu, and Xu, 2024, the authors propose a multi-grid operator for solving linear PDEs while W. Hao, X. Liu, and Y. Yang, 2024; Fabiani et al., 2025 introduce an operator learning algorithm that emulates the iterative updates of Newton’s method for solving PDEs. In Georgiou, Siettos, and Yannacopoulos, 2025a; Georgiou, Siettos, and Yannacopoulos, 2025b, the authors introduce Fredholm neural networks, which solve Fredholm integral equations of the

second kind by unrolling an iterative fixed-point scheme into a feed-forward architecture. Several PDEs, including the Helmholtz equation, admit such formulations. In Georgiou, Siettos, and Yannacopoulos, 2025b, they report small interior errors and near machine precision accuracy on the boundary for selected benchmarks. By contrast, Fabiani et al., 2025 proposes *RandONets*: shallow, one-hidden-layer operator networks that first embed inputs via random projections and then learn only the output weights with linear solvers. For certain linear operators with aligned data, this approach attains near machine precision. The work W. Hao, X. Liu, and Y. Yang, 2024 is the closest method to ours. Methodologically, the method in W. Hao, X. Liu, and Y. Yang, 2024 learns a neural operator that directly approximates the nonlinear Newton update map and obtains a solution by iterating this learned map, guided by a Newton-informed loss. In contrast, our CHONKNORIS framework regresses Cholesky factors of the underlying linear elliptic operators and hard-wires a regularized Newton–Kantorovich residual iteration as the network architecture. The CHONKNORIS framework therefore does not approximate the full nonlinear Newton step, but only the linear elliptic operator entering the Newton–Kantorovich linearization. Under standard assumptions ensuring that the Newton–Kantorovich iteration based on the exact elliptic operator is contractive, and provided that the learned operator is sufficiently close to this exact operator so as to preserve contraction, increasing the iteration budget systematically reduces the error down to the accuracy level dictated by the underlying numerical discretization.

Gauss–Newton algorithms, along with other quasi-Newton algorithms as well as their function space extensions, the Newton–Kantorovich method, have been widely used for solving nonlinear PDEs Quarteroni and Valli, 1994; Zeidler, 1986 and inverse problems Haber, Ascher, and Oldenburg, 2000; Haber, 2004. Such algorithms are also central to the design of recent RKHS methods for solving PDEs Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b as well as boosting Physics Informed Neural Net (PINN)-type methods Jnini, Vella, and Zeinhofer, 2024. The wide applicability and simple abstract formulation of quasi-Newton algorithms suggests that they can be emulated or approximated using Machine Learning (ML) models, further motivating our exposition in light of algorithm unrolling ideas in the design of ANN architectures Monga, Y. Li, and Eldar, 2021; Q. Yang, Sadeghi, and G. Wang, 2022.

Finally, we note that operator learning is deeply related to older ideas in scientific computing and applied mathematics such as computer model emulation Kennedy

and O’Hagan, 2001, operator compression Feischl and Peterseim, 2020, polynomial chaos expansions Xiu and G. E. Karniadakis, 2002b, and model order reduction Schilders, Van der Vorst, and Rommes, 2008 among others. For more historical remarks in this direction we refer the reader to the literature review in Batlle, Matthieu Darcy, Hosseini, et al., 2024c.

Outline

The remainder of this article is organized as follows. Section 6.2 details the operator learning problem (Section 6.2), the Newton–Kantorovich method (Section 6.2), the Newton–Kantorovich method with Tikhonov regularization (Section 6.2), the proposed CHONKNORIS method (Section 6.2), and the proposed FONKNORIS foundation model (Section 6.2). Section 6.3 details our numerical experiments including benchmarking CHONKNORIS against existing operator learning methods (Section 6.3), forward problem modeling with CHONKNORIS (Section 6.3), foundation modeling with FONKNORIS (Section 6.3), and inverse problem modeling with CHONKNORIS (Section 6.3). Section 6.4 provides theoretical guarantees of the CHONKNORIS method. Finally, Section 6.5 gives a brief conclusion and outlook of future work.

6.2 Operator Learning

The operator learning problem

Let $\mathcal{U}$ and $\mathcal{V}$ be two separable Banach function spaces. We are interested in learning the operator $\mathcal{G} : \mathcal{U} \rightarrow \mathcal{V}$ that is implicitly defined by $\mathcal{F} : \mathcal{U} \times \mathcal{V} \rightarrow \mathcal{W}$:

$$\mathcal{G} : u \mapsto \mathcal{G}(u) := v \quad \text{such that} \quad \mathcal{F}(u, \mathcal{G}(u)) = 0 \quad \text{for all } u \in \mathcal{U}.$$

In this work, we assume to know and have full access to the operator $\mathcal{F}$, and thus to its Fréchet derivative $\frac{\delta \mathcal{F}}{\delta v} : \mathcal{U} \times \mathcal{V} \rightarrow \mathcal{L}(\mathcal{V}, \mathcal{W})$ as well as its adjoint $\frac{\delta \mathcal{F}}{\delta v}^* : \mathcal{U} \times \mathcal{V} \rightarrow \mathcal{L}(\mathcal{W}, \mathcal{V})$.

Nonlinear elliptic PDE as a running example

As a running example, we consider the following nonlinear elliptic equation supplemented with periodic boundary conditions. We will use this PDE throughout the paper to ground our ideas and provide intuition for our abstract derivations:

$$\begin{cases} -\Delta v(x) + \kappa v(x)^3 = u(x) & \text{for } x \in (0, 1) \\ v(0) = v(1) \end{cases}. \quad (6.1)$$

The corresponding differential operator and its derivative are

$$\mathcal{F}(u, v) = -\Delta v + \kappa v^3 - u \quad \text{and} \quad \left[\frac{\delta \mathcal{F}}{\delta v}(u, v) \right] (h) = [-\Delta + 3\kappa v^2](h) \quad (6.2)$$

respectively. Our task is to learn an approximation of $\mathcal{G}$ such that for $v = \mathcal{G}(u)$ the above equation is satisfied in some suitable sense, e.g., in the classical or weak sense. In other words, we aim to learn the solution operator that maps the right hand side u to the solution v of (6.1).

A suitable analytical framework may set $\mathcal{U} = \mathcal{W} = C_{per}(\mathbb{T})$, the space of periodic continuous functions on $\mathbb{T}$, and either $\mathcal{V} = C_{per}^2(\mathbb{T})$, the space of periodic twice differentiable functions on Ω for classical solutions, or $\mathcal{V} := H_{per,0}^1(\mathbb{T}) = \left\{ u \in H_{per}^1(\mathbb{T}) : \int_0^1 u = 0 \right\}$ for weak solutions where $W := V^* \simeq H_{per,0}^{-1}(\mathbb{T})$ with $\|u\|_V := \|u'\|_{L^2(0,1)}$.

The Newton–Kantorovich method

The Newton–Kantorovich (NK) method Polyak, 2006 is a generalization of Newton’s method to Banach spaces. First, we outline the method with the nonlinear elliptic PDE defined in (6.1) and (6.2). To solve this equation, we start from an initial guess v_0 , and iteratively perturb the approximation using successive linearizations around each estimate. Writing v_n for the current estimate and $v_{n+1} := v_n + \delta v_n$ for the next estimate, we have

$$\begin{aligned} 0 &= -\Delta(v_n + \delta v_n) + \kappa(v_n + \delta v_n)^3 - u \\ &= [-\Delta(v_n) + \kappa(v_n)^3 - u] + (-\Delta\delta v_n + 3\kappa v_n^2\delta v_n) + o(\delta v_n) \\ &= \mathcal{F}(u, v_n) + (-\Delta + 3\kappa v_n^2)\delta v_n + o(\delta v_n). \end{aligned}$$

Ignoring higher orders of approximation, this defines the update δv_n as the solution of a v_n -dependent linear PDE. Our critical insight is that learning to solve this linearized PDE is much simpler than directly solving the original nonlinear PDE. This methodology can be generalized to any operator equation $\mathcal{F}(u, v) = 0$. We have

$$0 = \mathcal{F}(u, v_n + \delta v_n) \approx \mathcal{F}(u, v_n) + \left[\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right] (\delta v_n), \quad (6.3)$$

where again $\frac{\delta \mathcal{F}}{\delta v}$ is the Fréchet derivative of $\mathcal{F}$. The NK method proceeds to find δv_n by solving the linear problem in (6.3) giving

$$\delta v_n = - \left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right)^{-1} \mathcal{F}(u, v_n). \quad (6.4)$$

Therefore to emulate the NK flow, it is sufficient to learn the term $\left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right)^{-1}$.

The Newton–Kantorovich method with Tikhonov regularization

Limited data and the potential ill-conditioning of the operator $\frac{\delta \mathcal{F}}{\delta v}$ can lead to inaccuracies and instabilities if we try to directly use the iteration in (6.4). To address this issue, it is necessary to regularize the problem. We therefore consider a Tikhonov-regularized variant of the NK method by defining

$$\delta v_n = \arg \min_{\delta v \in \mathcal{V}} \left\{ \left\| \mathcal{F}(u, v_n) + \frac{\delta \mathcal{F}}{\delta v}(u, v_n)[\delta v] \right\|_{\mathcal{W}}^2 + \lambda \|\delta v\|_{\mathcal{V}}^2 \right\}, \quad (6.5)$$

where $\lambda > 0$ is the regularization parameter that balances robustness and accuracy; larger values of λ slow down convergence and improve robustness while smaller values lead to faster algorithms that are less stable. Assuming that the underlying spaces are Hilbert, the first-order optimality condition for (6.5) yields a (linear) normal equations that can be solved explicitly. This results in the following iterative scheme

$$\delta v_n = - \left(\left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right)^* \frac{\delta \mathcal{F}}{\delta v}(u, v_n) + \lambda I \right)^{-1} \left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right)^* \mathcal{F}(u, v_n), \quad (6.6)$$

$$v_{n+1} = v_n + \delta v_n, \quad (6.7)$$

where $\frac{\delta \mathcal{F}}{\delta v}^*$ denotes the Hilbert adjoint of the Fréchet derivative of $\mathcal{F}$. This is precisely the Hilbert-space analogue of the Levenberg–Marquardt step Nocedal and Stephen J Wright, 2006a. We note that this iterative scheme leads to a connection with the attention mechanism from transformers Vaswani et al., 2017; Calvello et al., 2024 that we highlight in Section E.7, leaving further investigation to future work.

In Banach spaces, a (unique) minimizer of (6.5) still exists under standard assumptions (e.g., uniform convexity/smoothness and suitable conditions on $\mathcal{F}$). The corresponding optimality conditions replaces inner products by duality pairings and involves the duality mappings J_V and J_W , leading to a nonlinear equation of the form

$$\left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right)^* J_W \left(\mathcal{F}(u, v_n) + \frac{\delta \mathcal{F}}{\delta v}(u, v_n) \delta v_n \right) + \lambda J_V(\delta v_n) = 0,$$

which can be solved by standard monotone-operator or Newton-type methods. For clarity, we confine our analysis to the Hilbert-space setting.

Since we assumed that the adjoint operator $\frac{\delta \mathcal{F}}{\delta v}^*$ is known, the most significant computational burden in computing the update δv_n is, in general, to compute $Q(u, v_n) = \left(\left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right)^* \frac{\delta \mathcal{F}}{\delta v}(u, v_n) + \lambda I \right)^{-1}$. For that reason, we only learn the parametric operator $(u, v) \mapsto Q(u, v)$. This operator is symmetric, positive definite and linear, but it depends nonlinearly on both u and v_n . The next remark justifies learning Q , as it can be done efficiently.

Remark 13 (Reduction to parametric elliptic operator learning). *Let us write $\mathcal{L}_+(\mathcal{V}, \mathcal{V})$ for the space of symmetric positive linear operators from $\mathcal{V}$ to $\mathcal{V}$. We notice that Q is indeed a mapping from $\mathcal{U} \times \mathcal{V}$ to $\mathcal{L}_+(\mathcal{V}, \mathcal{V})$. Then, assuming that the adjoint operator $\frac{\delta \mathcal{F}}{\delta v}^*$ is known, we can compute $Q \frac{\delta \mathcal{F}}{\delta v}^*$. The operator Q^{-1} is indeed an elliptic differential operator whose solution map Q can be learned efficiently Schafer and Houman Owhadi, 2024; J. Chen, Schafer, and Desbrun, 2024.*

The subsequent remark shows that our analogue of the Levenberg–Marquardt algorithm, as in the finite-dimensional case, interpolates between gradient descent and the Newton–Kantorovich method.

Remark 14 (Interpolation between Gradient Descent and NK). *In the Hilbert space setting, we can formally write*

$$\left\| \mathcal{F}(u, v_n) + \left[\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right] (\delta v) \right\|_{\mathcal{W}}^2 = \|\mathcal{F}(u, v_n)\|_{\mathcal{W}}^2 + 2 \left\langle \delta v, \left[\frac{\delta \mathcal{F}}{\delta v}(u, v_n) \right]^* \mathcal{F}(u, v_n) \right\rangle_{\mathcal{V}} + O(\|\delta v\|_{\mathcal{V}}^2), \quad (6.8)$$

and see that δv_n given by (6.6) approximates the NK increment (6.4) as $\lambda \downarrow 0$ while it is proportional to a gradient descent update for the left-hand side of (6.8) as $\lambda \uparrow \infty$. Therefore, the regularization parameter λ allows us to interpolate between the NK algorithm and gradient descent, enabling a controlled balance between convergence stability and update accuracy.

CHONKNORIS

We are now ready to introduce CHONKNORIS based on the idea of using operator learning to efficiently learn the solution operator Q as mentioned in Remark 13. More precisely, CHONKNORIS uses the same update rule as in (6.7), but replaces the costly operator inversion in the definition of Q with a learned approximation. In practice, we work with an arbitrary discretization of input and output spaces $\mathcal{U}$, $\mathcal{V}$, and $\mathcal{W}$. This discretization can be derived from various numerical methods such as finite-elements, finite-differences, spectral methods, or any other discretization method. Thus, choosing an appropriate parametrization of our input and output functions, CHONKNORIS can be made discretization-invariant following the optimal recovery approach in Batlle, Matthieu Darcy, Hosseini, et al., 2024c or the Fourier neural operator (FNO) framework in Zongyi Li, N. Kovachki, et al., 2020b. Consequently, $Q(u, v_n)$ is discretized as a positive definite matrix. We choose to approximate the Cholesky factors of Q to enforce positivity, which stabilizes learn-

ing, guarantees descent directions, reduces the number of learned parameters, and allows for efficient triangular solves. We train a surrogate operator $\hat{\mathcal{R}}$, parameterized for instance as a neural operator or a kernel-based model, such that

$$\hat{\mathcal{R}}(u, v)\hat{\mathcal{R}}(u, v)^T \approx Q(u, v) = \left[\left(\frac{\delta \mathcal{F}}{\delta v}(u, v) \right)^* \left(\frac{\delta \mathcal{F}}{\delta v}(u, v) \right) + \lambda I \right]^{-1} \quad (6.9)$$

$$\hat{\mathcal{R}}(u, v) \text{ is upper triangular.} \quad (6.10)$$

To that end, we use a traditional NK solver to generate training data for our method. Using input data $u^{(1)}, \dots, u^{(M)} \sim \mu$ that are sampled from a probability measure μ supported on $\mathcal{U}$, we run the NK solver for $n_{\text{warm}} \in \mathbb{N}$ steps to generate flow data $v_k^{(m)}$ where $m = 1, \dots, M$ and $k = 0, \dots, n_{\text{warm}}$. This flow data lives along the *true* NK method trajectory to mitigate the curse of dimensionality and avoid generating data that is not seen in the NK iteration scheme. We can then compute the Cholesky factors of $Q(u^{(m)}, v_k^{(m)})$ and train $\hat{\mathcal{R}}$ to approximate them using a the usual regularized mean squared error loss. Note that it is also possible to vary the Tikhonov regularization λ , introducing an additional dependency for $\hat{\mathcal{R}}$, i.e., $\hat{\mathcal{R}}(u, v, \lambda)$. This is particularly useful for ill-posed problems, such as inverse problems, to speed up the convergence and obtaining higher accuracy.

Once we have trained $\hat{\mathcal{R}}$, the CHONKNORIS approximation is $\hat{\mathcal{G}} : u \mapsto \hat{v}_N$, where

$$\hat{v}_{n+1} = \hat{v}_n - \alpha_n \hat{\mathcal{R}}(u, \hat{v}_n, \lambda_n) \hat{\mathcal{R}}(u, \hat{v}_n, \lambda_n)^T \left(\frac{\delta \mathcal{F}}{\delta v}(u, \hat{v}_n) \right)^* \mathcal{F}(u, \hat{v}_n), \quad n = 0, \dots, N-1. \quad (6.11)$$

For a sufficiently large number of inference iterations N , we expect $\hat{\mathcal{G}}(u) = \hat{v}_N \approx \mathcal{G}(u)$. Here, α_n is a learning rate schedule and λ_n the Tikhonov regularization schedule. Both schedules are chosen using line-search. Thus, CHONKNORIS is an emulator of the *true* NK method. We summarize the resulting algorithm in Algorithm 2 and give a visual depiction of the post-training operator in Figure 6.1.

FONKNORIS

In this section, we introduce FONKNORIS, a foundational model variant of CHONKNORIS. We illustrate this method using our running example of a nonlinear elliptic PDE. First, we notice that in 1D the Fréchet derivative of the nonlinear elliptic PDE in (6.2), $\frac{\delta \mathcal{F}}{\delta v}(u, v) = (-\partial_{xx} + 3\kappa v^2)$, can be expressed as

$$\left[\frac{\delta \mathcal{F}}{\delta v}(u, v) \right] (h)(x) = [a(u, v)\partial_{xx} + b(u, v)\partial_x + c(u, v)] h(x), \quad (6.12)$$

Algorithm 2 CHONKNORIS/FONKNORIS

- 1: **Inputs:** training data size M ; residual map $\mathcal{F}$; Jacobian $J(u, v) = \frac{\delta \mathcal{F}}{\delta v}(u, v)$; adjoint J^* ; measure μ ; NK warm-up steps n_{warm} ; model $\hat{\mathcal{R}}_\theta$ (lower-triangular, $\text{diag} > 0$); flow relaxation λ_{flow} ; training relaxation set λ_{train}
 - 2: **Mode:** choose *parameterization* Φ and *Jacobian builder* BuildJ:
 - CHONKNORIS: $\Phi(u, v) = (u, v)$, BuildJ(u, v) = $J(u, v)$
 - FONKNORIS: $\Phi(u, v) = (a(u, v), b(u, v), c(u, v))$, BuildJ(a, b, c) = $a \partial_{xx} + b \partial_x + c$
 - 3: **Data (offline):**
 - 4: **for** $m = 1, \dots, M$ **do**
 - 5: sample $u^{(m)} \sim \mu$
 - 6: initialize v_0
 - 7: **for** $i = 0, \dots, n_{\text{warm}}$ **do** ▷ solver warmup
 - 8: $z_i \leftarrow \Phi(u^{(m)}, v_i)$ ▷ $z_i = (u^{(m)}, v_i)$ in CHONKNORIS, or $z_i = (a, b, c)$ in FONKNORIS
 - 9: $J_i \leftarrow \text{BuildJ}(z_i)$
 - 10: $v_{i+1} \leftarrow v_i - (J_i^* J_i + \lambda_{\text{flow}} I)^{-1} J_i^* \mathcal{F}(u^{(m)}, v_i)$
 - 11: **for** $\lambda \in \lambda_{\text{train}}$ **do**
 - 12: $R = (\text{chol}_{\text{lower}}(J_i^* J_i + \lambda I))^{-*}$ ▷ $RR^* = (J_i^* J_i + \lambda I)^{-1}$
 - 13: store training data (z_i, λ, R)
 - 14: **end for**
 - 15: **end for**
 - 16: **end for**
 - 17: **Learn (offline):** $\min_{\theta} \sum \|\hat{\mathcal{R}}_\theta(z, \lambda) - R\|_F^2$; enforce lower-triangular with positive diagonal.
 - 18: **Evaluate (online):** given u, v_0 , budget N ; initial values $\alpha_{\text{predict}}/\lambda_{\text{train}}$.
 - 19: **for** $n = 0, \dots, N - 1$ **do**
 - 20: $r \leftarrow \mathcal{F}(u, v_n)$
 - 21: $z_n \leftarrow \Phi(u, v_n)$
 - 22: $J_n \leftarrow \text{BuildJ}(z_n)$
 - 23: $R \leftarrow \hat{\mathcal{R}}_\theta(z_n, \lambda_n)$
 - 24: $\delta v \leftarrow -\alpha_n R^* R J_n^* r$
 - 25: $v_{n+1} \leftarrow v_n + \delta v$
 - 26: choose α_n, λ_n by repeating the above steps until $\|\mathcal{F}(u, v_{n+1})\| \ll \|r\|$ ▷ e.g., using line search
 - 27: **Stop** if $\|r\|$ and/or $\|\delta v\|$ below given tolerances
 - 28: **end for**
 - 29: **Return** $\hat{v} = v_N$
-

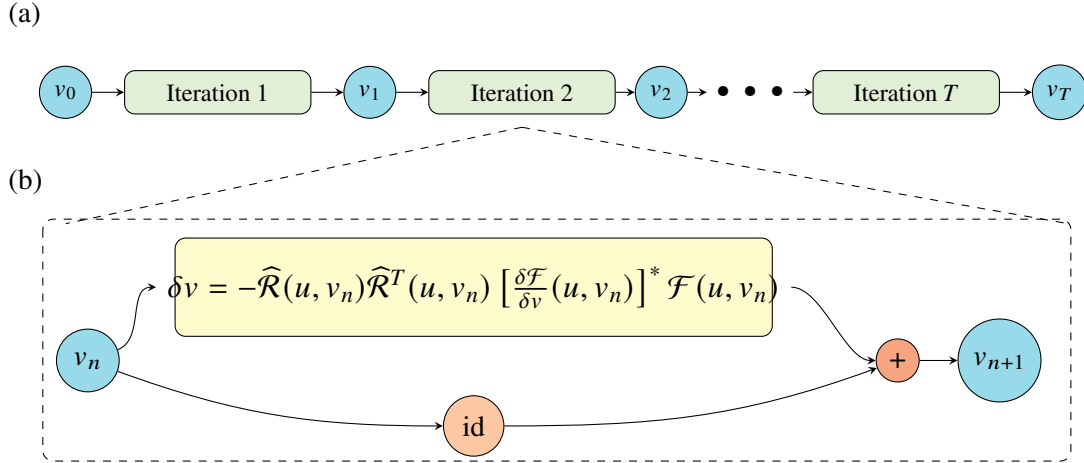


Figure 6.1: CHONKNORIS. (a) An initial guess v_0 for the true solution $v = \mathcal{G}(u)$ is iteratively refined by adding a correction term. (b) Each iteration consists of two steps: First, compute the correction term $\delta v = -(\hat{\mathcal{R}} \hat{\mathcal{R}}^* \left[\frac{\delta \mathcal{F}}{\delta v} \right]^* \mathcal{F})(u, v_k)$, where $\hat{\mathcal{R}}$ is a learned surrogate for the Cholesky factors of $(\frac{\delta \mathcal{F}}{\delta v} \frac{\delta \mathcal{F}}{\delta v}^T + \lambda I)^{-1}$, $\mathcal{F}$ is the forward map with its Fréchet derivative $\frac{\delta \mathcal{F}}{\delta v}$, v_k is the current approximation of the desired function $v = G(u)$, and u is the input for which we seek the solution. Next, update the current approximation via $v_{k+1} = v_k + \delta v$.

for coefficient functions $a = -1$, $b = 0$, $c = 3\kappa v^2$. While CHONKNORIS learns to predict the Cholesky factors of the Tikhonov-regularized inverse of this operator as a function of v_n and u , i.e., $\hat{\mathcal{R}}(u, \hat{v}_n)$, FONKNORIS aims to learn the same Cholesky factors as a function of the coefficient functions a, b , and c , i.e., $\hat{\mathcal{R}}(a, b, c)$. This implies that, once the model $\hat{\mathcal{R}}(a, b, c)$ is trained on a sufficiently large and diverse dataset of (a, b, c) combinations, we are able to predict the Cholesky factors arising from any PDE whose Fréchet derivative is of the form (6.12), i.e., any local PDE of second order. We note that the coefficient functions a, b , and c depend on and are uniquely determined by the operator $\mathcal{F}$.

To this end, FONKNORIS seeks to train a data-driven surrogate operator $\hat{\mathcal{R}}$, such as a neural operator or a kernel-based model, where

$$\hat{\mathcal{R}}(a, b, c) \hat{\mathcal{R}}(a, b, c)^T \approx ([a \partial_{xx} + b \partial_x + c] [a \partial_{xx} + b \partial_x + c]^* + \lambda I)^{-1},$$

$\hat{\mathcal{R}}(a, b, c)$ is lower triangular.

This model is *trained once, and generalizes to any equation* with a Jacobian of the form (6.12). Then, the FONKNORIS approximation is $\hat{\mathcal{G}} : u \mapsto \hat{v}_N$, where

$$\hat{v}_{n+1} = \hat{v}_n - \alpha_n \hat{\mathcal{R}}(a_n, b_n, c_n) \hat{\mathcal{R}}(a_n, b_n, c_n)^T [a_n \partial_{xx} + b_n \partial_x + c_n]^* \mathcal{F}(u, \hat{v}_n)$$

where $a_n = a(u, v_n)$, $b_n = b(u, v_n)$, $c_n = c(u, v_n)$.

Again, for a sufficiently large number of iterations N , we expect $\hat{\mathcal{G}}(u) = \hat{v}_N \approx \mathcal{G}(u)$. Note that for FONKNORIS, we add an intermediate step that, given u and v_n , computes the coefficient functions $a_n = a(u, v_n)$, $b_n = b(u, v_n)$, $c_n = c(u, v_n)$ and passes them to the model $\hat{\mathcal{R}}$, and hence we assume knowledge of these functions.

Furthermore, we note that this 1D example can be easily extended to any dimension and to any nonlinear operator. For local differential operators of order $k \in \mathbb{N}$, we observe that

$$\frac{\partial \mathcal{F}}{\partial v}(u, v) = \sum_{|\alpha| \leq k} \beta^\alpha(u, v) D^\alpha,$$

where $\alpha \in \mathbb{N}^k$ is a multi-index, D^α denotes a linear differential operator, and β^α denotes the coefficient functions. For nonlinear operators that are nonlocal, the Fréchet derivative also contains linear nonlocal operators. However, the principle remains the same: the Fréchet derivative can still be parameterized in terms of the coefficient functions. FONKNORIS can be regarded as a *foundation model* for local differential equations, as it is trained once on a diverse collection of PDEs through their coefficient functions β^α and generalizes to unseen equations without retraining. By learning a universal mapping from operator coefficients to inverse operators, FONKNORIS provides a reusable model that captures the shared structure underlying broad classes of PDEs. The FONKNORIS algorithm is summarized in Algorithm 2.

6.3 Numerical Experiments

In this section we present various numerical experiments that verify the ability of CHONKNORIS and FONKNORIS in emulating various PDE and inverse problem solution maps to machine precision.

Benchmarking Summary

To assess the performance of the CHONKNORIS method, we benchmark against the kernel/Gaussian process (GP) operator learning framework of Batlle, Matthieu Darcy, Hosseini, et al., 2024c, Fourier neural operators (FNOs) Zongyi Li, N. Kovachki, et al., 2020b, and transformer neural operator (TNO) Calvello et al., 2024. We deploy these three models as purely data-driven baselines which learn the operator $\mathcal{G}$ directly from input-output pairs (u, v) . Note that v is given by the last iterate of the Newton–Kantorovich method, so the baseline methods do not have access to the intermediate steps of the solver. These baselines are not intended as a comprehensive benchmark; rather, they serve as a control to verify that conventional

operator-learning frameworks do not attain machine precision on these problems. Furthermore, we note that the benchmark methods are trained in the low-data regime detailed in the following subsections, justifying the varying performance of the baselines. We also note that the vanilla FNO and vanilla TNO do not support the different input and output domains which arise in our benchmark inverse problems as detailed in Section 6.3; hence, we do not apply these benchmarks to the inverse problems we test here. Table 6.1 gives a performance comparison between these benchmark operator learning methods and our proposed CHONKNORIS approach. The details of the benchmark problems are outlined in the remainder of this section. Additional details on the GP baselines are given in Section E.1.

	GP	FNO	TNO	CHONKNORIS (ours)
Nonlinear elliptic	5.1e-6	1.3e-3	5.7e-3	8.9e-16
Burgers'	1.1e-1	8.8e-3	2.2e-2	5.1e-16
Nonlinear Darcy	1.8e-3	4.8e-3	3.9e-3	9.6e-16
Calderón	1.6e-2	—	—	3.2e-15
Inverse wave scattering	1.5e-2	—	—	9.2e-13
Seismic imaging 5×5	2.3e-2	—	—	2.0e-14
Seismic imaging 7×7	4.4e-2	—	—	3.0e-12
Seismic imaging 10×10	5.7e-2	—	—	1.2e-03

Table 6.1: Summary of median relative L^2 losses over multiple forward and inverse problem benchmarks.

Forward Problems

Table 6.2 summarizes the forward problems we consider and their parameterizations. The following subsections provide additional details on each of these forward problems.

Nonlinear Elliptic Equation

The first example is the 1D nonlinear elliptic PDE (6.1) with $\kappa = 50$ as introduced in Section 6.2.

The differential operator is given in (6.2) and the corresponding solution operator is given by $\mathcal{G}(u) = v$ mapping the right hand side to the solution of the boundary value problem Section 6.2. The random coefficient u is sampled from a zero-mean GP with a periodic kernel

$$K(x, x') = \exp(-2/\ell \sin^2(\pi/p(x - x')))$$

forward problem	Nonlinear Elliptic 1D	Burgers' 1D time-dependent	Darcy 2D
u distrib.	GP-periodic	sum of sines	GP-invLaplacian
FD grid	$N_x = 63$	$(N_t, N_x) = (151, 127)$	$(N_{x_1}, N_{x_2}) = (20, 20)$
$(R_{\text{train}}, R_{\text{val}})$	$(896, 128)$	$(448, 64)$	$(896, 128)$
$(n_{\text{warm}}, \lambda_{\text{flow}}, \lambda_{\text{train}})$	$(5, 0, 0)$	$(5, 0, 10^{-2})$	$(6, 0, 10^{-3})$
Hessian model	GP-Gaussian	MLP-Tanh $(127, 500, 1000, 8128)$	GP-Gaussian
Section	Section 6.3	Section 6.3	Section 6.3
inverse problem	Calderón	Inverse Wave Scattering	Seismic Imaging FWI
u distrib.	GP-invLaplacian	GP-invLaplacian	OpenFWI dataset (C. Deng et al., 2022)
FD grid	$(N_{x_1}, N_{x_2}) = (9, 9)$	$(N_{x_1}, N_{x_2}) = (7, 7)$	$(N_{x_1}, N_{x_2}) \in \{(5, 5), (7, 7), (10, 10), (14, 14)\}$
$(R_{\text{train}}, R_{\text{val}})$	$(7500, 2500)$	$(9750, 250)$	$(800, 200)$
$(n_{\text{warm}}, \lambda_{\text{flow}}, \lambda_{\text{train}})$	$(0, 10^{-10}, 10^{-10})$	$(0, 10^{-4}, 10^{-4})$	$(400, \text{adaptive}, \text{adaptive})$
Hessian model	GP-Gaussian	GP-Gaussian	GP-Gaussian
Section	Section 6.3	Section 6.3	Section 6.3

Table 6.2: Forward and inverse problem parameters. u distrib. is the distribution of random coefficients. FD grid contains the regular grid sizes for the finite difference scheme. $(R_{\text{train}}, R_{\text{val}})$ are the number of training and validation realizations respectively. n_{warm} denotes the number of Newton–Kantorovich iterations used to generate training data along the flow. λ_{flow} is the relaxation used in the NK method, while λ_{train} is the relaxation shown to the Hessian prediction model. CHONKNORIS training is performed with $\Lambda_{\text{train}} = \{\lambda_{\text{flow}}\}$ in the context of Algorithm 2. GP-kernel denotes a Gaussian process with the given kernel, with the invLaplacian kernel given by $5(-\Delta + 1/100)^{-2}$ where $-\Delta$ denotes the Laplacian. MLP-nonlinearity is a multi-layer perceptron neural network architecture where the tuple specifies layer sizes, including input and output layers, and the given nonlinearity is applied to all hidden layers. Note that for Burgers' equation, $n_{\text{warm}} = 5$ NK steps were run per time step.

with period length $p = 1/2$ and lengthscale $\ell = 10$. Here we fit two operator learning models. The first is an end-to-end operator learning model whose prediction is used as an initial guess for the NK/CHONKNORIS method. The second is our CHONKNORIS predictor model for the Cholesky factor. For both models we use vector-valued GP regression with Matérn kernels with smoothness parameter $\nu = 5/2$ (which we abbreviate as 5/2 Matérn) and squared exponential kernels, all with tuned lengthscales.

Burgers' Equation

The second example is the 1D time-dependent Burgers' equation, supplemented with periodic boundary conditions and a random initial condition, which may be written as

$$\begin{cases} \partial_t f = \nu \partial_{xx} f - f \partial_x f, & (x, t) \in \mathbb{T} \times [0, T], \\ f(0, t) = f(1, t), & t \in [0, T], \\ f(x, 0) = f_0(x), & x \in \mathbb{T} \end{cases}. \quad (6.13)$$

Here $\nu = 1/50$, f_0 is the initial condition and ∂_t , ∂_x , ∂_{xx} are partial derivatives. Here, we discretize the PDE in time and learn the solution operator of the time-discrete problem. More precisely, we discretize $[0, T]$ using the M -point uniform grid $\{t_i\}_{i=0}^{M-1} := \{i\Delta t\}_{i=0}^{M-1}$, $\Delta t = T/(M-1)$, and apply an implicit Euler time discretization leading to

$$\frac{f^{i+1}(x) - f^i(x)}{\Delta t} = \nu \partial_{xx} f^{i+1}(x) - f^{i+1}(x) \partial_x f^{i+1}(x),$$

where $f^i(x) = f(x, t_i)$ and $\Delta t = T/(M-1)$. Thus, the time marching discretization scheme defines the next time step f^{i+1} given f^i so that $\mathcal{F}(f^i, f^{i+1}) = 0$ with

$$\mathcal{F}(u, v) = v - \Delta t (\nu \partial_{xx} v - v \partial_x v) - u \quad \text{and} \quad \frac{\delta \mathcal{F}}{\delta v}(u, v)[h] = h - \Delta t (\nu \partial_{xx} h - \partial_x v h - v \partial_x h). \quad (6.14)$$

Notice that $\delta \mathcal{F} / \delta v$ does *not* depend on u . The resulting solution operator is given by $\mathcal{G}(u) = v$ mapping the previous time step $u = f^i$ to the next time step $v = f^{i+1}$. Here, CHONKNORIS approximates the solution operator $\mathcal{G}$ which in turn is used iteratively to obtain the solution for all time steps. We take our random initial condition to be

$$f_0(x) = \sum_{k=1}^3 a_k \sin(\pi k x), \quad (a_1, \dots, a_3) \sim \mathcal{N}(0, 1).$$

Nonlinear Darcy Flow

The third example is the 2D Darcy flow equation, supplemented with homogeneous Dirichlet boundary conditions, which may be written as

$$\begin{cases} -\nabla \cdot (e^u \nabla v) + v^3 = f, & x \in [0, 1]^2 \\ v = 0, & x \in \partial[0, 1]^2 \end{cases} \quad (6.15)$$

with forcing term f and conductivity e^u . Expanding $-\nabla \cdot (e^u \nabla v) = -e^u [\nabla u \cdot \nabla v + \Delta v]$, we have

$$\mathcal{F}(u, v) = -e^u [\nabla u \cdot \nabla v + \Delta v] + \kappa v^3 - f \quad \text{and} \quad \left[\frac{\delta \mathcal{F}}{\delta v}(u, v) \right](h) = -e^u [\nabla u \cdot \nabla h + \Delta h] + 3\kappa v^2 h. \quad (6.16)$$

Here, the solution operator is given by $\mathcal{G}(u, f) = v$ which maps the forcing f and the conductivity u to the solution v . In that sense, CHONKNORIS is learning a parametric family of solution operators $\mathcal{G}_u(\cdot) = \mathcal{G}(u, \cdot)$. We take the fixed forcing term f to be a single draw from a zero-mean GP with a 5/2 Matérn kernel and constant lengthscale of 3/10 across both dimension.

Summary of the numerical results for CHONKNORIS on the forward problems

We used CHONKNORIS to emulate and solve the three problems above. The domains were discretized using regular grids, and derivatives were approximated using standard finite differences. A traditional NK solver was used to generate the training data, and the reference solutions with which the errors were computed. The Cholesky factors for the nonlinear elliptic PDE in Section 6.3 and the Darcy flow problem in Section 6.3 were predicted using a GP, while for the Burgers' equation in Section 6.3 we used a Multi-Layer Perceptron (MLP).

In all tested instances of the three problems, CHONKNORIS consistently achieves machine precision error. We study the speed of convergence in the nonlinear elliptic PDE in Section 6.3, see Figure 6.2 (a), and found that CHONKNORIS starting from an initial guess of 0 typically converges in 10 iterations while predicting the approximate Hessian, whereas the traditional NK solver needs only 4 iterations using the precise approximate Hessian. We also observed that the convergence of the RMSE residual $\|\mathcal{F}(u, v_n)\|$, which we could track, follows the desired convergence of L^2 relative errors $\|v_n - v\|$ to machine precision (which is not generally known except when the true solution v is known as is the case here). We observed that providing an initial guess from an end-to-end operator learning approach gives modest speedups of one to two iterations to both the NK and CHONKNORIS iterations.

We tested the robustness of our method with the Burgers' problem in Section 6.3, see Figure 6.2 (b), and found that our method can achieve machine precision even when solutions contain shocks. Finally, the Darcy flow problem in Section 6.3 was the most challenging of the forward benchmarks as it required a lot of data to learn the parametric Cholesky factors. This experiment showcases the capacity of our method to utilize additional compute to achieve convergence even in difficult settings. As shown in Figure 6.2 (c), within 10 iterations CHONKNORIS usually reaches L^2 relative errors on the order of 10^{-3} , putting it on par with existing operator learning models. After 100 iterations, errors are typically on the order of 10^{-6} or better. Continuing to increase the number of CHONKNORIS iterations to 1000 enables convergence to machine precision in 95% of all cases. In this example, we observed that realizations from a rougher distribution require significantly more CHONKNORIS iterations for exact recovery.

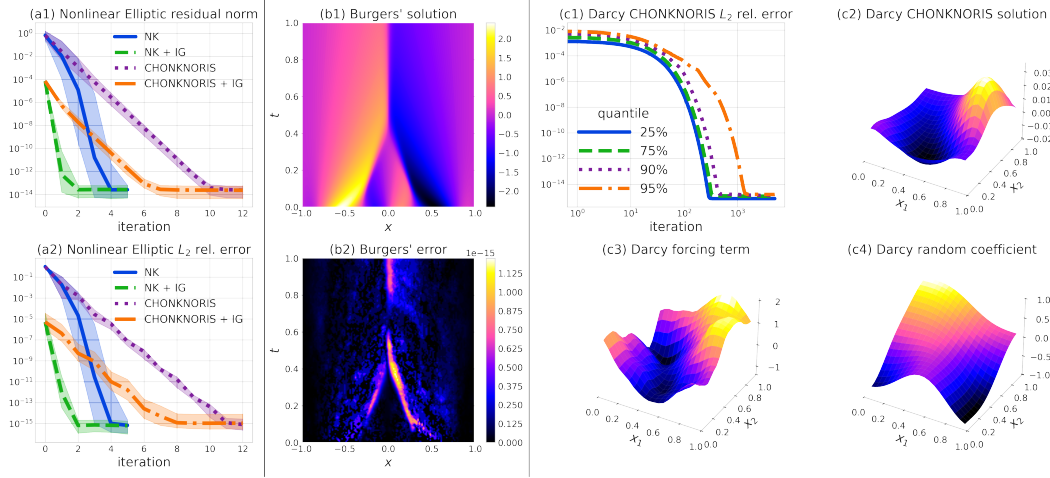


Figure 6.2: Forward problems. (a) Results for the nonlinear elliptic PDE problem. Quantiles of 10% – 90% are shown across test realizations. Our CHONKNORIS method is able to achieve machine precision accuracy in around 10 iterations. (b) Results for Burgers' equation. CHONKNORIS was able to achieve machine precision error in recovering the discretized solution which contained shocks. (c) Results for the Darcy flow PDE: (c1) shows that more challenging realizations require more CHONKNORIS iterations. (c4) shows a single realization of the random coefficient with the corresponding solution in (c2). (c3) shows the fixed forcing term.

Klein–Gordon and Sine–Gordon

For our last two forward problem examples, we consider the 1D Klein–Gordon and the Sine–Gordon equations. We will use these problems as a held out validation problem for testing the generalization error of FONKNORIS. When supplemented with initial and boundary condition, both PDEs take the form of

$$\begin{cases} \partial_{tt}f = \kappa_1 \partial_{xx}f - \kappa_2 \tau(f), & (x, t) \in \mathbb{T} \times [0, T], \\ f(0, t) = f(1, t), & t \in [0, T], \\ f(x, 0) = f_0(x), & x \in \mathbb{T} \end{cases}. \quad (6.17)$$

Here $x \in \Omega = \mathbb{T}$ and $t \in [0, T]$, f_0 is the initial condition, and τ is a nonlinearity. For Klein–Gordon, $\tau(f) = f^3$, $\kappa_1 = 0.1$, $\kappa_2 = 10$, while for Sine–Gordon $\tau(f) = \sin(f)$ and $\kappa_1 = \kappa_2 = 1$. Similar to the time marching scheme for Burgers' equation in Section 6.3, we discretize $[0, T]$ and approximate $\partial_{tt}f \approx (f^{i+2} - 2f^{i+1} + f^i)/\Delta t^2$. Knowing the state $u := (u_1, u_2) := (f^{i+1}, f^i)$, we want to compute the next time

step $v := f^{i+2}$ defined as $\mathcal{F}((f^{i+1}, f^i), f^{i+2}) = 0$ with

$$\mathcal{F}(u, v) := v - 2u_1 + u_2 - (\Delta t)^2(v\partial_{xx}v - \kappa_2\tau(v)), \quad (6.18)$$

$$\left[\frac{\delta \mathcal{F}}{\delta v}(u, v) \right] (h) = h - (\Delta t)^2 \left(\kappa_1 \partial_{xx} h - \kappa_2 \frac{\delta \tau}{\delta v}(v)[h] \right), \quad (6.19)$$

where $u = (u_1, u_2)$. For Klein–Gordon, $\frac{\delta \tau}{\delta v}(v)[h] = 3v^2h$, and for Sine–Gordon $\frac{\delta \tau}{\delta v}(v)[h] = \cos(v)h$. We note again, that $\delta \mathcal{F}/\delta v$ is independent of u . The resulting solution operators are in both cases given by $\mathcal{G}(u) = v$ which map the previous time steps $u = (u_1, u_2) = (f^{i+1}, f^i)$ to the next time step $v = f^{i+2}$. Again, CHONKNORIS approximates the solution operator $\mathcal{G}$ which is used in the very same time marching scheme to obtain the solution for all time steps.

FONKNORIS	Partial differential equation	a	b	c
training PDEs	Nonlinear elliptic (6.2)	-1	0	$3\kappa v^2$
	Burgers’ (6.14)	$-(\Delta t)v$	$(\Delta t)v$	$1 + (\Delta t)\nabla v$
	Nonlinear Darcy flow (6.16)	$-e^u$	$-e^u \nabla u$	$3\kappa v^2$
testing PDEs	Sine–Gordon (6.18)/(6.19)	$-\kappa_1(\Delta t)^2$	0	$1 + \kappa_2(\Delta t)^2 \cos(v)$
	Klein–Gordon (6.18)/(6.19)	$-\kappa_1(\Delta t)^2$	0	$1 + 3\kappa_2(\Delta t)^2 v^2$

Table 6.3: FONKNORIS coefficients.

Foundation Model - FONKNORIS

For the FONKNORIS model, we train sub-model experts, each specific to a given PDE, and aggregate individual CHONKNORIS sub-models within mixture of experts Rulliere et al., 2018. The aggregation of the sub-models is done by the so-called nested Kriging method Rulliere et al., 2018, see also Section E.2. This method relies on the fact that each sub-model has the same underlying Gaussian process and thus share the same kernel. In our FONKNORIS prediction framework, we employ Gaussian processes to model and predict the behavior of specific PDEs for each expert model. Constructing a single foundational Gaussian process model capable of representing a large class of PDEs would require an extensive dataset, which would quickly exceed typical computational and memory limits. To overcome this limitation, we train each expert model independently and subsequently aggregate the individual experts within a foundational (meta) model.

Data Generation and Training

In our numerical experiments for FONKNORIS, we trained separate instances of the model $\hat{\mathcal{R}}_\theta$ for the one-dimensional nonlinear elliptic equation, Burgers’ equation,

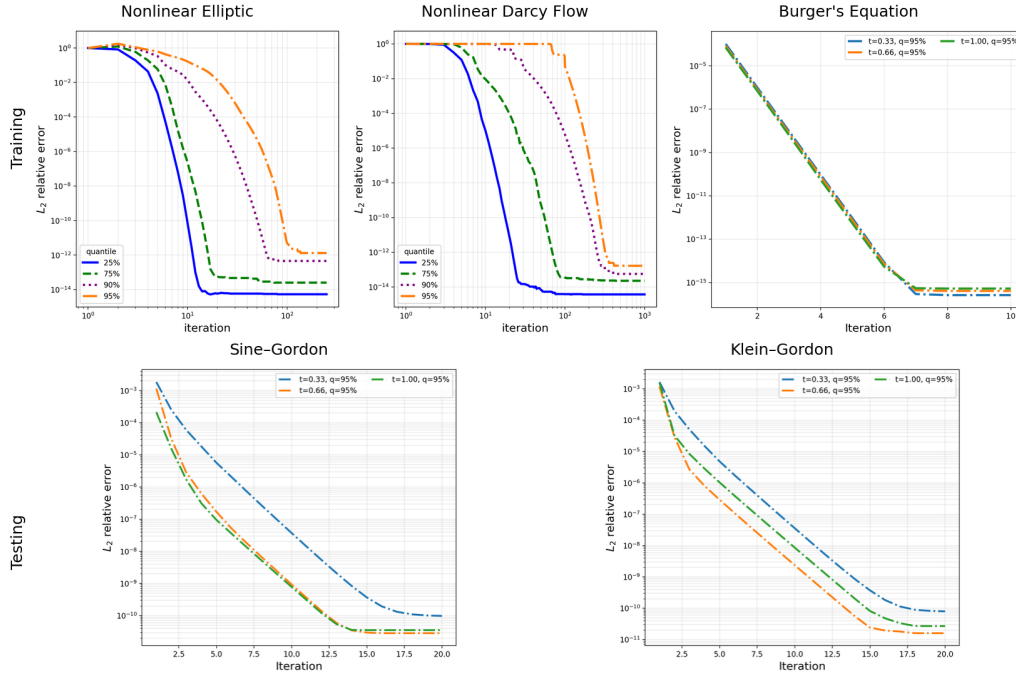


Figure 6.3: Quantiles of predictions of FONKNORIS for 100 realizations of initial conditions, external forces, and conductivities using a mixture of experts consisting of GPs for nonlinear elliptic, nonlinear darcy flow, and Burgers' equation and testing it for withheld Sine–Gordon and Klein–Gordon equations.

and the one-dimensional nonlinear Darcy flow, using Matérn kernels with learned length scales ℓ_1 , ℓ_2 , and ℓ_3 for the input distributions μ . For the aggregated model, we used the arithmetic mean of the length scales, $\ell_{\mathcal{A}} = \frac{1}{p} \sum_{i=1}^p \ell_i$. The Sine–Gordon and Klein–Gordon equations were held out for testing. All PDEs were discretized with $N_x = 64$ collocation points. In the Newton–Kantorovich warm-up step of Algorithm 2 we used a fixed flow relaxation parameter $\lambda_{\text{flow}} = 0.01$ and the same value in the training relaxation set, $\lambda_{\text{train}} = 0.01$.

For the nonlinear elliptic equation and Burgers' equation, we generated $M_{\text{ell}} = 5,000$ and $M_{\text{Burg}} = 5,000$ offline realizations, respectively. The realizations for the nonlinear elliptic equation were generated by drawing 1,000 external forcings f from a periodic kernel defining μ and running $n_{\text{warm}} = 5$ Newton–Kantorovich iterations. The realizations for Burgers' equation were generated by drawing 1,000 initial conditions u_0 from a periodic kernel and again running $n_{\text{warm}} = 5$ Newton–Kantorovich iterations. For the nonlinear Darcy flow, we generated $M_{\text{Darcy}} = 10,000$ realizations by drawing 2,000 pairs of conductivities a and external forcings f from a periodic kernel and running $n_{\text{warm}} = 5$ Newton–Kantorovich iterations for each

pair (a_i, f_i) . The aggregation model was tested on the Klein–Gordon and Sine–Gordon equations by drawing realizations of the initial condition u_0 from a periodic kernel and choosing $v_0 = 0$ in the online phase of Algorithm 2. The data for training and testing were generated from the same distributions.

Summary of the numerical results for FONKNORIS on the forward problems

We applied FONKNORIS to the above problems, using the nonlinear elliptic, the Burgers’, and the nonlinear Darcy flow equations as training problems, and testing on all five problems, including the Klein–Gordon and Sine–Gordon equations. In Figure 6.3, we see that the aggregated model $M(x)$ not only achieves machine precision on the trained instances, but also achieves near machine precision for the withheld Klein–Gordon and Sine–Gordon equation. Table 6.3 shows the FONKNORIS coefficients for the aforementioned forward problems. The advantage of this approach is its simplicity and generating training data does not rely on a specific PDE as the coefficients a, b, c can be generated from a certain distribution, optimally from various distributions. The main difficulty is that the model can become very large and the coefficients a, b, c can easily get out of distribution when computing each Newton–Kantorovich step; one deals with the curse of dimensionality as the generated data might not inherit the statistics of a given PDE. Therefore, we generate the training data from the first three problems and generalize it to all problems including the Sine–Gordon and Klein–Gordon equation. Here, our mixture-of-experts Gaussian process model was capable of handling the large scale data necessary to generalize across PDE problems. This ability to generalize to unseen problems at machine precision is a state-of-the-art achievement that is due to the unique design of the FONKNORIS model.

Inverse Problems

In this section, we introduce various inverse problems which will be used to further test the performance of CHONKNORIS as an emulator for PDE constrained optimization. These include the Calderón problem (Section 6.3), the inverse wave scattering problem (Section 6.3), and a problem in seismic imaging full waveform inversion (Section 6.3). Specific parameterizations for each of these problems are given in Table 6.2. The following subsections further detail our setup.

The Calderón problem

Let $\Omega = [0, 1]^2$ and assume $v \in L^\infty(\Omega)$ is a real-valued conductivity with $v(x) > 0$ for almost every $x \in \Omega$. Consider the boundary value problem

$$\begin{cases} \nabla \cdot (v(x) \nabla c(x)) = 0 & \text{for } x \in \Omega, \\ c(x) = g(x) & \text{for } x \in \partial\Omega, \end{cases} \quad (6.20)$$

where on the boundary, we prescribe a boundary voltage pattern $g \in H^{\frac{1}{2}}(\partial\Omega)$. We define the *Dirichlet-to-Neumann* (DtN) map $\Lambda_g : H^{\frac{1}{2}}(\partial\Omega) \rightarrow H^{-\frac{1}{2}}(\partial\Omega)$ that maps the boundary voltage pattern g to the current flux $v \frac{\partial c}{\partial n} \Big|_{\partial\Omega}$. Note that $\Lambda_g \in \mathcal{L}(H^{\frac{1}{2}}(\partial\Omega), H^{-\frac{1}{2}}(\partial\Omega))$ is a linear bounded operator. The *Calderón problem* is the task of recovering the conductivity v from a given DtN map Λ_g . Denote with $\tilde{\mathcal{F}} : L^\infty(\Omega) \rightarrow \mathcal{L}(H^{\frac{1}{2}}(\partial\Omega), H^{-\frac{1}{2}}(\partial\Omega))$ the forward operator that maps the conductivity to the Dirichlet-to-Neumann map. Then, the operator of interest is given by

$$\mathcal{F}(\Lambda_g, v) = \tilde{\mathcal{F}}(v) - \Lambda_g.$$

CHONKNORIS aims to approximate the operator $\mathcal{G} : \mathcal{L}(H^{\frac{1}{2}}(\partial\Omega), H^{-\frac{1}{2}}(\partial\Omega)) \rightarrow L^\infty(\Omega)$ that maps the DtN map to the conductivity, i.e., $\mathcal{G}(\Lambda_g) = \tilde{\mathcal{F}}^{-1}(\Lambda_g) = v$. We note that, in practice, we only have access to pairs of observations $(g_i, v \frac{\partial c_i}{\partial n} \Big|_{\partial\Omega})_{i=1}^N$ on a finite number of sensor points on the boundary $\partial\Omega$, where c_i denotes the solution to the (6.20) with conductivity v and boundary condition prescribed by g_i . Thus the equation used in practice is

$$\hat{\mathcal{F}}(u, v) = \left(\tilde{\mathcal{F}}(v) g_i - v \frac{\partial c_i}{\partial n} \Big|_{\partial\Omega} \right)_{i=1}^N = 0,$$

where $u = (g_i, v \frac{\partial c_i}{\partial n} \Big|_{\partial\Omega})_{i=1}^N$.

We take the fixed forcing term to be a single draw from a zero-mean GP with 5/2 Matérn kernel and constant lengthscale of 3/10 across both dimension.

Inverse Wave Scattering

Let $\Omega = [0, 1]^2$ and assume that $a \in L^\infty(\Omega)$ is a real-valued material property satisfying $a(x) > 0$ for almost every $x \in \Omega$. We study the following elliptic boundary value problem:

$$\begin{cases} -\Delta u(x) - \omega^2 a(x) u(x) = 0, & x \in \Omega, \\ u(x) = g(x), & x \in \partial\Omega, \end{cases} \quad (6.21)$$

where $g \in H^{\frac{1}{2}}(\partial\Omega)$ denotes the prescribed boundary excitation.

We define the *Dirichlet-to-Neumann* (DtN) map $\Lambda_a : H^{\frac{1}{2}}(\partial\Omega) \rightarrow H^{-\frac{1}{2}}(\partial\Omega)$ that maps the boundary input g to the corresponding flux

$$\Lambda_a(g) = \frac{\partial u}{\partial n} \Big|_{\partial\Omega}.$$

Note that $\Lambda_a \in \mathcal{L}(H^{\frac{1}{2}}(\partial\Omega), H^{-\frac{1}{2}}(\partial\Omega))$ is a bounded linear operator depending on the material coefficient a .

The *inverse wave scattering problem* is the task of recovering the coefficient a from a given DtN map Λ_a . Denote with $\tilde{\mathcal{F}} : L^\infty(\Omega) \rightarrow \mathcal{L}(H^{\frac{1}{2}}(\partial\Omega), H^{-\frac{1}{2}}(\partial\Omega))$ the forward operator that maps the material property a to the DtN map. Then, the operator of interest is defined as

$$\mathcal{F}(\Lambda_a, a) = \tilde{\mathcal{F}}(a) - \Lambda_a.$$

Again, CHONKNORIS aims to approximate the operator $\mathcal{G} : \mathcal{L}(H^{\frac{1}{2}}(\partial\Omega), H^{-\frac{1}{2}}(\partial\Omega)) \rightarrow L^\infty(\Omega)$ that maps the DtN map to the material property, i.e., $\mathcal{G}(\Lambda_a) = \tilde{\mathcal{F}}^{-1}(\Lambda_a) = a$.

In practice, we only have access to finitely many boundary input-output pairs $(g_i, \frac{\partial u_i}{\partial n}|_{\partial\Omega})_{i=1}^N$ measured on sensor locations along $\partial\Omega$, where u_i denotes the solution to (6.21) corresponding to the boundary excitation g_i . Hence, the equation used in practice is

$$\hat{\mathcal{F}}(u, a) = \left(\tilde{\mathcal{F}}(a)g_i - \frac{\partial u_i}{\partial n} \Big|_{\partial\Omega} \right)_{i=1}^N = 0,$$

where $u = (g_i, \frac{\partial u_i}{\partial n}|_{\partial\Omega})_{i=1}^N$.

We take the fixed forcing term to be a single draw from a zero-mean Gaussian process with 5/2 Matérn kernel and constant lengthscale of 2/10 across both dimension.

Seismic Imaging

Let $\Omega = [0, 1]^2$ and let $T > 0$ denote the final observation time. Assume that $v \in L^\infty(\Omega)$ is a real-valued velocity coefficient satisfying $v(x) > 0$ for almost every $x \in \Omega$. We study the following time-dependent acoustic wave equation:

$$\begin{cases} \Delta p(t, x) - \frac{1}{v^2(x)} p_{tt}(t, x) = s(t, x), & (t, x) \in (0, T) \times \Omega, \\ p(0, x) = 0, \quad p_t(0, x) = 0, & x \in \Omega, \end{cases} \quad (6.22)$$

where p denotes the pressure variation and s represents the source term which we take to be a Ricker wavelet Yanghua Wang, 2015.

The reflected and refracted wavefields are measured on the surface $\mathcal{S} := \{(x_1, x_2) \in \Omega \mid x_2 = 0\} \subset \partial\Omega$, yielding the boundary observations $p|_{[0,T] \times \mathcal{S}}$. We define the *Source-to-Receiver* (StR) map $\Lambda_v : L^2((0, T) \times \Omega) \rightarrow L^2(0, T; H^{\frac{1}{2}}(\mathcal{S}))$ that maps a source s to the measured surface signal $\Lambda_v(s) = p|_{[0,T] \times \mathcal{S}}$. Note that $\Lambda_v \in \mathcal{L}(L^2((0, T) \times \Omega), L^2(0, T; H^{\frac{1}{2}}(\mathcal{S})))$ is a bounded linear operator depending on the velocity coefficient v .

The *seismic inverse problem* or *full wave inversion problem* is the task of recovering the coefficient v from a given Source-to-Receiver map Λ_v Virieux and Operto, 2009. Denote with $\tilde{\mathcal{F}} : L^\infty(\Omega) \rightarrow \mathcal{L}(L^2((0, T) \times \Omega), L^2(0, T; H^{\frac{1}{2}}(\mathcal{S})))$ the forward operator that maps the velocity coefficient v to the corresponding StR map. Then, the operator of interest is defined as

$$\mathcal{F}(\Lambda_v, v) = \tilde{\mathcal{F}}(v) - \Lambda_v(s).$$

Similar to the inverse wave scattering and Calderón problems, CHONKNORIS aims to approximate the operator

$$\mathcal{G} : \mathcal{L}(L^2((0, T) \times \Omega), L^2(0, T; H^{\frac{1}{2}}(\mathcal{S}))) \rightarrow L^\infty(\Omega),$$

that maps the Source-to-Receiver map to the velocity coefficient, i.e., $\mathcal{G}(\Lambda_v) = \tilde{\mathcal{F}}^{-1}(\Lambda_v) = v$.

In practice, we only have access to finitely many input-output pairs $(s_i, p_i|_{[0,T] \times \mathcal{S}})_{i=1}^N$ measured at discrete sensor locations along $\mathcal{S}$, where p_i denotes the solution to (6.22) corresponding to the source s_i . Hence, the equation used in practice is

$$\hat{\mathcal{F}}(u, v) = \left(\tilde{\mathcal{F}}(v)s_i - p_i|_{[0,T] \times \mathcal{S}} \right)_{i=1}^N = 0,$$

where $u = (s_i, p_i|_{[0,T] \times \mathcal{S}})_{i=1}^N$. We use data from the OpenFWI dataset (C. Deng et al., 2022) and the forward solver from X. Wang, 2023 (a 2 – 4 finite difference scheme with 2nd-order accuracy in time and 4th-order in space). Convergence of the exact NK method for a single 14×14 resolution velocity map is shown in Figure 6.4 (a1-a5).

While our previous experiments predicted the inverse Cholesky of the approximate Hessian, for this problem we obtained better performance by directly predicting the Cholesky factors of the approximate Hessian and then use triangular solves to

determine each increment. We note this does not change the computational complexity of our method. We also found it necessary to simultaneously tune both the learning rate α in the line search and the Tikhonov relaxation λ in order to converge to machine precision in a reasonable number of iterations (see Algorithm 2). The tuning scheme we used is described in Section E.5.

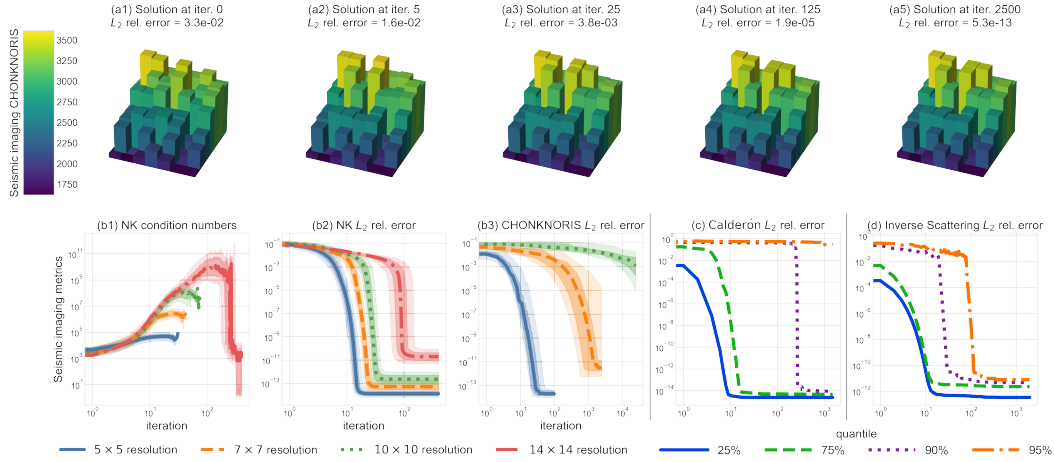


Figure 6.4: Inverse Problems. (a) Results for the seismic imaging problem, showing the iterative solutions and the relative L^2 error across CHONKNORIS iteration. (b) For the seismic imaging problem, evolution of the adaptive regularization term in the Newton–Kantorovich iterations for different resolutions, and comparison of the relative L^2 error between the Newton–Kantorovich method and our CHONKNORIS method. (c) Results for the Calderón problem. (d) Results for the inverse wave scattering problem.

Summary of the numerical results for CHONKNORIS on the Inverse Problems

We use CHONKNORIS to solve the three inverse problems above. The domains are discretized using regular grids. A traditional NK solver is used to generate training data where the converged iterates are used as reference solutions. These inverse problems are significantly harder to solve than the forward problems due to the non-locality of the operator $\mathcal{F}$ and the ill-posedness of the inverse problem. The results for the Calderón problem in Figure 6.4 (d) show that running the CHONKNORIS method for 10^3 iterations was able to recover solutions to machine precision for over 75% of withheld test realizations. The results for the inverse wave scattering problem in Figure 6.4 (e) shows that running the CHONKNORIS method for 40 iterations is able to recover solutions to machine precision. For the rough velocity maps encountered in the seismic imaging problem, increasing the resolution increases the

condition numbers of the relaxed approximate Hessian. For example, Figure 6.4 shows that the seismic imaging problem with just a 10×10 velocity map resolution encounters condition numbers around 10^8 for near-convergence iterations. CHONKNORIS is unable to exactly predict the Cholesky factor of ill-conditioned matrices, and thus resorts to inferring gradient descent steps which can make CHONKNORIS inference slow to converge. As expected, we also observe that the relaxation is decreased as NK nears convergence, indicating a smooth transition from gradient descent to Gauss–Newton updates. This behavior is more difficult to replicate with CHONKNORIS as the approximate Hessian near the solution is ill-conditioned and therefore difficult for CHONKNORIS to predict.

6.4 Theoretical results

In this section, we want to present several results on the convergence properties of the CHONKNORIS and FONKNORIS algorithms. This is accomplished by combining an inexact Newton–Kantorovich method with Tikhonov-regularized inverse and the kernel-based operator learning method.

Analytic Setting

Assume that we are in the operator learning framework of Section 6.2, i.e., let $(\mathcal{U}, \langle \cdot, \cdot \rangle_{\mathcal{U}}, \|\cdot\|_{\mathcal{U}})$, $(\mathcal{V}, \langle \cdot, \cdot \rangle_{\mathcal{V}}, \|\cdot\|_{\mathcal{V}})$, and $(\mathcal{W}, \langle \cdot, \cdot \rangle_{\mathcal{W}}, \|\cdot\|_{\mathcal{W}})$ be separable Hilbert spaces. For notational convenience, we suppress the dependence of $\mathcal{F}$ on the first argument, i.e., for each fixed $u \in \mathcal{U}$, we define $\mathcal{F}(\cdot) := \mathcal{F}(u, \cdot) : \mathcal{V} \rightarrow \mathcal{W}$ and its Fréchet derivative $\mathcal{F}'(v) := \frac{\delta \mathcal{F}}{\delta v}(u, v) \in \mathcal{L}(\mathcal{V}, \mathcal{W})$ with Hilbert adjoint $\mathcal{F}'(v)^* \in \mathcal{L}(\mathcal{W}, \mathcal{V})$.

For $\lambda > 0$ and $v \in \mathcal{V}$, define the *Tikhonov resolvent* and the *Tikhonov right inverse*

$$R_{\lambda}(v) := (\lambda I_{\mathcal{V}} + \mathcal{F}'(v)^* \mathcal{F}'(v))^{-1} \in \mathcal{L}(\mathcal{V}, \mathcal{V}), \quad B_{\lambda}(v) := R_{\lambda}(v) \mathcal{F}'(v)^* \in \mathcal{L}(\mathcal{W}, \mathcal{V}).$$

Let $\widehat{R}_{\lambda}(v) \in \mathcal{L}(\mathcal{V}, \mathcal{V})$ be a (learned) surrogate and set $\widehat{B}_{\lambda}(v) := \widehat{R}_{\lambda}(v) \mathcal{F}'(v)^*$. Given $v_0 \in \mathcal{V}$, consider the iteration

$$v_{k+1} = v_k + \delta v_k, \quad \delta v_k := -\widehat{B}_{\lambda_k}(v_k) \mathcal{F}(v_k) = -\widehat{R}_{\lambda_k}(v_k) \mathcal{F}'(v_k)^* \mathcal{F}(v_k),$$

for a sequence of parameters $\lambda_k > 0$. Furthermore, we make the following assumptions.

Assumption 10. Let $D \subset \mathcal{V}$ be open and convex and $\mathcal{F} : D \rightarrow \mathcal{W}$ be C^1 . Fix

$v_0 \in D$ and $R > 0$ with $B := \overline{B(v_0, R)} \subset D$. Assume:

(A1) $\mathcal{F}'(v_0)$ is invertible and $\|\mathcal{F}'(v_0)^{-1}\|_{\mathcal{L}(\mathcal{W}, \mathcal{V})} \leq \beta \in \mathbb{R}_{>0}$,

(A2) There exists $L > 0$ s.t. $\|\mathcal{F}'(u) - \mathcal{F}'(v)\|_{\mathcal{L}(\mathcal{V}, \mathcal{W})} \leq L \|u - v\|_{\mathcal{V}} \quad \forall u, v \in B$,

(A3) $M := \sup_{v \in B} \|\mathcal{F}'(v)\|_{\mathcal{L}(\mathcal{V}, \mathcal{W})} < +\infty$, $\sigma_* := \inf_{v \in B} \sigma_{\min}(\mathcal{F}'(v)) = \inf_{v \in B} \|\mathcal{F}'(v)^{-1}\|_{\mathcal{L}(\mathcal{W}, \mathcal{V})}^{-1} > 0$.

Now, we are in a position to state the main theorem on the Tikhonov-regularized inexact Newton–Kantorovich iteration method based on a classical inexact Newton–Kantorovich method Dembo, Eisenstat, and Steihaug, 1982. The proofs in this section are postponed to the appendix Section E.8.

Theorem 11 (Tikhonov–inexact Newton–Kantorovich). *Let Assumption 10 be satisfied. Then, for each v_k with $v_k \in \overline{B(v_0, R)}$, the linearized residual satisfies*

$$\frac{\|\mathcal{F}'(v_k)\delta v_k + \mathcal{F}(v_k)\|_{\mathcal{W}}}{\|\mathcal{F}(v_k)\|_{\mathcal{W}}} \leq \frac{\lambda_k}{\lambda_k + \sigma_*^2} + M^2 \varepsilon_{\lambda_k} =: \theta_k, \quad (6.23)$$

where

$$\varepsilon_{\lambda} := \sup_{v \in \overline{B(v_0, R)}} \|\widehat{R}_{\lambda}(v) - R_{\lambda}(v)\|_{\mathcal{L}(\mathcal{V}, \mathcal{V})}$$

denotes the design/learning error. If $\sup_k \theta_k \leq \bar{\theta} < 1$ and all iterates remain in $\overline{B(v_0, R)}$, then the inexact Newton–Kantorovich conclusions hold on $\overline{B(v_0, R)}$ with

$$\tilde{L} := \frac{L}{1 - \bar{\theta}}, \quad \tilde{h} := \beta \tilde{L} \eta, \quad \eta := \|\mathcal{F}'(v_0)^{-1} \mathcal{F}(v_0)\|_{\mathcal{V}}.$$

In particular, if $\tilde{h} \leq \frac{1}{2}$ and $t_* = (1 - \sqrt{1 - 2\tilde{h}})/(\beta \tilde{L}) \leq R$, then the iterates are well-defined, remain in $\overline{B(v_0, t_*)}$, and converge to the unique zero $v_* \in \overline{B(v_0, t_*)}$, with the Kantorovich majorant bounds

$$\|v_k - v_*\|_{\mathcal{V}} \leq t_* - t_k, \quad \|v_{k+1} - v_k\|_{\mathcal{V}} \leq t_{k+1} - t_k,$$

where $t_{k+1} = t_k - \frac{\phi(t_k)}{\phi'(t_k)}$ and $\phi(t) = \eta - t + \frac{1}{2} \beta \tilde{L} t^2$.

Moreover, if $\lambda_k \rightarrow 0$ and $\varepsilon_{\lambda_k} \rightarrow 0$, then $\theta_k \rightarrow 0$ and the local rate approaches the quadratic rate of exact Newton.

The following corollary makes the final statement of the preceding Theorem 11 on the convergence more precise. In particular, it shows for which Tikhonov parameters and design errors we obtain linear, superlinear, quadratic convergence.

Corollary 2 (Convergence Rates). *Let $e_k := \|v_k - v_*\|_{\mathcal{V}}$. Under the hypotheses of Theorem 11 and for all k with $v_k \in \overline{B(v_0, R)}$,*

$$e_{k+1} \leq \frac{1}{1 - \bar{\theta}} \left(\frac{\beta L}{2} e_k^2 + \bar{\theta} e_k \right). \quad (6.24)$$

Then, there exist constants C_1, C_2, C_3, C_4 such that

- (i) *If $\bar{\theta} \in (0, 1)$ is fixed, then $\limsup_{k \rightarrow \infty} \frac{e_{k+1}}{e_k} \leq \frac{\bar{\theta}}{1 - \bar{\theta}}$ (at least Q -linear).*
- (ii) *If $\theta_k \rightarrow 0$, then $e_{k+1} \leq \frac{\beta L}{2} e_k^2 (1 + o(1))$ and the rate is Q -superlinear.*
- (iii) *If $\theta_k \leq C_1 \|\mathcal{F}(v_k)\|_{\mathcal{W}}^\alpha$ for some $\alpha > 0$, then $e_{k+1} \leq C_2 e_k^{1+\alpha} + C_3 e_k^2$ for k large, so the local order is $\min\{2, 1 + \alpha\}$.*
- (iv) *If $\theta_k = O(\|\mathcal{F}(v_k)\|_{\mathcal{W}})$, then $e_{k+1} \leq C_4 e_k^2$ and the convergence is Q -quadratic.*

Based on the preceding corollary, we can propose a schedule for the Tikhonov parameter λ_k and the design error ε_{λ_k} in order to achieve the desired convergence.

Remark 15 (Scheduling λ_k to reach superlinear/quadratic convergence). *Let $\theta_{\text{tik},k} = \frac{\lambda_k}{\lambda_k + \sigma_*^2}$ and $\theta_{\text{des},k} = M^2 \varepsilon_{\lambda_k}$ so that $\theta_k = \theta_{\text{tik},k} + \theta_{\text{des},k}$.*

- (a) *If $\lambda_k \rightarrow 0$ and $\varepsilon_{\lambda_k} \rightarrow 0$, then $\theta_k \rightarrow 0$ and the rate is superlinear.*
- (b) *If there exist $c_1, c_2 > 0$ such that*

$$\lambda_k \leq c_1 \|\mathcal{F}(v_k)\|_{\mathcal{W}}, \quad \varepsilon_{\lambda_k} \leq c_2 \|\mathcal{F}(v_k)\|_{\mathcal{W}},$$

then $\theta_k = O(\|\mathcal{F}(v_k)\|_{\mathcal{W}})$ and the convergence is Q -quadratic.

- (c) *Suppose for some $C > 0$ one has the regularity proxy $\varepsilon_\lambda \approx C \lambda^{-2}$. Consider $\phi(\lambda) := \frac{\lambda}{\lambda + \sigma_*^2} + C \lambda^{-2}$. Then any λ_k chosen near the minimizer $\lambda_* \asymp (C \sigma_*^2)^{1/3}$ balances Tikhonov bias and model error early on; subsequently decreasing $\lambda_k \downarrow 0$ (as the model improves and the iterates approach v_*) drives $\bar{\theta} \rightarrow 0$ and approaches quadratic convergence.*

Remark 16 (Variant: learning the full $B_\lambda(v)$ directly). *If the surrogate model directly learns $B_\lambda(v)$ and achieves $\sup_{v \in B} \|\widehat{B}_\lambda(v) - B_\lambda(v)\|_{\mathcal{L}(\mathcal{W}, \mathcal{V})} \leq \varepsilon_\lambda$, then the residual bound becomes*

$$\bar{\theta} \leq \underbrace{\frac{\lambda}{\lambda + \sigma_*^2}}_{\theta_{\text{tik}}} + \underbrace{M \varepsilon_\lambda}_{\theta_{\text{des}}},$$

since $\|I - \mathcal{F}' \widehat{B}_\lambda\| \leq \|I - \mathcal{F}' B_\lambda\| + \|\mathcal{F}'\| \|\widehat{B}_\lambda - B_\lambda\|$. All conclusions of Theorem 11 remain unchanged with this $\bar{\theta}$.

Example 1 (Application to nonlinear elliptic PDE).

In this example, we want to apply the convergence theorem to the nonlinear elliptic PDE considered in Section 6.3. Let $\mathbb{T} = \mathbb{R}/\mathbb{Z}$ and

$$\mathcal{V} := H_{\text{per},0}^1(\mathbb{T}) = \left\{ u \in H_{\text{per}}^1(\mathbb{T}) : \int_0^1 u = 0 \right\}, \quad \mathcal{W} := \mathcal{V}^* \simeq H_{\text{per},0}^{-1}(\mathbb{T}),$$

with $\|u\|_{\mathcal{V}} := \|u'\|_{L^2(0,1)}$. Let $f \in L^2(0,1)$ be fixed and define

$$\mathcal{F} : \mathcal{V} \rightarrow \mathcal{W}, \quad \mathcal{F}(u) = -u'' + u^3 - f, \quad \mathcal{F}'(u)v = -v'' + 3u^2v.$$

Now, we present the main convergence result with explicit constants by verifying the conditions of Theorem 11.

Theorem 12 (Main convergence theorem with explicit constants). *Let $v_0 = 0$ and $\lambda > 0$ be the Tikhonov parameter. Assume that the iterates $\{v_k\}$ remain in $B_r := \{v \in \mathcal{V} : \|v\|_{\mathcal{V}} \leq r\}$ for some $r > 0$. Define*

$$\eta := \|\mathcal{F}'(0)^{-1} \mathcal{F}(0)\|_{\mathcal{V}} = \|f\|_{H^{-1}}, \quad L = \frac{3}{2\pi^2} r, \quad M = 1 + \frac{3}{4\pi^2} r^2.$$

Then all the conclusions of Theorem 11 hold true for this problem. In particular, the Lipschitz constant in Theorem 11 can be chosen as

$$\widetilde{L} := \frac{L}{1 - \bar{\theta}} = \frac{\frac{3}{2\pi^2} r}{1 - \frac{\lambda}{1+\lambda} - M^2 \varepsilon_\lambda},$$

where ε_λ is the design/learning error, and $\bar{\theta} := \frac{\lambda}{1+\lambda} + M^2 \varepsilon_\lambda$ bounds the forcing terms θ_k in (6.23) for all v_k with $v_k \in B_r$.

6.5 Conclusion and Outlook

In this paper, we introduced CHONKNORIS, a novel operator learning methodology rooted in the Newton–Kantorovich method for solving infinite-dimensional systems. Crucially, CHONKNORIS removes the longstanding ceiling in accuracy typically observed in operator learning frameworks, setting a new standard in precision. To our knowledge, this is the first instance of an operator learning framework that achieves machine precision in solving a variety of nonlinear partial differential equations (PDEs) and inverse problems.

The central innovation of CHONKNORIS lies in employing an iterative scheme based on the Newton–Kantorovich method, where we explicitly learn the dependence of the Cholesky factors of the matrix $(\lambda I + (\frac{\partial \mathcal{F}}{\partial v})^* \frac{\partial \mathcal{F}}{\partial v})^{-1}$ on coefficients u and solution estimates v . The explicit integration of the underlying equation within these iterations significantly enhances the accuracy and interpretability of the learned solutions. It also makes it uniquely suited for inverse problems, as the general equation $\mathcal{F}$ may incorporate observed data. The ability to tackle inverse problems directly within the operator learning framework is a significant improvement from traditional approaches, which depend on repeated iterations of forward solvers and often struggle in high-dimensional settings. Building on CHONKNORIS, we introduced FONKNORIS, a foundational model motivated by the observation that $\frac{\partial \mathcal{F}}{\partial v}$ always defines a linear PDE. While CHONKNORIS learns to solve the linearized PDEs associated with the Fréchet derivative of one equation $\mathcal{F}$, FONKNORIS extends this concept by learning to solve any linearized PDE. By plugging this learned linear PDE solver into the CHONKNORIS framework, we obtain a universal operator learner capable of solving a broad class of nonlinear PDEs.

We validated our methodology on diverse forward problems, including a nonlinear elliptic equation, Burgers’ equation, a Darcy flow equation, the Sine–Gordon equation, and Klein–Gordon equation as well as inverse problems, including Calderón’s problem, an inverse wave scattering problem, and a problem from seismic imaging full waveform inversion. Both CHONKNORIS and FONKNORIS achieve machine precision in these cases, demonstrating their robustness and accuracy. Notably, FONKNORIS attained near machine precision on the Klein–Gordon and Sine–Gordon equations, despite not being trained on these equations, demonstrating remarkable generalization capabilities.

Despite these advances, our method is still constrained by the computational cost and occasional ill-conditioning that arise when computing and storing Cholesky factors

in high dimensions. In practice, avoiding explicit computation and inversion of the Fréchet derivative is crucial. For example, rather than forming the derivative exactly, one can use the approximation described in Section E.6. Additional improvements in high-dimensional scalability depend on the surrogate model used to solve the linear subproblem. We note that CHONKNORIS and FONKNORIS are agnostic to the underlying machine learning algorithm used to approximate the Cholesky factors, which may allow one to solve the computational challenges associated with high-dimensional problems. In particular, this adaptability enables the integration of various deep learning techniques, such as Fourier Neural Operators or Deep Operator Networks, as well as scalability enhancements for Gaussian process regression through inducing points or nested Kriging. In Section E.3, we show that a sparse-Cholesky strategy can scale and accelerate CHONKNORIS for Gaussian-process surrogates, albeit with an accuracy-speed trade-off: faster run-times come at the cost of reduced precision. Even though, we don't make use of the sparse-Cholesky algorithm, we do exploit the rank-revealing and sparsity properties of the max-min ordering; see Section E.3. Further ways of scaling and accelerating CHONKNORIS are by using nested kriging and approximating the Fréchet derivative; see Section E.2 and Section E.6, respectively. We also experimented with learning Newton–Kantorovich increments directly to reduce overhead; however, this has thus far yielded only limited gains. Finally, incorporating multi-fidelity, multi-scale, and hierarchical approaches offer promising directions for future research.

6.6 Code and data availability

An open source Python implementation of all experiments presented in this work is available at <https://github.com/ArasBacho/CHONKNORIS>. We primarily utilize the Python packages PyTorch Paszke et al., 2019, PyTorch Lightning Falcon and The PyTorch Lightning team, 2019, GPyTorch Gardner et al., 2021, and FastGPs A. G. Sorokin, 2023. Our implementation uses double-precision floating point arithmetic for which machine precision is 2.2×10^{-16} .

All synthetic datasets used in the forward and inverse PDE experiments are provided in the repository. For the seismic imaging (full waveform inversion) experiments we use the publicly available OpenFWI dataset C. Deng et al., 2022.

*Chapter 7***ONE-SHOT LEARNING OF STOCHASTIC DIFFERENTIAL
EQUATIONS WITH DATA ADAPTED KERNELS**

7.1 Introduction

Forecasting a stochastic or a deterministic time series is a fundamental problem in, e.g., Econometrics or Dynamical Systems, which is commonly solved by learning and/or inferring a stochastic or a deterministic dynamical system model from the observed data, respectively; see, e.g., Kantz and Schreiber, 1997; Casdagli, 1989; Chattopadhyay et al., 2019; Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz, 2016b; Pathak, Z. Lu, et al., 2017; Nielsen, 2019; Abarbanel, 2012; Kleinhans and Rudolf Friedrich, 2007; Archambeau et al., 2007; Yildiz et al., 2018; Infante et al., 2016; Dridi, Drumetz, and Fablet, 2021; Siegert, Friedrich, and Peinke, 1998; Klimontovich, 1993; Rudolf Friedrich et al., 2011; Dietrich et al., 2021; Oppen, 2019; Hyndman and Athanasopoulos, 2021, among many others.

On the kernel methods to forecasting time series

Among the various learning-based approaches, methods based on kernels hold potential for considerable advantages in terms of theoretical analysis, numerical implementation, regularization, guaranteed convergence, automatization, and interpretability over, e.g., methods based on variants of Artificial Neural Networks (ANNs); see, e.g., Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b; Houman Owhadi, 2022a. In particular, Reproducing Kernel Hilbert Spaces (RKHS) Felipe Cucker and Steve Smale, 2002 have provided a strong mathematical foundations for studying dynamical systems J. Bouvrie and B. Hamzi, 2010b; B. Haasdonk et al., 2018; B. Haasdonk et al., 2021; Giesl et al., 2019; Bittracher et al., 2019; Boumediene Hamzi and Colonius, 2019; Stefan Klus, Nuske, and Boumediene Hamzi, 2020; Stefan Klus, Nuske, et al., 2020; Alexander and Giannakis, 2020; Jake Bouvrie and Boumediene Hamzi, 2012; Jake Bouvrie and Boumediene Hamzi, 2017b; Jake Bouvrie and Boumediene Hamzi, 2017c; Boumediene Hamzi, Kuehn, and Sameh Mohamed, 2019 and surrogate modeling (see, e.g., Gabriele Santin and Bernard Haasdonk, 2019 for a survey). However, these emulators' accuracy hinges on the kernel's choice; nonetheless, the problem of selecting a good kernel has received less attention so far. Numerical experiments have recently shown that when the time series is regularly Boumediene Hamzi and Houman Owhadi, 2021a or is irregularly sampled Lee, Brouwer, et al., 2021, simple kernel methods can successfully reconstruct the dynamics of prototypical chaotic dynamical systems when kernels are also learned from data via Kernels Flows (KF), a variant of cross-validation H. Owhadi and G. R. Yoo, 2019. KF approach has then been applied to complex, large-scale systems, including geophysical data B. H. , R. M. H. Owhadi,

2021; Prasanth, Z. S. Haddad, et al., 2021; Jouni Susiluoto, Braverman, P. G. Brodrick, et al., 2021, and to learning non-parametric kernels for dynamical systems M. Darcy, B. Hamzi, Susiluo, et al., 2021.

On the learning of Stochastic Differential Equations (SDEs)

While time series produced by deterministic dynamical systems offer a direct observation of the vector-field (i.e., of the drift) driving those systems, those produced by SDEs only present an indirect observation of the underlying drift, diffusion, and stochastic forcing terms. A popular approach employed to recover the drift and the diffusion of an SDE is the so-called Kramers-Moyal expansion; see, e.g., Risken and Haken, 1989; Rudolf Friedrich et al., 2011. In this manuscript, we formulate the problem of learning stochastic dynamical systems described by SDEs as that of completing a computation graph Houman Owhadi, 2022a, which represents the functional dependencies between the observed increments of the time-series and the unknown quantities. Our approach to solving this Computational Graph Completion (CGC) problem can be summarized as (1) replacing unknown functions and variables with Gaussian Processes (GPs) and (2) approximating those functions by the Maximum a Posteriori (MAP) estimator of those GPs given available data. The covariance kernels of these GPs are learned from data via a randomized cross-validation procedure.

Outline of the article

Section 7.2 describes the problem we focus on in this manuscript and our proposed solution. Section 7.3 describes the MAP estimator for the GPs in the one dimensional case. Section 7.4 describes the algorithm we propose to learn the kernels and the hyper-parameters. Section 7.5 provides numerical results, with an additional discussion of the impact of the choice of kernels in Section 7.6 and the impact of time discretization in Section 7.6. We conclude with a brief discussion in Section 7.8. Section F.1 displays additional plots.

7.2 Statement of the problem and proposed solution

We first describe the type of SDEs used here. We consider SDEs of the form:

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t \quad (7.1)$$

with initial condition $X_0 = x_0$. In the previous equation, $(W_t)_{t \in [0, T]}$ denotes a Wiener process. We assume that the process in Equation (7.1) is observed at discrete times $t_n, n = 1 \dots N$, such that the time intervals $\Delta t_n := (t_{n+1} - t_n)$ between observations

$X_n := X_{t_n}$ of the time series are not small enough so that $\sigma(X_t)$ can be efficiently approximated by estimating the quadratic variation of X_t near t , but small enough so that the following approximation holds:

$$X_{n+1} = X_n + f(X_n)\Delta t_n + \sigma(X_n)\sqrt{\Delta t_n}\xi_n + \varepsilon_n, \quad (7.2)$$

where the *i.i.d.* random variables $\xi_n \stackrel{d}{\sim} \mathcal{N}(0, 1)$ represent Brownian Motion increments and the *i.i.d.* random variables $\varepsilon_n \stackrel{d}{\sim} \mathcal{N}(0, \lambda)$ represent discretization noise/misspecification¹; henceforth, the notation “ $\stackrel{d}{\sim}$ ” stands for “distributed as”. We seek to recover/approximate the unknown functions f and σ from the data $(X, Y) = \{(X_n, Y_n)\}_{1 \leq n \leq N}$, where

$$Y_n := X_{n+1} - X_n.$$

Therefore, the relation between X_n and Y_n is given by our modeling assumption

$$Y_n = f(X_n)\Delta t_n + \sigma(X_n)\sqrt{\Delta t_n}\xi_n + \varepsilon_n. \quad (7.3)$$

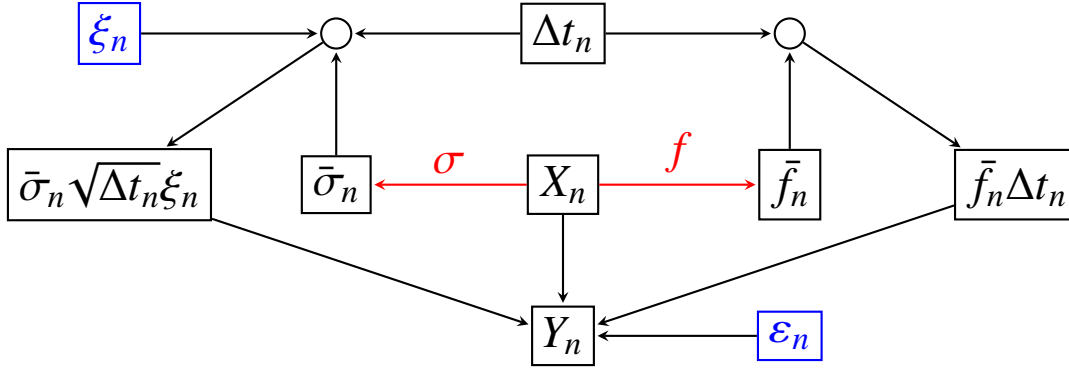
The Computational Graph Completion problem

In general, a *computational graph* is defined as a graph representing functional dependencies between a finite number of (not necessarily random) variables and functions. We will use nodes to represent variables and arrows to represent functions. We will color known functions in black and unknown functions in red. Random variables are drawn in blue and primary variables as squares. We will distinguish nodes used to aggregate variables by drawing them as circles. Multiple incoming arrows into a square node are interpreted as a sum. Now, let $\bar{f}_n := f(X_n)$ and $\bar{\sigma}_n := \sigma(X_n)$ be two intermediate (unobserved) variables. Equation (7.3) can thus be rewritten as:

$$Y_n = \bar{f}_n\Delta t_n + \bar{\sigma}_n\sqrt{\Delta t_n}\xi_n + \varepsilon_n. \quad (7.4)$$

In particular, it can be represented as the following computational graph:

¹While this assumption is not well justified by theory, the Euler-Maruyama method yields a practical scaling for the variance λ of the error. Because the Euler-Maruyama method has a strong order of convergence $O(\Delta t)$, it is therefore reasonable to choose $\lambda = C\Delta t$. In practice, we find that the constant C should be small and use $C = 0.01$.



We now formulate the learning problem in this manuscript as the problem of completing the just displayed computational graph; see Houman Owhadi, 2022a. Let $X_1, \dots, X_N$, $Y_1, \dots, Y_N$ and $\Delta t_1, \dots, \Delta t_N$ be the N observations data: our goal is to approximate the unknown functions f and σ from these observations. In order to solve this problem, we will use the GP framework: we replace σ and f with GPs and approximate them via MAP estimation given the data. More precisely, we assume that f and σ are mutually independent GPs, with centered Gaussian priors $f \stackrel{d}{\sim} \mathcal{GP}(\mathbf{0}, \mathbf{K})$, $\sigma \stackrel{d}{\sim} \mathcal{GP}(\mathbf{0}, \mathbf{G})$ defined by the covariance functions/kernels $\mathbf{K}$ and $\mathbf{G}$.

Remark 17. *Although in this case the computational graph serves mostly as a visual aid, its underlying formalism (i.e., draw the graph, replace unknown functions by GPs, and compute their MAP from the data to complete the graph) is a simple pathway to generalize the method to learning more complex systems than SDEs.*

7.3 MAP estimator

Write $\bar{f}$ for the vector with entries $\{\bar{f}_n\}_{1 \leq n \leq N}$ and $\bar{\sigma}$ for the vector with entries $\{\bar{\sigma}_n\}_{1 \leq n \leq N}$. Observe that given $\bar{\sigma}$ and $\bar{f}$, the identification of the functions f and σ reduces to two separate simple kernel regression problems. We will therefore first focus on the estimation of $\bar{f}$ and $\bar{\sigma}$. Since f and σ are independent, $\bar{f} = f(X)$ and $\bar{\sigma} = \sigma(X)$ are conditionally (on X) independent. Using the shorthand notations $p(Y|X)$ for $p(A|X)$ where A is the event $\{Y_n = f(X_n)\Delta t_n + \sigma(X_n)\sqrt{\Delta t_n}\xi_n + \varepsilon_n\}$, for $1 \leq n \leq N\}$, we deduce that

$$p(\bar{f}, \bar{\sigma}|Y, X) = p(Y|\bar{f}, \bar{\sigma}, X) \frac{p(\bar{f}|X)p(\bar{\sigma}|X)}{p(Y|X)}.$$

It follows that a MAP estimator of $(\bar{f}, \bar{\sigma})$ is a minimizer of the loss $-\ln(p(Y|X, \bar{f}, \bar{\sigma})p(\bar{f}|X)p(\bar{\sigma}|X))$, which up to a constant ($\log \det K(X, X) + \log \det G(X, X)$) and a multiplicative factor

1/2 is equal to

$$\begin{aligned} \mathcal{L}(\bar{f}, \bar{\sigma}) := & (Y - \Lambda \bar{f})^T (\Sigma + \lambda I)^{-1} (Y - \Lambda \bar{f}) + \sum_{n=1}^N \ln(\bar{\sigma}_n^2 \Delta t_n + \lambda) \\ & + \bar{f}^T K(X, X)^{-1} \bar{f} + \bar{\sigma}^T G(X, X)^{-1} \bar{\sigma}. \end{aligned} \quad (7.5)$$

where $K(X, X)$ is the $N \times N$ matrix with entries $K(X_i, X_j)$, $G(X, X)$ is the $N \times N$ matrix with entries $G(X_i, X_j)$, Σ is the diagonal $N \times N$ matrix with diagonal entries $\bar{\sigma}_n^2 \Delta t_n$, and Λ is the diagonal $N \times N$ matrix with diagonal entries Δt_n .

Recovery of f .

First, observe that given $\bar{\sigma}$, (7.5) is quadratic in $\bar{f}$ and its minimizer in $\bar{f}$ is

$$\bar{f}^*(\bar{\sigma}) = K(X, X) \Lambda \left(\Lambda K(X, X) \Lambda + \Sigma + \lambda I \right)^{-1} Y. \quad (7.6)$$

Therefore $f \stackrel{d}{\sim} \mathcal{GP}(\mathbf{O}, \mathbf{K})$ conditioned on $(X, f(X) = \bar{f})$ is normally distributed with (conditional) mean

$$f^*(x) := K(x, X) \Lambda \left(\Lambda K(X, X) \Lambda + \Sigma + \lambda I \right)^{-1} Y, \quad (7.7)$$

and (conditional) covariance

$$K(x, x) - K(x, X) \Lambda \left(\Lambda K(X, X) \Lambda + \Sigma + \lambda I \right)^{-1} \Lambda K(X, x). \quad (7.8)$$

We therefore estimate f with $f^* = (7.7)$. Note that (7.7) and (7.8) can also be recovered by observing that given $\bar{\sigma}$, Equation (7.4) corresponds to a noisy regression problem, with noise coming from two independent Gaussian variables. This is proved in Appendix F.2 and it is an easy modification of the proof presented in Bishop, 2006, Page 306.

Recovery of σ .

Taking $\bar{f} = \bar{f}^* = (7.7)$ in (7.5), the estimation of $\bar{\sigma}$ reduces to the minimization of the loss

$$\begin{aligned} \mathcal{L}(\bar{f}^*(\bar{\sigma}), \bar{\sigma}) = & (Y - \Lambda \bar{f}^*(\bar{\sigma}))^T (\Sigma + \lambda I)^{-1} (Y - \Lambda \bar{f}^*(\bar{\sigma})) + \sum_{n=1}^N \ln(\bar{\sigma}_n^2 \Delta t_n + \lambda) \\ & + \bar{f}^*(\bar{\sigma})^T K(X, X)^{-1} \bar{f}^*(\bar{\sigma}) + \bar{\sigma}^T G(X, X)^{-1} \bar{\sigma}, \end{aligned} \quad (7.9)$$

Write $\bar{\sigma}^\dagger$ for a minimizer of (7.9) obtained through a numerical optimization algorithm (e.g., gradient descent). Because the numerical approximation of $\bar{\sigma}^\dagger$ is noisy, we then further estimate $\bar{\sigma}$ as the mean of the Gaussian vector $\sigma(X)$ conditioned on $\sigma(X) = \bar{\sigma}^\dagger + Z$ where the entries Z_i of the (noise) vector Z are *i.i.d.* Gaussian with variance γ . We therefore approximate $\bar{\sigma}$ with

$$\bar{\sigma}^* = \mathbb{E}[\sigma(X) | \sigma(X) + Z = \bar{\sigma}^\dagger] = G(X, X)(G(X, X) + \gamma I)^{-1} \bar{\sigma}^\dagger, \quad (7.10)$$

and σ with

$$\sigma^*(x) = G(x, X)(G(X, X) + \gamma I)^{-1} \bar{\sigma}^\dagger.$$

Minimization of $\mathcal{L}$

The natural approach to identification of $\bar{\sigma}^\dagger$ is to minimize (with respect to $\bar{\sigma}$) (7.9) with $\bar{f}^*$ defined as the function (7.6) of $\bar{\sigma}$. The minimization of (7.9) is difficult given that this is a non-convex, non-linear problem. We, therefore use a gradient descent algorithm with a step size chosen such that the perturbation is no more than $p\%$ in norm. We set $p = 1$ initially and increment to $p \leftarrow 0.9p$ if the resulting perturbation does not reduce the loss. We optimize until $p < 10^{-20}$ or a maximum $N = 10^5$ iterations. We also use Newton method with step-size chosen using Armijo's condition Nocedal and Stephen J. Wright, 2006c until $\|\nabla \mathcal{L}\| < 10^{-8}$ for a maximum of $N = 1000$ iterations. The gradient descent algorithm is slow compared to Newton's method (10-20 seconds compared to 10-15 minutes in wall clock time) but consistently produces better results and is less sensitive to changes in the initial condition. In our gradient descent algorithm, we ensure that the loss is decreased at every step and we therefore converge to a local minimum. However, there is no guarantee that we converge to the global minima since the loss is non-convex.

Initial condition.

The initialization of the optimization problem requires the specification of an initial condition for $\bar{\sigma}_{\text{init}}$. We use an estimate of quadratic variation of the process given the data:

$$(\bar{\sigma}_{\text{init}})_i = \frac{(X_i - X_{i-1})^2}{\Delta t_i}. \quad (7.11)$$

We smooth out this noisy estimate using the mean of the Gaussian process σ conditioned on these values:

$$\bar{\sigma}_{\text{init}} = \mathbb{E}[\sigma(X) | \sigma(X) + Z = \bar{\sigma}_{\text{init}}] = G(X, X)(G(X, X) + \gamma I)^{-1} \bar{\sigma}_{\text{init}}. \quad (7.12)$$

Extension to the multivariate case

The extension to the multivariate case is relatively straightforward and is presented in Appendix F.5.

7.4 Learning the kernels and the hyperparameters

Motivation

The computational graph completion approach relies on a choice of prior and hence on a choice of kernel. There are many possible such choices that encode varying priors on the functions f and σ . For example isotropic kernels such as the Gaussian and Matérn kernel are used to model the covariance functions of stationary processes Rasmussen and Williams, 2005. Dot product-based kernels, such as the polynomial kernel, on the other hand are non-stationary Rasmussen and Williams, 2005. Moreover, each family of kernels is parameterized by one or several parameters, which can have a large impact on the recovery of the function. This motivates the development of a method to select the optimal kernel. In the next section, we present a cross-validation-based method to select the best kernel based on the data. This method is able to not only select the best parameter but also enables ill-specified kernels to perform comparably to well-specified kernels.

To illustrate this point, consider the Geometric Brownian Motion

$$dX_t = \mu X_t dt + \sigma X_t dW_t \quad (7.13)$$

which has both non-stationary drift and diffusion functions. In section 7.6, we will show (see figure 7.1) how our hyper-parameter learning algorithm enables a non-perfectly-adapted kernel (e.g., a Matérn kernel for approximating the drift and volatility of (7.13)) to perform comparably to a perfectly adapted kernel (e.g., a linear kernel for approximating the drift and volatility of (7.13)). We also note that learning the kernel improves out-of-sample predictions.

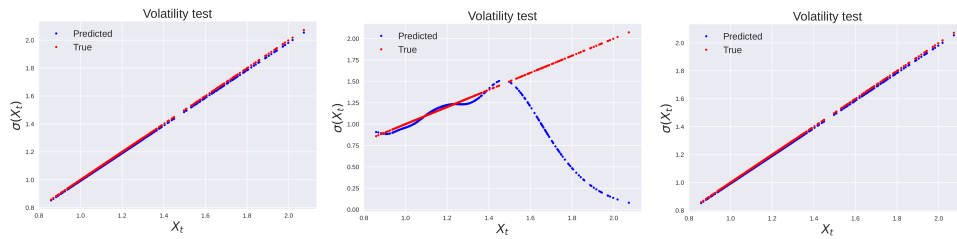


Figure 7.1: Predicted volatility: well-specified kernel (left), ill specified kernel (middle), and well-specified kernel with learned parameters (right).

Methodology

We now describe how to select the kernels K, G in a family of kernels parameterized by θ_k, θ_g , which we learn from data using a cross-validation approach. Writing

$$\boldsymbol{\theta} := (\theta_k, \theta_g), \quad (7.14)$$

for the vector formed by all the hyperparameters of our approach, we learn θ through a robust-learning cross-validation approach which we will now describe. Consider the set of all sets of possible partitions of the training data $\mathcal{D}_{\mathcal{T}}$ with indices $\mathcal{I}$ into two mutually disjoint subsets of equal size.

$$\mathcal{A} = \{(\Pi, \Pi^c) | \Pi \cup \Pi^c = \mathcal{I}, \Pi \cap \Pi^c = \emptyset, |\Pi| = \frac{|\mathcal{I}|}{2}\}. \quad (7.15)$$

Write $\mathcal{D}_{\Pi} = (x_j, y_j)_{j \in \Pi}$ for the set of points belonging to the first partition and $\mathcal{D}_{\Pi^c} = (x_j, y_j)_{j \in \Pi^c}$ for the set of points belonging to the second set. Write $\mathbb{E}_{(\Pi, \Pi^c)}$ for the uniform distribution over $\mathcal{A}$. For $(\Pi, \Pi^c) \in \mathcal{A}$ write $\mathcal{L}(\mathcal{D}_{\Pi^c}, \bar{f}, \bar{\sigma})$ for the MAP loss (7.5) calculated with dataset $\mathcal{D}_{\Pi^c}$. Write

$$\begin{aligned} \mathcal{L}_{\text{CV}}(\boldsymbol{\theta}; \bar{f}^*, \bar{\sigma}^*, \mathcal{D}_{\Pi}) &= -\ln p(Y_{\Pi} | \bar{f}^*, \bar{\sigma}^*, X_{\Pi}) \\ &= \sum_i \frac{(Y_i - \bar{f}_i^* \Delta t_i)^2}{2(\bar{\sigma}_i^{*2} \Delta t_i + \lambda)} + \frac{1}{2} \ln(\bar{\sigma}_i^{*2} \Delta t_i + \lambda), \end{aligned} \quad (7.16)$$

for the negative log-likelihood of the validation data $\mathcal{D}_{\Pi}$ given $\bar{f}^*, \bar{\sigma}^*$. Our proposed cross-validation approach is then to select $\boldsymbol{\theta}^*$ as

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \min_{\boldsymbol{\theta}} \mathbb{E}_{(\Pi, \Pi^c)} \{\mathcal{L}_{\text{CV}}(\boldsymbol{\theta}; \bar{f}^*, \bar{\sigma}^*, \mathcal{D}_{\Pi})\} \\ \text{subject to } \bar{f}^*, \bar{\sigma}^* &= \arg \min_{\bar{f}, \bar{\sigma}} \mathcal{L}(\mathcal{D}_{\Pi^c}, \bar{f}, \bar{\sigma}). \end{aligned} \quad (7.17)$$

Note that $\bar{f}^*, \bar{\sigma}^*$ are selected as described in Section 7.3. In practice, the computation of the exact value of $\mathbb{E}_{(\Pi, \Pi^c)} \{\mathcal{L}_{\text{CV}}(\boldsymbol{\theta}; \bar{f}^*, \bar{\sigma}^*, \mathcal{D}_{\Pi})\}$ is intractable. Therefore, we instead approximate $\mathbb{E}_{(\Pi, \Pi^c)} \{\mathcal{L}_{\text{CV}}\}$ with the empirical average

$$\frac{1}{M} \sum_{i=1}^M \mathcal{L}_{\text{CV}}(\boldsymbol{\theta}; \bar{f}_i^*, \bar{\sigma}_i^*, \mathcal{D}_{\Pi_i}). \quad (7.18)$$

where the (Π_i, Π_i^c) are i.i.d. samples from $\mathbb{P}_{(\Pi, \Pi^c)}$ and $\bar{f}_i^*, \bar{\sigma}_i^* = \arg \min_{\bar{f}, \bar{\sigma}} \mathcal{L}(\mathcal{D}_{\Pi_i^c}, \bar{f}, \bar{\sigma})$.

We use a gradient free optimization algorithm to minimize (7.18) (see Subsection 7.4). This algorithm only uses noisy observations (7.18) of the true loss.

The proposed cross-validation algorithm can be summarized as follows:

1. Select a gradient-free optimization algorithm.
2. At each iteration, given the hyperparameters θ^k , select M divisions of the data $\mathcal{D}_{\mathcal{T}}$ into a training set $\mathcal{D}_{\Pi_i^c}$ and a validation set $\mathcal{D}_{\Pi_i}$.
3. For each $1 \leq i \leq M$, recover $\tilde{f}_i^*, \tilde{\sigma}_i^*$ using training data $\mathcal{D}_{\Pi_i^c}$ using hyperparameters θ^k .
4. For each $1 \leq i \leq M$, compute the loss $\mathcal{L}(\theta^k; \tilde{f}_i^*, \tilde{\sigma}_i^*, \mathcal{D}_{\Pi_i})$ and the empirical average (7.18).
5. Minimize (7.18) with the gradient free optimization algorithm to select θ^{k+1} .

The active learning algorithm

We choose the Bayesian optimization algorithm Snoek, Larochelle, and Adams, 2012, where the loss function is modeled using a Gaussian Process with Matern kernel Rasmussen and Williams, 2005, implemented in the scikit-optimize library in Python *Bayesian optimization with skopt* n.d. In our case, we set $M = 1$ when using gradient descent minimization and $M = 10$ when using Newton's method for minimization. The maximum number of iterations for the Bayesian optimization algorithm is set to $K = 75$ when using gradient descent minimization and $K = 150$ when using Newton's method for minimization. While we can use a greater number of samples and a greater number of iterations with Newton's method because of its greater speed, in practice, the gradient descent method offered better performance in our tests.

7.5 Experimental results

We first illustrate the effectiveness of our methods by considering two systems with non-linear drift or volatility. The two processes we consider are:

$$dX_t = \mu X_t dt + b \exp(-X_t^2) dW_t \quad \text{Exponential decay volatility.} \quad (7.19)$$

$$dX_t = \sin(2k\pi X_t) dt + b \cos(2k\pi X_t) dW_t \quad \text{Trigonometric process.} \quad (7.20)$$

In both cases, we generate trajectories using a Euler-Maruyama discretization. We use 500 points for training and 500 points for testing.

Metrics.

To measure the performance of each method, we use three metrics. The first is the likelihood of the model given the data of the test set defined as

$$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* | X, Y) = -\log(p(Y | \bar{f}^*, \bar{\sigma}^*, X)) \propto \sum_{i=1}^M \frac{(Y_i - \bar{f}^* \Delta t_i)^2}{2(\bar{\sigma}_i^{*2} \Delta t_i + \lambda)} + \frac{1}{2} \ln(\bar{\sigma}_i^{*2} \Delta t_i + \lambda).$$

The other two metrics are the relative error of the test drift and volatility at the test points:

$$\delta_f = \frac{\|f - \bar{f}\|}{\|f\|}$$

$$\delta_\sigma = \frac{\|\sigma - \bar{\sigma}\|}{\|\sigma\|},$$

where f is the vector of drift values at the test points $(f)_i = f(X_i)$ and $\bar{f}$ is the vector of prediction $(\bar{f})_i = \bar{f}(X_i)$ (likewise for $\sigma, \bar{\sigma}$). Note that in practice, only $\mathcal{L}(\bar{f}^*, \bar{\sigma}^* | X, Y)$ may be computed without access to the true drift f and true volatility σ . We still compute δ_f, δ_σ to illustrate how a lower loss on the recovery of the drift and volatility yields a lower loss on the likelihood.

Choice of kernels

In all experiments, we use the Matérn Kernel Rasmussen and Williams, 2005 with smoothness parameter $\nu = \frac{5}{2}$ defined as

$$K_{\text{Matérn}}(x, y) = \sigma^2 \left(1 + \frac{\sqrt{5} \|x - y\|}{l} + \frac{5 \|x - y\|^2}{3l^2} \right) \exp \left(- \frac{\sqrt{5} \|x - y\|}{l} \right). \quad (7.21)$$

We learn the parameters (σ, l) of the kernel (the smoothness parameter ν is not learned). Note that the Matérn kernel is not well specified for many of the processes we will consider as the processes it defines are only three times differentiable (in the mean square sense Rasmussen and Williams, 2005) and hence it is overly general. A better specified of kernel can in many cases significantly improve the performance. We choose this kernel because it is very general and allows us to illustrate how one can obtain good results with few assumptions and little domain knowledge.

Benchmarks.

We compare our method and optimized parameters with two baselines. A first baseline that uses our method and unoptimized parameters. For kernels using a lengthscale (such as the Matérn kernel (7.21)), it is set to be the average ℓ^2 distance

between data points:

$$l_{\text{unopt}} = \frac{1}{N(N-1)} \sum_{i \neq j} \|X_i - X_j\|.$$

All other paramters are set to 1.0. This method is labeled as “non-learned kernel”.

The second baseline does not use our method to recover the drift and diffusion separately. Instead we assume that Y is the sum of two Gaussian processes:

$$y(x) = \xi(x) + W(x), \quad (7.22)$$

where ξ is a smooth Gaussian process (such as a Gaussian process with Matérn covariance function) and W is a white noise Gaussian process with covariance matrix

$$K_{\text{wn}}(x_i, x_j) = c\delta(x_i - x_j). \quad (7.23)$$

The posterior distribution y conditioned on the data provides a prediction for the conditional mean (the drift of the SDE) and the conditional variance (the volatility of the SDE)². The parameters of the kernel are optimized through the minimization of the negative log marginal likelihood. This method is labeled as “benchmark” and uses the implementation present in Pedregosa et al., 2011 and the details are presented in Appendices F.2 and F.3. Note that a major drawback of this method is the assumption that the noise $\sigma(X_t)$ is identically distributed Gaussian noise, modeled through the white noise kernel $\delta(x_i - x_j)$. This assumption is valid for only a restricted class of SDEs.

Exponential decay volatility.

The discretization of the exponential decay volatility is given by

$$X_{n+1} - X_n = \mu X_n \Delta t + b \exp(-X_n^2) \sqrt{\Delta t} \xi_n, \quad X_0 = x_0, \quad (7.24)$$

with timestep $\Delta t = 0.01$. Observe that this process is pushed towards the origin by its drift value, where the volatility is maximized. Moreover, the volatility decreases exponentially fast away from the origin. We generate two trajectories with the same drift parameter $\mu = 5$ and different volatility parameter $b = 0.5, 1.0$. The results for the learned and unlearned kernel are reported in table 7.1. The training and testing data are illustrated in Figure 7.2 and the prediction of the drift and volatility are presented in Figure 7.3 (see also Figures F.1, F.2 in the appendix).

²Note that this is the usual Gaussian process regression method

	Trajectory 1			Trajectory 2		
	$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* X, Y)$	δ_f	δ_σ	$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* X, Y)$	δ_f	δ_σ
Benchmark	-2.547	0.394	0.033	-1.887	0.322	0.085
Non-learned kernel	-2.532	0.506	0.081	-1.887	0.472	0.062
Learned kernel	-2.546	0.388	0.048	-1.900	0.249	0.035

Table 7.1: Results for the exponential volatility process.

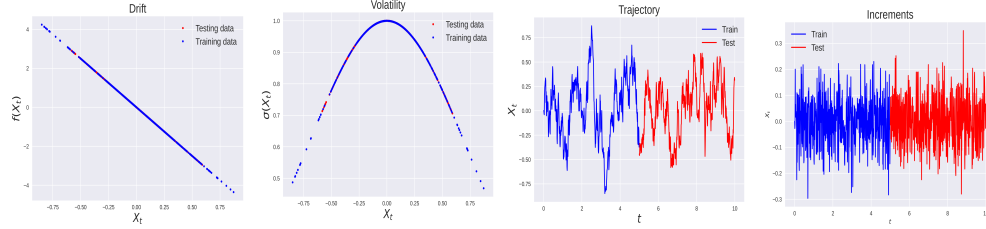


Figure 7.2: From left to right: drift function, volatility function, sample trajectory, and sample increments of the exponential volatility process.

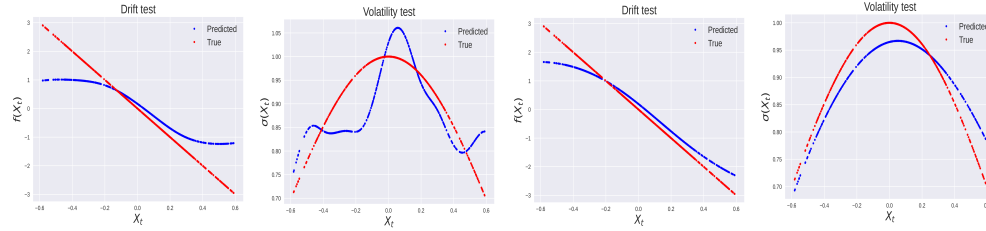


Figure 7.3: Predicted drift and volatility on the testing set for trajectory 2 of the exponential volatility process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel) .

Trigonometric process

The discretization of the trigonometric process is given by

$$X_{n+1} - X_n = \sin(2k\pi X_t)\Delta t + b \cos(2k\pi X_t)\sqrt{\Delta t}\xi_n, \quad X_0 = x_0. \quad (7.25)$$

In this case, both the drift and volatility functions are non-linear functions of X_t . We generate two trajectories with volatility parameter $b = 1.0, 0.5$. For $b = 0.5$, we set $\Delta t = 0.01$. For the second trajectory, $b = 0.5$ and the timestep is set $\Delta t = 0.001$. In this case, the testing data contains points outside the training distribution. Hence the problem of prediction is more challenging than the problem of recovery at the training points. The results for the learned and unlearned kernel are reported in Table 7.2. The training and testing data are illustrated in Figure 7.4. For trajectory 2, the optimization of the hyper-parameters of the kernel yield a better recovery and

a better prediction of the volatility outside the training distribution 7.4, 7.5 (also see Figure 7.5 in the appendix).

	Trajectory 1			Trajectory 2		
	$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* X, Y)$	δ_f	δ_σ	$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* X, Y)$	δ_f	δ_σ
Benchmark	-3.411	0.327	0.442	-3.808	3.454	0.198
Non-learned kernel	-3.675	0.157	0.093	-2.405	0.832	0.665
Learned kernel	-3.678	0.269	0.088	-3.642	1.481	0.242

Table 7.2: Results for the trigonometric process.

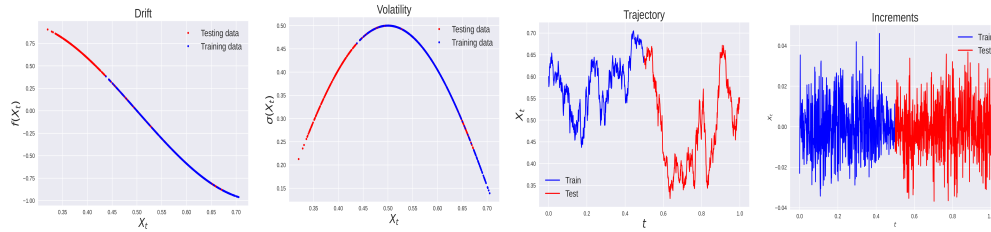


Figure 7.4: From left to right: drift function, volatility function, sample trajectory and sample increments of the trigonometric process (trajectory 2).

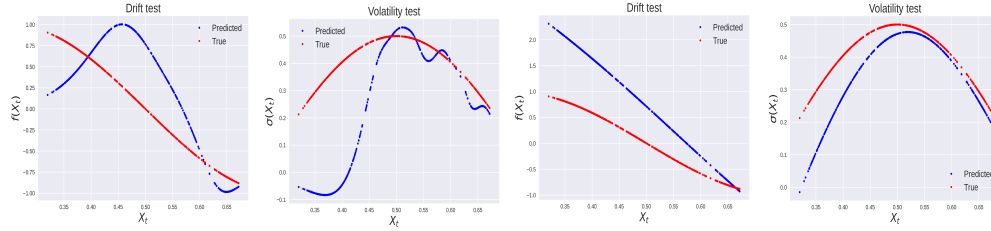


Figure 7.5: Predicted drift and volatility on the testing set for trajectory 2 of the trigonometric process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel) .

Benchmark comparison

Generally our method with optimized parameters outperforms our selected benchmark both in recovery of the drift and the volatility as measured by our metric. Because the standard Gaussian process regressor uses a white noise kernel, it generally does not capture well a non-constant volatility. Our method on the other hand is able to capture non-constant volatility models. Compare for example figures 7.5 and 7.7.

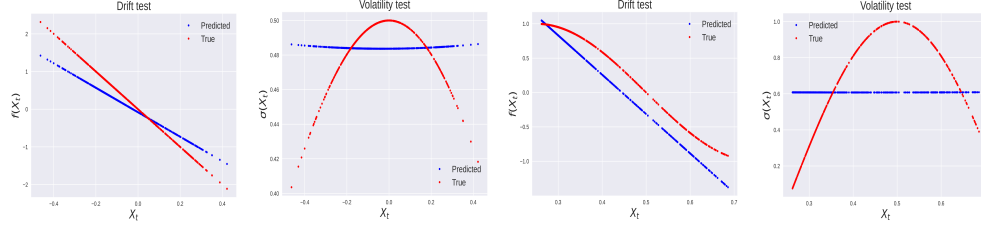


Figure 7.6: Exponential volatility process benchmark prediction. From left to right: drift (trajectory 1), volatility (trajectory 1), drift (trajectory 2), volatility (trajectory 2).

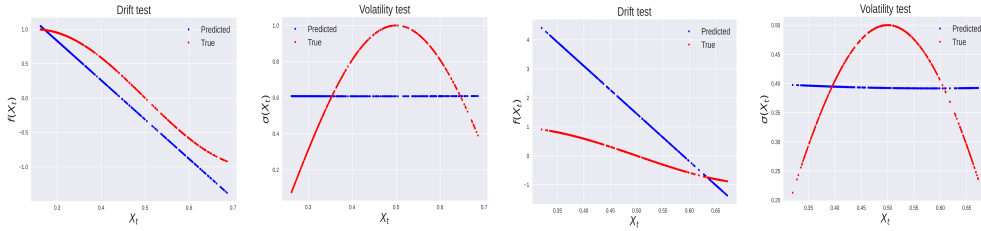


Figure 7.7: Trigonometric process benchmark prediction. From left to right: drift (trajectory 1), volatility (trajectory 1), drift (trajectory 2), volatility (trajectory 2).

7.6 Experimental results: a comparison between perfectly specified and non perfectly specified kernels.

We now illustrate how the choice of the kernel can have a significant impact on the recovery of the drift and volatility. The following examples illustrate how choosing from the correct parametric family of kernels can improve both the recovery and the prediction of the functions of interest.

Overcoming non perfectly specified kernels

We consider the Geometric Brownian Motion process defined as

$$dX_t = \mu X_t dt + \sigma X_t dW_t \quad \text{Geometric Brownian motion (GBM)}. \quad (7.26)$$

We generate one trajectory with parameters $\mu = 2.0, \sigma = 1.0$ and initial condition $X_0 = 1.0$, using the Euler-Maruyama discretization with timestep $\Delta t = 0.001$. We focus primarily on the recovery of the volatility (the recovery of the drift is very difficult at a fine time-scale). The generated data is illustrated in figure 7.8.

We compare the performance of the Matern kernel (7.21) family with the family of linear kernels Rasmussen and Williams, 2005 defined as

$$K_{\text{linear}}(x, y) = \sigma^2(x^\top y + c) \quad (7.27)$$

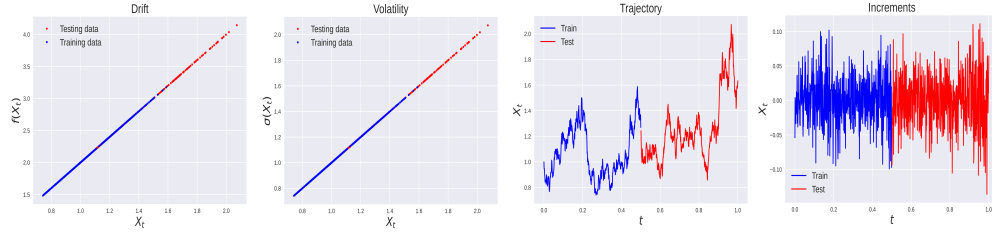


Figure 7.8: From left to right: drift function, volatility function, sample trajectory, and sample increments of GBM.

and parameterized by (σ, c) which are learned. Note that this kernel induces the Reproducing Kernel Hilbert Space of linear functions and therefore is perfectly specified for GBM. The results are reported in table 7.3. The linear kernel is able to both recover and predict the volatility with or without learning the hyper-parameters, as it is perfectly specified for this problem. In contrast, without learning the hyper-parameters, the Matérn kernel is unable to accurately predict the future, reverting to the prior mean 0. Learning the hyper-parameters, however, enables the Matérn kernel to correctly predict future values. These results are illustrated in figure 7.9. We also observe that in all cases, our proposed approach outperforms the Gaussian Process Regressor benchmark (see figure 7.10).

	Linear kernel			Matern kernel		
	$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* X, Y)$	δ_f	δ_σ	$\mathcal{L}(\bar{f}^*, \bar{\sigma}^* X, Y)$	δ_f	δ_σ
Benchmark	-2.755	2.077	0.236	-2.755	2.018	0.237
Non-learned kernel	-2.800	0.763	0.015	-1.420	0.962	0.355
Learned kernel	-2.800	0.672	0.008	-2.800	0.500	0.010

Table 7.3: Comparison between the linear kernel and the Matern kernel on GBM.

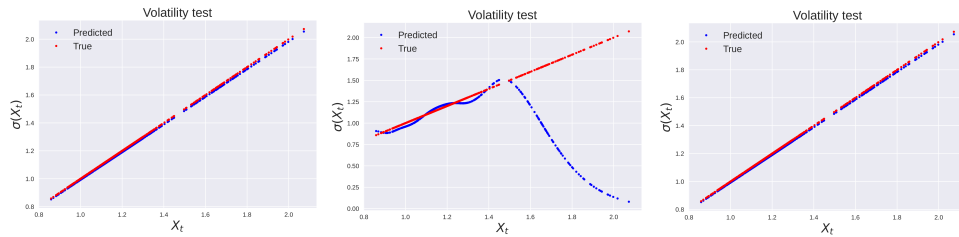


Figure 7.9: Predicted volatility on the testing set for GBM. From left to right: linear kernel, Matern kernel (non-learned parameters), and Matern Kernel (learned parameters).

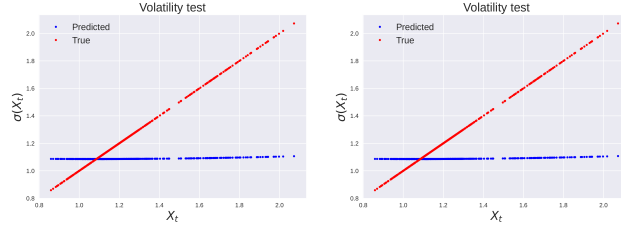


Figure 7.10: GBM benchmark prediction. From left to right: volatility (linear kernel) and volatility (Matern kernel).

A failure case

We now present a case where our methodology fails because of a poor choice of prior. We consider the Ornstein–Uhlenbeck process defined as

$$dX_t = -\mu X_t dt + \sigma dW_t, \quad \text{Ornstein–Uhlenbeck (OU).}$$

We discretize the process with parameters $\mu = -5$ and $\sigma = 1.0$ and initial condition $X_0 = 1.0$ using the Euler-Maruyama method with a time discretization of $\Delta t = 0.001$. One of the trajectories is illustrated in figure 7.11.

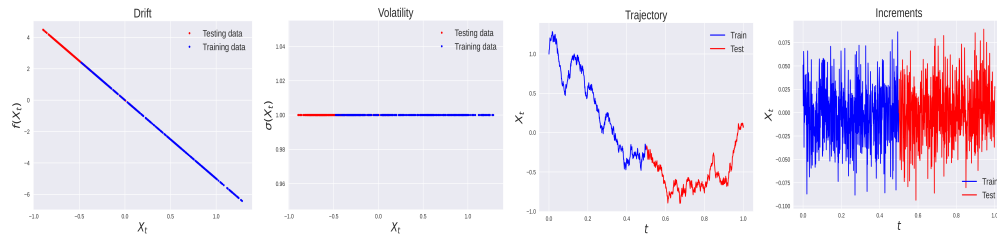


Figure 7.11: From left to right: drift function, volatility function, sample trajectory, and sample increments of the OU process.

In this case, the drift is a linear function, and the volatility is a constant function. Hence a Matérn kernel prior is not perfectly specified. More precisely, the best prior for the volatility is a white noise kernel

$$K_{\text{wn}}(x, y) = \sigma^2 \delta(x - y). \quad (7.28)$$

Therefore, in this case, the benchmark method (Gaussian process regression) is not only better specified than our method, but it is also optimal. We again use both the Matérn 7.21 and linear kernels (7.27). The results are presented in Table 7.4. In this case, the benchmark always outperforms our computational graph completion approach in the recovery of the volatility. With a Matérn kernel our method outperforms the benchmark, but with a perfectly specified linear kernel, the

benchmark outperforms our method. Moreover, Figure 7.12 shows that the Matérn kernel does not capture the overall shape of the drift and volatility even if the relative error is low. We do note, however, that our cross-validation method improves the performance.

	Matérn kernel			Linear kernel		
	$\mathcal{L}(\tilde{f}^*, \tilde{\sigma}^* X, Y)$	δ_f	δ_σ	$\mathcal{L}(\tilde{f}^*, \tilde{\sigma}^* X, Y)$	δ_f	δ_σ
Benchmark	-2.973	1.799	0.006	-2.977	0.212	0.006
Non-learned kernel	-2.971	1.038	0.065	-2.957	0.510	0.011
Learned kernel	-2.973	0.459	0.012	-2.978	1.066	0.013

Table 7.4: Results for the OU process.

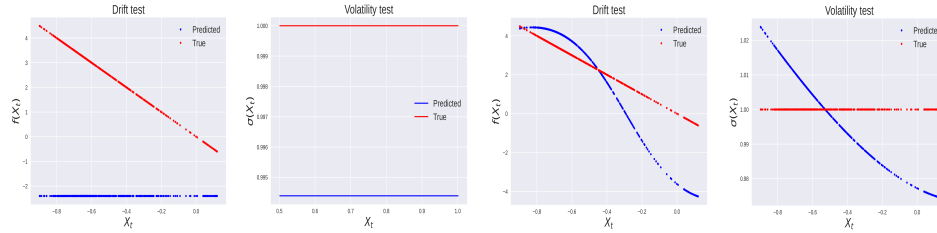


Figure 7.12: Prediction on the OU process with the Matérn kernel. From left to right: benchmark prediction (drift), benchmark prediction (volatility), CGC prediction (drift), CGC prediction (volatility).

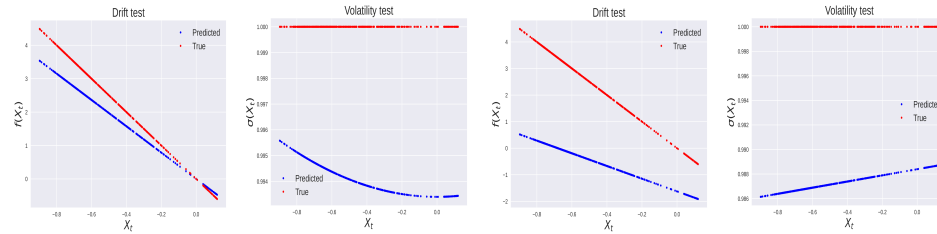


Figure 7.13: Prediction on the OU process with the linear kernel. From left to right: benchmark prediction (drift), benchmark prediction (volatility), CGC prediction (drift), CGC prediction (volatility).

These results illustrate that choosing a perfectly specified kernel can improve the performance of the proposed method. Therefore, integrating prior knowledge such as the non-stationarity of the process into the choice of the prior can significantly improve performance. However, these results also illustrate how learning the parameters of a general kernel (such as the Matérn kernel) can yield similar performance

to a well-specified kernel. A potential solution to this problem could be to learn a kernel which is the sum of a mix of stationary and non-stationary kernels such as

$$K_{\text{sum}}(x, y) = \sum_{i=1}^N \alpha_i K_i(x, y), \quad (7.29)$$

where the weights α_i are learned. Learning the weights of a sum of kernels is a well-established problem in the area of Support Vector Machines (see, for example Lanckriet et al., 2002). Future avenues of research could therefore focus on efficient learning of hyper-parameters for this problem.

7.7 Experimental results: the effect of time discretization

We now examine the effect of time discretization. In all previous experiments, we observed the exact simulation of the trajectory. We now consider the cases where the observations do not exactly match the dynamics. More precisely, we generate a trajectory $(X_i)_{i=1}^M$ separated by regular time steps Δt . We then consider the sub-trajectories $(X_{ik})_{i=1}^N$ for $k = 1, 2, \dots, 10$ where the indexes ik are chosen such that $(X_{ik})_{i=1}^N$ are separated by timesteps $k \times \Delta t$. This allows us to measure the impact of the discretization error on the effectiveness of our method. A priori, we would expect that our method would perform more poorly as k grows larger, given that our modeling assumption is no longer fulfilled exactly.

We consider both our method with the initial guess of parameters (as detailed in the benchmark section 7.5) and with the optimized version of parameters. We also choose a base λ which performs well for $k = 1$ and set $\lambda_k = k\lambda$ for other values of k .

Exponential decay volatility.

We first consider the exponential decay volatility process described by (7.24) with time discretization $\Delta t = 0.01$. For each k , we set $N = 500$ for both the training and test sets. The results for k are presented in table 7.5 and illustrative examples are presented in figures 7.14 and 7.15. We note that the choice of the Matérn kernel for the drift function induces a large error. A better choice of the kernel (such as the linear kernel discussed in section 7.6) would yield a better performance.

	Non optimized parameters			Optimized parameters		
	$\mathcal{L}(\tilde{f}^*, \tilde{\sigma}^* X, Y)$	δ_f	δ_σ	$\mathcal{L}(\tilde{f}^*, \tilde{\sigma}^* X, Y)$	δ_f	δ_σ
k = 1	-1.887	0.472	0.062	-1.887	0.889	0.033
k = 2	-1.872	0.669	0.299	-1.843	0.897	0.181
k = 3	-1.896	0.702	0.419	-1.888	0.881	0.415
k = 4	-1.871	0.886	0.505	-1.922	0.902	0.502
k = 5	37.778	0.805	0.370	-1.928	0.855	0.574
k = 6	-1.802	0.901	0.605	-1.928	0.982	0.590
k = 7	-0.356	0.880	0.641	-1.818	0.865	0.638
k = 8	-1.774	0.913	0.505	-1.496	0.913	0.228
k = 9	-1.044	0.878	0.249	-1.881	0.919	0.659
k = 10	-1.648	0.940	0.439	-1.883	0.952	0.695

Table 7.5: Results for the exponential decay process for different time step Δt observations.

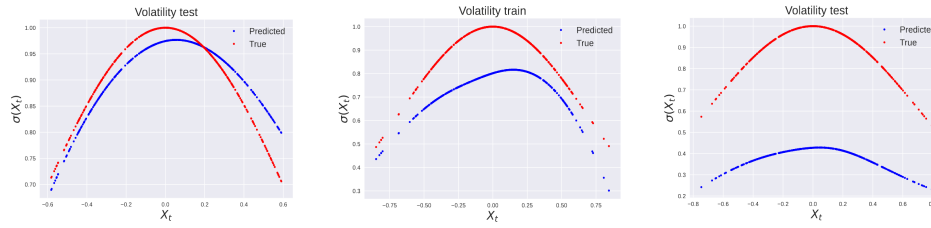


Figure 7.14: Exponential decay volatility process: prediction of the volatility for different values of k . From left to right: $k = 1, 2, 5$.

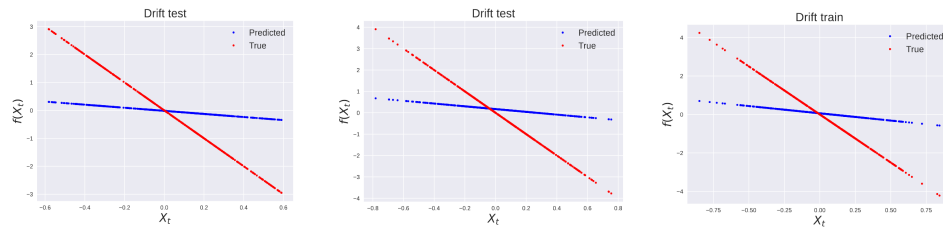


Figure 7.15: Exponential decay volatility process: prediction of the drift for different values of k . From left to right: $k = 1, 2, 5$.

Trigonometric process

We now consider the trigonometric process described by (7.25) with time discretization $\Delta t = 0.001$. For each k , we set $N = 500$ for both the training and test sets. The results for k are presented in table 7.6 and illustrative examples are presented in figures 7.16 and 7.17.

	Non optimized parameters			Optimized parameters		
	$\mathcal{L}(\tilde{f}^*, \tilde{\sigma}^* X, Y)$	δ_f	δ_σ	$\mathcal{L}(\tilde{f}^*, \tilde{\sigma}^* X, Y)$	δ_f	δ_σ
k = 1	-3.675	0.259	0.093	-3.674	0.163	0.092
k = 2	-3.883	0.717	0.300	-3.807	0.792	0.299
k = 3	-3.729	0.702	0.419	-3.725	0.654	0.434
k = 4	-3.784	0.836	0.499	-3.798	0.848	0.495
k = 5	-4.027	0.904	0.587	-4.019	0.852	0.593
k = 6	-3.910	0.911	0.594	-3.942	0.952	0.597
k = 7	-3.909	0.954	0.636	-3.562	0.925	0.211
k = 8	-3.652	0.772	0.668	-3.930	0.767	0.669
k = 9	-3.939	0.880	0.657	-3.424	0.935	0.145
k = 10	-3.505	0.888	0.254	-3.926	0.880	0.705

Table 7.6: Results for the trigonometric process for different time step Δt observations.

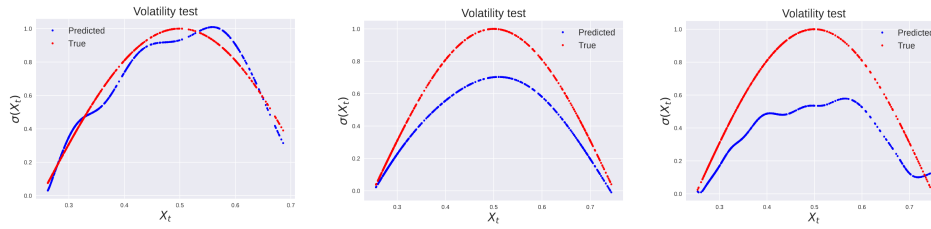


Figure 7.16: Trigonometric process: prediction of the volatility for different values of k . From left to right: $k = 1, 2, 3$.

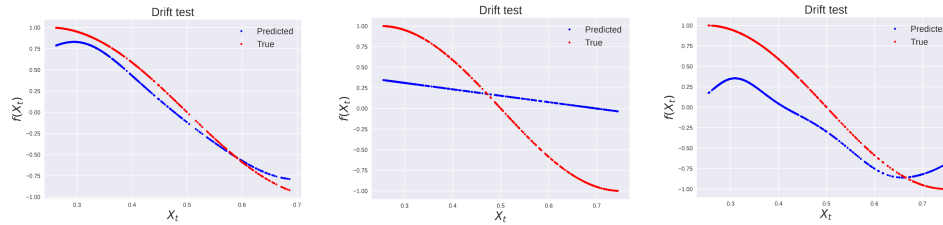


Figure 7.17: Trigonometric process: prediction of the drift for different values of k . From left to right: $k = 1, 2, 5$.

In general, we note that, as expected, as k gets larger, our method has worse performance. Moreover, for values of $k \neq 1$, our cross-validation method does not perform as well, suggesting the need to adapt the cross-validation procedure to learning the parameter λ . There are, however some exceptions, such $k = 8, 9$ for the exponential volatility process and $k = 7, 9$ for the trigonometric process where one of the functions is better recovered. This suggests that a good choice of parameters

via cross-validation leads to a good recovery even if the modeling assumption is incorrect, consistent with observations that cross-validation methods are somewhat robust to model misspecification Y. Chen, Houman Owhadi, and A. Stuart, 2021.

7.8 Experimental results: discussion

We make the following observations regarding our results.

First, we observe that our method provides comparable or greater performance compared to simple kernel regression with hyper-parameters optimized through the minimization of the log marginal likelihood. Which method is preferable depends on the underlying SDE. We observe that for SDEs with constant volatility functions, such as the Ornstein–Uhlenbeck process, the simple kernel regression generally outperforms our method as measured by the likelihood. This is likely due to the better recovery of the volatility as the modeling assumption of the kernel regression (i.i.d. noise) better captures the true model (constant volatility). Nonetheless, our method does occasionally outperform kernel regression in predicting the drift of the SDE. Our method, however, notably outperforms kernel regression for SDEs with non-constant volatility, such as the trigonometric process and the exponential volatility process.

Second, we observe that the randomized cross-validation algorithm for hyper-parameter optimization reliably improves the performance of our method. In all cases, the selected parameters have better performance compared to the initial guess, as measured by the likelihood of the model. Moreover, as the Geometric Brownian Motion example illustrates, learning the parameters of the kernels can enable ill-specified kernels to perform comparably to well-specified kernels.

We also observe that a better likelihood generally implies a better capture of the drift and volatility. Hence, while these quantities are unobserved, better performance as measured by the likelihood generally implies that the model captures well both the drift and volatility.

Finally, we note that as the observations increasingly depart from the true dynamics, our method performs more poorly. This suggests the need for a better model for the discretization error term, either through learning the parameter λ or a better modeling assumption than the *i.i.d* Gaussian noise presented in Section 7.2.

*Chapter 8***LEARNING DYNAMICAL SYSTEMS FROM DATA WITH
NON-PARAMETRIC KERNEL FLOWS**

8.1 Introduction

The ubiquity of time series in many domains of science has led to the development of diverse statistical and machine learning forecasting methods Kantz and Schreiber, 1997; Casdagli, 1989; Chattopadhyay et al., 2019; Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz, 2016b; Pathak, Z. Lu, et al., 2017; Nielsen, 2019; Abarbanel, 2012.

Amongst various learning-based approaches, kernel-based methods hold potential for considerable advantages in terms of theoretical analysis, numerical implementation, regularization, guaranteed convergence, automatization, and interpretability Y. Chen, Hosseini, Houman Owhadi, and Andrew M. Stuart, 2021b; Houman Owhadi, 2022a. Indeed, reproducing kernel Hilbert spaces (RKHS) Felipe Cucker and Steve Smale, 2002 have provided strong mathematical foundations for analyzing dynamical systems Lengyel et al., 2024; Lee, Boumediene Hamzi, et al., 2024; B. Haasdonk et al., 2018; B. Haasdonk et al., 2021; Giesl et al., 2019; Boumediene Hamzi and Colonius, 2019; Stefan Klus, Nuske, and Boumediene Hamzi, 2020; Stefan Klus, Nuske, et al., 2020; Alexander and Giannakis, 2020; Jake Bouvrie and Boumediene Hamzi, 2012; Jake Bouvrie and Boumediene Hamzi, 2017a; Jake Bouvrie and Boumediene Hamzi, 2017c; Boumediene Hamzi, Kuehn, and Sameh Mohamed, 2019; Akian et al., 2022; B. H. , R. M. H. Owhadi, 2021; Jake Bouvrie and Boumediene Hamzi, 2010; Hudson et al., 1990; Y. Lu et al., 2023; Boumediene Hamzi and Houman Owhadi, 2021a; Boumediene Hamzi, Houman Owhadi, and Paillet, 2023; Lu Yang et al., 2024; Boumediene Hamzi and Houman Owhadi, 2021a; Lee, Brouwer, et al., 2021; B. H. , R. M. H. Owhadi, 2021; Prasanth, Z. S. Haddad, et al., 2021; M. Darcy, Tavallali, et al., 2021, and surrogate modeling (cf. Gabriele Santin and Bernard Haasdonk, 2019 for a survey). Yet the accuracy of these emulators depends on the kernel, and the problem of selecting a good kernel has received less attention. Recently, the experiments by Hamzi and Owhadi Boumediene Hamzi and Houman Owhadi, 2021a show that when the time series is regularly sampled, or is irregularly sampled Lee, Brouwer, et al., 2021, kernel flows H. Owhadi and G. R. Yoo, 2019 (an RKHS technique) can successfully reconstruct the dynamics of some prototypical chaotic dynamical systems. KFs have subsequently been applied to complex, large-scale systems, including climate data B. H. , R. M. H. Owhadi, 2021; Prasanth, Z. S. Haddad, et al., 2021. A KFs version for SDEs is at M. Darcy, Tavallali, et al., 2021.

In previous work Boumediene Hamzi and Houman Owhadi, 2021a; Lee, Brouwer,

et al., 2021; B. H. , R. M. H. Owhadi, 2021, we used the parametric variant of kernel flows H. Owhadi and G. R. Yoo, 2019 (KF) to tune the kernel used to learn chaotic dynamical systems¹.

Given a family of kernels parameterized by α , and some input/output data, KF (in its original form H. Owhadi and G. R. Yoo, 2019) finds a good α (a good kernel) by minimizing (with respect to α , via stochastic gradient descent) the loss

$$\rho = \mathbb{E}[\|u - v\|_{K_\alpha}^2 / \|u\|_{K_\alpha}^2], \quad (8.1)$$

where $\|\cdot\|_{K_\alpha}$ is the RKHS norm defined by K_α , u interpolates a random subset of the data, v interpolates a random subset of the previous subset of half-size and $\mathbb{E}$ represents the average with respect to subsampling. KF is a variant of cross-validation in the sense that it operates under the premise that a kernel must be good if the number of points used to interpolate the data can be halved without significant loss in accuracy. The method presented in H. Owhadi and G. R. Yoo, 2019 uses the regression relative error between two interpolants (measured in the RKHS norm of the kernel) as the quantity to minimize.

In this paper, we use the non-parametric version of KFs with this metric along a second metric based on the Maximum Mean Discrepancy (MMD) that is computed from two different samples of a time series or between a sample and a subsample of half-length that was introduced in Boumediene Hamzi and Houman Owhadi, 2021a. The non-parametric version of KFs is essentially a method of kernel warping where samples are displaced along the direction that minimizes a certain metric which in our current work corresponds to the relative error in H. Owhadi and G. R. Yoo, 2019 or the metric based on the MMD introduced in Boumediene Hamzi and Houman Owhadi, 2021a.

Kernels of the form $K(\phi(x), \phi(x'))$ defined by a warping of the space ϕ have been employed in numerical homogenization Houman Owhadi and L. Zhang, 2007b

¹A variant of KF based on Lyapunov exponents, in order to capture long-term behavior of the system, can be found at Boumediene Hamzi and Houman Owhadi, 2021a, another one based on the Maximum Mean Discrepancy that allows capturing the statistical properties of the system, and that we conjecture to be related to the properties of the Frobenius-Perron operator, can be found at Boumediene Hamzi and Houman Owhadi, 2021a, and another one based on the Hausdorff distance, that allows reconstructing attractors, can be found at Lu Yang et al., 2023. See also Y. Lu et al., 2023 for the method of Sparse Kernel Flows where we modified (8.1) by adding an L_1 penalty term. An Algorithmic Information Theoretic point of view on KF, and particularly Sparse Kernel Flows, can be found at Boumediene Hamzi, M. Hutter, and Houman Owhadi, 2024 where the problem of learning kernels from data is viewed as a problem of data-compression and where it is shown that KF is the “natural approach” to learn kernels from data.

(where they enable upscaling with non separated scales), and in spatial statistics Sampson and Guttorp, 1992; Perrin and Monestiez, 1999; Schmidt and O'Hagan, 2003; Zammit-Mangion et al., 2019 where they enable the nonparametric estimation of nonstationary and anisotropic spatial covariance structures.

Parametric vs. non-parametric. Statistical inference has an inherent tradeoff between robustness and accuracy; these are two conflicting requirements Houman Owhadi and Scovel, 2017; Houman Owhadi and Scovel, 2016; Bajgiran et al., 2022. Indeed a model with a small number of parameters promotes accuracy when well-specified at the cost of a lack of robustness to misspecification. Conversely, a model with a large number of parameters is less likely to be misspecified at the cost of a loss of accuracy compared to a well-specified model with a small number of parameters. We observe this tradeoff when comparing the results of the non-parametric variants of KF to those obtained from parametric variants. Boumediene Hamzi and Houman Owhadi, 2021a observed highly accurate learning of the underlying dynamical systems by using a parametric family of kernels able to capture long and short-range correlations. The warping kernels employed here do not have such information but are expressive to capture it.

Our contributions. The main contributions of this paper are as follows.

- We show that by using the Gaussian kernel and initially choosing the variance that characterizes it, combining (non-parametric) KF with the kriging of the vector field improves the accuracy of the prediction of chaotic time series.
- Training KFs based on the MMD as introduced in Boumediene Hamzi and Houman Owhadi, 2021a gives better results than training KFs with the relative error in H. Owhadi and G. R. Yoo, 2019.
- The choice of the step size in training is important. If it's too large, the relative error diverges. If it's too small, convergence will be very slow.

Structure of the paper. The remainder of the manuscript is structured as follows. We describe the problem in Section 8.2 and describe two cross-validation metrics to learn the parameters of the kernel used for approximating the vector field of the dynamical system. In Section 8.3, we investigate the performance of these methods for the Bernoulli map, the Hénon map, and the Lorenz system.

8.2 The problem and its proposed cross-validation solutions

Let $x_1, \dots, x_k, \dots$ be a time series in $\mathbb{R}^d$. Our goal is to forecast $x_{n+1}, \dots, x_{n+m}$ given the observation of $x_1, \dots, x_n$. We work under the assumption that this time series can be approximated by a solution of a dynamical system of the form

$$z_{k+1} = f^\dagger(z_k, \dots, z_{k-\tau^\dagger+1}), \quad (8.2)$$

where $\tau^\dagger \in \mathbb{N}^*$ and $f^\dagger$ may be unknown. Given $\tau \in \mathbb{N}^*$, the approximation of the dynamical can then be recast as that of interpolating $f^\dagger$ from pointwise measurements

$$f^\dagger(X_k) = Y_k \text{ for } k = 1, \dots, N, \quad (8.3)$$

with $X_k := (x_{k+\tau-1}, \dots, x_k)$, $Y_k := x_{k+\tau}$ and $N = n - \tau$. Given a reproducing kernel Hilbert space² of candidates $\mathcal{H}$ for $f^\dagger$, and using the relative error in the RKHS norm $\|\cdot\|_{\mathcal{H}}$ as a loss, the regression of the data (X_k, Y_k) with the kernel K associated with $\mathcal{H}$ provides a minimax optimal approximation Houman Owhadi and Scovel, 2019b of $f^\dagger$ in $\mathcal{H}$. This regressor (in the presence of measurement noise of variance $\lambda > 0$) is

$$f(x) = K(x, X)(K(X, X) + \lambda I)^{-1}Y, \quad (8.4)$$

where $X = (X_1, \dots, X_N)$, $Y = (Y_1, \dots, Y_N)$, $k(X, X)$ is the $N \times N$ matrix with entries $k(X_i, X_i)$, $k(x, X)$ is the N vector with entries $k(x, X_i)$ and I is the identity matrix. This regressor has also a natural interpretation in the setting of Gaussian process (GP) regression: (i.) (8.4) is the conditional mean of the centered GP $\xi \sim \mathcal{N}(0, K)$ with covariance function K conditioned on $\xi(X_k) = Y_k + \sqrt{\lambda}Z_k$ where the Z_k are centered i.i.d. normal random variables.

Evidently, the accuracy of the proposed approach depends on the kernel K , and one of our goals is to also learn that kernel from the data (X_k, Y_k) with Kernel Flows (KF) H. Owhadi and G. R. Yoo, 2019.

Since the motivation of learning a dynamical system from data is not necessarily about only making predictions but also about emulating the qualitative behaviour of the dynamical systems, we also use a metric based on the Maximum Mean Discrepancy (MMD) to train KFs Boumediene Hamzi and Houman Owhadi, 2021a. The MMD Gretton et al., 2012 is a distance on the space of probability measures with a representer theorem for empirical distributions which we recall in the appendix.

²A brief overview of RKHSs is given in part I of this paper Boumediene Hamzi and Houman Owhadi, 2021a.

Our strategy for learning the kernel K will then simply be to minimize the MMD

$$\rho_{\text{MMD}} = \mathbb{E} \text{MMD}(S_1, S_2), \quad (8.5)$$

between two different samples³, $S_1 = x_{\sigma_1}, \dots, x_{\sigma_m}$ and $S_2 = x_{\mu_1}, \dots, x_{\mu_m}$, of the time series ($\mathbb{E}$ represents, a possibly approximated, average with respect to the subsampling). This metric characterizes the statistical properties of the dynamical system and is, intuitively, related to the Frobenius-Perron operator.

Non-parametric KFs

Write $X := (X_1, \dots, X_N)$ and $Y := (Y_1, \dots, Y_N)$ for the input/output training data. Our goal is to learn a kernel of the form

$$K^\phi(x, x') = K(\phi(x, 1), \phi(x', 1)), \quad (8.6)$$

where K is a standard kernel (e.g. Gaussian or Matérn kernels) and ϕ maps the input space into itself. The warping of the input space ϕ satisfies the following ODE

$$\begin{cases} \dot{\phi}(x, t) = v(\phi(x, t), t) \\ \phi(x, 0) = x \end{cases} \quad (8.7)$$

with

$$v(x, t) = \Gamma(x, q)\Gamma(q, q)^{-1}\dot{q},$$

and

$$\dot{q} = -\nabla[\rho(q)], \quad (8.8)$$

where q corresponds to position variables in $\mathcal{X}^N$ started from $q(0) = X = (X_1, \dots, X_N)$, Γ is an operator/vector-valued kernel, $\Gamma(q, q)$ is an $N \times N$ matrix with entries $\Gamma(q_i, q_j)$, $\Gamma(x, q)$ is a $1 \times N$ vector with entries $\Gamma(x, q_i)$, and ρ is the kernel flow loss (possibly randomized through-subsampling) (8.1) or (8.5) associated with the input/output data (q, Y) .

Using an explicit Euler scheme to integrate (8.7) and regularizing with a nugget $\lambda > 0$ leads to an iteration of the form

$$\phi_{n+1}(x) = \phi_n(x) + \epsilon v_n(\phi_n(x)) \quad (8.9)$$

with $\phi_0(x) = x$. Writing $X = (X_1, \dots, X_N)$ for the training points and $q_n := \phi_n(X) := (\phi_n(X_1), \dots, \phi_n(X_N))$, the discretized equations take the form

$$q_{n+1} = q_n - \epsilon \nabla \rho(q_n) \quad (8.10)$$

³One could also consider the MMD between a sample S_1 of size m and a subsample of S_1 of size $m/2$.

and

$$v_n(x) = \Gamma(x, q_n) (\Gamma(q_n, q_n) + \lambda I)^{-1} (q_{n+1} - q_n) / \epsilon. \quad (8.11)$$

Note that $\rho(q_n)$ is the kernel flow loss (8.1) or (8.5) associated with the input/output data (q_n, Y) where the averaging operation $\mathbb{E}$ can be approximated (via Monte-Carlo sampling from the uniform distribution without replacement) using a finite number of subsamples (possibly reduced to one per iteration in n).

Note also that the proposed (Kernel Flow) algorithm produces a flow ϕ_n (randomized through a sampling of the training data) in the input space, a (stochastic) dynamical system $K(\phi_n(x), \phi_n(y))$ in kernel space and a (stochastic) dynamical system q_n in input space.

8.3 Numerical experiments

We now numerically investigate the efficacy of the approach described in the previous section in learning chaotic dynamical systems. In this section, we report one-step ahead predictions and the corresponding errors. We expect the method of non-parametric kernel flows to not perform well for multi-step ahead predictions.

Example: Bernoulli map

We consider the Bernoulli dynamical system defined by

$$x(k+1) = 2x(k) \mod 1. \quad (8.12)$$

For training, we use a 200 point trajectory with $x_0 = \pi/3$. We choose both K and Γ to be the Gaussian kernel ($K = \Gamma$)

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{\sigma^2}\right). \quad (8.13)$$

We set σ^2 to the average squared distance between all training points and the regularization (nugget) to $\lambda = 10^{-5}$. For testing we use three trajectories with initial values $x_0 = 0.1, 0.99, \pi/10$. We generate 500 points from each trajectory and measure the RMSE. The RMSE for each of the above initialization are denoted R_1, R_2, R_3 . Our baseline is Kernel Regression with the kernel K . We first train Kernel Flows non-parametric (with ρ loss function) for 100,000 iterations with learning rate $\epsilon = 2 \times 10^{-4}$. The learning rate ϵ must be chosen small enough so that the explicit Euler scheme (8.10) is stable and the flow of the dynamical system is

Table 8.1: Bernoulli map: 200 training points.

Method	R_1	R_2	R_3
Kernel Regression	0.107 (0.0%)	0.119(0.0%)	0.114 (0.0%)
Kernel Flows ρ	0.102 (-5.3%)	0.112 (-5.7 %)	0.103 (-9.3 %)

averaged through subsampling and large enough so that the discretization remains computationally tractable (an adaptive integrator may be used here to adjust ϵ to the integration of the potential flow of $\rho(q)$). At inference time, we set $\lambda = 5 \times 10^{-3}$. We set the batch size to $N_b = 100$ (50% of the training set).

The results are presented in table 8.1. We put the change compared to the baseline in parentheses (the baseline is presented in the first line) with the best result highlighted. We repeat the experiment with a larger training set of 800 points. The batch size is set to $N_b = 200$ (25% of the training set). The results are recorded in table 8.2. We also increase the training iterations to 500,000 with a lower learning rate of $\varepsilon = 1 \times 10^{-4}$. Finally, we run the same experiment with the ρ_{MMD} loss function. We train the kernel over 50,000 iterations, with a training size of 800 ($N_b = 200$) and a learning rate of $\varepsilon = 1 \times 10^{-5}$. This numerical experiment illustrates how the ρ_{MMD} can achieve better results with fewer iterations. Figure 8.1 illustrates the time dynamics of the first trajectory (for R_2 and R_3 , see figure G.3 in the appendix).

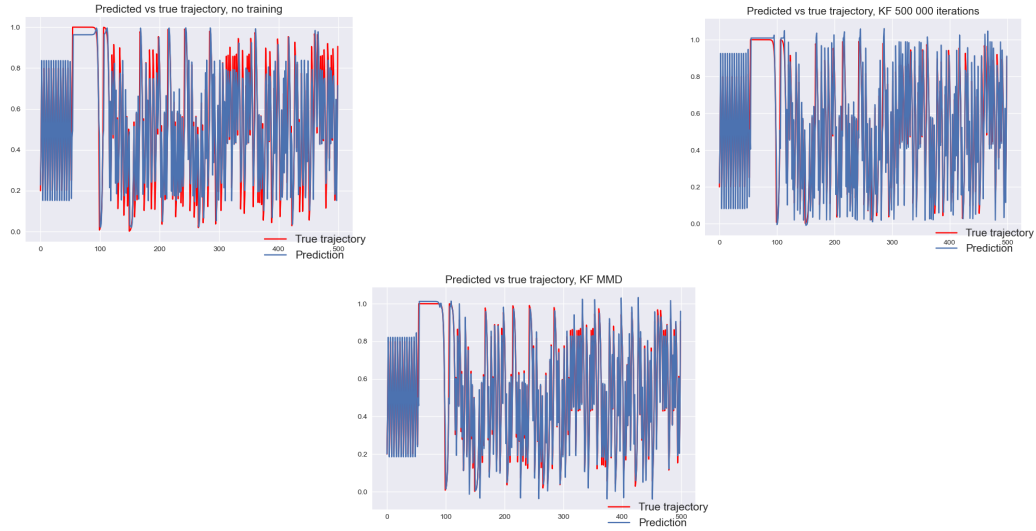
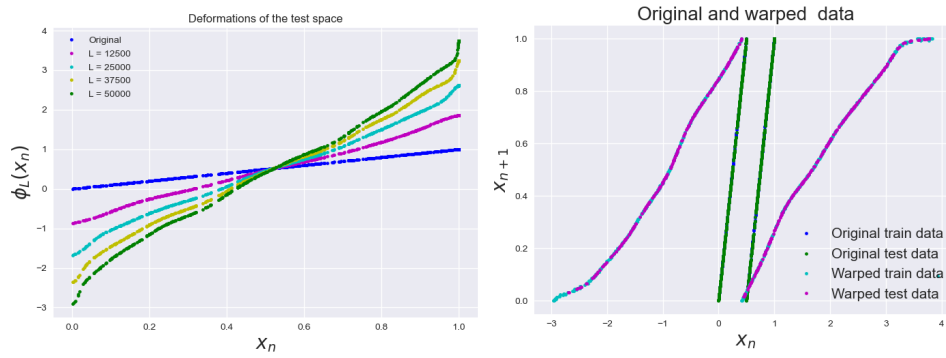
We also illustrate the deformation of the input space X by the flow function ϕ_t for different values of t in figure . We observe that in this simple 1-dimensional case, the flow rotates the data round the x axis (figure 8.2), which amounts to learning to separate between the two maps:

$$T(x) = \begin{cases} 2x, & \text{if } 0 \leq x < \frac{1}{2} \\ 2x - 1, & \frac{1}{2} \leq x \leq 1. \end{cases}$$

Finally, we illustrate the convergence properties of both loss functions in figure 8.3. We observe that while both loss functions converge, ρ_{MMD} does so much more quickly than the original ρ , possibly due to the lower noise in the loss. Hence, larger learning rates ε can be used for quicker convergence in a lower number of iterations. The effect of the learning rate is further discussed in section 8.3.

Table 8.2: Bernoulli map: 800 training points.

Method	R_1	R_2	R_3
Kernel Regression	0.1053 (0.0%)	0.1168 (0.0%)	0.1125 (0.0%)
Kernel Flows $\rho 1 \times 10^5$ iterations	0.0990 (-6.0%)	0.1010 (-13.5 %)	0.0994 (-11.7 %)
Kernel Flows $\rho 5 \times 10^5$ iterations	0.0923 (-12.3%)	0.0924 (-20.8 %)	0.0900 (-20.0 %)
Kernel Flows $\rho_{MMD} 5 \times 10^4$ iterations	0.0562 (-46.6%)	0.0528 (-54.8 %)	0.0518 (-53.9 %)

Figure 8.1: One step ahead times series predictions results for the Bernoulli map R_1 .Figure 8.2: Deformation of input case for different iterations of the flow function ϕ_L (left) and deformed final data (right).

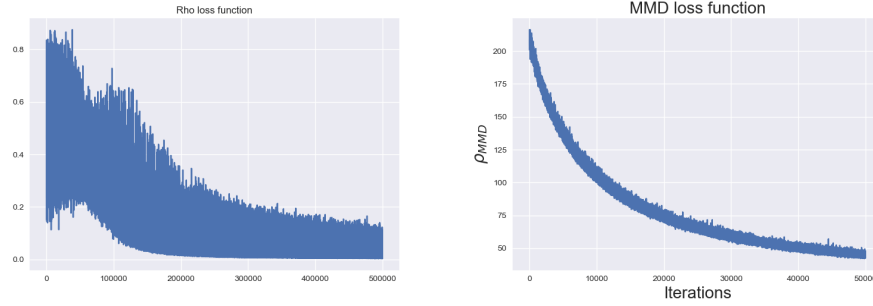


Figure 8.3: Convergence of the ρ and ρ_{MMD} losses (horizontal axis stands for the number of iterations).

Example: The Lorenz system

Consider the Lorenz system

$$\frac{dx}{dt} = s(y - x), \quad (8.14)$$

$$\frac{dy}{dt} = rx - y - xz, \quad (8.15)$$

$$\frac{dz}{dt} = xy - bz, \quad (8.16)$$

with $s = 10$, $r = 28$, $b = 10/3$. We use the initial condition $(x(0), y(0), z(0)) = (0.0, 1.0, 1.05)$ and generate 3000 (training) points with a time step $h = 0.01$. The test set consists of another trajectory of the same size with initial condition $(x(0), y(0), z(0)) = (0.5, 1.5, 2.5)$. For the Lorenz system, the inference task is separated into the prediction of the three functions:

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \mapsto x_{n+1}, \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \mapsto y_{n+1}, \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \mapsto z_{n+1}. \quad (8.17)$$

We train and predict using Kernel Flows separately for each of these three functions, and the RMSE for each of the above functions are denoted as R_1, R_2, R_3 . For training we use the Gaussian kernel (8.13) for K with $\sigma^2 = 1.0$ and regularization (nugget) of $\lambda = 10^{-5}$, with a batch size of $N_b = 300$ (10% of the training set). We choose Γ to be the vector valued kernel $\Gamma = KI_3$ obtained by multiplying K with I_3 , the 3×3 identity matrix.

We train KF with both the ρ loss function and ρ_{MMD} (without the averaging operation $\mathbb{E}$). The first is trained with a learning rate of $\epsilon = 10^{-6}$, the second with a learning rate of 10^{-2} . We compare the performance of the two on all three dimensions of the Lorenz system. Our baseline is again Kernel Regression. The results

of the experiment are recorded in Table 8.3 with the best performance highlighted. The recovered dynamics for the first dimension is provided in figure 8.4 (see figure 8.7 for dimensions 2 and 3), and the error dynamics is provided in figure 8.8.

We observed that in the case of the Lorenz system, the ρ_{MMD} loss function consistently outperformed the original loss ρ and the base estimator. These improvements may be due to the better convergence properties of ρ_{MMD} (in the absence of the averaging operation $\mathbb{E}$): for all three variables, Kernel Flows with ρ_{MMD} converged quickly whereas with a ρ loss function, no convergence was observed (see figure 8.6 for illustrations). Because of the lack of convergence, KF with ρ loss function does not improve over the base model, but it is possible that with a better learning rate schedule, the ρ loss function will yield better results. However, the choice ρ_{MMD} has the advantage of being much less sensitive to the choice of learning rate and will converge to a good solution for multiple values.

We also considered another way of running KFs with the ρ loss function. Since the Lorenz system has multivariate outputs, we apply a standard recipe to normalize the data so that it is zero mean and unit variance. More specifically, after subtracting the mean, we compute the eigendecomposition of the training (input) data covariance matrix and project our inputs on the three eigenvectors. We then scale each transformed dimension to have unit variance. The warping of the space with training data, and the predictions (warping training data and performing regression for the warped inputs) after that are both run in this transformed space. This requires carrying out the transformation operation also for the testing data. Since we don't have a maximum covariance parameter in our model, we also perform the same transformation for the training data labels. That way, the marginal variance of our data is consistent with that of the kernel.

As in the other examples, we use a spherical Gaussian kernel. The length scale of the kernel is computed from the average Euclidean distance between the training inputs in the transformed space, and comes down to 2^{-16} . We use small nuggets for training and prediction, $1e^{-7}$ and $1e^{-6}$, respectively.

The warping is computed in non-parametric KF from the instantaneous gradients of the loss function with respect to the training inputs. In some cases, a few gradients dominate; we use gradient clipping to regularize the flow and clip the magnitudes of the gradient vectors at the 0.8th quantile of all gradient magnitudes. The learning rate is scaled adaptively so that at each step, the maximum distance traveled by any of the inputs is $1e^{-5}$ in the transformed space. We perform 2,000 iterations of KF

in this experiment. The results are reported in Table 8.3.

Table 8.3: Lorenz system: original data.

Method	R_1	R_2	R_3
Kernel regression.	2.15 (0.0%)	2.57(0.0%)	5.45 (0.0%)
Kernel Flows ρ	2.15 (0.0%)	2.57(0.0%)	3.81 (-30%)
Kernel flows ρ_{MMD}	1.60 (-26%)	2.29 (-11%)	3.81 (-30%)

Table 8.4: Lorenz system: pre-processed data.

Variable	R'_1	R'_2	R'_3	R'_4
x	0.25788	0.29009	0.01351	0.01714
y	0.86578	1.14942	0.04364	0.05669
z	0.84734	1.23247	0.01543	0.02203

R'_1 is the RMSE in the warped coordinates, R'_2 : is the RMSE in the original coordinates, R'_3 is the relative error in the warped coordinates, R'_4 is the relative error in the original coordinates. The relative errors are reduced by NPKF training by 21.179, 23.0148, 29.943 percents, due to the training.

Learning rate and convergence

We now discuss one possible advantage of the ρ_{MMD} loss function over the ρ loss function. As was seen in the previous sections, for a particular dynamical system (the Lorenz system), Kernel Flows with ρ did not improve over the base estimator in our tests, whereas Kernel Flows with ρ_{MMD} achieved significant improvements. For the Bernoulli map, while both versions improved over the base estimator, ρ_{MMD} yielded a better performance and converged faster. One possible explanation is that the algorithm is more sensitive to the choice of learning rate ε with ρ compared to ρ_{MMD} .

To illustrate this sensitivity, we retrain the network on the Bernoulli dynamical system with the same parameters as in section 8.3, but with higher learning rates of 2×10^{-4} , 3×10^{-4} for both loss functions and compare the results with the originally trained kernel in tables 8.5 and 8.6. For the ρ loss function, we note that higher learning rates do not necessarily yield better results, even though convergence of the loss is achieved in all cases. In some cases, the performance significantly worsened compared to the base estimator.

We suspect that this is due to the instability of the explicit Euler scheme (8.10) in integrating (8.8). Indeed there are two potential sources of instabilities when ρ is defined as in (8.1) and the time steps are too large: (1) The instability of the explicit Euler scheme itself, (2) the fact ρ is not averaged but randomized at each time step (which produces an effective dynamic with the correct average drift when the time steps are small enough but does add instability when they are not).

Due to these instabilities and the inherent stochasticity of the loss function, as observed in H. Owhadi and G. R. Yoo, 2019, it is possible for the learned kernel to converge to so-called "degenerate" kernels if the learning rate is too high. This may explain why Kernel Flows with ρ did not perform well on the Lorenz system: a high learning rate leads to convergence to a poor solution. We hypothesize that a better choice of learning rate (such as an adaptive learning rate) is required for the kernel to converge to a good solution.

On the other hand, for ρ_{MMD} increasing the learning rate improves the results as long as the kernel achieves convergences. Hence, ρ_{MMD} yields a more stable system which is less sensitive to small changes in hyper-parameter choices. We suspect that this is due to the fact that ρ_{MMD} is not inherently stochastic.

Table 8.5: Bernoulli map learned with ρ for different learning rates

Learning rate	R_1	R_2	R_3
$\varepsilon = 1 \times 10^{-4}$	0.0923 (-12.3%)	0.0924 (-20.8%)	0.112 (-20.0%)
$\varepsilon = 2 \times 10^{-4}$	0.0940 (-10.9%)	0.159(+36.2%)	0.0938 (-16.6%)
$\varepsilon = 3 \times 10^{-4}$	0.1030 (-2.1%)	0.105(-10.1%)	0.104 (-7.5%)

Table 8.6: Bernoulli map learned with ρ_{MMD} for different learning rates

Learning rate	R_1	R_2	R_3
$\varepsilon = 1 \times 10^{-4}$	0.0562 (-46.6%)	0.0528 (-54.8 %)	0.0518 (-53.9 %)
$\varepsilon = 2 \times 10^{-4}$	0.04883 (-53.6%)	0.0425(-63.6%)	0.0420 (-62.4%)
$\varepsilon = 3 \times 10^{-4}$	0.0441 (-58.1%)	0.0375 (-67.9%)	0.0370 (-67.1%)

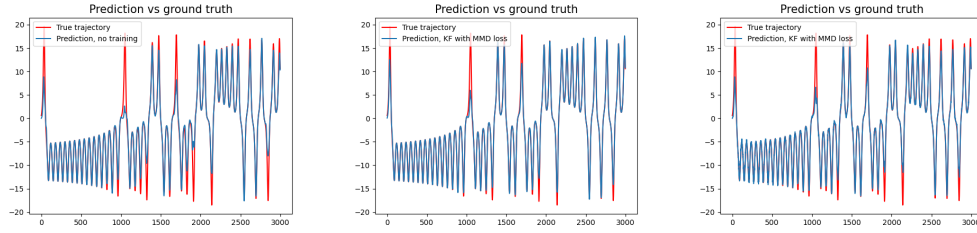


Figure 8.4: Comparison between the different predictions of the first variables for the baseline, ρ training and ρ_{MMD} training (left, middle, right). The horizontal axis stands for the number of iterations.

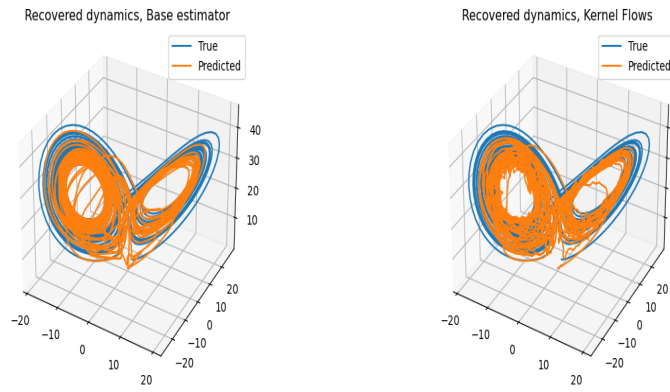


Figure 8.5: One step prediction in the phase space: base estimator (left), kernel flows with ρ_{MMD} (right).

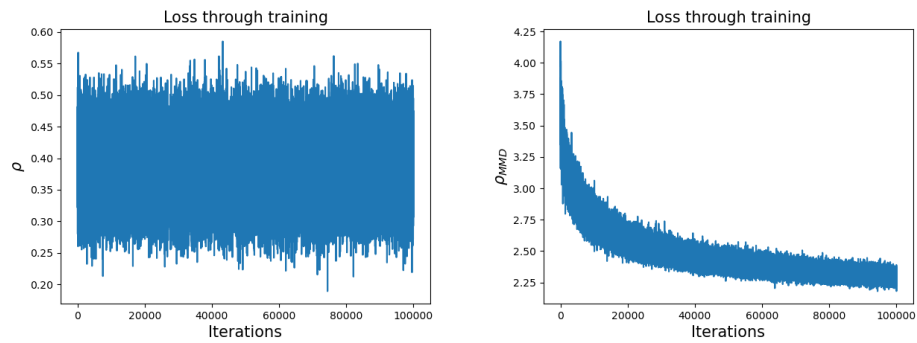


Figure 8.6: Loss functions through training of the first variable for ρ (left) and ρ_{MMD} (right).

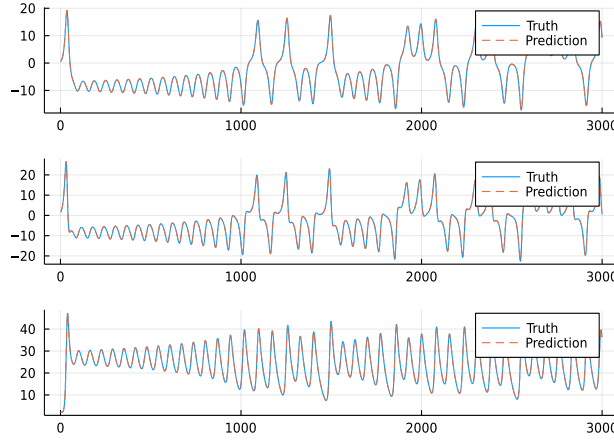


Figure 8.7: Times series predictions. Top: x , middle: y , bottom: z .

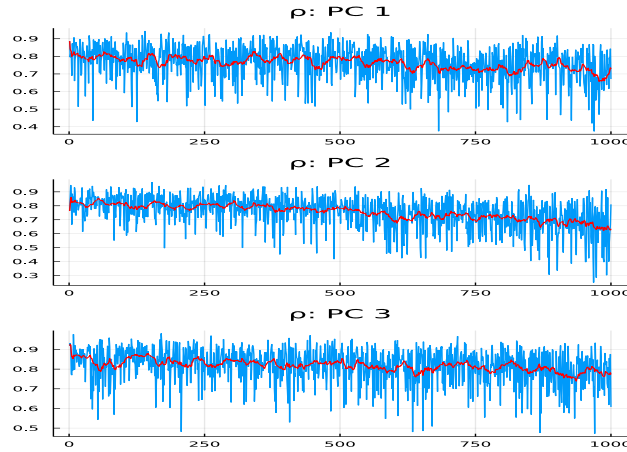


Figure 8.8: Losses. Relative errors ρ for the different variables (blue) and averages (red)- Top: x , middle: y , bottom: z .

Remarks

- i. Convergence results that characterize the error estimates of the difference between a dynamical system and its approximation from data using kernel methods can be found in Jake Bouvrie and Boumediene Hamzi, 2017c; Giesl et al., 2019.
- ii. In the case of very large datasets, it is possible to reduce the number of points during training by considering greedy techniques as in Schafer, Katzfuss, and Houman Owhadi, 2021.
- iii. It is possible to include new measurements when approximating the dynamics from data without repeating the learning process. This can be done by working

in Newton basis as in Pazouki and Schaback, 2011 (check also section 4 of Gabriele Santin and Bernard Haasdonk, 2019). The Newton basis is just another basis for the space spanned by the kernel on the points, i.e., $\text{span}\{k(\cdot, x_1), \dots, k(\cdot, x_N)\} = \text{span}\{v_1, \dots, v_N\}$.

The kernel expansion of f writes as $f(x) = \sum_{i=1}^N c_i K(x, x_i) = \sum_{i=1}^N b_i v_i(x)$ with $\langle v_i, v_j \rangle_H = \delta_{ij}$ (i.e., the basis is orthonormal in the RKHS inner product).

If we add a new point $x_{N+1}, \dots, x_{N+m}$, we'll have corresponding elements $v_{N+1}, \dots, v_{N+m}$ of the Newton basis, still orthonormal to the previous ones. So we will have a new interpolant $f_{\text{new}}(x) = \sum_{i=1}^{N+m} b_i v_i(x)$ that can be rewritten in terms of the old interpolant as

$$f_{\text{new}}(x) = \sum_{i=1}^{N+m} c_i v_i(x) = f(x) + \sum_{i=N+1}^{N+m} c_i v_i(x),$$

where f can still be written in terms of the basis K , but with different coefficients c' .

If A is the kernel matrix on the first N points, one can compute a Cholesky factorization $A = LL^T$ with L lower triangular. Let $B := L^{-T}$, then $v_j(x) = \sum_{i=1}^N (B)_{ij} K(x, x_i)$.

When we add new points, we have an updated kernel matrix A' , and the Cholesky factor of A can be easily updated to the one of A' .

- iv. In our simulations, the kernels in (8.6) and (8.2) are both Gaussian, a possible extension is to consider that they are different kernels and to learn them using parametric KFs.

8.4 Conclusion

In this paper, numerical experiments show that by using the Gaussian kernel and initially choosing the variance that characterizes it, combining two variants of non-parametric KFs with the kriging of the vector field improves the accuracy of the prediction of chaotic time series. Extensions of this approach to other classes of dynamical systems remain open.

BIBLIOGRAPHY

- Abarbanel, H. (2012). *Analysis of Observed Chaotic Data*. Institute for Nonlinear Science. Springer New York. ISBN: 9781461207634.
- Akian, Jean-Luc et al. (Dec. 2022). “Learning “Best” Kernels from Data in Gaussian Process Regression. With Application to Aerodynamics”. In: *Journal of Computational Physics* 470, p. 111595. ISSN: 00219991. DOI: 10.1016/j.jcp.2022.111595.
- Akiba, Takuya et al. (2019a). “Optuna: A Next-Generation Hyperparameter Optimization Framework”. In: *The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631.
- (2019b). *Optuna: A Next-generation Hyperparameter Optimization Framework*. arXiv: 1907.10902 [cs.LG]. URL: <https://arxiv.org/abs/1907.10902>.
- Alaoglu, Leonidas (1940). “Weak topologies of normed linear spaces”. In: *Annals of Mathematics* 41.2, pp. 252–267.
- Alexander, Romeo and Dimitrios Giannakis (2020). “Operator-theoretic framework for forecasting nonlinear time series with kernel analog techniques”. In: *Physica D: Nonlinear Phenomena* 409, p. 132520. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2020.132520>. URL: <http://www.sciencedirect.com/science/article/pii/S016727891930377X>.
- Allen, Samuel M. and John W. Cahn (1979). “A microscopic theory for antiphase boundary motion and its application to antiphase domain coarsening”. In: *Acta Metallurgica* 27.6, pp. 1085–1095. ISSN: 0001-6160. DOI: [https://doi.org/10.1016/0001-6160\(79\)90196-2](https://doi.org/10.1016/0001-6160(79)90196-2). URL: <https://www.sciencedirect.com/science/article/pii/0001616079901962>.
- Almroth, Bo O, Perry Stern, and Frank A Brogan (1978). “Automatic choice of global shape functions in structural analysis”. In: *Aiaa Journal* 16.5, pp. 525–528.
- Altmann, Robert, Patrick Henning, and Daniel Peterseim (2021). “Numerical homogenization beyond scale separation”. In: *Acta Numerica* 30, pp. 1–86.
- Alvarez, Mauricio A, Lorenzo Rosasco, Neil D Lawrence, et al. (2012). “Kernels for vector-valued functions: A review”. In: *Foundations and Trends in Machine Learning* 4.3, pp. 195–266.
- Amari, Shun-ichi (1999). “Natural gradient works efficiently in learning”. In: *Unsupervised Learning*. USA: Bradford Company, pp. 177–202. ISBN: 026258168X.
- Amsallem, David and Charbel Farhat (2008). “Interpolation method for adapting reduced-order models and application to aeroelasticity”. In: *AIAA journal* 46.7, pp. 1803–1813.

- Angelopoulos, Anastasios N. and Stephen Bates (2022). *A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification*. arXiv: 2107.07511 [cs.LG]. URL: <https://arxiv.org/abs/2107.07511>.
- Arcangeli, Remi, Maria Cruz Lopez de Silanes, and Juan Jose Torrens (Aug. 2007a). “An extension of a bound for functions in Sobolev spaces, with applications to (m, s)-spline interpolation and smoothing”. In: *Numerische Mathematik* 107.2, pp. 181–211. ISSN: 0945-3245. DOI: 10.1007/s00211-007-0092-z. URL: <https://doi.org/10.1007/s00211-007-0092-z>.
- (2007b). “An extension of a bound for functions in Sobolev spaces, with applications to (m, s)-spline interpolation and smoothing”. In: *Numerische Mathematik* 107.2, pp. 181–211.
- Archambeau, Cedric et al. (Dec. 2007). “Gaussian Process Approximations of Stochastic Differential Equations”. In: *Gaussian Processes in Practice*. Ed. by Neil D. Lawrence, Anton Schwaighofer, and Joaquin Quiñero Candela. Vol. 1. Proceedings of Machine Learning Research. Bletchley Park, UK: PMLR, pp. 1–16. URL: <https://proceedings.mlr.press/v1/archambeau07a.html>.
- Aronszajn, Nachman (1950). “Theory of reproducing kernels”. In: *Transactions of the American Mathematical Society* 68, pp. 337–404.
- B.Haasdonk et al. (2021). “Kernel methods for center manifold approximation and a weak data-based version of the Center Manifold Theorems”. In: *Physica D*.
- Bacho, Aras, Aleksei G. Sorokin, Xianjin Yang, Theo Bourdais, et al. (2025a). *Operator Learning at Machine Precision*. arXiv: 2511.19980. URL: <https://arxiv.org/abs/2511.19980>.
MD ran the benchmarks, and wrote and edited the manuscript.
- (2025b). *Operator Learning at Machine Precision*. arXiv: 2511.19980 [cs.LG]. URL: <https://arxiv.org/abs/2511.19980>.
- Bain, Alan and Dan Crisan (2009). *Fundamentals of Stochastic Filtering*. Vol. 60. Stochastic Modelling and Applied Probability. Springer New York, NY. ISBN: 978-0-387-76895-3. DOI: 10.1007/978-0-387-76896-0.
- Bajgiran, Hamed Hamze et al. (2022). “Uncertainty quantification of the 4th kind; optimal posterior accuracy-uncertainty tradeoff with the minimum enclosing ball”. In: *Journal of Computational Physics* 471, p. 111608. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2022.111608>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999122006714>.
- Baptista, Ricardo et al. (2025a). *Solving Roughly Forced Nonlinear PDEs via Misspecified Kernel Methods and Neural Networks*. arXiv: 2501.17110 [math.NA]. URL: <https://arxiv.org/abs/2501.17110>.
- (2025b). *Solving Roughly Forced Nonlinear PDEs via Misspecified Kernel Methods and Neural Networks*. arXiv: 2501.17110 [math.NA]. URL: <https://arxiv.org/abs/2501.17110>.

Baptista, Ricardo et al. (2026). “Solving Roughly Forced Nonlinear PDEs via Misspecified Kernel Methods and Neural Networks”. In: *AMS: Mathematics of Computation (to appear)*.

MD was the lead author, developed and implemented the methodology, developed the theoretical results, ran the numerical experiments, and wrote the manuscript.

Barth, Andrea, Christoph Schwab, and Nathaniel Zollinger (2011). “Multi-level Monte Carlo Finite Element method for elliptic PDEs with stochastic coefficients”. In: *Numerische Mathematik* 119.1, pp. 123–161. ISSN: 0945-3245. DOI: [10.1007/s00211-011-0377-0](https://doi.org/10.1007/s00211-011-0377-0). URL: <https://doi.org/10.1007/s00211-011-0377-0>.

Bartolucci, Francesca et al. (2023). “Representation equivalent neural operators: a framework for alias-free operator learning”. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS ’23. New Orleans, LA, USA: Curran Associates Inc.

Bartolucci, Francesco et al. (2024). “Convolutional Neural Operators”. In: *Advances in Neural Information Processing Systems*. NeurIPS; official metadata available via the NeurIPS proceedings.

Bastos, Leonardo S and Anthony O’hagan (2009). “Diagnostics for Gaussian process emulators”. In: *Technometrics* 51.4, pp. 425–438.

Battle, Pau, Yifan Chen, et al. (2025a). “Error analysis of kernel/GP methods for nonlinear and parametric PDEs”. In: *Journal of Computational Physics* 520, p. 113488. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2024.113488>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999124007368>.

– (2025b). “Error analysis of kernel/GP methods for nonlinear and parametric PDEs”. In: *Journal of Computational Physics* 520, p. 113488. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2024.113488>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999124007368>.

Battle, Pau, Matthieu Darcy, Bamdad Hosseini, et al. (2024a). “Kernel methods are competitive for operator learning”. In: *Journal of Computational Physics* 496, p. 112549. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2023.112549>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999123006447>.

MD developed and implemented the methodology, ran the numerical experiments, and wrote the manuscript.

– (2024b). “Kernel methods are competitive for operator learning”. In: *Journal of Computational Physics* 496, p. 112549. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2023.112549>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999123006447>.

– (2024c). “Kernel methods are competitive for operator learning”. In: *Journal of Computational Physics* 496, p. 112549.

- Battle, Pau, Matthieu Darcy, Bamdad Hosseini, et al. (2024d). “Kernel methods are competitive for operator learning”. In: *Journal of Computational Physics* 496, p. 112549. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2023.112549>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999123006447>.
- Battle, Pau, Matthieu Darcy, Matthew Levine, et al. (2025). “Benchmarking Gaussian Processes for Prediction and Data Assimilation of Alzheimer’s Disease Progression”. In: *Journal of Alzheimer’s Disease*.
- Baxter, Rodney J (2016). *Exactly solved models in statistical mechanics*. Elsevier.
- Bayesian optimization with skopt* (n.d.). https://scikit-optimize.github.io/stable/auto_examples/bayesian-optimization.html. Accessed: 2021-09-07.
- Beck, Joakim et al. (2012). “On the optimal polynomial approximation of stochastic PDEs by Galerkin and collocation methods”. In: *Mathematical Models and Methods in Applied Sciences* 22.09, p. 1250023.
- Bendsoe, Martin Philip and Ole Sigmund (2003). *Topology optimization: theory, methods, and applications*. Springer Science & Business Media.
- Berges, Juergen, Nikolaos Tetradis, and Christof Wetterich (2002). “Non-perturbative renormalization flow in quantum field theory and statistical physics”. In: *Physics Reports* 363.4-6, pp. 223–386.
- Bergstra, James, Remi Bardenet, et al. (2011). “Algorithms for Hyper-Parameter Optimization”. In: *Advances in Neural Information Processing Systems*. Ed. by J. Shawe-Taylor et al. Vol. 24. Curran Associates, Inc. URL: <https://proceedings.neurips.cc/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf>.
- Bergstra, James and Yoshua Bengio (Feb. 2012). “Random Search for Hyper-Parameter Optimization”. In: *J. Mach. Learn. Res.* 13.null, pp. 281–305. ISSN: 1532-4435.
- Bhattacharya, Kaushik, Bamdad Hosseini, Nikola B Kovachki, et al. (2021a). “Model reduction and neural networks for parametric PDEs”. In: *The SMAI journal of computational mathematics* 7, pp. 121–157.
- (2021b). “Model Reduction And Neural Networks For Parametric PDEs”. en. In: *The SMAI Journal of computational mathematics* 7, pp. 121–157.
- Bhattacharya, Kaushik, Nikola Kovachki, et al. (2023). *Learning Homogenization for Elliptic Operators*. arXiv: 2306.12006 [math.NA].
- Billingsley, Patrick (1999). *Convergence of probability measures*. Wiley.
- Binev, Peter et al. (2011). “Convergence rates for greedy algorithms in reduced basis methods”. In: *SIAM journal on mathematical analysis* 43.3, pp. 1457–1472.

- Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.
- Bittracher, Andreas et al. (2019). “Dimensionality Reduction of Complex Metastable Systems via Kernel Embeddings of Transition Manifolds”. In: <https://arxiv.org/abs/1904.08622>.
- Blondel, Mathieu et al. (2021). “Efficient and Modular Implicit Differentiation”. In: *arXiv preprint arXiv:2105.15183*.
- Bodnar, Cristian et al. (2025). “A foundation model for the Earth system”. In: *Nature* 641.8023, pp. 1180–1187. DOI: 10.1038/s41586-025-09005-y.
- Bogachev, Vladimir I. (1998a). *Gaussian Measures*. Vol. 62. Mathematical Surveys and Monographs. American Mathematical Society.
- (1998b). *Gaussian Measures*. Vol. 62. Mathematical Surveys and Monographs. American Mathematical Society. ISBN: 978-1-4704-1869-4.
- Bohmer, Klaus and Robert Schaback (2013). “A nonlinear discretization theory”. In: *Journal of Computational and Applied Mathematics* 254. Nonlinear Elliptic Differential Equations, Bifurcation, Local Dynamics of Parabolic Systems and Numerical Methods, pp. 204–219. ISSN: 0377-0427. DOI: <https://doi.org/10.1016/j.cam.2013.03.029>. URL: <https://www.sciencedirect.com/science/article/pii/S0377042713001374>.
- Boncoraglio, Gabriele and Charbel Farhat (2021). “Active manifold and model-order reduction to accelerate multidisciplinary analysis and optimization”. In: *AIAA Journal* 59.11, pp. 4739–4753.
- Bonev, Boris et al. (2023). *Spherical Fourier Neural Operators: Learning Stable Dynamics on the Sphere*. arXiv: 2306.03838 [cs.LG]. URL: <https://arxiv.org/abs/2306.03838>.
- Bonito, Andrea et al. (2024). *Convergence and error control of consistent PINNs for elliptic PDEs*. arXiv: 2406.09217 [math.NA]. URL: <https://arxiv.org/abs/2406.09217>.
- Borchardt, Julia and Benjamin Knorr (2015). “Global solutions of functional fixed point equations via pseudospectral methods”. In: *Phys. Rev. D* 91.10, p. 105011. DOI: 10.1103/PhysRevD.91.105011.
- (2016). “Solving functional flow equations with pseudospectral methods”. In: *Phys. Rev. D* 94.2, p. 025027. DOI: 10.1103/PhysRevD.94.025027.
- Boulle, Nicolas and Alex Townsend (2022). “Learning elliptic partial differential equations with randomized linear algebra”. In: *Foundations of Computational Mathematics*, pp. 1–31.
- (2024). “A mathematical guide to operator learning”. In: *Handbook of Numerical Analysis*. Vol. 25. Elsevier, pp. 83–125.

- Bourdais, Th'eo et al. (2024). "Codiscovering graphical structure and functional relationships within data: A Gaussian Process framework for connecting the dots". In: *Proceedings of the National Academy of Sciences* 121.32, e2403449121. DOI: 10.1073/pnas.2403449121. eprint: <https://www.pnas.org/doi/pdf/10.1073/pnas.2403449121>. URL: <https://www.pnas.org/doi/abs/10.1073/pnas.2403449121>.
- Bourdais, Theo et al. (2024). "Codiscovering graphical structure and functional relationships within data: A Gaussian Process framework for connecting the dots". In: *Proceedings of the National Academy of Sciences* 121.32, e2403449121.
- Bouvrie, J. and B. Hamzi (2010a). "Balanced Reduction of Nonlinear Control Systems in Reproducing Kernel Hilbert Space". In: *Proc. 48th Annual Allerton Conference on Communication, Control, and Computing*. <http://arxiv.org/abs/1011.2952>, pp. 294–301.
- (2010b). "Balanced Reduction of Nonlinear Control Systems in Reproducing Kernel Hilbert Space". In: *Proc. 48th Annual Allerton Conference on Communication, Control, and Computing*. <https://arxiv.org/abs/1011.2952>, pp. 294–301.
 - (2017). "Kernel Methods for the Approximation of Some Key Quantities of Nonlinear Systems". In: *Journal of Computational Dynamics* 1. <http://arxiv.org/abs/1204.0563>.
- Bouvrie, Jake and Boumediene Hamzi (2010). "Balanced reduction of nonlinear control systems in reproducing kernel Hilbert space". In: *48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 294–301. DOI: 10.1109/ALLERTON.2010.5706920.
- (2012). "Empirical Estimators for Stochastically Forced Nonlinear Systems: Observability, Controllability and the Invariant Measure". In: *Proc. of the 2012 American Control Conference*. <https://arxiv.org/abs/1204.0563v1>, pp. 294–301.
 - (Jan. 2017a). "Kernel Methods for the Approximation of Nonlinear Systems". In: *SIAM Journal on Control and Optimization* 55.4. <https://arxiv.org/abs/1108.2903>, pp. 2460–2492. ISSN: 0363-0129, 1095-7138. DOI: 10.1137/14096815X.
 - (2017b). "Kernel methods for the approximation of nonlinear systems". In: *SIAM J. Control and Optimization*. <https://arxiv.org/abs/1108.2903>.
 - (2017c). "Kernel methods for the approximation of some key quantities of nonlinear systems". In: *Journal of Computational Dynamics* 1. <http://arxiv.org/abs/1204.0563>.
- Boyd, Stephen and Lieven Vandenberghe (2004). *Convex optimization*. Cambridge university press.

- Bradbury, James et al. (2018a). *JAX: composable transformations of Python+NumPy programs*. Version 0.3.13. URL: <http://github.com/google/jax>.
- (2018b). *JAX: composable transformations of Python+NumPy programs*. Version 0.3.13. URL: <http://github.com/jax-ml/jax>.
- Breit, Dominic (2018). “An Introduction to Stochastic Navier–Stokes Equations”. In: *New Trends and Results in Mathematical Description of Fluid Flows*. Ed. by Miroslav Bulicek, Eduard Feireisl, and Milan Pokorný. Cham: Springer International Publishing, pp. 1–51. ISBN: 978-3-319-94343-5. DOI: 10.1007/978-3-319-94343-5_1. URL: https://doi.org/10.1007/978-3-319-94343-5_1.
- Brezis, Haim (2010). *Functional analysis, Sobolev spaces and partial differential equations*. Springer-Verlag.
- Brunton, Steven L, Joshua L Proctor, and J Nathan Kutz (2016a). “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. In: *Proceedings of the national academy of sciences* 113.15, pp. 3932–3937.
- (2016b). “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. In: *Proceedings of the National Academy of Sciences* 113.15, pp. 3932–3937. ISSN: 0027-8424. DOI: 10.1073/pnas.1517384113. eprint: <https://www.pnas.org/content/113/15/3932.full.pdf>. URL: <https://www.pnas.org/content/113/15/3932>.
- C. Berg J. P. R. Christensen, P. Ressel (1984). *Harmonic Analysis on Semigroups: Theory of Positive Definite and Related Functions*. Graduate texts in mathematics 100. Springer-Verlag Berlin and Heidelberg GmbH & Co. K. ISBN: 3540909257; 9783540909255.
- Cai, Zhiqiang et al. (2024). “A structure-guided gauss-newton method for shallow ReLU neural network”. In: *arXiv preprint arXiv:2404.05064*.
- Calvello, Edoardo et al. (2024). “Continuum Attention for Neural Operators”. In: *arXiv preprint arXiv:2406.06486*.
- Cao, Shuhao (2021). “Choose a Transformer: Fourier or Galerkin”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., pp. 24924–24940.
- Cardaliaguet, Pierre et al. (2019). *The master equation and the convergence problem in mean field games*. Princeton University Press.
- Carmona, Rene, Francois Delarue, et al. (2018). *Probabilistic theory of mean field games with applications I-II*. Vol. 3. Springer.
- Casdagli, Martin (1989). “Nonlinear prediction of chaotic time series”. In: *Physica D: Nonlinear Phenomena* 35.3, pp. 335–356. ISSN: 0167-2789. DOI: [https://doi.org/10.1016/0167-2789\(89\)90074-2](https://doi.org/10.1016/0167-2789(89)90074-2). URL: <http://www.sciencedirect.com/science/article/pii/0167278989900742>.

- Chandra, Anirban et al. (2025). “Fourier Neural Operator based surrogates for CO_2 storage in realistic geologies”. In: *arXiv preprint arXiv:2503.11031*.
- Chattopadhyay, Ashesh et al. (2019). “Data-driven prediction of a multi-scale Lorenz 96 chaotic system using a hierarchy of deep learning methods: Reservoir computing, ANN, and RNN-LSTM”. In: *CoRR* abs/1906.08829. arXiv: 1906.08829. URL: <http://arxiv.org/abs/1906.08829>.
- Chen, Jiong, Florian Schafer, and Mathieu Desbrun (July 2024). “Lightning-fast Method of Fundamental Solutions”. In: *ACM Trans. Graph.* 43.4. ISSN: 0730-0301. DOI: 10.1145/3658199. URL: <https://doi.org/10.1145/3658199>.
- Chen, Jiong, Florian Schafer, Jin Huang, et al. (July 2021). “Multiscale Cholesky preconditioning for ill-conditioned problems”. In: *ACM Trans. Graph.* 40.4. ISSN: 0730-0301. DOI: 10.1145/3450626.3459851. URL: <https://doi.org/10.1145/3450626.3459851>.
- Chen, Ricky T. Q. et al. (2019). *Neural Ordinary Differential Equations*. arXiv: 1806.07366 [cs.LG]. URL: <https://arxiv.org/abs/1806.07366>.
- Chen, Tianping and Hong Chen (1995a). “Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks”. In: *IEEE Transactions on Neural Networks* 6.4, pp. 904–910.
- (1995b). “Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems”. In: *IEEE Transactions on Neural Networks* 6.4, pp. 911–917.
- Chen, Yifan, Ethan N Epperly, et al. (2023). “Randomly pivoted Cholesky: Practical approximation of a kernel matrix with few entry evaluations”. In: *Communications on Pure and Applied Mathematics*.
- Chen, Yifan, Bamdad Hosseini, Houman Owhadi, and Andrew M Stuart (2021a). *Solving and Learning Nonlinear PDEs with Gaussian Processes*.
- (2024a). “Gaussian Measures Conditioned on Nonlinear Observations: Consistency, MAP Estimators, and Simulation”. In: *arXiv preprint arXiv:2405.13149*.
- (2024b). *Gaussian Measures Conditioned on Nonlinear Observations: Consistency, MAP Estimators, and Simulation*. arXiv: 2405.13149 [stat.ML]. URL: <https://arxiv.org/abs/2405.13149>.
- (2024c). *Gaussian Measures Conditioned on Nonlinear Observations: Consistency, MAP Estimators, and Simulation*. arXiv: 2405.13149 [stat.ML]. URL: <https://arxiv.org/abs/2405.13149>.
- (2021b). “Solving and learning nonlinear PDEs with Gaussian processes”. In: *Journal of Computational Physics* 447, p. 110668. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2021.110668>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999121005635>.

- Chen, Yifan, Houman Owhadi, and Florian Schafer (2024a). *Sparse Cholesky Factorization for Solving Nonlinear PDEs via Gaussian Processes*. arXiv: 2304.01294 [math.NA]. URL: <https://arxiv.org/abs/2304.01294>.
- (2024b). “Sparse Cholesky factorization for solving nonlinear PDEs via Gaussian processes”. In: *Mathematics of Computation*.
- Chen, Yifan, Houman Owhadi, and Andrew Stuart (2021). “Consistency of empirical Bayes and kernel flow for hierarchical parameter estimation”. In: *Mathematics of Computation*.
- Chen, Yifan, Houman Owhadi, and Andrew M. Stuart (2020). “Consistency of Empirical Bayes And Kernel Flow For Hierarchical Parameter Estimation”. In: <https://arxiv.org/abs/2005.11375>.
- (2021). *Consistency of Empirical Bayes And Kernel Flow For Hierarchical Parameter Estimation*. arXiv: 2005.11375 [math.ST]. URL: <https://arxiv.org/abs/2005.11375>.
- Cheng, Mulin, Thomas Y. Hou, and Zhiwen Zhang (2013a). “A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations I: Derivation and algorithms”. In: *Journal of Computational Physics* 242, pp. 843–868. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2013.02.033>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999113001526>.
- (2013b). “A dynamically bi-orthogonal method for time-dependent stochastic partial differential equations II: Adaptivity and generalizations”. In: *Journal of Computational Physics* 242, pp. 753–776. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2013.02.020>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999113001393>.
- Chkifa, Abdellah, Albert Cohen, Ronald DeVore, et al. (Nov. 2012). “Sparse adaptive Taylor approximation algorithms for parametric and stochastic elliptic PDEs”. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 47.1, pp. 253–280.
- Chkifa, Abdellah, Albert Cohen, and Christoph Schwab (2014). “High-dimensional adaptive sparse polynomial interpolation and applications to parametric PDEs”. In: *Foundations of Computational Mathematics* 14.4, pp. 601–633.
- Chung, Jui-Hui and Ying-Jer Kao (June 2021). “Neural Monte Carlo renormalization group”. In: *Phys. Rev. Res.* 3 (2), p. 023230. DOI: [10.1103/PhysRevResearch.3.023230](https://doi.org/10.1103/PhysRevResearch.3.023230). URL: <https://link.aps.org/doi/10.1103/PhysRevResearch.3.023230>.
- Cliffe, K. A. et al. (2011). “Multilevel Monte Carlo methods and applications to elliptic PDEs with random coefficients”. In: *Computing and Visualization in Science* 14.1, pp. 3–15. ISSN: 1433-0369. DOI: [10.1007/s00791-011-0160-x](https://doi.org/10.1007/s00791-011-0160-x). URL: <https://doi.org/10.1007/s00791-011-0160-x>.

- Cockayne, Jon et al. (2017). *Probabilistic Numerical Methods for Partial Differential Equations and Bayesian Inverse Problems*. arXiv: 1605.07811 [stat.ME]. URL: <https://arxiv.org/abs/1605.07811>.
- Cohen, Albert and Ronald DeVore (2015). “Approximation of high-dimensional parametric PDEs”. In: *Acta Numerica* 24, pp. 1–159.
- Cucker, F. and S. Smale (2002). “On the mathematical foundations of learning”. In: *Bulletin of the American mathematical society* 39.1, pp. 1–49.
- Cucker, Felipe and Steve Smale (2002). “On the mathematical foundations of learning”. In: *Bulletin of the American Mathematical Society* 39, pp. 1–49.
- Cuomo, Salvatore et al. (2022). “Scientific machine learning through physics-informed neural networks: Where we are and what’s next”. In: *Journal of Scientific Computing* 92.3, p. 88.
- Dacorogna, B. (1989). *Direct Methods in the Calculus of Variations*. Applied Mathematical Sciences. Springer Berlin Heidelberg. ISBN: 9783540504917.
- Darcy, M., B. Hamzi, J. Susiluo, et al. (2021). “Learning dynamical systems from data: a simple cross-validation perspective, part II: nonparametric kernel flows”. In: *preprint*.
- Darcy, M., B. Hamzi, J. Susiluoto, et al. (2021). “Learning dynamical systems from data: a simple cross-validation perspective, Part II: nonparametric kernel flows”. In: *submitted*.
- Darcy, M., P. Tavallali, et al. (2021). “Learning dynamical systems from data: a simple cross-validation perspective, Part IV: Robust Kernel Learning of SDEs”. In: *preprint*.
- Darcy, Matthieu, Boumediene Hamzi, Giulia Livieri, et al. (2023a). “One-shot learning of stochastic differential equations with data adapted kernels”. In: *Physica D: Nonlinear Phenomena* 444, p. 133583. DOI: <https://doi.org/10.1016/j.physd.2022.133583>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278922002871>.
MD was the lead author, and developed and coded the methodology, ran the numerical experiments, and wrote the manuscript.
- (2023b). “One-shot learning of stochastic differential equations with data adapted kernels”. In: *Physica D: Nonlinear Phenomena* 444, p. 133583. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2022.133583>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278922002871>.
- Darcy, Matthieu, Boumediene Hamzi, Jouni Susiluoto, et al. (2025a). “Learning dynamical systems from data: A simple cross-validation perspective, part II: Nonparametric kernel flows”. In: *Physica D: Nonlinear Phenomena* 476, p. 134641. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2025.134641>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278925001204>.

MD developed and implemented the methodology, ran the numerical experiments, and wrote the manuscript.

Darcy, Matthieu, Boumediene Hamzi, Jouni Susiluoto, et al. (2025b). “Learning dynamical systems from data: A simple cross-validation perspective, part II: Non-parametric kernel flows”. In: *Physica D: Nonlinear Phenomena* 476, p. 134641. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2025.134641>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278925001204>.

Darcy, Matthieu and Houman Owhadi (2026). *Sparse Inducing points and Variational Inference I for Forward and Inverse Problems in PDEs*.
MD was the lead author, and developed and implemented the methodology, wrote the theoretical results, and ran the numerical experiments.

Dashti, Masoumeh and Andrew Stuart (June 2017). “The Bayesian Approach to Inverse Problems”. In: Springer International Publishing, pp. 311–428. DOI: 10.1007/978-3-319-12385-1_7.

De Hoop, Maarten et al. (2022). “The cost-accuracy trade-off in operator learning with neural networks”. In: *arXiv preprint arXiv:2203.13181*.

De Ryck, Tim and Siddhartha Mishra (2022). “Generic bounds on the approximation error for physics-informed (and) operator learning”. In: *Advances in Neural Information Processing Systems* 35, pp. 10945–10958.

– (2024). “Numerical analysis of physics-informed neural networks and related models in physics-informed machine learning”. In: *Acta Numerica* 33, pp. 633–713. DOI: 10.1017/S0962492923000089.

De Ryck, Tim, Siddhartha Mishra, and Roberto Molinaro (2024). “wPINNs: Weak Physics Informed Neural Networks for Approximating Entropy Solutions of Hyperbolic Conservation Laws”. In: *SIAM Journal on Numerical Analysis* 62.2, pp. 811–841. DOI: 10.1137/22M1522504. eprint: <https://doi.org/10.1137/22M1522504>. URL: <https://doi.org/10.1137/22M1522504>.

Delamotte, Bertrand (2012). “An Introduction to the Nonperturbative Renormalization Group”. In: *Renormalization Group and Effective Field Theory Approaches to Many-Body Systems*. Springer Berlin Heidelberg, pp. 49–132. ISBN: 9783642273209. DOI: 10.1007/978-3-642-27320-9_2. URL: http://dx.doi.org/10.1007/978-3-642-27320-9_2.

Dembo, Ron S., Stanley C. Eisenstat, and Trond Steihaug (1982). “Inexact Newton Methods”. In: *SIAM Journal on Numerical Analysis* 19.2, pp. 400–408. DOI: 10.1137/0719025. eprint: <https://doi.org/10.1137/0719025>. URL: <https://doi.org/10.1137/0719025>.

Deng, Beichuan et al. (2022). “Approximation rates of DeepONets for learning operators arising from advection–diffusion equations”. In: *Neural Networks* 153, pp. 411–426.

- Deng, Chengyuan et al. (2022). “OpenFWI: Large-scale multi-structural benchmark datasets for full waveform inversion”. In: *Advances in Neural Information Processing Systems* 35, pp. 6007–6020.
- Di Sante, Domenico, Matija Medvidovic, et al. (Sept. 2022a). “Deep Learning the Functional Renormalization Group”. In: *Phys. Rev. Lett.* 129 (13), p. 136402. DOI: 10.1103/PhysRevLett.129.136402. URL: <https://link.aps.org/doi/10.1103/PhysRevLett.129.136402>.
- (2022b). “Deep Learning the Functional Renormalization Group”. In: *Phys. Rev. Lett.* 129, p. 136402. DOI: 10.1103/PhysRevLett.129.136402.
- Diaconis, Persi (1988). “Bayesian Numerical Analysis”. In: *Statistical Decision Theory and Related Topics IV*. Ed. by Shanti S. Gupta and James O. Berger. New York, NY: Springer New York, pp. 163–175. ISBN: 9781461387701 9781461387688. DOI: 10.1007/978-1-4613-8768-8_20. URL: http://link.springer.com/10.1007/978-1-4613-8768-8_20 (visited on 04/15/2026).
- Dietrich, Felix et al. (2021). *Learning effective stochastic differential equations from microscopic simulations: combining stochastic numerics and deep learning*. arXiv: 2106.09004 [physics.comp-ph].
- Dosovitskiy, Alexey et al. (2021). “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.
- Dridi, Noura, Lucas Drumetz, and Ronan Fablet (2021). “Learning stochastic dynamical systems with neural networks mimicking the Euler-Maruyama scheme”. In: *CoRR* abs/2105.08449. arXiv: 2105.08449. URL: <https://arxiv.org/abs/2105.08449>.
- Dupuis, Nicolas et al. (2021). “The nonperturbative functional renormalization group and its applications”. In: *Physics Reports* 910, pp. 1–114.
- E, Weinan (Apr. 2021). “The Dawning of a New Era in Applied Mathematics”. en. In: *Notices of the American Mathematical Society* 68.04, p. 1. ISSN: 0002-9920, 1088-9477. DOI: 10.1090/noti2259. URL: <https://www.ams.org/notices/202104/rnoti-p565.pdf> (visited on 04/15/2026).
- E, Weinan and Bing Yu (Mar. 2018). “The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems”. In: *Communications in Mathematics and Statistics* 6.1, pp. 1–12. DOI: 10.1007/s40304-018-0127-z. URL: <https://doi.org/10.1007/s40304-018-0127-z>.
- E. J. Allen, S. J. Novosel and Z. Zhang (1998). “Finite element and difference approximation of some linear stochastic partial differential equations”. In: *Stochastics and Stochastic Reports* 64.1-2, pp. 117–142. DOI: 10.1080/17442509808834159. eprint: <https://doi.org/10.1080/17442509808834159>. URL: <https://doi.org/10.1080/17442509808834159>.

- Economon, Thomas D et al. (2016). “SU2: An open-source suite for multiphysics simulation and design”. In: *Aiaa Journal* 54.3, pp. 828–846.
- Fabiani, Gianluca et al. (2025). “RandONets: Shallow networks with random projections for learning linear and nonlinear operators”. In: *Journal of Computational Physics* 520, p. 113433. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2024.113433>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999124006818>.
- Falcon, William and The PyTorch Lightning team (Mar. 2019). *PyTorch Lightning*. Version 1.4. DOI: 10.5281/zenodo.3828935. URL: <https://github.com/Lightning-AI/lightning>.
- Fan, Yuwei, Cindy Orozco Bohorquez, and Lexing Ying (2019). “BCR-Net: A neural network based on the nonstandard wavelet form”. In: *Journal of Computational Physics* 384, pp. 1–15.
- Fan, Yuwei, Jordi Feliu-Faba, et al. (2019). “A multiscale neural network based on hierarchical nested bases”. In: *Research in the Mathematical Sciences* 6.2, pp. 1–28.
- Fan, Yuwei, Lin Lin, et al. (2019). “A multiscale neural network based on hierarchical matrices”. In: *Multiscale Modeling & Simulation* 17.4, pp. 1189–1213.
- Fang, Shikai, Madison Cooley, Da Long, Shibo Li, Mike Kirby, et al. (2024). “Solving High Frequency and Multi-Scale PDEs with Gaussian Processes”. In: *The Twelfth International Conference on Learning Representations*. URL: <https://openreview.net/forum?id=q4AEBLHuA6>.
- Fang, Shikai, Madison Cooley, Da Long, Shibo Li, Robert Kirby, et al. (2023). *Solving High Frequency and Multi-Scale PDEs with Gaussian Processes*. arXiv: 2311.04465 [cs.LG].
- (2024). *Solving High Frequency and Multi-Scale PDEs with Gaussian Processes*. arXiv: 2311.04465 [cs.LG]. URL: <https://arxiv.org/abs/2311.04465>.
- Fasshauer, Gregory E. and Qi Ye (June 2011). “Reproducing kernels of generalized Sobolev spaces via a Green function approach with distributional operators”. In: *Numerische Mathematik* 119.3, pp. 585–611. DOI: 10.1007/s00211-011-0391-2. URL: <https://doi.org/10.1007%2Fs00211-011-0391-2>.
- Feischl, Michael and Daniel Peterseim (2020). “Sparse compression of expected solution operators”. In: *SIAM Journal on Numerical Analysis* 58.6, pp. 3144–3164.
- Feng, Berthy T. et al. (2023). *Score-Based Diffusion Models as Principled Priors for Inverse Imaging*. arXiv: 2304.11751 [cs.CV]. URL: <https://arxiv.org/abs/2304.11751>.

- Feyel, Frederic and Jean-Louis Chaboche (2000). “FE2 multiscale approach for modelling the elastoviscoplastic behaviour of long fibre SiC/Ti composite materials”. In: *Computer methods in applied mechanics and engineering* 183.3-4, pp. 309–330.
- Filippo Gazzola Hans-Christoph Grunau, Guido Sweers (2010). *Polyharmonic Boundary Value Problems*. Lecture Notes in Mathematics. Springer. DOI: <https://doi.org/10.1007/978-3-642-12245-3>.
- Fischbacher, Thomas and Franziska Synatschke-Czerwonka (Aug. 2013). “FlowPy- A numerical solver for functional renormalization group equations”. In: *Computer Physics Communications* 184.8, pp. 1931–1945. ISSN: 0010-4655. DOI: [10.1016/j.cpc.2013.03.002](https://doi.org/10.1016/j.cpc.2013.03.002). URL: <http://dx.doi.org/10.1016/j.cpc.2013.03.002>.
- Fish, Jacob et al. (1997). “Computational plasticity for composite structures based on mathematical homogenization: Theory and practice”. In: *Computer methods in applied mechanics and engineering* 148.1-2, pp. 53–73.
- Franceschi, Luca et al. (2017). *Forward and Reverse Gradient-Based Hyperparameter Optimization*. arXiv: 1703.01785 [stat.ML].
- Friedrich, Rudolf et al. (2011). “Approaching complexity by stochastic methods: From biological systems to turbulence”. In: *Physics Reports* 506.5, pp. 87–162. ISSN: 0370-1573. DOI: <https://doi.org/10.1016/j.physrep.2011.05.003>. URL: <https://www.sciencedirect.com/science/article/pii/S0370157311001530>.
- Garcia-Ojalvo, Jordi and Jose M. Sancho (1999). *Noise in Spatially Extended Systems*. Institute for Nonlinear Science. Springer New York, NY. ISBN: 978-0-387-98855-9. DOI: [10.1007/978-1-4612-1536-3](https://doi.org/10.1007/978-1-4612-1536-3).
- Gardner, Jacob R. et al. (2021). *GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration*. arXiv: 1809.11165 [cs.LG]. URL: <https://arxiv.org/abs/1809.11165>.
- Georgiou, Kyriakos, Constantinos Siettos, and Athanasios N. Yannacopoulos (2025a). “Fredholm Neural Networks”. In: *SIAM Journal on Scientific Computing* 47.4, pp. C1006–C1031. DOI: [10.1137/24M1686991](https://doi.org/10.1137/24M1686991). eprint: <https://doi.org/10.1137/24M1686991>. URL: <https://doi.org/10.1137/24M1686991>.
- (2025b). *Fredholm Neural Networks for forward and inverse problems in elliptic PDEs*. arXiv: 2507.06038 [math.NA]. URL: <https://arxiv.org/abs/2507.06038>.
- Ghanem, Roger G and Pol D Spanos (2003). *Stochastic finite elements: a spectral approach*. Dover Publications.
- (1991). *Stochastic Finite Elements: A Spectral Approach*. 1st ed. Softcover ISBN: 978-1-4612-7795-8, eBook ISBN: 978-1-4612-3094-6. Springer New York, NY, pp. X, 214. DOI: [10.1007/978-1-4612-3094-6](https://doi.org/10.1007/978-1-4612-3094-6).

- Giesl, P. et al. (2019). “Approximation of Lyapunov functions from noisy data”. In: *Journal of Computational Dynamics*. <https://arxiv.org/abs/1601.01568>. DOI: 10.3934/jcd.2020003.
- Gin, Craig R et al. (2021). “DeepGreen: deep learning of Green’s functions for nonlinear boundary value problems”. In: *Scientific reports* 11.1, p. 21614.
- Glimm, James and Arthur Jaffe (2012). *Quantum physics: a functional integral point of view*. Springer Science & Business Media.
- Gneiting, Tilmann, Fadoua Balabdaoui, and Adrian E. Raftery (Mar. 2007). “Probabilistic Forecasts, Calibration and Sharpness”. In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 69.2, pp. 243–268. ISSN: 1369-7412. DOI: 10.1111/j.1467-9868.2007.00587.x. eprint: https://academic.oup.com/jrsssb/article-pdf/69/2/243/49794500/jrsssb_69_2_243.pdf. URL: <https://doi.org/10.1111/j.1467-9868.2007.00587.x>.
- Gopakumar, Vignesh et al. (2024). “Plasma surrogate modelling using Fourier neural operators”. In: *Nuclear Fusion* 64.5, p. 056025.
- Goswami, Somdatta et al. (2023). “Physics-informed deep neural operator networks”. In: *Machine Learning in Modeling and Simulation: Methods and Applications*. Springer, pp. 219–254.
- Grecksch, W. and P.E. Kloeden (1996). “Time-discretised Galerkin approximations of parabolic stochastic PDEs”. In: *Bulletin of the Australian Mathematical Society* 54.1, p. 7985. DOI: 10.1017/S0004972700015094.
- Gretton, Arthur et al. (2012). “A Kernel Two-Sample Test”. In: *Journal of Machine Learning Research* 13.25, pp. 723–773. URL: <http://jmlr.org/papers/v13/gretton12a.html>.
- Grohs, Philipp, Samuel Lanthaler, and Margaret Trautner (2025). “Theory-to-practice gap for neural networks and neural operators”. In: *arXiv preprint arXiv:2503.18219*.
- Grossi, Eduardo and Nicolas Wink (2023). “Resolving phase transitions with Discontinuous Galerkin methods”. In: *SciPost Phys. Core* 6.4, p. 071. DOI: 10.21468/SciPostPhysCore.6.4.071.
- Gunzburger, Max D, Clayton G Webster, and Guannan Zhang (2014). “Stochastic finite element methods for partial differential equations with random input data”. In: *Acta Numerica* 23, pp. 521–650.
- Gupta, Gaurav, Xiongye Xiao, and Paul Bogdan (2021). “Multiwavelet-based operator learning for differential equations”. In: *Advances in neural information processing systems* 34, pp. 24048–24062.
- Gyongy, Istvan and Annie Millet (2009). “Rate of Convergence of Space Time Approximations for Stochastic Evolution Equations”. In: *Potential Analysis* 30.1, pp. 29–64. ISSN: 1572-929X. DOI: 10.1007/s11118-008-9105-5. URL: <https://doi.org/10.1007/s11118-008-9105-5>.

- Haasdonk, B. et al. (2018). “Greedy Kernel Methods for Center Manifold Approximation”. In: *Proc. of ICOSAHOM 2018, International Conference on Spectral and High Order Methods* 1. <https://arxiv.org/abs/1810.11329>.
- Haber, Eldad (2004). “Quasi-Newton methods for large-scale electromagnetic inverse problems”. In: *Inverse problems* 21.1, p. 305.
- Haber, Eldad, Uri M Ascher, and Doug Oldenburg (2000). “On optimization techniques for solving nonlinear inverse problems”. In: *Inverse problems* 16.5, p. 1263.
- Hairer, M. (Mar. 2014). “A theory of regularity structures”. In: *Inventiones mathematicae* 198.2, p. 269504. ISSN: 1432-1297. DOI: 10.1007/s00222-014-0505-4. URL: <http://dx.doi.org/10.1007/s00222-014-0505-4>.
- Hairer, M., A. M. Stuart, and J. Voss (2007). “Analysis of SPDEs arising in path sampling part II: The nonlinear case”. In: *The Annals of Applied Probability* 17.5-6, pp. 1657–1706. DOI: 10.1214/07-AAP441. URL: <https://doi.org/10.1214/07-AAP441>.
- Hairer, M., A. M. Stuart, J. Voss, and P. Wiberg (2005). “Analysis of SPDEs arising in path sampling. Part I: The Gaussian case”. In: *Communications in Mathematical Sciences* 3.4, pp. 587–603.
- Hairer, Martin (2012). *Solving the KPZ equation*. arXiv: 1109.6811 [math.PR]. URL: <https://arxiv.org/abs/1109.6811>.
- (July 3, 2023). *An Introduction to Stochastic PDEs*. DOI: 10.48550/arXiv.0907.4178. arXiv: 0907.4178. URL: <http://arxiv.org/abs/0907.4178> (visited on 04/15/2026).
- (n.d.). URL: <https://hairer.org/notes/SPDEs.pdf>.
- Halko, Nathan, Per-Gunnar Martinsson, and Joel A Tropp (2011). “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions”. In: *SIAM review* 53.2, pp. 217–288.
- Hamzi, Boumediene and Fritz Colonius (2019). “Kernel methods for the approximation of discrete-time linear autonomous and control systems”. In: *SN Applied Sciences* 1.7, pp. 1–12.
- Hamzi, Boumediene, Marcus Hutter, and Houman Owhadi (2024). “Bridging Algorithmic Information Theory and Machine Learning: A new approach to kernel learning”. In: *Physica D: Nonlinear Phenomena* 464, p. 134153. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2024.134153>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278924001040>.
- Hamzi, Boumediene, Christian Kuehn, and Sameh Mohamed (2019). “A note on kernel methods for multiscale systems with critical transitions”. In: *Mathematical Methods in the Applied Sciences* 42.3, pp. 907–917. DOI: <https://doi.org/10.1002/mma.5394>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/mma.5394>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/mma.5394>.

- Hamzi, Boumediene, Romit Maulik, and Houman Owhadi (2021). “Simple, low-cost and accurate data-driven geophysical forecasting with learned kernels”. In: *Proceedings of the Royal Society A* 477.2252, p. 20210326.
- Hamzi, Boumediene and Houman Owhadi (2021a). “Learning dynamical systems from data: A simple cross-validation perspective, part I: Parametric kernel flows”. In: *Physica D: Nonlinear Phenomena* 421, p. 132817. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2020.132817>. URL: <https://www.sciencedirect.com/science/article/pii/S0167278920308186>.
- (2021b). “Learning dynamical systems from data: a simple cross-validation perspective, part I: parametric kernel flows”. In: *Physica D: Nonlinear Phenomena* 421, p. 132817.
- Hamzi, Boumediene, Houman Owhadi, and Leo Paillet (2023). “A note on microlocal kernel design for some slowfast stochastic differential equations with critical transitions and application to EEG signals”. In: *Physica A: Statistical Mechanics and its Applications* 616, p. 128583. ISSN: 0378-4371. DOI: <https://doi.org/10.1016/j.physa.2023.128583>. URL: <https://www.sciencedirect.com/science/article/pii/S0378437123001383>.
- Hao, Wenrui, Xinliang Liu, and Yahong Yang (2024). “Newton informed neural operator for solving nonlinear partial differential equations”. In: *Advances in neural information processing systems* 37, pp. 120832–120860.
- Hao, Zhongkai, Chang Su, et al. (2024). “DPOT: auto-regressive denoising operator transformer for large-scale PDE pre-training”. In: *Proceedings of the 41st International Conference on Machine Learning*. ICML’24. Vienna, Austria: JMLR.org.
- Hao, Zhongkai, Zhengyi Wang, et al. (2023). “GNOT: A General Neural Operator Transformer for Operator Learning”. In: *Proceedings of the 40th International Conference on Machine Learning*. ICML’23. Honolulu, Hawaii, USA: JMLR.org.
- Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. Springer. ISBN: 978-0-387-84857-0. DOI: <https://doi.org/10.1007/978-0-387-84858-7>.
- He, Juncai, Xinliang Liu, and Jinchao Xu (2024). “MgNO: Efficient Parameterization of Linear Operators via Multigrid”. In: *Proceedings of the Twelfth International Conference on Learning Representations (ICLR)*. <https://arxiv.org/abs/2310.19809>.
- Hennig, Philipp, Michael A. Osborne, and Hans Kersting (2022). *Probabilistic numerics: computation as machine learning*. Cambridge New York, NY Melbourne New Delhi Singapore: Cambridge University Press. 398 pp. ISBN: 9781107163447.
- Henry, Daniel (2006). *Geometric theory of semilinear parabolic equations*. Vol. 840. Springer.

- Hensman, James, Nicolo Fusi, and Neil D. Lawrence (2013). *Gaussian Processes for Big Data*. arXiv: 1309.6835 [cs.LG]. URL: <https://arxiv.org/abs/1309.6835>.
- Hensman, James, Magnus Rattray, and Neil D. Lawrence (2012). *Fast Variational Inference in the Conjugate Exponential Family*. arXiv: 1206.5162 [cs.LG]. URL: <https://arxiv.org/abs/1206.5162>.
- Herde, Maximilian et al. (2024). “Poseidon: Efficient Foundation Models for PDEs”. In: *Advances in Neural Information Processing Systems*. Vol. 37. DOI: 10.52202/079017–2311.
- Hesthaven, J.S. and S. Ubbiali (2018). “Non-intrusive reduced order modeling of nonlinear problems using neural networks”. In: *Journal of Computational Physics* 363, pp. 55–78.
- Hesthaven, Jan S, Gianluigi Rozza, Benjamin Stamm, et al. (2016). *Certified reduced basis methods for parametrized partial differential equations*. Vol. 590. Springer.
- Ho, Kenneth L. and Lexing Ying (2016). “Hierarchical interpolative factorization for elliptic operators: integral equations”. In: *Comm. Pure Appl. Math.* 69.7, pp. 1314–1353. ISSN: 0010-3640. DOI: 10.1002/cpa.21577. URL: <https://doi.org/10.1002/cpa.21577>.
- Huang, Daniel Zhengyu, Tapio Schneider, and Andrew M Stuart (2022). “Iterated Kalman methodology for inverse problems”. In: *Journal of Computational Physics* 463, p. 111262.
- Hudes, Matthew and Gregory Eyink (2025). “Probabilistic Interpretation of FRG”. In progress.
- Hudson, John L. et al. (1990). “Nonlinear signal processing and system identification: Applications to time series from electrochemical reactions”. In: *Chemical Engineering Science* 45.8, pp. 2075–2081. ISSN: 0009-2509. DOI: [https://doi.org/10.1016/0009-2509\(90\)80079-T](https://doi.org/10.1016/0009-2509(90)80079-T). URL: <https://www.sciencedirect.com/science/article/pii/000925099080079T>.
- Hutter, Frank, Holger H. Hoos, and Kevin Leyton-Brown (2011). “Sequential Model-Based Optimization for General Algorithm Configuration”. In: *Learning and Intelligent Optimization*. Ed. by Carlos A. Coello Coello. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 507–523. ISBN: 978-3-642-25566-3.
- Hyndman, Rob and G. Athanasopoulos (2021). *Forecasting: Principles and Practice*. English. 3rd. Australia: OTexts.
- I. Bogachev, Vladimir (1998). *Gaussian measures*. American Mathematical Society.
- Infante, Saba et al. (Dec. 2016). “Approximations of the solutions of a stochastic differential equation using Dirichlet process mixtures and Gaussian mixtures”. In: *Statistics, Optimization & Information Computing* 4.4, pp. 289–307. DOI: 10.19139/soic.v4i4.242. URL: <http://www.iapress.org/index.php/soic/article/view/20161202>.

- Jalalian, Yasamin et al. (2025). “Data-Efficient Kernel Methods for Learning Differential Equations and Their Solution Operators: Algorithms and Error Analysis”. In: *arXiv preprint arXiv:2503.01036*.
- Jentzen, Arnulf and Michael Rockner (2012). “A Milstein scheme for stochastic partial differential equations”. In: *Journal of Differential Equations* 252.1, pp. 114–136. DOI: 10.1016/j.jde.2011.08.050.
- Jnini, Anas, Elham Kiyani, et al. (2026). *Curvature-Aware Optimization for High-Accuracy Physics-Informed Neural Networks*. arXiv: 2604.05230 [cs.LG]. URL: <https://arxiv.org/abs/2604.05230>.
- Jnini, Anas, Flavio Vella, and Marius Zeinhofer (2024). “Gauss-newton natural gradient descent for physics-informed computational fluid dynamics”. In: *arXiv preprint arXiv:2402.10680*.
- Kadri, Hachem, Emmanuel Duflos, Philippe Preux, Stephane Canu, et al. (2015). “Operator-valued Kernels for Learning from Functional Response Data”. In.
- (2016a). “Operator-valued kernels for learning from functional response data”. In.
- (2016b). “Operator-valued kernels for learning from functional response data”. In: *Journal of Machine Learning Research* 17.20, pp. 1–54.
- Kantz, Holger and Thomas Schreiber (1997). *Nonlinear Time Series Analysis*. USA: Cambridge University Press. ISBN: 0521551447.
- Kardar, Mehran, Giorgio Parisi, and Yi-Cheng Zhang (1986). “Dynamic scaling of growing interfaces”. In: *Physical Review Letters* 56.9, p. 889.
- Karnakov, Petr, Sergey Litvinov, and Petros Koumoutsakos (Jan. 2024). “Solving inverse problems in physics by optimizing a discrete loss: Fast and accurate learning without neural networks”. In: *PNAS Nexus* 3.1, pgae005. ISSN: 2752-6542. DOI: 10.1093/pnasnexus/pgae005. eprint: <https://academic.oup.com/pnasnexus/article-pdf/3/1/pgae005/56347037/pgae005.pdf>. URL: <https://doi.org/10.1093/pnasnexus/pgae005>.
- Karniadakis, George Em, Ioannis G Kevrekidis, et al. (2021a). “Physics-informed machine learning”. In: *Nature Reviews Physics* 3.6, pp. 422–440.
- (May 24, 2021b). “Physics-informed machine learning”. In: *Nature Reviews Physics* 3.6, pp. 422–440. ISSN: 2522-5820. DOI: 10.1038/s42254-021-00314-5. URL: <https://www.nature.com/articles/s42254-021-00314-5> (visited on 04/14/2026).
- Kennedy, Marc C and Anthony O’Hagan (2001). “Bayesian calibration of computer models”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.3, pp. 425–464.

- Khoo, Yuehaw, Jianfeng Lu, and Lexing Ying (2021). “Solving parametric PDE problems with artificial neural networks”. In: *European Journal of Applied Mathematics* 32.3, pp. 421–435.
- Khoo, Yuehaw and Lexing Ying (2019). “SwitchNet: a neural network model for forward and inverse scattering problems”. In: *SIAM Journal on Scientific Computing* 41.5, A3182–A3201.
- Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG]. URL: <https://arxiv.org/abs/1412.6980>.
- Kingma, Diederik P., Tim Salimans, et al. (2017). *Improving Variational Inference with Inverse Autoregressive Flow*. arXiv: 1606.04934 [cs.LG]. URL: <https://arxiv.org/abs/1606.04934>.
- Kissas, Georgios et al. (2022). “Learning operators with coupled attention”. In: *Journal of Machine Learning Research* 23.215, pp. 1–63.
- Kleinhans, David and Rudolf Friedrich (2007). “Quantitative Estimation of Drift and Diffusion Functions from Time Series Data”. In: *Wind Energy*. Ed. by Joachim Peinke, Peter Schaumann, and Stephan Barth. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 129–133. ISBN: 978-3-540-33866-6.
- Klenke, Achim (2008). *Probability theory*. URL: <https://link.springer.com/book/10.1007/978-1-84800-048-3>.
- Klimontovich, Yu (Feb. 1993). “The Reconstruction of the Fokker-Planck and Master Equations on the Basis of Experimental Data: H-Theorem and S-Theorem”. In: *International Journal of Bifurcation and Chaos* 3, pp. 113–. DOI: 10.1142/S0218127493000076.
- Klus, S. et al. (2020). “Data-driven approximation of the Koopman generator: Model reduction, system identification, and control”. In: *Physica D: Nonlinear Phenomena* 406, p. 132416. DOI: 10.1016/j.physd.2020.132416.
- Klus, Stefan, Feliks Nuske, and Boumediene Hamzi (2020). “Kernel-Based Approximation of the Koopman Generator and Schrodinger Operator”. In: *Entropy* 22. <https://www.mdpi.com/1099-4300/22/7/722>.
- Klus, Stefan, Feliks Nuske, et al. (2020). “Data-driven approximation of the Koopman generator: Model reduction, system identification, and control”. In: *Physica D: Nonlinear Phenomena* 406, p. 132416.
- Kossaiifi, Jean et al. (Jan. 2026). *Demystifying Data-Driven Probabilistic Medium-Range Weather Forecasting*. en. URL: <https://arxiv.org/abs/2601.18111v1> (visited on 04/15/2026).
- Kovachki, Nikola, Samuel Lanthaler, and Siddhartha Mishra (2021). “On universal approximation and error bounds for fourier neural operators”. In: *The Journal of Machine Learning Research* 22.1, pp. 13237–13312.

- Kovachki, Nikola, Zongyi Li, et al. (2023). “Neural operator: Learning maps between function spaces with applications to pdes”. In: *Journal of Machine Learning Research* 24.89, pp. 1–97.
- Kovachki, Nikola, Burigede Liu, et al. (2022). “Multiscale modeling of materials: Computing, data science, uncertainty and goal-oriented optimization”. In: *Mechanics of Materials* 165, p. 104156.
- Kovachki, Nikola B, Samuel Lanthaler, and Andrew M Stuart (2024). “Operator learning: Algorithms and analysis”. In: *Handbook of Numerical Analysis* 25, pp. 419–467.
- Krischer, K et al. (1993). “Model identification of a spatiotemporally varying catalytic reaction”. In: *AIChE Journal* 39.1, pp. 89–98.
- Kropfl, Fabian, Roland Maier, and Daniel Peterseim (2022). “Operator compression with deep neural networks”. In: *Advances in Continuous and Discrete Models* 2022.1, pp. 1–23.
- Laing, Carlo and Gabriel J Lord (Sept. 2009). *Stochastic Methods in Neuroscience*. Oxford University Press. ISBN: 9780199235070. DOI: 10.1093/acprof:oso/9780199235070.001.0001. URL: <https://doi.org/10.1093/acprof:oso/9780199235070.001.0001>.
- Lam, Remi et al. (2023). “Learning skillful medium-range global weather forecasting”. In: *Science* 382.6677, pp. 1416–1421.
- Lan, Shiwei et al. (Dec. 2025). “Spatiotemporal Besov Priors for Bayesian Inverse Problems”. In: *Journal of the American Statistical Association*, pp. 1–15. ISSN: 1537-274X. DOI: 10.1080/01621459.2025.2560688. URL: <http://dx.doi.org/10.1080/01621459.2025.2560688>.
- Lanckriet, Gert et al. (Jan. 2002). “Learning the Kernel Matrix with Semi-Definite Programming.” In: vol. 5, pp. 323–330.
- Lanthaler, Samuel, Andrew M Stuart, and Margaret Trautner (2024). “Discretization error of Fourier neural operators”. In: *arXiv preprint arXiv:2405.02221*.
- Larkin, F.M. (Sept. 1972a). “Gaussian measure in Hilbert space and applications in numerical analysis”. In: *Rocky Mountain Journal of Mathematics* 2.3. ISSN: 0035-7596. DOI: 10.1216/RMJ-1972-2-3-379. URL: <https://projecteuclid.org/journals/rocky-mountain-journal-of-mathematics/volume-2/issue-3/Gaussian-measure-in-Hilbert-space-and-applications-in-numerical-analysis/10.1216/RMJ-1972-2-3-379.full> (visited on 04/14/2026).
- (1972b). “Gaussian measure in Hilbert space and applications in numerical analysis”. In: *The Rocky Mountain Journal of Mathematics*, pp. 379–421.
- Ledoux, Michel (1996). “Isoperimetry and gaussian analysis”. In: *Lecture Notes in Mathematics*, pp. 165–294. DOI: 10.1007/bfb0095676.

- Lee, Jonghyeon, Edward De Brouwer, et al. (2021). *Learning dynamical systems from data: A simple cross-validation perspective, part III: Irregularly-Sampled Time Series*. arXiv: 2111.13037 [stat.ML].
- Lee, Jonghyeon, Boumediene Hamzi, et al. (2024). *Gaussian Processes simplify differential equations*. <https://arxiv.org/abs/2410.03003>. arXiv: 2410.03003 [math.DS]. URL: <https://arxiv.org/abs/2410.03003>.
- Lengyel, Daniel et al. (2024). *Kernel Sum of Squares for Data Adapted Kernel Learning of Dynamical Systems from Data: A global optimization approach*. <https://arxiv.org/abs/2408.06465>. arXiv: 2408.06465 [cs.LG]. URL: <https://arxiv.org/abs/2408.06465>.
- Li, Shuo-Hui and Lei Wang (Dec. 2018). “Neural Network Renormalization Group”. In: *Phys. Rev. Lett.* 121 (26), p. 260601. DOI: 10.1103/PhysRevLett.121.260601. URL: <https://link.aps.org/doi/10.1103/PhysRevLett.121.260601>.
- Li, Wei et al. (2025). “LMO: Linear Mamba Operator for MRI Reconstruction”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 5112–5122.
- Li, Zijie, Kazem Meidani, and Amir Barati Farimani (2023). “Transformer for Partial Differential Equationsextrateright Operator Learning”. In: *Transactions on Machine Learning Research*. ISSN: 2835-8856.
- Li, Zongyi, Nikola Kovachki, et al. (2020a). *Fourier Neural Operator for Parametric Partial Differential Equations*.
- (2020b). “Fourier neural operator for parametric partial differential equations”. In: *arXiv preprint arXiv:2010.08895*.
- Li, Zongyi, Miguel Liu-Schiaffini, et al. (2021). “Learning dissipative dynamics in chaotic systems”. In: *arXiv preprint arXiv:2106.06898*.
- Li, Zongyi, Hongkai Zheng, et al. (2023a). *Physics-Informed Neural Operator for Learning Partial Differential Equations*. arXiv: 2111.03794 [cs.LG]. URL: <https://arxiv.org/abs/2111.03794>.
- (2023b). *Physics-Informed Neural Operator for Learning Partial Differential Equations*. arXiv: 2111.03794 [cs.LG]. URL: <https://arxiv.org/abs/2111.03794>.
- (May 2024a). “Physics-Informed Neural Operator for Learning Partial Differential Equations”. In: *ACM / IMS J. Data Sci.* 1.3. DOI: 10.1145/3648506. URL: <https://doi.org/10.1145/3648506>.
- (2024b). “Physics-informed neural operator for learning partial differential equations”. In: *ACM/JMS Journal of Data Science* 1.3, pp. 1–27.

- Lin, Daozhe and Qiang Du (2025). *Nonlinear Optimal Recovery in Hilbert Spaces*. arXiv: 2506.00704 [math.NA]. URL: <https://arxiv.org/abs/2506.00704>.
- Litim, Daniel F (2001). “Optimized renormalization group flows”. In: *Physical Review D* 64.10, p. 105007.
- Long, Da et al. (2024). “A kernel framework for learning differential equations and their solution operators”. In: *Physica D: Nonlinear Phenomena* 460, p. 134095.
- Lord, Gabriel J., Catherine E. Powell, and Tony Shardlow (2014). *An Introduction to Computational Stochastic PDEs*. Cambridge Texts in Applied Mathematics. Cambridge University Press. doi: 10.1017/CBO9781139017329.
- Lord, Gabriel J. and Antoine Tambue (2019). “Stochastic exponential integrators for a finite element discretisation of SPDEs with additive noise”. In: *Applied Numerical Mathematics* 136, pp. 163–182. ISSN: 0168-9274. doi: <https://doi.org/10.1016/j.apnum.2018.10.008>. URL: <https://www.sciencedirect.com/science/article/pii/S0168927418302344>.
- Lu, Lu, Pengzhan Jin, and George Em Karniadakis (2019). “Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators”. In: *arXiv preprint arXiv:1910.03193*.
- Lu, Lu, Pengzhan Jin, Guofei Pang, et al. (Mar. 2021a). “Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators”. In: *Nature Machine Intelligence* 3.3, pp. 218–229.
- (2021b). “Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators”. In: *Nature machine intelligence* 3.3, pp. 218–229.
- Lu, Lu, Xuhui Meng, et al. (2022). “A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data”. In: *Computer Methods in Applied Mechanics and Engineering* 393, p. 114778. ISSN: 0045-7825.
- Lu, Yang et al. (2023). “Learning Dynamical Systems from Data: A Simple Cross-Validation Perspective, Part V: Sparse Kernel Flows for 132 Chaotic Dynamical Systems”. In: *Physica D* 460.134070.
- Lucia, David J, Philip S Beran, and Walter A Silva (2004). “Reduced-order modeling: new approaches for computational physics”. In: *Progress in aerospace sciences* 40.1-2, pp. 51–117.
- Maclaurin, Dougal, David Duvenaud, and Ryan P. Adams (2015). *Gradient-based Hyperparameter Optimization through Reversible Learning*. arXiv: 1502.03492 [stat.ML].
- Maday, Yvon, Anthony T Patera, and Gabriel Turinici (2002). “A priori convergence theory for reduced-basis approximations of single-parameter elliptic partial differential equations”. In: *Journal of Scientific Computing* 17, pp. 437–446.

- Malqvist, Axel and Daniel Peterseim (2014). “Localization of elliptic multiscale problems”. In: *Mathematics of Computation* 83.290, pp. 2583–2603.
- Mardani, Morteza et al. (2023). *A Variational Perspective on Solving Inverse Problems with Diffusion Models*. arXiv: 2305.04391 [cs.LG]. URL: <https://arxiv.org/abs/2305.04391>.
- Martin, James et al. (2012). “A stochastic Newton MCMC method for large-scale statistical inverse problems with application to seismic inversion”. In: *SIAM Journal on Scientific Computing* 34.3, A1460–A1487.
- Martins, Joaquim RRA and Andrew B Lambe (2013). “Multidisciplinary design optimization: a survey of architectures”. In: *AIAA journal* 51.9, pp. 2049–2075.
- Marzari, Nicola et al. (2012). “Maximally localized Wannier functions: Theory and applications”. In: *Reviews of Modern Physics* 84.4, p. 1419.
- Meng, Rui and Xianjin Yang (2023a). “Sparse Gaussian processes for solving nonlinear PDEs”. In: *Journal of Computational Physics* 490, p. 112340.
- (Oct. 2023b). “Sparse Gaussian processes for solving nonlinear PDEs”. In: *Journal of Computational Physics* 490, p. 112340. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2023.112340. URL: <http://dx.doi.org/10.1016/j.jcp.2023.112340>.
- Metzner, Walter et al. (2012). “Functional renormalization group approach to correlated fermion systems”. In: *Rev. Mod. Phys.* 84, pp. 299–352. DOI: 10.1103/RevModPhys.84.299.
- Micchelli, Charles and Massimiliano Pontil (2004). “Kernels for Multi-task Learning”. In: *Advances in neural information processing systems* 17.
- Mishra, Siddhartha and Roberto Molinaro (Apr. 2022). “Estimates on the generalization error of physics-informed neural networks for approximating a class of inverse problems for PDEs”. In: *IMA Journal of Numerical Analysis* 42.2, pp. 981–1022. DOI: 10.1093/imanum/drab032.
- Miller, Jennifer L. and Samuli Siltanen (2012). *Linear and Nonlinear Inverse Problems with Practical Applications*. USA: Society for Industrial and Applied Mathematics. ISBN: 1611972337.
- Molinaro, Roberto et al. (23–29 Jul 2023). “Neural Inverse Operators for Solving PDE Inverse Problems”. In: vol. 202. *Proceedings of Machine Learning Research*. PMLR, pp. 25105–25139. URL: <https://proceedings.mlr.press/v202/molinaro23a.html>.
- Monga, Vishal, Yuelong Li, and Yonina C Eldar (2021). “Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing”. In: *IEEE Signal Processing Magazine* 38.2, pp. 18–44.
- Montanelli, Hadrien (n.d.). *Machine Learning for Scientific Computing and Numerical Analysis*.

- Mora, Carlos et al. (2025). “Operator learning with Gaussian processes”. In: *Computer Methods in Applied Mechanics and Engineering* 434, p. 117581.
- Mou, Chencheng, Xianjin Yang, and Chao Zhou (2022). “Numerical methods for mean field games based on Gaussian processes and Fourier features”. In: *Journal of Computational Physics* 460, p. 111188.
- Mukherjee, S., E. Osuna, and F. Girosi (1997). “Nonlinear prediction of chaotic time series using support vector machines”. In: *Neural Networks for Signal Processing VII. Proceedings of the 1997 IEEE Signal Processing Society Workshop*, pp. 511–520.
- Murphy, Kevin P. (2025). *Probabilistic machine learning: an introduction*. eng. Third printing. Adaptive computation and machine learning. Cambridge, Massachusetts London, England: The MIT Press. ISBN: 9780262046824.
- Nelsen, Nicholas H, Houman Owhadi, et al. (2025). “Bilevel optimization for learning hyperparameters: Application to solving PDEs and inverse problems with Gaussian processes”. In: *arXiv preprint arXiv:2510.05568*.
- Nelsen, Nicholas H and Andrew M Stuart (2021). “The random feature model for input-output maps between banach spaces”. In: *SIAM Journal on Scientific Computing* 43.5, A3212–A3243.
- (2024). “Operator learning using random features: A tool for scientific computing”. In: *SIAM Review* 66.3, pp. 535–571.
- Nelsen, Nicholas H. et al. (2025). *Bilevel optimization for learning hyperparameters: Application to solving PDEs and inverse problems with Gaussian processes*. arXiv: 2510.05568 [stat.ML]. URL: <https://arxiv.org/abs/2510.05568>.
- Nielsen, A. (2019). *Practical Time Series Analysis: Prediction with Statistics and Machine Learning*. O’Reilly Media. ISBN: 9781492041627.
- Nobile, Fabio, Raul Tempone, and Clayton G Webster (2008a). “A sparse grid stochastic collocation method for partial differential equations with random input data”. In: *SIAM Journal on Numerical Analysis* 46.5, pp. 2309–2345.
- (2008b). “An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data”. In: *SIAM Journal on Numerical Analysis* 46.5, pp. 2411–2442.
- Nocedal, Jorge and Stephen J Wright (2006a). *Numerical optimization*. Springer.
- (Dec. 2006b). *Numerical Optimization*. 2nd ed. Springer Series in Operations Research and Financial Engineering. Springer New York, NY. ISBN: 978-0-387-30303-1. DOI: 10.1007/978-0-387-40065-5. URL: <https://doi.org/10.1007/978-0-387-40065-5>.
- (2006c). *Numerical Optimization*. 2e. New York, NY, USA: Springer.

- Noor, Ahmed K and Jeanne M Peters (1980). “Reduced basis technique for nonlinear analysis of structures”. In: *Aiaa journal* 18.4, pp. 455–462.
- O’hagan, A (Aug. 1992). “Some Bayesian Numerical Analysis”. en. In: *Bayesian Statistics 4*. Ed. by J M Bernardo et al. Oxford University PressOxford, pp. 345–363. ISBN: 9780198522669 9781383023756. DOI: 10.1093/oso/9780198522669.003.0019. URL: <https://academic.oup.com/book/54041/chapter/422209719> (visited on 04/15/2026).
- Opper, Manfred (2019). “Variational Inference for Stochastic Differential Equations”. In: *Annalen der Physik* 531.3, p. 1800233. DOI: <https://doi.org/10.1002/andp.201800233>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/andp.201800233>. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1002/andp.201800233>.
- Owhadi, Boumediene Hamzi , Romit Maulik, Houman (2021). “Simple, low-cost and accurate data-driven geophysical forecasting with learned kernels”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 477(2252). DOI: 10.1098/rspa.2021.0326..
- Owhadi, H. and G. R. Yoo (2019). “Kernel Flows: From learning kernels from data into the abyss”. In: *Journal of Computational Physics* 389, pp. 22–47. DOI: 10.1016/j.jcp.2019.03.040.
- Owhadi, Houman (Jan. 2015a). “Bayesian Numerical Homogenization”. en. In: *Multiscale Modeling & Simulation* 13.3, pp. 812–828. ISSN: 1540-3459, 1540-3467. DOI: 10.1137/140974596. URL: <http://epubs.siam.org/doi/10.1137/140974596> (visited on 04/15/2026).
- (2015b). “Bayesian numerical homogenization”. In: *Multiscale Modeling & Simulation* 13.3, pp. 812–828.
 - (2017). “Multigrid with rough coefficients and multiresolution operator decomposition from hierarchical information games”. In: *SIAM Review* 59.1, pp. 99–149.
 - (2020). *Do ideas have shape? Idea registration as the continuous limit of artificial neural networks*.
 - (2022a). “Computational Graph Completion”. In: *Research in the Mathematical Sciences* 9.27.
 - (2022b). “Computational graph completion”. In: *Research in the Mathematical Sciences* 9.2, p. 27.
 - (2023a). “Do ideas have shape? Idea registration as the continuous limit of artificial neural networks”. In: *Physica D: Nonlinear Phenomena* 444, p. 133592.
 - (2023b). “Do ideas have shape? Idea registration as the continuous limit of artificial neural networks”. In: *Physica D: Nonlinear Phenomena* 444. arXiv:2008.03920, 2020, p. 133592.

- Owhadi, Houman and Clint Scovel (2016). “Brittleness of Bayesian inference and new Selberg formulas”. In: *Commun. Math. Sci.* 14.1, pp. 83–145. ISSN: 1539-6746. DOI: [10.4310/CMS.2016.v14.n1.a5](https://doi.org/10.4310/CMS.2016.v14.n1.a5). URL: <https://doi.org/10.4310/CMS.2016.v14.n1.a5>.
- (2017). “Qualitative robustness in Bayesian inference”. In: *ESAIM: Probability and Statistics* 21, pp. 251–274.
 - (2018). “Conditioning Gaussian Measure on Hilbert Space”. In: *Journal of Mathematical and Statistical Analysis* 1.109.
 - (2019a). *Operator adapted wavelets, fast solvers, and numerical homogenization: from a game theoretic approach to numerical approximation and algorithm design*. eng. Cambridge monographs on applied and computational mathematics 35. Cambridge New York, NY: Cambridge University Press. ISBN: 9781108594967.
 - (2019b). *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization: From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press. doi: [10.1017/9781108594967](https://doi.org/10.1017/9781108594967).
 - (2019c). *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization: From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*. Vol. 35. Cambridge University Press.
- Owhadi, Houman, Clint Scovel, and Florian Schafer (2019). “Statistical numerical approximation”. In: *Notices of the AMS* 66.10.
- Owhadi, Houman, Clint Scovel, and Gene Ryan Yoo (2021a). *Kernel Mode Decomposition and the programming of kernels*. Springer.
- (2021b). *Kernel Mode Decomposition and the programming of kernels*. Springer.
- Owhadi, Houman and Gene Ryan Yoo (2019a). “Kernel Flows: From learning kernels from data into the abyss”. In: *Journal of Computational Physics* 389, pp. 22–47.
- (2019b). “Kernel flows: from learning kernels from data into the abyss”. In: *Journal of Computational Physics* 389, pp. 22–47.
- Owhadi, Houman and Lei Zhang (2007a). “Metric-based upscaling”. In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 60.5, pp. 675–723.
- (2007b). “Metric-based upscaling”. In: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 60.5, pp. 675–723.
 - (2017). “Gamblits for opening the complexity-bottleneck of implicit schemes for hyperbolic and parabolic ODEs/PDEs with rough coefficients”. In: *Journal of Computational Physics* 347, pp. 99–128.

- Paszke, Adam et al. (2019). *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. arXiv: 1912.01703 [cs.LG]. URL: <https://arxiv.org/abs/1912.01703>.
- Pathak, Jaideep, Zhixin Lu, et al. (2017). “Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 27.12, p. 121102. DOI: 10.1063/1.5010300. URL: <https://doi.org/10.1063/1.5010300>.
- Pathak, Jaideep, Shashank Subramanian, et al. (Feb. 2022). *FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators*. en. URL: <https://arxiv.org/abs/2202.11214v1> (visited on 04/15/2026).
- Pawlowski, Jan M (2007). “Aspects of the functional renormalisation group”. In: *Annals of Physics* 322.12, pp. 2831–2915.
- Pazouki, Maryam and Robert Schaback (2011). “Bases for kernel-based spaces”. In: *Journal of Computational and Applied Mathematics* 236.4. International Workshop on Multivariate Approximation and Interpolation with Applications (MAIA 2010), pp. 575–588. ISSN: 0377-0427. DOI: <https://doi.org/10.1016/j.cam.2011.05.021>. URL: <http://www.sciencedirect.com/science/article/pii/S0377042711002688>.
- Pedregosa, F. et al. (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830.
- Perrin, Olivier and Pascal Monestiez (1999). “Modelling of non-stationary spatial structure using parametric radial basis deformations”. In: *GeoENV II Geostatistics for Environmental Applications*. Springer, pp. 175–186.
- Pfortner, Marvin et al. (2022). “Physics-Informed Gaussian Process Regression Generalizes Linear PDE Solvers”. In: *arXiv preprint*.
- Polyak, Boris Teodorovich (2006). “Newton-Kantorovich method and its global convergence”. In: *Journal of Mathematical Sciences* 133, pp. 1513–1523.
- Posner, E. (July 1975). “Random coding strategy for minimum entropy”. In: *IEEE Transactions on Information Theory* 21.4, pp. 388–391. DOI: 10.1109/tit.1975.1055416.
- Prasanth, Sai, Ziad Haddad, et al. (2021). “Kernel flows to infer the structure of convective storms from satellite passive microwave observations”. In: *AGU Fall Meeting Abstracts*. Vol. 2021, A55F–1445.
- Prasanth, Sai, Ziad S Haddad, et al. (2021). “Kernel flows to infer the structure of convective storms from satellite passive microwave observations”. In: *preprint*.
- Printems, Jacques (2001). “On the discretization in time of parabolic stochastic partial differential equations”. In: *ESAIM: Mathematical Modelling and Numerical Analysis* 35.6, p. 10551078. DOI: 10.1051/m2an:2001148.

- Quarteroni, Alfio and Alberto Valli (1994). *Numerical approximation of partial differential equations*. Springer.
- Rahman, Md Ashiqur et al. (2024). “Pretraining Codomain Attention Neural Operators for Solving Multiphysics PDEs”. In: *arXiv preprint arXiv:2403.12553*.
- Raissi, M., P. Perdikaris, and G.E. Karniadakis (2019a). “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378, pp. 686–707. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2018.10.045>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999118307125>.
- (2019b). “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378, pp. 686–707. ISSN: 0021-9991. DOI: <https://doi.org/10.1016/j.jcp.2018.10.045>. URL: <https://www.sciencedirect.com/science/article/pii/S0021999118307125>.
- (Feb. 2019c). “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational Physics* 378, pp. 686–707. ISSN: 00219991. DOI: [10.1016/j.jcp.2018.10.045](https://doi.org/10.1016/j.jcp.2018.10.045). URL: <https://linkinghub.elsevier.com/retrieve/pii/S0021999118307125> (visited on 04/14/2026).
- Rancon, Adam et al. (2025). “Dreaming up scale invariance via inverse renormalization group”. In: *arXiv preprint arXiv:2506.04016*.
- Raonic, Bogdan et al. (2023). “Convolutional neural operators for robust and accurate learning of PDEs”. In: *Advances in Neural Information Processing Systems* 36, pp. 77187–77200.
- Rasmussen, Carl Edward and Christopher K. I. Williams (2005). *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press. ISBN: 026218253X.
- (2006). *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, pp. I–XVIII, 1–248. ISBN: 026218253X.
- (2008). *Gaussian processes for machine learning*. eng. 3. print. Adaptive computation and machine learning. Cambridge, Mass.: MIT Press. ISBN: 9780262182539.
- Reinhardt, Niklas, Sven Wang, and Jakob Zech (2024). “Statistical learning theory for neural operators”. In: *arXiv preprint arXiv:2412.17582*.
- Rezende, Danilo Jimenez and Shakir Mohamed (2016). *Variational Inference with Normalizing Flows*. arXiv: 1505.05770 [stat.ML]. URL: <https://arxiv.org/abs/1505.05770>.
- Risken, H. and H. Haken (1989). *The Fokker-Planck Equation: Methods of Solution and Applications Second Edition*. Springer.

- Rudy, Samuel et al. (2019). “Data-driven identification of parametric partial differential equations”. In: *SIAM Journal on Applied Dynamical Systems* 18.2, pp. 643–660.
- Rulliere, Didier et al. (2018). “Nested Kriging predictions for datasets with a large number of observations”. In: *Statistics and Computing* 28.4, pp. 849–867. DOI: 10.1007/s11222-017-9766-2.
- Saksman, Matti Lassas. Eero and Samuli Siltanen (2009). *Discretization-invariant Bayesian inversion and Besov space priors*. arXiv: 0901.4220 [math.PR]. URL: <https://arxiv.org/abs/0901.4220>.
- Salimbeni, Hugh, Stefanos Eleftheriadis, and James Hensman (2018). *Natural Gradients in Practice: Non-Conjugate Variational Inference in Gaussian Process Models*. arXiv: 1803.09151 [stat.ML]. URL: <https://arxiv.org/abs/1803.09151>.
- Sampson, Paul D and Peter Guttorp (1992). “Nonparametric estimation of non-stationary spatial covariance structure”. In: *Journal of the American Statistical Association* 87.417, pp. 108–119.
- Santin, G. and B. Haasdonk (2019). “Kernel methods for surrogate modelling”. In: *ArXiv e-prints arXiv:1907.10556*. <https://arxiv.org/abs/1907.10556>.
- Santin, Gabriele and Bernard Haasdonk (2019). “Kernel Methods for Surrogate Modeling”. In: <https://arxiv.org/abs/1907.10556>.
- Sattler, Franz R and Jan M Pawłowski (2024a). “DiFfRG: A Discretisation Framework for functional Renormalisation Group flows”. In: *arXiv preprint arXiv:2412.13043*.
- (2024b). “DiFfRG: A Discretisation Framework for functional Renormalisation Group flows”. In: *arXiv*. DOI: 10.48550/arXiv.2412.13043. arXiv: 2412.13043 [hep-ph].
- Schaafer, Florian, Matthias Katzfuss, and Houman Owhadi (2021). “Sparse Cholesky Factorization by Kullback–Leibler Minimization”. In: *SIAM Journal on scientific computing* 43.3, A2019–A2046.
- Schafer, Florian, Matthias Katzfuss, and Houman Owhadi (2021). “Sparse Cholesky Factorization by Kullback–Leibler Minimization”. In: *SIAM Journal on Scientific Computing* 43.3, A2019–A2046. DOI: 10.1137/20M1336254. eprint: <https://doi.org/10.1137/20M1336254>. URL: <https://doi.org/10.1137/20M1336254>.
- Schafer, Florian and Houman Owhadi (2021). “Sparse recovery of elliptic solvers from matrix-vector products”. In: *arXiv preprint arXiv:2110.05351*.
- (2024). “Sparse Recovery of Elliptic Solvers from Matrix-Vector Products”. In: *SIAM Journal on Scientific Computing* 46.2, A998–A1025. DOI: 10.1137/22M154226X. eprint: <https://doi.org/10.1137/22M154226X>. URL: <https://doi.org/10.1137/22M154226X>.

- Schafer, Florian, Timothy John Sullivan, and Houman Owhadi (2021). “Compression, inversion, and approximate PCA of dense kernel matrices at near-linear computational complexity”. In: *Multiscale Modeling & Simulation* 19.2, pp. 688–730.
- Schilders, Wilhelmus HA, Henk A Van der Vorst, and Joost Rommes (2008). *Model order reduction: theory, research aspects and applications*. Vol. 13. Springer.
- Schmidt, Alexandra M and Anthony O’Hagan (2003). “Bayesian inference for non-stationary spatial covariance structure via spatial deformations”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 65.3, pp. 743–758.
- Scholkopf, Bernhard, Ralf Herbrich, and Alex J. Smola (2001a). “A Generalized Representer Theorem”. In: *Computational Learning Theory*. Ed. by David Helmbold and Bob Williamson. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 416–426. ISBN: 978-3-540-44581-4.
- (2001b). “A Generalized Representer Theorem”. In: *Computational Learning Theory*. Ed. by David Helmbold and Bob Williamson. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 416–426. ISBN: 978-3-540-44581-4.
- Siebert, S, R Friedrich, and J Peinke (July 1998). “Analysis of data sets of stochastic systems”. In: *Physics Letters A* 243.5-6, pp. 275–280. ISSN: 0375-9601. DOI: 10.1016/S0375-9601(98)00283-7. URL: [http://dx.doi.org/10.1016/S0375-9601\(98\)00283-7](http://dx.doi.org/10.1016/S0375-9601(98)00283-7).
- Snoek, Jasper, Hugo Larochelle, and Ryan P. Adams (2012). *Practical Bayesian Optimization of Machine Learning Algorithms*. arXiv: 1206.2944 [stat.ML].
- Sorokin, Aleksei, Pieterjan Robbe, and Fred J Hickernell (Jan. 2025). “Fast Gaussian process regression for high dimensional functions with derivative information”. In: *Proceedings of the First International Conference on Probabilistic Numerics*. Ed. by Motonobu Kanagawa et al. Vol. 271. Proceedings of Machine Learning Research. PMLR, pp. 35–49. URL: <https://proceedings.mlr.press/v271/sorokin25a.html>.
- Sorokin, Aleksei G. (2023). *Fast Gaussian Process Regression Models in Python*. URL: <https://alegresor.github.io/fastgps/>.
- Stuart, A. M. (2010). “Inverse problems: A Bayesian perspective”. In: *Acta Numerica* 19, pp. 451–559. DOI: 10.1017/S0962492910000061.
- Stuart, Andrew M., Jochen Voss, and Petter Wilberg (2004). “Conditional Path Sampling of SDEs and the Langevin MCMC Method”. In: *Communications in Mathematical Sciences* 2.4, pp. 685–697.
- Sudret, Bruno, Stefano Marelli, and Joe Wiart (2017). “Surrogate models for uncertainty quantification: An overview”. In: *2017 11th European conference on antennas and propagation (EUCAP)*. IEEE, pp. 793–797.

- Susiluoto, Jouni, Amy Braverman, Philip Brodrick, et al. (2021). “Radiative transfer emulation for hyperspectral imaging retrievals with advanced kernel flows-based gaussian process emulation”. In: *AGU Fall Meeting Abstracts*. Vol. 2021, NG25A–0506.
- Susiluoto, Jouni, Amy Braverman, Philip G. Brodrick, et al. (2021). “Radiative transfer emulation for hyperspectral imaging retrievals with advanced Kernel Flows-based Gaussian Process emulation”. In: *preprint*.
- Taylor, Michael E. (2011). *Partial differential equations I basic theory*. Springer New York.
- Titsias, Michalis and Neil D Lawrence (2010). “Bayesian Gaussian process latent variable model”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 844–851.
- Touvron, Hugo et al. (Feb. 2023). *LLaMA: Open and Efficient Foundation Language Models*. arXiv:2302.13971 [cs]. DOI: 10.48550/arXiv.2302.13971. URL: <http://arxiv.org/abs/2302.13971> (visited on 01/16/2025).
- Vaart, A. W. van der and J. H. van Zanten (2008). “Reproducing kernel hilbert spaces of gaussian priors”. In: *Institute of Mathematical Statistics Collections*, pp. 200–222. DOI: 10.1214/0749217080000000156.
- Vaswani, Ashish et al. (2017). “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc.
- Virieux, Jean and Stephane Operto (2009). “An overview of full-waveform inversion in exploration geophysics”. In: *Geophysics* 74.6, WCC1–WCC26.
- Walsh, John B. (1981). “A stochastic model of neural response”. In: *Advances in Applied Probability* 13.2, pp. 231–281. DOI: 10.2307/1426607.
- Wang, Junyang et al. (July 2021). “Bayesian numerical methods for nonlinear partial differential equations”. In: *Statistics and Computing* 31.5, p. 55. ISSN: 1573-1375. DOI: 10.1007/s11222-021-10030-w. URL: <https://doi.org/10.1007/s11222-021-10030-w>.
- Wang, Sifan, Shyam Sankaran, et al. (2023). *An Expert’s Guide to Training Physics-informed Neural Networks*. arXiv: 2308.08468 [cs.LG].
- Wang, Sifan, Hanwen Wang, and Paris Perdikaris (Sept. 2021a). “Learning the solution operator of parametric partial differential equations with physics-informed DeepONets”. In: *Science Advances* 7.40. ISSN: ISSN 2375-2548. DOI: 10.1126/sciadv.abi8605. URL: <https://www.osti.gov/biblio/1904189>.
- (2021b). “Learning the solution operator of parametric partial differential equations with physics-informed deeponets”. In: *Science advances* 7.40, eabi8605.

- Wang, Sifan, Hanwen Wang, and Paris Perdikaris (2021c). “On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks”. In: *Computer Methods in Applied Mechanics and Engineering* 384, p. 113938. ISSN: 0045-7825. DOI: <https://doi.org/10.1016/j.cma.2021.113938>. URL: <https://www.sciencedirect.com/science/article/pii/S0045782521002759>.
- (2022). “Improved architectures and training algorithms for deep operator networks”. In: *Journal of Scientific Computing* 92.2, p. 35.
- Wang, Xin (2023). *2D 2-8 FD Acoustic Modeling Lab*. <https://csim.kaust.edu.sa/files/SeismicInversion/Chapter.FD/lab.FD2.8/lab.html>. Accessed: 2025-10-07.
- Wang, Yanghua (2015). “Frequencies of the Ricker wavelet”. In: *Geophysics* 80.2, A31–A37.
- Wang, Yongji et al. (2025). *Discovery of Unstable Singularities*. arXiv: 2509.14185 [math.AP]. URL: <https://arxiv.org/abs/2509.14185>.
- Weinan, E (2011). *Principles of multiscale modeling*. Cambridge University Press.
- Wendland, Holger (2004). *Scattered Data Approximation*. Cambridge, UK: Cambridge University Press. ISBN: 9780521830470. DOI: 10.1017/CB09780511617539.
- Wilson, Kenneth G (1971). “Renormalization group and critical phenomena. I. Renormalization group and the Kadanoff scaling picture”. In: *Physical review B* 4.9, p. 3174.
- Wu, Kaiwen and Jacob R. Gardner (2024). *Understanding Stochastic Natural Gradient Variational Inference*. arXiv: 2406.01870 [cs.LG]. URL: <https://arxiv.org/abs/2406.01870>.
- Wu, Zong-min and Robert Schaback (1993). “Local error estimates for radial basis function interpolation of scattered data”. In: *IMA journal of Numerical Analysis* 13.1, pp. 13–27.
- Xiu, Dongbin (2010). *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton University Press.
- Xiu, Dongbin and George Em Karniadakis (2002a). “The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations”. In: *SIAM Journal on Scientific Computing* 24.2, pp. 619–644. DOI: 10.1137/S1064827501387826. eprint: <https://doi.org/10.1137/S1064827501387826>. URL: <https://doi.org/10.1137/S1064827501387826>.
- (2002b). “The Wiener–Askey polynomial chaos for stochastic differential equations”. In: *SIAM journal on scientific computing* 24.2, pp. 619–644.
- Xiu, Dongbin and Jie Shen (2009). “Efficient stochastic Galerkin methods for random diffusion equations”. In: *Journal of Computational Physics* 228.2, pp. 266–281.

- Yan, Yubin (2005a). “Galerkin Finite Element Methods for Stochastic Parabolic Partial Differential Equations”. In: *SIAM Journal on Numerical Analysis* 43.4, pp. 1363–1384. DOI: 10.1137/040605278. eprint: <https://doi.org/10.1137/040605278>. URL: <https://doi.org/10.1137/040605278>.
- (2005b). “Galerkin Finite Element Methods for Stochastic Parabolic Partial Differential Equations”. In: *SIAM Journal on Numerical Analysis* 43.4, pp. 1363–1384. DOI: 10.1137/040605278. eprint: <https://doi.org/10.1137/040605278>. URL: <https://doi.org/10.1137/040605278>.
- Yang, Liu and Stanley J Osher (2024). “Pde generalization of in-context operator networks: A study on 1d scalar nonlinear conservation laws”. In: *Journal of Computational Physics* 519, p. 113379.
- Yang, Lu et al. (2023). “Learning dynamical systems from data: A simple cross-validation perspective, Part VI: Hausdorff-metric based Kernel Flows to learn Attractors and Invariant Sets”. In: *Physica D*.
- (2024). “Hausdorff metric based training of kernels to learn attractors with application to 133 chaotic dynamical systems”. In: *Physica D: Nonlinear Phenomena* 466, p. 134192. ISSN: 0167-2789. DOI: <https://doi.org/10.1016/j.physd.2024.134192>. URL: <https://www.sciencedirect.com/science/article/pii/S016727892400143X>.
- Yang, Qiuling, Alireza Sadeghi, and Gang Wang (2022). “Data-driven priors for robust PSSE via Gauss-Newton unrolled neural networks”. In: *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 12.1, pp. 172–181.
- Yang, Xianjin, Matthieu Darcy, et al. (2025). *Solving Functional PDEs with Gaussian Processes and Applications to Functional Renormalization Group Equations*. arXiv: 2512.20956 [cs.LG]. URL: <https://arxiv.org/abs/2512.20956>. MD developed the methodology, implemented early numerical examples, and wrote the manuscript.
- Yang, Xianjin and Houman Owhadi (2024). *A Mini-Batch Method for Solving Nonlinear PDEs with Gaussian Processes*. arXiv: 2306.00307 [math.NA]. URL: <https://arxiv.org/abs/2306.00307>.
- Ye, Zhanhong et al. (2024). “PDEformer: Towards a Foundation Model for One-Dimensional Partial Differential Equations”. In: *arXiv preprint arXiv:2402.12652*. URL: <https://arxiv.org/abs/2402.12652>.
- Yildiz, Cagatay et al. (2018). “LEARNING STOCHASTIC DIFFERENTIAL EQUATIONS WITH GAUSSIAN PROCESSES WITHOUT GRADIENT MATCHING”. In: *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6. DOI: 10.1109/MLSP.2018.8516991.
- Yokota, Takeru (June 2024a). “Physics-informed neural networks for solving functional renormalization group on a lattice”. In: *Phys. Rev. B* 109 (21), p. 214205. DOI: 10.1103/PhysRevB.109.214205. URL: <https://link.aps.org/doi/10.1103/PhysRevB.109.214205>.

- Yokota, Takeru (2024b). “Physics-informed neural networks for solving functional renormalization group on a lattice”. In: *Phys. Rev. B* 109, p. 214205. DOI: 10.1103/PhysRevB.109.214205.
- Yu, Guangting and Shiwei Lan (2025). “Solving and Learning Partial Differential Equations with Variational Q-Exponential Processes”. In: *The Thirty-ninth Annual Conference on Neural Information Processing Systems*. URL: <https://openreview.net/forum?id=WjhS0EpJH7>.
- Zammit-Mangion, Andrew et al. (2019). “Deep compositional spatial models”. In: *arXiv preprint arXiv:1906.02840*.
- Zeidler, Eberhard (1986). *Nonlinear functional analysis and its applications: I: Fixed-point Theorems*. Springer Science & Business Media.
- Zhang, Dongkun, Ling Guo, and George Em Karniadakis (2020). “Learning in Modal Space: Solving Time-Dependent Stochastic PDEs Using Physics-Informed Neural Networks”. In: *SIAM Journal on Scientific Computing* 42.2, A639–A665. DOI: 10.1137/19M1260141. eprint: <https://doi.org/10.1137/19M1260141>. URL: <https://doi.org/10.1137/19M1260141>.
- Zhang, Zhongqiang and George Em Karniadakis (2017). *Numerical Methods for Stochastic Partial Differential Equations with White Noise*. Applied Mathematical Sciences. Springer International Publishing. ISBN: 978-3-319-57510-0. DOI: 10.1007/978-3-319-57511-7.
- Zhu, Yinhao and Nicholas Zabaras (2018). “Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification”. In: *Journal of Computational Physics* 366, pp. 415–447.
- Zorbach, Niklas et al. (2025). “Lattice Monte Carlo meets the lattice functional renormalization group: A quantitative comparison”. In: *Physical Review D* 112.7, p. 076036.

Appendix A

CHAPTER 2

A.1 Auxiliary proofs

Convergence for the Continuum Poisson Problem

Consider the Poisson equation of the form (2.10). For fixed regularizer $\gamma \in \mathbb{R}^+$, we consider the continuum problem as in (2.5), denoting its solution by u^γ . We recall that u^* denotes the solution to (2.10). Letting $\mathcal{H}_k \equiv H^q$ for some $q > -s + 2$, in the following we show for which $\alpha \in \mathbb{R}$, the following two conditions hold:

$$\|u^\gamma\|_{H^\alpha} < \infty \quad \text{and} \quad \lim_{\gamma \rightarrow 0} \|u^\gamma - u_0\|_{H^\alpha} = 0.$$

To show this, we first find an expression for solution u^γ in terms of the eigenvalues $\{\lambda_j\}_{j \in \mathbb{N}}$ of the negative Laplacian operator. We recall that any $u \in H_0^q$ can be expressed as

$$u = \sum_{j=1}^{\infty} u_j \varphi_j, \quad \text{with } u_j = \langle u, \varphi_j \rangle,$$

where $\{\varphi_j\}_{j \in \mathbb{N}}$ is the set L^2 orthonormal generalized eigenvectors of the negative Laplacian operator, subject to Dirichlet boundary conditions. Similarly we recall that it is possible to express $\xi \in H^{-s}$ from (2.10) as

$$\xi = \sum_{j=1}^{\infty} \xi_j \varphi_j, \quad \text{with } \xi_j = [\xi, \varphi_j],$$

where $[\cdot, \cdot]$ is the H^s duality product. Given these expressions, we have by the definition of the H^{-s} norm eq. (2.12) (which can formally be taken as the L^2 integral inner product).

$$\|-\Delta u - \xi\|_{H^{-s}}^2 = \sum_{j=1}^{\infty} (\lambda_j u_j - \xi_j)^2 \lambda_j^{-s}, \quad \|u\|_{H_0^q}^2 = \sum_{j=1}^{\infty} u_j^2 \lambda_j^q,$$

from which we can reformulate the optimization problem as

$$(u^\gamma)_j = \arg \min_{v \in \mathbb{R}} (\lambda_j v - \xi_j)^2 \lambda_j^{-s} + \gamma v^2 \lambda_j^q \quad \text{for each } j \in \mathbb{N}. \quad (\text{A.1})$$

It is readily found that the quadratic optimization problem (A.1) yields

$$u^\gamma = \sum_{j=1}^{\infty} \frac{\xi_j}{\lambda_j (1 + \gamma \lambda_j^{q+s-2})} \varphi_j.$$

Proposition 13. *For any $\alpha \leq 2 - s$, it holds that*

$$\|u^\gamma\|_{H^\alpha} < \infty \quad \text{and} \quad \lim_{\gamma \rightarrow 0} \|u^\gamma - u_0\|_{H^\alpha} = 0. \quad (\text{A.2})$$

Proof. We divide the proof into two parts. Indeed we first consider for which $\alpha \in \mathbb{R}$ it holds that $\|u^\gamma\|_{H^\alpha} < \infty$. Using the growth rate of the eigenvalues of the negative Laplacian operator $\lambda_j \sim j^{2/d}$, we have that

$$\|u^\gamma\|_{H^\alpha}^2 = \sum_{j=1}^{\infty} \left(\frac{\xi_j}{\lambda_j(1 + \gamma\lambda_j^{q+s-2})} \right)^2 \lambda_j^\alpha \quad (\text{A.3a})$$

$$\lesssim \sum_{j=1}^{\infty} \xi_j^2 \lambda_j^{-s} \frac{j^{2(s+\alpha-2)/d}}{(1 + \gamma j^{2(q+s-2)/d})^2} \quad (\text{A.3b})$$

$$\lesssim \gamma^{-2} \sum_{j=1}^{\infty} \xi_j^2 \lambda_j^{-s} \cdot j^{2(\alpha-2q-s+2)/d}, \quad (\text{A.3c})$$

since clearly,

$$(1 + \gamma j^{2(\alpha-2q-s+2)/d})^{-2} \leq (\gamma j^{2(\alpha-2q-s+2)/d})^{-2}.$$

Therefore, with the knowledge that $\|\xi\|_{H^{-s}}^2 < \infty$, (A.3c) gives that $\|u^\gamma\|_{H^\alpha} < \infty$ if $\alpha \leq 2q + s - 2$.

We now turn our attention to $\|u^\gamma - u_0\|_{H^\alpha}$. Again, using the growth rate of the eigenvalues of the negative Laplacian operator $\lambda_j \sim j^{2/d}$, we have that

$$\begin{aligned} \|u^\gamma - u_0\|_{H^\alpha}^2 &= \sum_{j=1}^{\infty} \left(\frac{\xi_j}{\lambda_j(1 + \gamma\lambda_j^{q+s-2})} - \frac{\xi_j}{\lambda_j} \right)^2 \lambda_j^\alpha \\ &\lesssim \sum_{j=1}^{\infty} \frac{\gamma^2 \xi_j^2 j^{2(2q+2s-6+\alpha)/d}}{(1 + \gamma j^{2(q+s-2)/d})^2} \\ &\lesssim \sum_{j=1}^{\infty} \gamma^2 \xi_j^2 \lambda_j^{-s} \frac{j^{2(2q+3s-6+\alpha)/d}}{(1 + \gamma j^{2(q+s-2)/d})^2} \\ &\lesssim \sum_{j=1}^{\infty} \gamma^2 \xi_j^2 \lambda_j^{-s} \frac{j^\kappa}{(1 + \gamma j^\tau)^2}, \end{aligned}$$

where we have defined

$$\kappa := \frac{2}{d}(2q + 3s - 6 + \alpha) \quad \text{and} \quad \tau := \frac{2}{d}(q + s - 2).$$

For ease of exposition we define

$$B_j^\gamma := \frac{\gamma^2 j^\kappa}{(1 + \gamma j^\tau)^2},$$

so that

$$\begin{aligned}\|u^\gamma - u_0\|_{H^\alpha}^2 &= \sum_{j=1}^{\infty} \xi_j^2 \lambda_j^{-s} B_j^\gamma \\ &= \sum_{j=1}^M \xi_j^2 \lambda_j^{-s} B_j^\gamma + \sum_{j>M}^{\infty} \xi_j^2 \lambda_j^{-s} B_j^\gamma,\end{aligned}$$

for any $M \in \mathbb{N}$. Clearly $B_j^\gamma \rightarrow 0$ as $\gamma \rightarrow 0$, therefore

$$\sum_{j=1}^M \xi_j^2 \lambda_j^{-s} B_j^\gamma \rightarrow 0 \quad \text{as } \gamma \rightarrow 0.$$

On the other hand, noting that

$$(1 + \gamma j^\tau)^{-2} \leq (\gamma j^\tau)^{-2},$$

we have that $B_j^\gamma \leq j^{\kappa-2\tau}$, and hence

$$\sum_{j>M}^{\infty} \xi_j^2 \lambda_j^{-s} B_j^\gamma \leq \sum_{j>M}^{\infty} \xi_j^2 \lambda_j^{-s} j^{\kappa-2\tau}.$$

By the fact that $\|\xi\|_{H^{-s}}^2 < \infty$, it holds that $\sum_{j>M}^{\infty} \xi_j^2 \lambda_j^{-s} B_j^\gamma < \infty$ if $\kappa - 2\tau \leq 0$. Indeed, for small enough γ , the sequence of partial sums converges uniformly, thus allowing the interchangeability of limits. Furthermore, if this holds, then

$$\sum_{j>M}^{\infty} \xi_j^2 \lambda_j^{-s} B_j^\gamma \rightarrow 0 \quad \text{as } \gamma \rightarrow 0,$$

since $B_j^\gamma \rightarrow 0$ as $\gamma \rightarrow 0$. Since $\kappa - 2\tau \leq 0$ if and only if $\alpha \leq 2 - s$, we conclude that for $\alpha \leq 2 - s$ it holds that

$$\lim_{\gamma \rightarrow 0} \|u^\gamma - u_0\|_{H^\alpha} = 0.$$

To finish the proof, we thus note that for $\alpha \leq \min\{2q + s - 2, 2 - s\}$, (A.2) holds, but since $q > 2 - s$, we have that $2q + s - 2 > 2 - s$, hence the conclusion. $\square$

Numerical method

Proof of proposition 1

Proof. We begin by noting the following equivalent characterizations:

$$\begin{aligned}|f|_{\Phi^N} &= \sup_{u \in \Phi^N} \frac{\int f u}{\|u\|_{H_0^s}} \\ &= \sup_{u \in \Phi^N, \|u\|_{H_0^s} = 1} \int f u.\end{aligned}$$

Moreover, observe that for $u \in \Phi^N$

$$\int f u = \sum_{i=1}^N c_i \int f \varphi_i = c^\top [f, \varphi],$$

for some $c \in \mathbb{R}^N$, and for $u \in \Phi^N$

$$\|u\|_{H_0^s} = \sqrt{\sum_{i=1}^N \sum_{j=1}^N c_i c_j \int \varphi_i (-\Delta)^s \varphi_j} = \sqrt{c^\top A c}.$$

Therefore we can write

$$\begin{aligned} |f|_{\Phi^N} &= \sup_{c \in \mathbb{R}^N} c^\top [f, \varphi] \\ \text{s.t. } &c^\top A c = 1, \end{aligned}$$

whose solution is

$$|f|_{\Phi^N} = \sqrt{[f, \varphi]^\top A^{-1} [f, \varphi]}.$$

Note that the matrix A is invertible because the φ_i are linearly independent. $\square$

Proof of proposition 2.

Proof. Under Assumption (D1) for all $\varepsilon > 0$, there is a $N = N(\varepsilon, u)$ such that

$$\|v^N - u\|_{H_0^s(\Omega)} < \varepsilon,$$

for some $v^N \in \Phi^N$. Now, by the definition of the dual norm, for all $\varepsilon > 0$, there is a $u \in H_0^s(\Omega)$ with $\|u\|_{H_0^s(\Omega)} = 1$ such that

$$\left| \int f u - \|f\|_{H_0^s(\Omega)} \right| < \frac{\varepsilon}{2}.$$

By the density of Φ^N , there is a N and a v^N such that

$$\|u - v^N\|_{H_0^s(\Omega)} \leq \frac{\varepsilon}{2 \|f\|_{H^{-s}(\Omega)}}.$$

Without loss of generality, we may assume that v^N has a unit norm since u has a unit norm. Therefore

$$\begin{aligned} \left| \int f u - \int f v^N \right| &\leq \|f\|_{H^{-s}(\Omega)} \|u - v^N\|_{H_0^s(\Omega)} \\ &\leq \frac{\varepsilon}{2}. \end{aligned}$$

Hence for all $\varepsilon > 0$, there is a $N = N(\varepsilon)$ and a corresponding $v^N \in \Phi^N$ with norm 1 such that

$$\left| \int f v^N - \|f\|_{H_0^s(\Omega)} \right| \leq \left| \int f u - \|f\|_{H_0^s(\Omega)} \right| + \left| \int f u - \int f v^N \right| < \varepsilon.$$

The assertion then follows. $\square$

Proof of theorem 3

Proof. Let $S := \text{span}\{K(\cdot, \chi_1), \dots, K(\cdot, \chi_N), K(\cdot, x_1), \dots, K(\cdot, x_M)\}$. Clearly, it holds that $\mathcal{H}_K = S \oplus S^\perp$. Therefore, we have that

$$\widehat{u} = \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j) + s,$$

where $s \in S^\perp$. By the reproducing kernel property, for any $1 \leq k \leq N$ we have that

$$\begin{aligned} \widehat{u}(\chi_k) &= \left\langle \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j), K(\cdot, \chi_k) \right\rangle + \langle s, K(\cdot, \chi_k) \rangle \\ &= \left\langle \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j), K(\cdot, \chi_k) \right\rangle, \end{aligned}$$

since $s \in S^\perp$. Similarly, for any $1 \leq k \leq M$, we have that

$$\begin{aligned} \widehat{u}(x_k) &= \left\langle \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j), K(\cdot, x_k) \right\rangle + \langle s, K(\cdot, x_k) \rangle \\ &= \left\langle \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j), K(\cdot, x_k) \right\rangle. \end{aligned}$$

Now, since s is orthogonal to $\sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j)$, we have that

$$\|\widehat{u}\|_{\mathcal{H}_K}^2 = \left\| \sum_{i=1}^N \alpha_i K(\cdot, \chi_i) + \sum_{j=1}^M \beta_j K(\cdot, x_j) \right\|_{\mathcal{H}_K}^2 + \|s\|_{\mathcal{H}_K}^2.$$

Hence, while s does not affect the first term in (2.22a), an s of smaller $\mathcal{H}_K$ norm decreases the second term. Therefore, any solution $\widehat{u}$ must have $s = 0$. $\square$

Proof of Proposition 3.

Proposition 14. *Using theorem 3 we reformulate the problem in eq. (2.20) as a quadratic minimization problem with linear inequality constraint. Each iteration n necessitates solution of*

$$\begin{aligned} \arg \inf_{c \in \mathbb{R}^{N+M}} & \frac{1}{2} \left[\sum_{i=1}^{M+N} c_i \mathcal{P}'(u) K(\cdot, \varphi) - r_n, \varphi \right]^\top A^{-1} \left[\sum_{i=1}^{M+N} c_i \mathcal{P}'(u) K(\cdot, \varphi_i) - r_n, \varphi \right] + \frac{\gamma}{2} \left\| \sum_{i=1}^{M+N} c_i K(\cdot, \varphi_i) \right\|_{\mathcal{H}_K}^2 \\ \text{s.t. } & \sum_{i=1}^{M+N} c_i K(x_j, \varphi_i) = g(x_j) \quad \text{for } j = 1, \dots, M. \end{aligned}$$

We may transform the above into a matrix expression:

$$\begin{aligned} \arg \inf_{c \in \mathbb{R}^{N+M}} & \frac{1}{2} \left(K(\chi, \varphi) c - [r_n, \varphi] \right)^\top A^{-1} \left(K(\chi, \varphi) c - [r_n, \varphi] \right) + \frac{\gamma}{2} c^\top K(\varphi, \varphi) c \\ \text{s.t. } & K(X, \varphi) c = g(X) \quad \text{for } j = 1, \dots, M. \end{aligned}$$

which may be formulated a standard convex quadratic minimization problem with equality constraints:

$$\begin{aligned} \arg \min_{c \in \mathbb{R}^{N+M}} & \frac{1}{2} c^\top \left(K(\varphi, \chi) A^{-1} K(\chi, \varphi) + \gamma K(\varphi, \varphi) \right) c - \left(K(\varphi, \chi) A^{-1} [g_n, \varphi] \right)^\top c \\ & + \frac{1}{2} [g_n, \varphi]^\top A^{-1} [g_n, \varphi] \\ \text{s.t. } & K(X, \varphi) c = g(X) \quad \text{for } j = 1, \dots, M \end{aligned}$$

The KKT system gives the explicit solution to this problem Boyd and Vandenberghe, 2004, p. 522:

$$\begin{bmatrix} K(\chi, \varphi)^\top A^{-1} K(\chi, \varphi) + \gamma K(\varphi, \varphi) & K(X, \varphi) \\ K(X, \varphi) & 0 \end{bmatrix} \begin{bmatrix} c \\ v \end{bmatrix} = \begin{bmatrix} K(\chi, \varphi)^\top A^{-1} [r_n, \varphi] \\ g(X), \end{bmatrix}$$

with $v \in \mathbb{R}^N$ are the Lagrange multipliers. Note that this system has a unique solution when the KKT matrix is non-singular, which is the case when the φ_i are linearly independent and the collocation points x_j are distinct.

New Concept of Solution

We now present a proof of the existence of a minimizer to the generalized problem (2.31) under suitable assumptions.

Assumption 13. *We assume the following compact embeddings:*

$$\begin{aligned} \mathcal{H}_K & \subset H^{s+i} \Subset \mathcal{H}_{\Gamma_i} \quad \text{for } i = 0, 1, 2. \\ \mathcal{H}_{K_i} & \Subset \mathcal{H}_{\Gamma_i} \quad \text{for } i = 0, 1, 2. \end{aligned}$$

Moreover, we assume that the nonlinear differential operator $\mathcal{P} : \mathcal{H}_{\Gamma_0} \times \mathcal{H}_{\Gamma_1} \times \mathcal{H}_{\Gamma_2} \rightarrow \mathcal{H}_{\Gamma_3}$ is continuous.

Assumption 14 (Continuity of $\mathcal{P}$). We assume that the nonlinear differential operator $\mathcal{P} : \mathcal{H}_{\Gamma_0} \times \mathcal{H}_{\Gamma_1} \times \mathcal{H}_{\Gamma_2} \rightarrow \mathcal{H}_{\Gamma_3}$ is continuous.

The next theorem is the equivalent of theorem 1 for the generalized problem.

Theorem 15. Let $\mathcal{H} := \mathcal{H}_K \times \mathcal{H}_{K_0} \times \mathcal{H}_{K_1} \times \mathcal{H}_{K_2}$ for some appropriate choices of RKHS $\mathcal{H}_K, \mathcal{H}_{K_i}$ satisfying Assumption 13. Let m be the optimal value of problem (2.32):

$$m := \inf_{u,v} L(u, v; \gamma).$$

Then $0 \leq m < \infty$ and there is a $(u^*, v^*) \in \mathcal{H}$ such that $m = L(u^*, v^*)$. Moreover, if (u^n, v^n) is a minimizing sequence, then there is a subsequence which converges strongly to (u^*, v^*) in $\mathcal{H}$, i.e.

$$\begin{aligned} u^n &\rightarrow u^* \quad \text{in } \mathcal{H}_K \\ v_i^n &\rightarrow v_i^* \quad \text{in } \mathcal{H}_{K_i} \quad \text{for } i = 0, 1, 2. \end{aligned}$$

Proof. To prove existence of a minimizer, we first show the boundedness of the minimizing sequence and then proceed by showing weak lower semicontinuity of the objective, thus proving the claim.

Step 1: existence of weak limit. Indeed, we first observe that

$$0 \leq L(u, v; \gamma) < \infty \quad \text{for all } (u, v) \in \mathcal{H},$$

and therefore $0 \leq m < \infty$. For ease of presentation, we will now omit the dependence on γ . Consider some minimizing sequence (u^n, v^n) :

$$L(u^n, v^n) \rightarrow m \quad \text{as } n \rightarrow \infty.$$

Then observe that by the definition of L , this implies that the minimizing sequence is bounded

$$\begin{aligned} \|u^n\|_{\mathcal{H}_K} &\leq C \\ \|v_i^n\|_{\mathcal{H}_{K_i}} &\leq C \quad \text{for } i = 0, 1, 2. \end{aligned}$$

By the Banach–Alaoglu theorem Alaoglu, 1940, since each $\mathcal{H}_{K_i}$ and $\mathcal{H}_K$ are reflexive, there is a subsequence (which we denote with the same index) such that $(u^n, v^n) \rightharpoonup (u^\infty, v^\infty)$ in $\mathcal{H}$.

Step 2: weak lower semicontinuity of the objective. By the compact embedding assumption 13, we deduce that

$$\begin{aligned} D^i u^n &\rightarrow D^i u^\infty \quad \text{in } \mathcal{H}_{\Gamma_i} \quad \text{for } i = 0, 1, 2 \\ v_i^n &\rightarrow v_i^\infty \quad \text{in } \mathcal{H}_{\Gamma_i} \quad \text{for } i = 0, 1, 2. \end{aligned}$$

Therefore

$$\|v_i^n - D^i u^n\|_{\Gamma_i}^2 \rightarrow \|v_i^\infty - D^i u^\infty\|_{\mathcal{H}_{\Gamma_i}}^2 \quad \text{for } i = 0, 1, 2.$$

Moreover by the continuity of $\mathcal{P}$ from $\mathcal{H}_{\Gamma_0} \times \mathcal{H}_{\Gamma_1} \times \mathcal{H}_{\Gamma_2}$ into $\mathcal{H}_{\Gamma_3}$, we obtain

$$\|\mathcal{P}(v_0^n, v_1^n, v_2^n) - f\|_{\mathcal{H}_{\Gamma_3}}^2 \rightarrow \|\mathcal{P}(v_0^\infty, v_1^\infty, v_2^\infty) - f\|_{\mathcal{H}_{\Gamma_3}}^2.$$

Hence the first part of the term of L is weakly continuous on $\mathcal{H}$. Since norms are lower-semicontinuous Brezis, 2010, p. 61, it follows that $L : \mathcal{H} \rightarrow \mathbb{R}$ is lower-semicontinuous. Therefore

$$(u^n, v^n) \rightharpoonup (u^\infty, v^\infty) \Rightarrow \liminf_{n \rightarrow \infty} L(u^n, v^n) \geq L(u^\infty, v^\infty),$$

and consequently

$$m = L(u^\infty, v^\infty, \mathbf{r}^\infty).$$

Step 3: strong convergence of the minimizing sequence. We now turn our attention to showing the existence of a subsequence of (u^n, v^n) that converges strongly to a minimizer. For ease of presentation, let

$$\begin{aligned} \Phi(u, v) &:= \|\mathcal{P}(v_0, v_1, v_2) - f\|_{\mathcal{H}_{\Gamma_3}}^2 + \sum_{i=0}^2 \|v_i - D^i u\|_{\mathcal{H}_{\Gamma_i}}^2 \\ \|(u, v)\|^2 &:= \|u\|_{\mathcal{H}_k}^2 + \sum_{i=0}^2 \|v_i\|_{\mathcal{H}_{K_i}}^2, \end{aligned}$$

so that

$$L(u, v) = \Phi(u, v) + \gamma \|(u, v)\|^2.$$

Using the polarization equality

$$\begin{aligned} \frac{\gamma}{4} \|(u, v)^l - (u, v)^k\|^2 &= \gamma \left(\frac{1}{2} \|(u, v)^l\|^2 + \frac{1}{2} \|(u, v)^k\|^2 - \left\| \frac{1}{2} ((u, v)^l + (u, v)^k) \right\|^2 \right) \\ &= \frac{1}{2} L((u, v)^k) + \frac{1}{2} L((u, v)^l) - L \left(\frac{1}{2} ((u, v)^l + (u, v)^k) \right) \\ &\quad - \frac{1}{2} \Phi((u, v)^l) - \frac{1}{2} \Phi((u, v)^k) + \Phi \left(\frac{1}{2} ((u, v)^l + (u, v)^k) \right). \end{aligned}$$

Letting $l, k \geq N_1(\delta)$ so that $L(u^k, v^k) \leq m + \delta$ and $L(u^l, v^l) \leq m + \delta$, we obtain

$$\begin{aligned} \frac{\gamma}{4} \|(u, v)^l - (u, v)^k\|^2 &\leq m + \delta - m - \frac{1}{2}\Phi((u, v)^l) - \frac{1}{2}\Phi((u, v)^k) + \Phi\left(\frac{1}{2}((u, v)^l + (u, v)^k)\right) \\ &= \delta - \frac{1}{2}\Phi((u, v)^l) - \frac{1}{2}\Phi((u, v)^k) + \Phi\left(\frac{1}{2}((u, v)^l + (u, v)^k)\right). \end{aligned}$$

Since $(u, v)^l$, $(u, v)^k$ and $\frac{1}{2}((u, v)^l + (u, v)^k)$ converge to $(u, v)^\infty$ in each $\mathcal{H}_{\Gamma_i}$, and since Φ is continuous on $\mathcal{H}_{\Gamma_0} \times \cdots \times \mathcal{H}_{\Gamma_3}$, we deduce that there is a $N_2(\delta)$ such that for $k, l \geq N_2(\delta)$:

$$\frac{\gamma}{4} \|(u, v)^l - (u, v)^k\|^2 \leq \delta + \delta = 2\delta.$$

Therefore the sequence $(u, v)^n$ is Cauchy in $\mathcal{H}_K \times \mathcal{H}_{K_0} \times \mathcal{H}_{K_1} \times \mathcal{H}_{K_3}$ and as a consequence converges strongly in those spaces:

$$\begin{aligned} u^n &\rightarrow u^\infty \quad \text{in } \mathcal{H}_K \\ v_i^n &\rightarrow v_i^\infty \quad \text{in } \mathcal{H}_{K_i} \quad \text{for } i = 0, 1, 2. \end{aligned}$$

□

A.2 Neural Network Architecture

Problem	RFF m	RFF Std σ	N° Layers	Neurons/layer	Total Parameters
1D Elliptic Linear	24	5.0	4	16	1,073
2D Semilinear Elliptic	32	5.0	4	64	8,385

Table A.1: Architecture for the neural networks used for the NeS-PINN approach.

table A.1 describes the architecture of our deep neural networks with N layers.

Random Fourier Features (RFF) As described in Sifan Wang, H. Wang, and Paris Perdikaris, 2021c, the random Fourier features add a non-trainable embedding at the start of the network:

$$\gamma(x) = \begin{bmatrix} \cos(Bx) \\ \sin(Bx) \end{bmatrix}, \quad (\text{A.15})$$

where $B \in \mathbb{R}^{m \times d}$ has entries sampled *i.i.d.* from a normal distribution $\mathcal{N}(0, \sigma^2)$. This embedding mitigates the spectral bias of neural networks.

A.3 Gaussian Measures and White Noise

We now briefly introduce the concepts of Gaussian measures, Gaussian spatial noise, and Gaussian space-time white noise. Our exposition will be informal and far from complete. For a more detailed explanation, see Gabriel J. Lord, Powell, and Shardlow, 2014.

Gaussian Measures

We introduce Gaussian measures on $L^2(\Omega)$.

Definition 1 (L^2 valued Gaussian measure). *An $L^2(\Omega)$ -valued random variable ξ is Gaussian if $\langle X, \phi \rangle$ is a Gaussian random variable for all $\phi \in L^2(\Omega)$.*

To each Gaussian random variable on $L^2(\Omega)$ one can associate a covariance operator.

Definition 2. *A linear operator $Q : L^2(\Omega) \rightarrow L^2(\Omega)$ is the covariance of an L^2 -valued random variable ξ if*

$$\langle Q\phi, \psi \rangle_{L^2} = \text{Cov}(\langle \xi, \phi \rangle_{L^2}, \langle \xi, \psi \rangle_{L^2}).$$

The above definitions imply that for any $\phi \in L^2(\Omega)$

$$\langle \xi, \phi \rangle_{L^2} \sim \mathcal{N}(\langle \mu, \phi \rangle_{L^2}, \langle Q\phi, \phi \rangle_{L^2}).$$

We will always consider centered Gaussian variables $\mathbb{E}[\xi] = 0$ so that $\langle \mu, \phi \rangle = 0$ for all $\phi \in L^2(\Omega)$. One can further show that the covariance operator Q is of trace class.

Proposition 15 (Corollary 4.41 Gabriel J. Lord, Powell, and Shardlow, 2014). *Let ξ be a $L^2(\Omega)$ -valued Gaussian random variable, then its covariance operator Q is of trace-class.*

This allows for the construction of ξ via the spectral decomposition of Q .

Proposition 16. *Let ξ be a centered $L^2(\Omega)$ -valued Gaussian random variable with covariance operator Q . Then ξ admits the representation*

$$\xi = \sum_{i=1}^{\infty} \sqrt{\lambda_i} Z_i e_i \quad Z_i \sim \mathcal{N}(0, 1) \text{ i.i.d.},$$

where $\{\lambda_i, e_i\}_{i=1}^{\infty}$ are the eigenvalues and eigenvectors of Q .

Spatial White Noise

We now briefly introduce the concept of spatial white noise. Formally, this extends the above definitions to the operator $Q = I$, which is not a trace-class operator. First, we consider the formal power series

$$\xi = \sum_{i=1}^{\infty} Z_i e_i \quad Z_i \sim \mathcal{N}(0, 1) \text{ i.i.d.}, \quad (\text{A.16})$$

where $\{e_i\}_{i=1}^{\infty}$ is any orthonormal basis of $L^2(\Omega)$. For each $\varphi \in L^2(\Omega)$, we may formally compute

$$\langle \xi, \varphi \rangle_{L^2} = \sum_{j=1}^{\infty} Z_j \langle e_j, \varphi \rangle_{L^2}.$$

Ignoring issues of convergence, we can view the above as normal random variable with mean and covariance

$$\begin{aligned} \mathbb{E}[\langle \xi, \varphi \rangle_{L^2}] &= \sum_{j=1}^{\infty} \mathbb{E}[Z_j] \langle e_j, \varphi \rangle_{L^2} = 0 \\ \mathbb{E}[\langle \xi, \varphi \rangle_{L^2} \langle \xi, \psi \rangle_{L^2}] &= \sum_{j=1}^{\infty} \sum_{i=1}^{\infty} \mathbb{E}[Z_j Z_i] \langle e_j, \varphi \rangle_{L^2} \langle e_i, \psi \rangle_{L^2} = \langle \varphi, \psi \rangle_{L^2}. \end{aligned}$$

From this, we may instead define ξ to be a stochastic process acting on elements of $L^2(\Omega)$ such that

$$[\xi, \varphi] \sim \mathcal{N}(0, \|\varphi\|_{L^2}^2).$$

The series (A.16) is not convergent in $L^2(\Omega)$. This can be fixed by considering the inclusion

$$i : L^2(\Omega) \rightarrow H^{-1}(\Omega),$$

which is Hilbert-Schmidt¹. The series (A.16) does converge in this space and the associated process is not a full Gaussian measure but instead a cylindrical measure on $L^2(\Omega)$. We refer to this process as spatial white noise which formally acts on $\varphi \in L^2(\Omega)$ by

$$[\xi, \varphi] \sim \mathcal{N}(0, \|\varphi\|_{L^2}^2).$$

¹One can make choices other than $H^{-1}(\Omega)$ and any space V such that the inclusion $i : L^2(\Omega) \rightarrow V$ is Hilbert-Schmidt is appropriate.

Space-time White Noise: the Cylindrical Wiener Process

We now define the cylindrical Wiener process. Let $(\mathcal{S}, \mathcal{F}, \mathcal{F}_t, \mathbb{P})$ be a filtered probability space.

Definition 3. *The cylindrical Wiener-process on $L^2(\Omega)$ if*

$$\xi(t) = \sum_{j=1}^{\infty} \beta_j(t) e_j, \quad (\text{A.17})$$

for any orthonormal basis $\{e_j\}_{j=1}^{\infty}$ and $\beta_j(t)$ are i.i.d $\mathcal{F}_t$ -Brownian motions.

One can show that

$$\xi(t) - \xi(s) \sim \mathcal{N}(0, (t-s)I) \quad \text{for all } 0 \leq s \leq t, \quad (\text{A.18})$$

so that the increments of the cylindrical Wiener process are spatial white-noise processes on $L^2(\Omega)$ (with a variance defined by time-step of the increments). Moreover, since the (formal) time-derivative of Brownian motion is white noise in time, this justifies the choice of $\dot{\xi}(t)$ as space-time white noise. Equivalently, one can see $\xi(t)$ as space-time white noise integrated in time.

Sampling From a Cylindrical Wiener Process

In order to solve SPDEs of the form eq. (2.29), we consider integration in time through an implicit or semi-implicit Euler-scheme (see Section 2.4). To this end, we need to sample increments of the cylindrical Wiener process $\xi(t) - \xi(s)$. The first option is to directly use the representation eq. (A.17) to obtain a truncated expansion

$$\Delta \xi^N := \sum_{j=1}^N \Delta \beta_j e_j, \quad (\text{A.19})$$

where $\Delta \beta_j \sim \Delta W_j = \mathcal{N}(0, \Delta t)$ are i.i.d. increments of Brownian motion. In the case where $L^2([0, 1]) \subseteq H^{-1}([0, 1])$ we would select $e_j(x) = \sin(2\pi jx)$. Other boundary conditions (such as Neumann and periodic boundary conditions) can also be considered by selecting the full Fourier basis $e^{i2\pi jx}$.

Another option is to use eq. (A.18) to show that for any collection $\Phi = \{\varphi_i\}_{i=1}^N \subset L^2(\Omega)$, the vector

$$[\Delta \xi, \varphi] \sim \mathcal{N}(0, \Delta t A), \quad (\text{A.20})$$

where $A_{ij} = \langle \varphi_i, \varphi_j \rangle_{L^2}$ is the stiffness matrix. If the φ_i are selected to be approximations to the identity around a point x , then one can informally view $[\Delta \xi, \varphi]$ as

approximating the value of $\Delta\xi$ at the point x (although we stress that action of $\Delta\xi$ against δ_x are not well defined as it does not have the required regularity). Likewise, while the truncated series eq. (A.19) has well-defined pointwise values, the limiting series eq. (A.17) does not.

An illustration of the two sampling methods based on expansion in (A.19) and the projection in (A.20) is shown in fig. A.1.

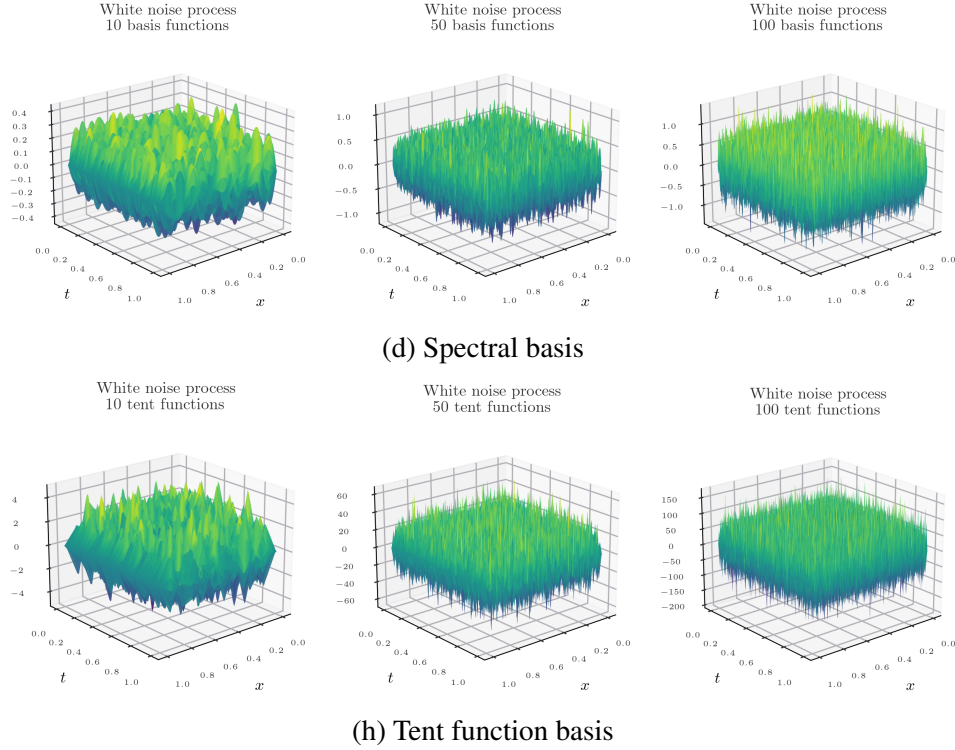


Figure A.1: Space-time white noise generated by two methods.

Appendix B

CHAPTER 3

B.1 Sparse variational inference derivation

The ELBO eq. (3.14) takes the form

$$J_{\text{ELBO}}(q) = \mathbb{E}_{\mathbf{z}_c \sim q} [-\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}_c))] + \text{KL}(q||p) \quad (\text{B.1})$$

The data is given by

$$\mathbb{E}_{\mathbf{z}_c \sim q} [-\log(\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}_c))] = \mathbb{E}_{\mathbf{z}_c \sim q} \left[\log \left(\int -\ell(f(\mathbf{x}_{\text{int}}), g(\mathbf{x}_{\text{bnd}})|\mathbf{z}) p(\mathbf{z}|\mathbf{z}_c) d\mathbf{z} \right) \right] \quad (\text{B.2})$$

$$= \mathbb{E}_{\mathbf{z}_c \sim q} \left[\log \left(\mathbb{E}_{\mathbf{z}|\mathbf{z}_c} \left[\frac{1}{2\sigma^2} (\|F(\mathbf{z}) - f(\mathbf{x}_{\text{int}})\|^2 + \|B(\mathbf{z}) - g(\mathbf{x}_{\text{bnd}})\|^2) \right] \right) \right] \quad (\text{B.3})$$

The inner expectation cannot be separated in the points $\mathbf{z}$ due to the logarithm and thus limits our ability to mini-batch.

B.2 Gaussian VI proofs

We will need the following useful inequality for Gaussian random variables.

Lemma 5 (Quadratic equality for Gaussian random variables). *Let $q = \mathcal{N}(m, S)$, then*

$$\frac{1}{2} \mathbb{E}_{u \sim q} [\|y - Au\|^2] = \frac{1}{2} \langle y - Am, y - Am \rangle + \frac{1}{2} \text{tr}(ASA^\top). \quad (\text{B.4})$$

Proof.

$$\frac{1}{2} \mathbb{E}_{u \sim q} [\|y - Au\|^2] = \mathbb{E}_{u \sim q} \left[\frac{1}{2} \langle y, y \rangle - \langle y, Au \rangle + \frac{1}{2} \langle Au, Au \rangle \right] \quad (\text{B.5})$$

$$= \frac{1}{2} \langle y, y \rangle - \langle y, Am \rangle + \frac{1}{2} \mathbb{E}_{u \sim q} \langle Au, Au \rangle \quad (\text{B.6})$$

$$= \frac{1}{2} \langle y, y \rangle - \langle y, Am \rangle + \frac{1}{2} \langle Am, Am \rangle + \frac{1}{2} \text{tr}(ASA^\top) \quad (\text{B.7})$$

$$= \frac{1}{2} \langle y - Am, y - Am \rangle + \frac{1}{2} \text{tr}(ASA^\top). \quad (\text{B.8})$$

□

SVGP

Proposition 17 (Marginal distribution u). *If the conditional distribution $u|\mathbf{z}_c$ is given by*

$$u|\mathbf{z}_c \sim \mathcal{GP}(A\mathbf{z}_c, \tilde{K}) \quad (\text{B.9a})$$

$$A\mathbf{z}_c = K(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} \mathbf{z}_c \quad (\text{B.9b})$$

$$\tilde{K} = K(\cdot, \cdot) - K(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} K(\phi_c, \cdot), \quad (\text{B.9c})$$

where $\mathbf{z}_c \sim \mathcal{N}(m, S)$, then u is distributed according to

$$u \sim \mathcal{GP}(\bar{u}_m, \tilde{Q}_S) \quad (\text{B.10})$$

$$\bar{u}_m = K(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} m \quad (\text{B.11})$$

$$\tilde{Q}_S = K(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} S K(\phi_c, \phi_c)^{-1} K(\phi_c, \cdot) + \tilde{K}. \quad (\text{B.12})$$

Proof. This follows from standard marginalization formulas (see for example Bishop, 2006). $\square$

Proof of eq. (3.35).

Proof. We apply lemma 5:

$$\mathbb{E}_{\mathbf{z}_c \sim q, u | \mathbf{z}_c} \left[\|\mathcal{F}'(\bar{u})u - r_n\|_{L^2(\Omega)}^2 \right] = \|r_n - \mathcal{F}'(\bar{u})\mathcal{K}(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} m\|_{L^2(\Omega)}^2 \quad (\text{B.13})$$

$$+ \frac{1}{2} \int_{\Omega} \mathcal{F}'(\bar{u}) \tilde{Q} \mathcal{F}'(\bar{u})^\top \quad (\text{B.14})$$

$$\mathbb{E}_{\mathbf{z}_c \sim q, u | \mathbf{z}_c} \left[\|\mathcal{B}'(\bar{u})u - s_n\|_{L^2(\partial\Omega)}^2 \right] = \|\mathcal{B}'(\bar{u})\mathcal{K}(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} m - s_n\|_{L^2(\partial\Omega)}^2 \quad (\text{B.15})$$

$$+ \frac{1}{2} \int_{\partial\Omega} \mathcal{B}'(\bar{u}) \tilde{Q} \mathcal{B}'(\bar{u})^\top. \quad (\text{B.16})$$

$\square$

The next lemma uses Fernique's theorem to show that we can uniformly bound all moments of a collection of Gaussian measures under uniform bounds on their means and covariances.

Lemma 6. Let $\mu_n = \mathcal{N}(\bar{u}_n, C_n)$ be a collection of Gaussian measures on a separable Banach space with

$$\sup_n \|\bar{u}_n\| \leq C_1 \quad (\text{B.17})$$

$$\sup_n \|C_n\| \leq C_2 \quad (\text{B.18})$$

then there is an $\alpha > 0$ such that for all n

$$\mathbb{E}_{u \sim \mu_n} [\exp(\alpha \|u\|^2)] < \infty. \quad (\text{B.19})$$

Moreover, there is a C independent of n such that

$$\mathbb{E}_{u \sim \mu_n} [\|u\|^{2p}] \leq C \frac{p!}{\alpha^p} (E_{u \sim \mu_n} [\|u\|])^{2p} \leq C \frac{p!}{\alpha^p} (E_{u \sim \mu_n} [\|u\|^2])^p \quad (\text{B.20})$$

Proof. We use Fernique's theorem Ledoux, 1996, Theorem 4.1 which states that for a centered Gaussian measure

$$\mathbb{E}_{u \sim \mu} [\exp(\alpha \|u\|^2)] < \infty \iff \alpha < \frac{1}{2\|C_n\|}. \quad (\text{B.21})$$

Therefore taking $\alpha < \frac{1}{2\sup_n \|C_n\|}$, by the triangle inequality

$$\mathbb{E}_{u \sim \mu_n} [\exp(\alpha \|u\|^2)] \leq \mathbb{E}_{u \sim \mu_n} [\exp(\alpha \|u - \bar{u}_n\|^2)] \exp(\alpha \|\bar{u}_n\|^2) \quad (\text{B.22})$$

$$\leq \mathbb{E}_{u \sim \mu_n} [\exp(\alpha \|u - \bar{u}_n\|^2)] \sup_n \exp(\alpha \|\bar{u}_n\|^2) \quad (\text{B.23})$$

$$< \infty. \quad (\text{B.24})$$

The second statement follows from Martin Hairer, n.d., Proposition 3.39 which implies that

$$\mathbb{E}_{u \sim \mu_n} [\exp(\alpha \|u\|^{2p})] \leq C \frac{p!}{\alpha^p} (E_{u \sim \mu_n} [\|u\|])^{2p} \leq C \frac{p!}{\alpha^p} (E_{u \sim \mu_n} [\|u\|^2])^p. \quad (\text{B.25})$$

The second inequality follows from Jensen's inequality. $\square$

Lemma 7. Consider a parametrized family of Gaussian processes $u_{m,S}$ on a separable Hilbert space H parametrized by (m, S) in the following manner:

$$u_{m,S} \sim \mathcal{GP}(\bar{u}_m, \tilde{Q}_S) \quad (\text{B.26})$$

$$\bar{u}_m = K(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} m \quad (\text{B.27})$$

$$\tilde{Q}_S = K(\cdot, \phi_c) K(\phi_c, \phi_c)^{-1} S K(\phi_c, \phi_c)^{-1} K(\phi_c, \cdot) + \tilde{K}. \quad (\text{B.28})$$

Consider the associated Gaussian measures $\mu_{m,S} = \mathcal{N}(\bar{u}_m, Q_S)$. Then convergence of the parameters $(m, S) \rightarrow (m^*, S^*) \in \mathbb{R}^d \times \mathbb{R}^{d \times d}$ implies weak convergence of the distributions

$$\mu_{m,S} \rightharpoonup \mu_{m^*, S^*}. \quad (\text{B.29})$$

Proof. By I. Bogachev, 1998, p. 3.8.15, a sequence of Gaussian measures on a separable Hilbert space μ^n converges weakly to μ^* if and only if

$$\lim_{n \rightarrow \infty} \|\bar{u}^n - \bar{u}^*\|_H = 0 \quad (\text{B.30})$$

$$\lim_{n \rightarrow \infty} \|\tilde{Q}^n - \tilde{Q}^*\|_{\mathcal{L}(H)} = 0 \quad (\text{B.31})$$

$$\lim_{n \rightarrow \infty} \mathbb{E}_{x \sim \mu^n} [\|x\|_H^2] = \mathbb{E}_{x \sim \mu^*} [\|x\|_H^2]. \quad (\text{B.32})$$

We will show that all conditions are implied by $(m, S) \rightarrow (m^*, S^*)$.

First observe that

$$\|\bar{u}_m - \bar{u}_{m^*}\|_H = \|K(\cdot, \phi_c)K(\phi_c, \phi_c)^{-1}(m - m^*)\|_H \quad (\text{B.33})$$

$$= \left\| \sum_{i,j} K(\cdot, \phi_{c,i}) \Theta_{i,j}^{-1} (m_j - m_j^*) \right\|_H \quad (\text{B.34})$$

$$\leq \sum_{i,j} \left\| K(\cdot, \phi_{c,i}) \Theta_{i,j}^{-1} \right\|_H |m_j - m_j^*| \quad (\text{B.35})$$

and hence $m \rightarrow m^*$ implies strong convergence of the means

$$\|\bar{u}_m - \bar{u}_{m^*}\|_H \rightarrow 0, \quad (\text{B.36})$$

showing eq. (B.30). Now consider the covariance. Observe that

$$\tilde{Q}_S - \tilde{Q}_{S^*} = K(\cdot, \phi_c)K(\phi_c, \phi_c)^{-1}(S - S^*)K(\phi_c, \phi_c)^{-1}K(\phi_c, \cdot) \quad (\text{B.37})$$

$$= \sum_{i,j,k,l} K(\cdot, \phi_{c,i}) \Theta_{i,j}^{-1} (S_{j,k} - S_{j,k}^*) \Theta_{k,l}^{-1} K(\phi_{c,l}, \cdot) \quad (\text{B.38})$$

$$= \sum_{i,j,k,l} \Theta_{i,j}^{-1} \Theta_{k,l}^{-1} (S_{j,k} - S_{j,k}^*) K(\cdot, \phi_{c,i}) K(\phi_{c,l}, \cdot) \quad (\text{B.39})$$

and hence

$$\tilde{Q}_S - \tilde{Q}_{S^*} = \sum_{i,j,k,l} \Theta_{i,j}^{-1} \Theta_{k,l}^{-1} (S_{j,k} - S_{j,k}^*) K(\cdot, \phi_{c,i}) \otimes K(\cdot, \phi_{c,l}). \quad (\text{B.40})$$

Therefore $S \rightarrow S^*$ implies that both

$$\text{tr}(\tilde{Q}_S - \tilde{Q}_{S^*}) = \sum_{i,j,k,l} \Theta_{i,j}^{-1} \Theta_{k,l}^{-1} (S_{j,k} - S_{j,k}^*) \text{tr}(K(\cdot, \phi_{c,i}) \otimes K(\cdot, \phi_{c,l})) \rightarrow 0 \quad (\text{B.41})$$

$$\|\tilde{Q}_S - \tilde{Q}_{S^*}\|_{\mathcal{L}(H)} \leq \sum_{i,j,k,l} |\Theta_{i,j}^{-1} \Theta_{k,l}^{-1}| |S_{j,k} - S_{j,k}^*| \|K(\cdot, \phi_{c,i}) \otimes K(\cdot, \phi_{c,l})\|_{\mathcal{L}(H)} \rightarrow 0, \quad (\text{B.42})$$

yielding eq. (B.31). Finally, using

$$\mathbb{E}_{x \sim \mu^*} [\|x\|_H^2] = \text{tr}(\tilde{Q}) + \|\bar{u}\|_H^2 \quad (\text{B.43})$$

yields eq. (B.32) (recalling that strong convergence of the means implies convergence of the norms as well). $\square$

Proof of lemma 2

Proof. The data term can be expressed as

$$I(q) = \mathbb{E}_{\mathbf{z}_c \sim q} \mathbb{E}_{u|\mathbf{z}_c} [\Phi(u)] = \mathbb{E}_{u \sim \mu_q} [\Phi(u)] \quad (\text{B.44})$$

with

$$\Phi(u) = \|\mathcal{F}(u) - f\|_{L^2(\Omega)}^2 + \|\mathcal{B}(u) - g\|_{L^2(\partial\Omega)}^2 \quad (\text{B.45})$$

and μ_q is given by proposition 17. By lemma 7 the weak convergence $q_n \rightharpoonup q$ implies that $\mu_{q_n} \rightharpoonup \mu_q$. Hence, we aim to show that

$$\lim_{n \rightarrow \infty} \mathbb{E}_{u \sim \mu_{q_n}} [\Phi(u)] = \mathbb{E}_{u \sim \mu_q} [\Phi(u)]. \quad (\text{B.46})$$

Moreover the continuity of $\mathcal{F}$ and $\mathcal{B}$ theorem 5, the random variables $\Phi(u_n) \rightharpoonup \Phi(u)$. Hence, we will show that the family $\Phi(u_n)$ is uniformly integrable, in which case weak convergence implies convergence of the means Billingsley, 1999, Theorem 3.5.

We now suppress the dependence on q and always assume that $u_n \sim \mu_{q_n}$ where μ_{q_n} is given by proposition 17. By the growth rate theorem 5

$$0 \leq \mathbb{E}[\Phi(u_n)] \leq C \mathbb{E}[1 + \|u_n\|_H^p]. \quad (\text{B.47})$$

We have that

$$\mathbb{E}\left[(1 + \|u_n\|_H^p) \mathbf{1}_{\|u_n\|^p \geq K}\right] = \mathbb{P}(\|u_n\|^p \geq K) + \mathbb{E}\left[\|u_n\|_H^p \mathbf{1}_{\|u_n\|^p \geq K}\right] \quad (\text{B.48})$$

We treat both terms separately. We will show that all the p-moments can be bounded uniformly by the second moment. By Markov's inequality

$$\mathbb{P}(\|u_n\|^p \geq K) \leq \frac{\mathbb{E}(\|u_n\|^p)}{K}. \quad (\text{B.49})$$

For Gaussian measures on separable Hilbert space, weak convergence measures implies that

$$\sup_n \|m_n\| < \infty \quad (\text{B.50})$$

$$\sup_n \|C_n\| \leq \sup_n \text{tr}(C_n) < \infty \quad (\text{B.51})$$

$$\sup_n \mathbb{E}[\|u_n\|^2] < \infty \quad (\text{B.52})$$

(see I. Bogachev, 1998, Chapter 3). By lemma 6, for any p there is some constant that only depends on p such that

$$\sup_n \mathbb{E}[\|u_n\|^p] \leq C_1 \quad (\text{B.53})$$

(the case for p even follows directly; the case for p odd follows by considering $p + 1$ and using the monotonicity of L^p norms on probability spaces). Hence

$$\sup_n \mathbb{P}(\|u_n\|^p \geq K) \leq \sup_n \frac{\mathbb{E}(\|u_n\|^p)}{K} \leq \frac{C_1}{K}. \quad (\text{B.54})$$

We now turn to the second term. Note that $\|u_n\|^p \geq K$ implies that $\|u_n\| \geq K^{\frac{1}{p}}$ which implies that $\frac{\|u_n\|^{p+1}}{K^{\frac{1}{p}}} \geq \|u_n\|^p$. Again leveraging uniform control over all moments of the Gaussians eq. (B.53), we obtain

$$\sup_n \mathbb{E} \left[\|u_n\|_H^p \mathbf{1}_{\|u_n\|^p \geq K} \right] \leq \frac{1}{K^{\frac{1}{p}}} \sup_n \mathbb{E}[\|u_n\|^{p+1}] \leq \frac{C_2}{K^{\frac{1}{p}}}. \quad (\text{B.55})$$

Hence we obtain

$$\lim_{K \rightarrow \infty} \sup_n \mathbb{E} \left[(1 + \|u_n\|_H^p) \mathbf{1}_{\|u_n\|^p \geq K} \right] \leq \lim_{K \rightarrow \infty} C \left(\frac{C_1}{K} + \frac{C_2}{K^{\frac{1}{p}}} \right) = 0. \quad (\text{B.56})$$

Therefore, the family $\Phi(u_n)$ is uniformly integrable and therefore weak convergence implies convergence of the means Billingsley, 1999, Theorem 3.5

$$\lim_{n \rightarrow \infty} \mathbb{E} \left[\Phi(u_n) \right] = \mathbb{E} \left[\Phi(u) \right] \quad (\text{B.57})$$

□

Proof of proposition 4.

Proof. We proceed by considering a minimizing subsequence of the objective eq. (3.22) and showing that a limit point of this sequence is a minimizer.

Step 1: existence of a convergent subsequence. Let $q_n = \mathcal{N}(m_n, S_n)$ be a minimizing sequence for $J(\cdot; \sigma)$, i.e.

$$\lim_{n \rightarrow \infty} J(q_n, \sigma) = \inf_{q \in \mathcal{Q}} J(q; \sigma). \quad (\text{B.58})$$

Since $J(q_0; \sigma) < \infty$, there exists $C < \infty$ such that $J(q_n; \sigma) \leq C$ uniformly in n , hence

$$\text{KL}(q_n \| p) \leq \frac{1}{\sigma^2} I(q_n) + \text{KL}(q_n \| p) \leq C.$$

The KL divergence can be written as

$$\text{KL}(q_n \| p) = \frac{1}{2} \left(\|m_n\|_{K(\phi_c, \phi_c)}^2 + R(S_n) \right), \quad (\text{B.59})$$

where

$$R(S_n) = \text{trace}(K(\phi_c, \phi_c)^{-1} S_n) + \log \frac{|K(\phi_c, \phi_c)|}{|S_n|} - d. \quad (\text{B.60})$$

Define $A_n = K(\phi_c, \phi_c)^{-\frac{1}{2}} S_n K(\phi_c, \phi_c)^{-\frac{1}{2}} \succ 0$. Observe that $S_n = K(\phi_c, \phi_c)^{\frac{1}{2}} A_n K(\phi_c, \phi_c)^{\frac{1}{2}}$ and hence $\log \frac{|K(\phi_c, \phi_c)|}{|S_n|} = \log(A_n)$. Additionally, by the cyclic property of the trace

$$R(S_n) = \text{trace}(A_n) - \log |A_n| - k \quad (\text{B.61})$$

$$= \sum_i \lambda_{i,n} - \log(\lambda_{i,n}) - 1 \quad (\text{B.62})$$

$$= \sum_i f(\lambda_{i,n}). \quad (\text{B.63})$$

where the $\lambda_{i,n}$ are the eigenvalues of A_n and $f(x) = x - \log(x) - 1$. From this, we may deduce that the sequence (m_n, S_n) is bounded in appropriate norms. For m_n , this follows from the above. For S_n , since $f(x) \rightarrow \infty$ as $x \rightarrow \infty$ or $x \rightarrow 0$ each of the eigenvalues remains in a bounded set $0 < \underline{\lambda} < \lambda_{i,n} < \bar{\lambda} < \infty$ uniformly in i, n . Therefore

$$\underline{\lambda} I \preceq A_n \preceq \bar{\lambda} I \iff \underline{\lambda} K(\phi_c, \phi_c) \preceq S_n \preceq \bar{\lambda} K(\phi_c, \phi_c). \quad (\text{B.64})$$

From this we deduce that the sequence S_n is uniformly bounded from above and below in any matrix norm. Since $m_n \in \mathbb{R}^d$, $S_n \in \mathbb{S}^d$ which are both finite dimensional spaces, it follows from the Bolzano-Weirstrass theorem that there is a convergent subsequence (which we denote using the same index n) $(m_n, S_n) \rightarrow (m_\infty, S_\infty)$. Moreover by eq. (B.64)

$$\underline{\lambda} K(\phi_c, \phi_c) \preceq \lim_{n \rightarrow \infty} S_n = S_\infty \preceq \bar{\lambda} K(\phi_c, \phi_c) \quad (\text{B.65})$$

and therefore S_∞ is symmetric positive definite matrix as well. Hence, since for finite dimensional Gaussian distributions weak convergence is equivalent to convergence in parameters, $q_n \rightarrow q_\infty = \mathcal{N}(m_\infty, S_\infty)$.

Step 2: the limit point is a minimizer of the objective. By lemma 2 and the weaklower-semicontinuity of the KL divergence Posner, 1975

$$\frac{1}{\sigma^2} I(q_\infty) + \text{KL}(q_\infty \| p) \leq \liminf_{n \rightarrow \infty} \frac{1}{\sigma^2} I(q_n) + \text{KL}(q_n \| p) \quad (\text{B.66})$$

$$\leq \lim_{n \rightarrow \infty} \frac{1}{\sigma^2} I(q_n) + \text{KL}(q_n \| p) \quad (\text{B.67})$$

$$= \inf_{q \in \mathcal{Q}} J(q; \sigma) \quad (\text{B.68})$$

and hence we conclude that q_∞ is a minimizer of the objective. $\square$

Zero noise limit

Proof of lemma 3

Proof. The proof follows a standard minimizing sequence argument. Consider some minimizing sequence $q_n \in \bar{Q}_d$ of I :

$$\lim_{n \rightarrow \infty} I(q_n) = \inf_{q \in \bar{Q}_d} I(q). \quad (\text{B.69})$$

Then $\sup_n I(q_n) \leq c$ which by theorem 6 implies that there is a weakly convergent subsequence (which we denote by the same index n) $q_n \rightharpoonup q_\infty \in \bar{Q}_d$. By lemma 2, this implies that

$$I(q_\infty) \leq \liminf_{n \rightarrow \infty} I(q_n) = \inf_{q \in \bar{Q}_d} I(q) \quad (\text{B.70})$$

i.e. $q_\infty \in \bar{Q}_d$ is a minimizer of the functional. $\square$

B.3 Dirac-SVGP proofs

Assumption 16 (Relative compactness of the data functional I). *For every $c < \infty$, the sublevel set*

$$\{m \in \mathbb{R}^d : I(q) \leq c\}$$

is relatively compact.

Lemma 8. *There is a $m^{\text{opt}} \in \mathbb{R}^d$ which minimizes the data term:*

$$m^{\text{opt}} = \arg \min_{m \in \mathbb{R}^d} I(m). \quad (\text{B.71})$$

Proof. The proof is identical to lemma 3. $\square$

Proof of proposition 6

Proof. We proceed in three step: first we show that the sequence of optimal parameters m_n has a convergent subsequence. Second we show that any limit point is an optimal solution to the data term (unregularized loss). Third we show that, moreover, any limit point has the minimal divergence property.

Step 1: the sequence m_n has a convergent subsequence. We have $J(m_n; \gamma_n) \leq J(m^{\text{opt}}; \gamma_n)$ and using lemma 8

$$0 \leq \|m_n\|_{K(\phi_c, \phi_c)}^2 \leq \frac{1}{\gamma_n} (I(q^{\text{opt}}) - I(q_n)) + \|m^{\text{opt}}\|_{K(\phi_c, \phi_c)}^2 \quad (\text{B.72})$$

$$\leq \|m^{\text{opt}}\|_{K(\phi_c, \phi_c)}^2 \quad (\text{B.73})$$

from the fact that $I(q^{\text{opt}}) \leq I(q_n)$. From this, we may deduce that the sequence m_n is bounded the norm $\|\cdot\|_{K(\phi_c, \phi_c)}$. Since $m_n \in \mathbb{R}^d$, it follows from the Bolzano-Weirstrass theorem that there is a convergent subsequence (which we denote using the same index n) $m_n \rightarrow m_\infty$.

Step 2: any limit point m_∞ ¹ of the sequence m_n is a minimizer of the data term (unregularized loss). By lemma 2

$$\lim_{n \rightarrow \infty} I(m_n) = I(q_\infty). \quad (\text{B.74})$$

Consider some arbitrary $q \in \mathcal{Q}$. Then by the definition of m_n and positivity of the KL divergence

$$\frac{1}{\sigma_n^2} I(m_n) \leq J(m_n; \gamma_n) \leq J(m; \gamma_n) \quad (\text{B.75})$$

which implies that

$$I(m_n) \leq I(m) + \gamma_n \|m\|_{K(\phi_c, \phi_c)}^2 \quad (\text{B.76})$$

Hence for a fixed m taking the limit $n \rightarrow \infty$ yields $\lim_{n \rightarrow \infty} I(m_n) = I(m_\infty) \leq I(m)$. Then further infimizing over all m

$$I(m_\infty) = \inf_{m \in \mathbb{R}^d} I(m). \quad (\text{B.77})$$

Step 3: any limit point of m_n further minimizes the norm $\|\cdot\|_{K(\phi_c, \phi_c)}$ among all minimizers of the data term. Consider some element $m^{\text{opt}} \in \arg \min_{m \in \mathbb{R}^d} I(m)$. We have that $I(m_\infty) = I(m^{\text{opt}}) \leq I(m_n)$ and $J(m_n; \gamma_n) \leq J(m^{\text{opt}}; \gamma_n)$. Together these imply that

$$I(m_\infty) + \gamma_n \|m_n\|_{K(\phi_c, \phi_c)} \leq I(m_n) + \gamma_n \|m_n\|_{K(\phi_c, \phi_c)} \quad (\text{B.78})$$

$$\leq I(m^{\text{opt}}) + \gamma_n \|m^{\text{opt}}\|_{K(\phi_c, \phi_c)} \quad (\text{B.79})$$

$$= I(m_\infty) + \gamma_n \|m^{\text{opt}}\|_{K(\phi_c, \phi_c)} \quad (\text{B.80})$$

and therefore that $\|m_n\|_{K(\phi_c, \phi_c)} \leq \|m^{\text{opt}}\|_{K(\phi_c, \phi_c)}$. Using the weak lower-semicontinuity of norms

$$\|m_\infty\|_{K(\phi_c, \phi_c)} \leq \liminf_{n \rightarrow \infty} \|m_n\|_{K(\phi_c, \phi_c)} \leq \|m^{\text{opt}}\|_{K(\phi_c, \phi_c)} \quad (\text{B.81})$$

proving the claim. □

¹Note that step 1 shows that there is *at least one* limit point, but step 2 applies to *any* limit point.

B.4 Model parameters and architecture

For the Gaussian process method, we use the squared exponential kernel Rasmussen and Williams, 2008

$$k(x, y) = \sigma^2 \exp\left(-\frac{\|x - y\|^2}{2l^2}\right) \quad (\text{B.82})$$

which has hyperparameters (σ, l) .

For the PINN approach, we use random feature embeddings and the tanh activation wang_experts_2023; wang_eigenvector_2020.

Parameter	Possible values	Description
rff_features	[1,6]	Number of Fourier Fourier Feature blocks.
rff_sigmas	(1e-2, 10.0)	Length-scale parameter for the random Fourier feature blocks. One frequency matrix B_i is created for each σ_i , with entries initialized as $\mathcal{N}(0, 1/\sigma_i^2)$.
n_features	[64, 2048]	Number of sampled frequencies per σ .
mlp_hidden	[64, 1024]	Width of the hidden layers in the MLP applied after the RFF embedding and skip connection.
mlp_size	[2,8]	Number of hidden layers in the MLP applied after the RFF embedding and skip connection.
activation	tanh	Activation function used in the MLP hidden layers.

Table B.1: Main parameters of the RFF-based PINN.

Appendix C

CHAPTER 4

C.1 Review of operator valued kernels and GPs

We review the theory of operator valued kernels and GPs Houman Owhadi, 2023a as these are utilized throughout the article. Operator-valued kernels were introduced in Kadri, Duflos, Preux, Stephane Canu, et al., 2016a as a generalization of vector-valued kernels Alvarez, Rosasco, Neil D Lawrence, et al., 2012.

Operator valued kernels

Let $\mathcal{U}$ and $\mathcal{V}$ be separable Hilbert spaces endowed with the inner products $\langle \cdot, \cdot \rangle_{\mathcal{U}}$ and $\langle \cdot, \cdot \rangle_{\mathcal{V}}$. Write $\mathcal{L}(\mathcal{V})$ for the set of bounded linear operators mapping $\mathcal{V}$ to $\mathcal{V}$.

Definition 4. We call $G : \mathcal{U} \times \mathcal{U} \rightarrow \mathcal{L}(\mathcal{V})$ an “operator-valued kernel” if

1. G is Hermitian, i.e. $G(u, u') = G(u', u)^T$ for all $u, u' \in \mathcal{U}$, writing A^T for the adjoint of the operator A with respect to $\langle \cdot, \cdot \rangle_{\mathcal{V}}$.
2. G is non-negative, i.e., for all $m \in \mathbb{N}$ and any set of points $(u_i, v_i)_{i=1}^m \subset \mathcal{U} \times \mathcal{V}$ it holds that $\sum_{i,j=1}^m \langle v_i, G(u_i, u_j) v_j \rangle_{\mathcal{V}} \geq 0$.

We call G non-degenerate if $\sum_{i,j=1}^m \langle v_i, G(u_i, u_j) v_j \rangle_{\mathcal{V}} = 0$ implies $v_i = 0$ for all i whenever $u_i \neq u_j$ for $i \neq j$.

RKHSs

Each non-degenerate, locally bounded and separately continuous operator-valued kernel G is in one to one correspondence with an RKHS $\mathcal{H}$ of continuous operators $\mathcal{G} : \mathcal{U} \rightarrow \mathcal{V}$ obtained as the closure of the linear span of the maps $z \mapsto G(z, u)v$ with respect to the inner product identified by the reproducing property

$$\langle g, G(\cdot, u)v \rangle_{\mathcal{H}} = \langle g(u), v \rangle_{\mathcal{V}}. \quad (\text{C.1})$$

Feature maps

Let $\mathcal{F}$ be a separable Hilbert space (with inner product $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ and norm $\| \cdot \|_{\mathcal{F}}$) and let $\psi : \mathcal{U} \rightarrow \mathcal{L}(\mathcal{V}, \mathcal{F})$ be a continuous function mapping $\mathcal{U}$ to the space of bounded linear operators from $\mathcal{V}$ to $\mathcal{F}$.

Definition 5. We say that $\mathcal{F}$ and $\psi : \mathcal{U} \rightarrow \mathcal{L}(\mathcal{V}, \mathcal{F})$ are a feature space and a feature map for the kernel G if, for all $(u, u', v, v') \in \mathcal{U}^2 \times \mathcal{V}^2$,

$$\langle v, G(u, u')v' \rangle = \langle \psi(u)v, \psi(u')v' \rangle_{\mathcal{F}}.$$

Write $\psi^T(u)$, for the adjoint of $\psi(u)$ defined as the linear function mapping $\mathcal{F}$ to $\mathcal{V}$ satisfying

$$\langle \psi(u)v, \alpha \rangle_{\mathcal{F}} = \langle v, \psi^T(u)\alpha \rangle_{\mathcal{V}}$$

for $u, v, \alpha \in \mathcal{U} \times \mathcal{V} \times \mathcal{F}$. Note that $\psi^T : \mathcal{U} \rightarrow \mathcal{L}(\mathcal{F}, \mathcal{V})$ is therefore a function mapping $\mathcal{U}$ to the space of bounded linear functions from $\mathcal{F}$ to $\mathcal{V}$. Writing $\alpha^T \alpha' := \langle \alpha, \alpha' \rangle_{\mathcal{F}}$ for the inner product in $\mathcal{F}$ we can ease our notations by writing

$$G(u, u') = \psi^T(u)\psi(u') \quad (\text{C.2})$$

which is consistent with the finite-dimensional setting and $v^T G(u, u')v' = (\psi(u)v)^T(\psi(u')v')$ (writing $v^T v'$ for the inner product in $\mathcal{V}$). For $\alpha \in \mathcal{F}$ write $\psi^T \alpha$ for the function $\mathcal{U} \rightarrow \mathcal{V}$ mapping $u \in \mathcal{U}$ to the element $v \in \mathcal{V}$ such that

$$\langle v', v \rangle_{\mathcal{V}} = \langle v', \psi^T(u)\alpha \rangle_{\mathcal{V}} = \langle \psi(u)v', \alpha \rangle_{\mathcal{F}} \text{ for all } v' \in \mathcal{V}.$$

We can, without loss of generality, restrict $\mathcal{F}$ to be the range of $(u, v) \rightarrow \psi(u)v$ so that the RKHS $\mathcal{H}$ defined by G is the closure of the pre-Hilbert space spanned by $\psi^T \alpha$ for $\alpha \in \mathcal{F}$. Note that the reproducing property (C.1) implies that for $\alpha \in \mathcal{F}$

$$\langle \psi^T(\cdot)\alpha, \psi^T(\cdot)\psi(u)v \rangle_{\mathcal{H}} = \langle \psi^T(u)\alpha, v \rangle_{\mathcal{V}} = \langle \alpha, \psi(u)v \rangle_{\mathcal{F}}$$

for all $u, v \in \mathcal{U} \times \mathcal{V}$, which leads to the following theorem.

Theorem 17. *The RKHS $\mathcal{H}$ defined by the kernel (C.2) is the linear span of $\psi^T \alpha$ over $\alpha \in \mathcal{F}$ such that $\|\alpha\|_{\mathcal{F}} < \infty$. Furthermore, $\langle \psi^T(\cdot)\alpha, \psi^T(\cdot)\alpha' \rangle_{\mathcal{H}} = \langle \alpha, \alpha' \rangle_{\mathcal{F}}$ and*

$$\|\psi^T(\cdot)\alpha\|_{\mathcal{H}}^2 = \|\alpha\|_{\mathcal{F}}^2 \text{ for } \alpha, \alpha' \in \mathcal{F}.$$

Interpolation

Let us consider the interpolation problem in operator valued RKHSs.

Problem 5. *Let $\mathcal{G}^\dagger$ be an unknown continuous operator mapping $\mathcal{U}$ to $\mathcal{V}$. Given the information¹ $\mathcal{G}^\dagger(\mathbf{u}) = \mathbf{v}$ with the data $(\mathbf{u}, \mathbf{v}) \in \mathcal{U}^N \times \mathcal{V}^N$, approximate $\mathcal{G}^\dagger$.*

Using the relative error in $\|\cdot\|_{\mathcal{H}}$ -norm as a loss, the minimax optimal recovery solution of Problem 5 is, by Houman Owhadi and Scovel, 2019b, Thm. 12.4, 12.5, given by

$$\begin{cases} \text{Minimize} & \|\mathcal{G}\|_{\mathcal{H}}^2 \\ \text{subject to} & \mathcal{G}(\mathbf{u}) = \mathbf{v}. \end{cases} \quad (\text{C.3})$$

¹For a N -vector $\mathbf{u} = (u_1, \dots, u_N) \in \mathcal{U}^N$ and a function $\mathcal{G} : \mathcal{U} \rightarrow \mathcal{V}$, write $\mathcal{G}(\mathbf{u})$ for the N vector with entries $(\mathcal{G}(u_1), \dots, \mathcal{G}(u_N))$.

The minimizer is then of the form $\mathcal{G}(\cdot) = \sum_{j=1}^N G(\cdot, u_j) w_j$, where the coefficients $w_j \in \mathcal{V}$ are identified by solving the system of linear equations $\sum_{j=1}^N G(u_i, u_j) w_j = v_i$ for all $i \in \{1, \dots, N\}$. Using our compressed notation we can rewrite this equation as $G(\mathbf{u}, \mathbf{u}) \mathbf{w} = \mathbf{v}$ where $\mathbf{w} = (w_1, \dots, w_N)$, $\mathbf{v} = (v_1, \dots, v_N) \in \mathcal{V}^N$ and $G(\mathbf{u}, \mathbf{u})$ is the $N \times N$ block-operator matrix² with entries $G(u_i, u_j)$. Therefore, writing $G(\cdot, \mathbf{u})$ for the vector $(G(\cdot, u_1), \dots, G(\cdot, u_N)) \in \mathcal{H}^N$, the optimal recovery interpolant is given by

$$\bar{\mathcal{G}}(\cdot) = G(\cdot, \mathbf{u}) G(\mathbf{u}, \mathbf{u})^{-1} \mathbf{v}, \quad (\text{C.4})$$

which implies that the value of (C.3) at the minimum is

$$\|\bar{\mathcal{G}}\|_{\mathcal{H}}^2 = \mathbf{v}^T G(\mathbf{u}, \mathbf{u})^{-1} \mathbf{v}, \quad (\text{C.5})$$

where $G(\mathbf{u}, \mathbf{u})^{-1}$ is the inverse of $G(\mathbf{u}, \mathbf{u})$, whose existence is implied by the non-degeneracy of G combined with $u_i \neq u_j$ for $i \neq j$.

Ridge regression

Let $\gamma > 0$. A ridge regression (approximate) solution to Problem 5 can be found as the minimizer of

$$\inf_{\mathcal{G} \in \mathcal{H}} \lambda \|\mathcal{G}\|_{\mathcal{H}}^2 + \gamma^{-1} \sum_{i=1}^N \|v_i - \mathcal{G}(u_i)\|_{\mathcal{V}}^2. \quad (\text{C.6})$$

This minimizer is given by the formula

$$\bar{\mathcal{G}}(u) = G(u, \mathbf{u}) (G(\mathbf{u}, \mathbf{u}) + \gamma I)^{-1} \mathbf{v}, \quad (\text{C.7})$$

writing I for the identity matrix. We can further compute directly

$$\|\bar{\mathcal{G}}\|_{\mathcal{H}}^2 = \mathbf{v}^T (G(\mathbf{u}, \mathbf{u}) + \gamma I)^{-1} \mathbf{v}.$$

Operator-valued GPs

The following definition of operator-valued Gaussian processes is a natural extension of scalar-valued Gaussian fields Houman Owhadi and Scovel, 2019b.

Definition 6. Houman Owhadi, 2023a, Def. 5.1 Let $G : \mathcal{U} \times \mathcal{U} \rightarrow \mathcal{L}(\mathcal{V})$ be an operator-valued kernel. Let m be a function mapping $\mathcal{U}$ to $\mathcal{V}$. We call $\xi :$

²For $N \geq 1$ let $\mathcal{V}^N$ be the N -fold product space endowed with the inner-product $\langle \mathbf{v}, \mathbf{w} \rangle_{\mathcal{V}^N} := \sum_{i,j=1}^N \langle v_i, w_j \rangle_{\mathcal{V}}$ for $\mathbf{v} = (v_1, \dots, v_N)$, $\mathbf{w} = (w_1, \dots, w_N) \in \mathcal{V}^N$. $\mathbf{A} \in \mathcal{L}(\mathcal{V}^N)$ given by $\mathbf{A} = \begin{pmatrix} A_{1,1} & \cdots & A_{1,N} \\ \vdots & & \vdots \\ A_{N,1} & \cdots & A_{N,N} \end{pmatrix}$ where $A_{i,j} \in \mathcal{L}(\mathcal{V})$, is called a block-operator matrix. Its adjoint $\mathbf{A}^T$ with respect to $\langle \cdot, \cdot \rangle_{\mathcal{V}^N}$ is the block-operator matrix with entries $(\mathbf{A}^T)_{i,j} = (A_{j,i})^T$.

$\mathcal{U} \rightarrow \mathcal{L}(\mathcal{V}, \mathbf{H})$ an operator-valued GP if ξ is a function mapping $u \in \mathcal{U}$ to $\xi(u) \in \mathcal{L}(\mathcal{V}, \mathbf{H})$ where $\mathbf{H}$ is a Gaussian space and $\mathcal{L}(\mathcal{V}, \mathbf{H})$ is the space of bounded linear operators from $\mathcal{V}$ to $\mathbf{H}$. Abusing notations we write $\langle \xi(u), v \rangle_{\mathcal{V}}$ for $\xi(u)v$. We say that ξ has mean m and covariance kernel G and write $\xi \sim \mathcal{N}(m, G)$ if $\langle \xi(u), v \rangle_{\mathcal{V}} \sim \mathcal{N}(m(u), v^T G(u, u) v)$ and

$$\text{Cov}(\langle \xi(u), v \rangle_{\mathcal{V}}, \langle \xi(u'), v' \rangle_{\mathcal{V}}) = v^T G(u, u') v'. \quad (\text{C.8})$$

We say that ξ is centered if it is of zero mean.

If $G(u, u)$ is trace class ($\text{Tr}[G(u, u)] < \infty$) then $\xi(u)$ defines a measure on $\mathcal{V}$, i.e. a $\mathcal{V}$ -valued random variable³.

Theorem 18. *Houman Owhadi, 2023a, Thm. 5.2 The law of an operator-valued GP is uniquely determined by its mean m and covariance kernel G . Conversely given m and G there exists an operator-valued GP having mean m and covariance kernel G . In particular if G has feature space $\mathcal{F}$ and map ψ , the e_i form an orthonormal basis of $\mathcal{F}$, and the Z_i are i.i.d. $\mathcal{N}(0, 1)$ random variables, then $\xi = m + \sum_i Z_i \psi^T e_i$ is an operator-valued GP with mean m and covariance kernel G .*

Theorem 19. *Houman Owhadi, 2023a, Thm. 5.3 Let ξ be a centered operator-valued GP with covariance kernel $G : \mathcal{U} \times \mathcal{U} \rightarrow \mathcal{L}(\mathcal{V})$. Let $\mathbf{u}, \mathbf{v} \in \mathcal{U}^N \times \mathcal{V}^N$. Let $Z = (Z_1, \dots, Z_N)$ be a random Gaussian vector, independent from ξ , with i.i.d. $\mathcal{N}(0, \gamma I_{\mathcal{V}})$ entries ($\gamma \geq 0$ and $I_{\mathcal{V}}$ is the identity map on $\mathcal{V}$). Then ξ conditioned on $\xi(\mathbf{u}) + Z$ is an operator-valued GP with mean*

$$\mathbb{E}[\xi(u) | \xi(\mathbf{u}) + Z = \mathbf{v}] = G(u, \mathbf{u})(G(\mathbf{u}, \mathbf{u}) + \gamma I_{\mathcal{V}})^{-1} \mathbf{v} = (\text{C.7}) \quad (\text{C.9})$$

and conditional covariance operator

$$G^\perp(u, u') := G(u, u') - G(u, \mathbf{u})(G(\mathbf{u}, \mathbf{u}) + \gamma I_{\mathcal{V}})^{-1} G(\mathbf{u}, u'). \quad (\text{C.10})$$

In particular, if G is trace class, then

$$\sigma^2(u) := \mathbb{E}[\|\xi(u) - \mathbb{E}[\xi(u) | \xi(\mathbf{u}) + Z = \mathbf{v}]\|_{\mathcal{V}}^2 | \xi(\mathbf{u}) + Z = \mathbf{v}] = \text{Tr}[G^\perp(u, u)]. \quad (\text{C.11})$$

³Otherwise it only defines a (weak) cylinder-measure in the sense of Gaussian fields.

Deterministic error estimates for operator-valued regression

The following theorem shows that the standard deviation (C.11) provides deterministic a priori error bounds on the accuracy of the ridge regressor (C.9) to $\mathcal{G}^\dagger$ in Problem 5. Local error estimates such as (C.12) below are classical in the Kriging literature Z.-m. Wu and Schaback, 1993 where $\sigma^2(u)$ is known as the power function/kriging variance; see also Houman Owhadi, 2015b[Thm. 5.1] for applications to PDEs.

Theorem 20. *Houman Owhadi, 2023a, Thm. 5.4 Let $\mathcal{G}^\dagger$ be the unknown function of Problem 5 and let $\mathcal{G}(u) = (C.9) = (C.7)$ be its ridge regressor. Let $\mathcal{H}$ be the RKHS associated with G and let $\mathcal{H}_\gamma$ be the RKHS associated with the kernel $G_\gamma := G + \gamma I_{\mathcal{V}}$. It holds true that*

$$\|\mathcal{G}^\dagger(u) - \mathcal{G}(u)\|_{\mathcal{V}} \leq \sigma(u) \|\mathcal{G}^\dagger\|_{\mathcal{H}} \quad (C.12)$$

and

$$\|\mathcal{G}^\dagger(u) - \mathcal{G}(u)\|_{\mathcal{V}} \leq \sqrt{\sigma^2(u) + \gamma \dim(\mathcal{V})} \|\mathcal{G}^\dagger\|_{\mathcal{H}_\gamma}, \quad (C.13)$$

where $\sigma(u)$ is the standard deviation (C.11).

C.2 An alternative regularization of operator regression

For $\gamma > 0$, the regularization implied by (4.27) is equivalent to adding noise on the $\varphi(\mathbf{v})$ measurements. If one could observe $\mathbf{v}$ (and not just $\varphi(\mathbf{v})$), then an alternative approach to regularizing the problem is to add noise to $\xi(\mathbf{u})$. To describe this let $Z' = (Z'_1, \dots, Z'_N)$ be a random block-vector, independent from ξ , with i.i.d. entries $Z'_j \sim \mathcal{N}(0, \gamma I_{\mathcal{V}})$ for $j = 1, \dots, N$ (where $I_{\mathcal{V}}$ denotes the identity map on $\mathcal{V}$). Then the GP ξ conditioned on $\xi(\mathbf{u}) = \mathbf{v} + Z'$ is a GP with conditional covariance kernel (C.10) and conditional mean $\tilde{\mathcal{G}}_\gamma = (C.7)$ that is also the minimizer of (C.6). Observing⁴ that $\varphi(Z'_i) \sim \mathcal{N}(0, \gamma K(\varphi, \varphi))$, we deduce that $\tilde{\mathcal{G}}_\gamma = \chi \circ \tilde{f}_\gamma \circ \phi$ where $\tilde{f}_\gamma$ minimizes

$$\begin{cases} \text{Minimize} & \|f\|_{\Gamma}^2 + \gamma^{-1} \sum_{i=1}^N (f(U_i) - V_i)^T K(\varphi, \varphi)^{-1} (f(U_i) - V_i). \\ \text{Over} & f \in \mathcal{H}_{\Gamma}. \end{cases} \quad (C.14)$$

Furthermore, the distribution of ξ conditioned on $\xi(\mathbf{u}) = \mathbf{v} + Z'$ is that of $\chi \circ \tilde{\zeta}^\perp \circ \phi$ where $\tilde{\zeta}^\perp \sim \mathcal{N}(\tilde{f}_\gamma, \tilde{\Gamma}^\perp)$ is the GP ζ conditioned on $\zeta(\mathbf{U}) = \mathbf{V} + \varphi(Z')$, whose mean is

⁴This follows from $\varphi(Z'_i) \sim \mathcal{N}(0, \gamma \varphi \varphi^T)$ where φ^T is the adjoint of φ identified as the linear map from $\mathbb{R}^m$ to $\mathcal{V}$ satisfying $\langle W, \varphi(w) \rangle_{\mathbb{R}^m} = \langle \varphi^\perp W, w \rangle_{\mathcal{V}}$ for $w \in \mathcal{V}$ and $W \in \mathbb{R}^m$ (i.e., $\varphi^T(W) = (K\varphi)W$).

$\tilde{f}_\gamma$ and conditional covariance kernel is $\tilde{\Gamma}^\perp(U, U') = \Gamma(U, U') - \Gamma(U, \mathbf{U})(\Gamma(\mathbf{U}, \mathbf{U}) + \gamma A)^{-1} \Gamma(\mathbf{U}, U')$ where A is a $N \times N$ block diagonal matrix with $K(\varphi, \varphi)$ as diagonal entries.

C.3 Expressions for the kernels used in experiments

Below we collect the expressions for the kernels that were referred to in the article or utilized for our numerical experiments. These can be found in many standard textbooks on GPs such as Rasmussen and Williams, 2006.

The linear kernel

The linear kernel has the simple expression $K_{\text{linear}}(x, x') = \langle x, x' \rangle$ and may be defined on any inner product space. It has no hyper-parameters.

The rational quadratic kernel

The rational quadratic kernel has the expression $K(x, x') = k_{\text{RQ}}(\|x - x'\|)$ where

$$k_{\text{RQ}}(r) = \left(1 + \frac{r^2}{2l^2}\right)^{-\alpha}. \quad (\text{C.15})$$

It has hyper-parameters $\alpha > 0$ and l .

The Matérn parametric family

The Matérn kernel family is of the form $K(x, x') = k(\|x - x'\|)$ where

$$k_\nu(r) = \exp\left(-\frac{\sqrt{2\nu}r}{l}\right) \frac{\Gamma(p+1)}{\Gamma(2p+1)} \sum_{i=0}^p \frac{(p+1)!}{i!(p-i)!} \left(\frac{\sqrt{8\nu}r}{l}\right)^{p-i}, \quad (\text{C.16})$$

for $\nu = p + \frac{1}{2}$. This kernel has hyper-parameters $p \in \mathbb{Z}_+$ and $l > 0$. In the limiting case where $\nu \rightarrow \infty$, the Matérn kernel, we obtain the Gaussian or squared exponential kernel:

$$k_\infty(r) = \exp\left(-\frac{r^2}{2l^2}\right), \quad (\text{C.17})$$

with hyper-parameter $l > 0$.

Appendix D

CHAPTER 5

D.1 Optimal recovery

Let $\mathcal{U}$ be a Banach space with norm $\|\cdot\|_{\mathcal{U}}$, and let $w : \mathcal{U} \rightarrow \mathbb{R}^m$ be a measurement map with components $w_i \in \mathcal{U}^*$ (the dual space of $\mathcal{U}$), so that $w(\varphi) = (w_1(\varphi), \dots, w_m(\varphi))^T$. Given data $z = w(\varphi)$, we reconstruct φ from z by optimal recovery. Namely, we take $\tilde{w}$ to be a data-consistent reconstruction rule that minimizes the worst-case relative error

$$\tilde{w} \in \underset{\substack{v: \mathbb{R}^m \rightarrow \mathcal{U} \\ w(v(z))=z}}{\operatorname{argmin}} \sup_{\varphi \in \mathcal{U} \setminus \{0\}} \frac{\|\varphi - v(w(\varphi))\|_{\mathcal{U}}}{\|\varphi\|_{\mathcal{U}}}.$$

Equivalently, for each z , $\tilde{w}(z)$ is the minimax-optimal interpolant among all $\psi \in \mathcal{U}$ satisfying $w(\psi) = z$. As shown in Houman Owhadi and Scovel, 2019c, this coincides with the minimum-norm interpolant:

$$\tilde{w}(z) := \underset{\psi \in \mathcal{U}}{\operatorname{argmin}} \|\psi\|_{\mathcal{U}}^2 \quad \text{subject to} \quad w(\psi) = z.$$

In this paper, we do not evaluate $\tilde{w}$ explicitly. Rather, it provides a useful conceptual link to optimal recovery and may serve as a starting point for a convergence analysis of the proposed algorithm, as discussed in Section 5.4.

D.2 Further Experiments

In this section, we present two additional numerical examples to illustrate our methods. The first concerns the exact Gaussian-model solution of the Wetterich–Morris equation (Subsection D.2). Here, we use an exponential regulator rather than the Litim regulator used in Section 5.3.

The second experiment investigates the Wetterich flow on the continuum interval $[0, 1]$ for a ϕ^4 model, whose bare action includes a quartic self-interaction term. Because of this nonlinear self-interaction, the corresponding flow equation cannot be solved in closed form, and the effective average action $\Gamma_k[\phi]$ is no longer available as an explicit analytic expression, in contrast to the Gaussian benchmarks. This experiment therefore examines the performance of our GP-based solver on a genuinely interacting, non-Gaussian flow without an explicit solution. We draw a randomized set of collocation functions and learn the flow on that set. We then evaluate the learned flow on spatially constant functions, which allows a direct comparison with the local-potential approximation (LPA) method (see Appendix D.5). Hence, although this example yields results similar to the LPA method, its purpose is to demonstrate that our approach can directly solve the Wetterich equation on non-constant fields, which the LPA cannot.

The Wetterich–Morris equation: Gaussian-model

We now turn to the Wetterich–Morris equation

$$\partial_\kappa \Gamma_\kappa[\phi] = \frac{1}{2} \int_0^1 \int_0^1 \partial_\kappa R_\kappa(x, y) \left(\Gamma_\kappa^{(2)}[\phi] + R_\kappa \right)^{-1}(x, y) dx dy, \quad (\text{D.1})$$

where $\Gamma_\kappa^{(2)}[\phi]$ denotes the linear operator defined by the second functional derivative with respect to ϕ . We consider (D.1) with effective average action Γ_κ given explicitly by

$$\Gamma_\kappa[\phi] = \frac{1}{2} \int_0^1 \left[\gamma (\nabla \phi(x))^2 + \gamma m^2 \phi(x)^2 \right] dx + \frac{1}{2} \ln \frac{\det[-\gamma \Delta + \gamma m^2 + R_\kappa]}{\det[-\gamma \Delta + \gamma m^2]}, \quad (\text{D.2})$$

where $m > 0$ is the mass and $\gamma > 0$ is a small prefactor in front of the quadratic term that moderates the influence of the Laplacian in the numerics. The bare action for (D.1) at the UV scale κ_{UV} is chosen by evaluating (D.2) at $\kappa = \kappa_{\text{UV}}$. We refer to (D.2) as the Gaussian model, since for every scale κ the corresponding bare action is a quadratic functional of ϕ (up to an additive constant).

As in the Wilson–Polchinski experiment, we work in $H_{\text{per}}^1([0, 1]; \mathbb{R})$ and in the truncated Fourier subspace V_P defined in Section 5.3, representing fields by their Fourier coefficients. The exponential regulator

$$R_\kappa(p) = \frac{|p|^{2\alpha}}{e^{(|p|/\kappa)^{2\alpha}} - 1}, \quad \alpha = 1,$$

enters both the logarithmic determinant in Γ_κ and the flow (D.1) through its real-space kernel $R_\kappa(x, y)$ and its scale derivative $\partial_\kappa R_\kappa(x, y)$ obtained by inverse Fourier transform.

We again place a GP prior on the restriction of Γ_κ to V_P , modeling Γ_κ as a zero-mean GP with the kernel (5.19).

Numerical setup

For this benchmark, we use the explicit solution in (D.2) with $\gamma = 10^{-3}$ and $m^2 = 1$. We reuse the same truncation P , the same Fourier representation, and the same sampling strategy as in Section 5.3. In particular, we work with $N = 1,000$ collocation fields $\Phi = \{\phi^{(i)}\}_{i=1}^N \subset V_P$ and an independent test ensemble $\tilde{\Phi} = \{\tilde{\phi}^{(j)}\}_{j=1}^{N_{\text{test}}} \subset V_P$ with $N_{\text{test}} = 200$, generated by the correlated-Gaussian and i.i.d. Gaussian procedures described in Section 5.3, including the same mild $1/|p|$ decay on high modes.

We use the explicit solution in (D.2) to evaluate the effective action at the ultraviolet scale $\kappa_{\text{UV}} = 10$, and take this as the initial condition for the flow. The associated ODE system in κ is then integrated from κ_{UV} down to $\kappa_{\text{IR}} = 10^{-10}$, and the values of Γ_κ are recorded on the same grid $\{\kappa_\ell\}_{\ell=0}^{N_T}$ as in the Wilson–Polchinski example. We denote the GP solution at scale κ by $\Gamma_\kappa^\dagger$. At each κ_ℓ we evaluate both the exact Γ_{κ_ℓ} and the surrogate $\Gamma_{\kappa_\ell}^\dagger$ on Φ and $\tilde{\Phi}$.

The collocation and test errors are defined exactly as in the Wilson–Polchinski experiment, but with $W_\kappa(J)$ replaced by $\Gamma_\kappa(\phi)$. For clarity we denote these by $\varepsilon_\Gamma^{\text{coll}}(\kappa_\ell)$, $\varepsilon_\Gamma^{\text{test}}(\kappa_\ell)$, and their averages $\bar{\varepsilon}_\Gamma^{\text{coll}}$, $\bar{\varepsilon}_\Gamma^{\text{test}}$, which are reported in Table D.1.

Numerical results and discussion

Figure D.2 summarizes the behavior of the GP-based solver for the Wetterich Gaussian model. Panel (A) displays three representative fields from the collocation set. Panel (B) compares, for the same fields, the trajectories $\kappa \mapsto \Gamma_\kappa[\phi^{(i)}]$ with the GP surrogate $\kappa \mapsto \Gamma_\kappa^\dagger[\phi^{(i)}]$. The curves demonstrate accuracy across the full RG range. Panel (C) shows the collocation errors $\varepsilon_\Gamma^{\text{coll}}(\kappa)$ as a function of κ ; they remain uniformly small along the flow.

Panels (D)–(F) repeat these diagnostics for three test fields and for the entire test ensemble $\tilde{\Phi}$. The out-of-sample trajectories $\Gamma_\kappa[\tilde{\phi}^{(j)}]$ and $\Gamma_\kappa^\dagger[\tilde{\phi}^{(j)}]$ remain close in absolute size, even though the test errors $\varepsilon_\Gamma^{\text{test}}(\kappa)$ are one to three orders of magnitude larger than the collocation errors and are still small on the scale of the function values. Together with the averages $\bar{\varepsilon}_\Gamma^{\text{coll}}$ and $\bar{\varepsilon}_\Gamma^{\text{test}}$ in Table D.1, these results show that the GP surrogate provides an accurate and stable approximation of the Wetterich flow for this Gaussian benchmark, both on and off the training set.

Table D.1: The Wetterich Gaussian Model: Average relative L^2 errors on collocation and test sets for varying sample sizes N .

	$N = 100$	$N = 200$	$N = 500$	$N = 1000$
Collocation set	1.31×10^{-9}	1.54×10^{-10}	5.40×10^{-11}	1.48×10^{-11}
Test set	1.27×10^{-6}	4.62×10^{-8}	3.07×10^{-9}	8.62×10^{-10}

The Wetterich equation: continuum ϕ^4 -model.

In this subsection, we consider the Wetterich–Morris equation

$$\partial_\kappa \Gamma_\kappa[\phi] = \frac{1}{2} \int \partial_\kappa R_\kappa(x, y) \left(\frac{\delta^2 \Gamma_\kappa[\phi]}{\delta \phi \delta \phi} + R_\kappa \right)^{-1} (x, y) dx dy, \quad (\text{D.3})$$

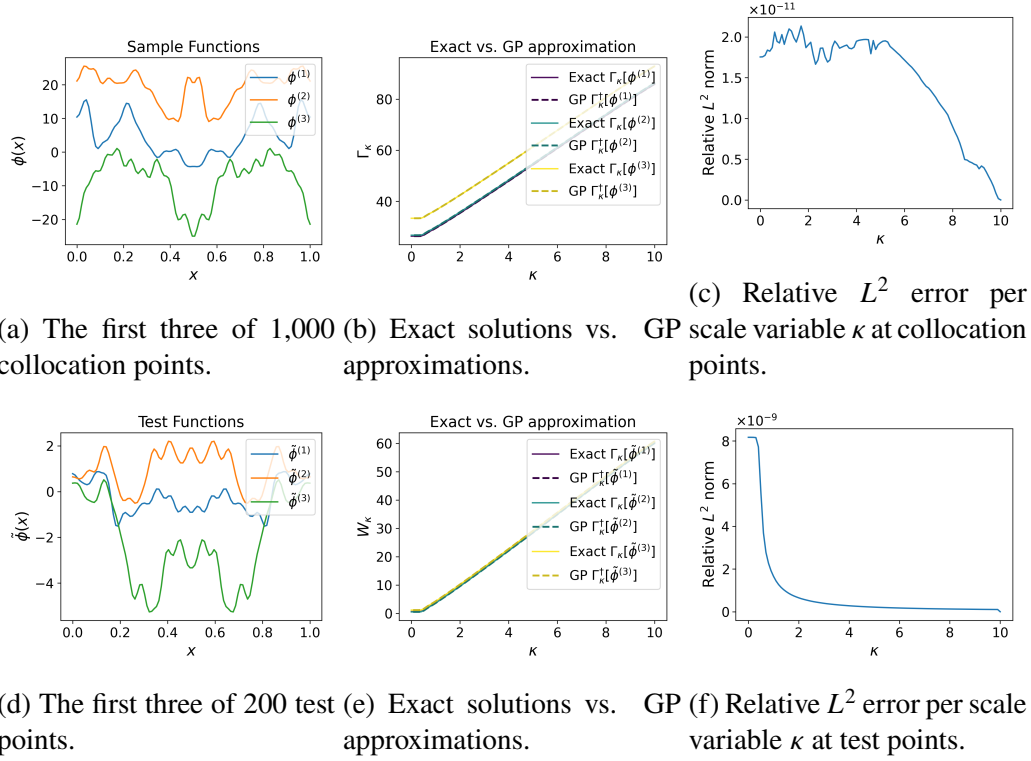


Figure D.1: Numerical results of the Wetterich Gaussian model. Subfigures (A)–(C) show results at collocation points, while (D)–(F) present corresponding results at testing points. For visualization, only the first 3 of 1,000 collocation points and the first 3 of 200 testing points are displayed. The relative L^2 error is evaluated for each scale variable κ .

with the ϕ^4 bare action on the periodic unit interval $[0, 1]$ at $\kappa = \kappa_{\text{UV}}$:

$$\Gamma_{\kappa_{\text{UV}}}[\phi] = \int_0^1 dx \left[\frac{1}{2} (\partial_x \phi)^2 + \frac{m^2}{2} \phi^2 + \frac{\lambda}{4!} \phi^4 \right], \quad \kappa_{\text{UV}} = 10, \quad m^2 = -1.5, \quad \lambda = 1, \quad (\text{D.4})$$

and the Litim regulator in Fourier representation

$$R_k(q^2) = (k^2 - q^2) \Theta(k^2 - q^2). \quad (\text{D.5})$$

We work in the periodic Sobolev space

$$H_{\text{per}}^1([0, 1]) = \{\phi \in H^1([0, 1]) : \phi(0) = \phi(1)\},$$

and use the standard real Fourier basis $\{e_\alpha\}_{\alpha=0}^\infty \subset H_{\text{per}}^1([0, 1])$ given by

$$e_0(x) = 1, \quad e_\alpha(x) = \begin{cases} \sqrt{2} \cos(2\pi p(\alpha) x), & \alpha \text{ odd}, \\ \sqrt{2} \sin(2\pi p(\alpha) x), & \alpha \text{ even}, \alpha > 0, \end{cases}$$

with $p(\alpha) = \lfloor (\alpha + 1)/2 \rfloor$. Any $\phi \in H_{\text{per}}^1([0, 1])$ then has the expansion

$$\phi(x) = \sum_{\alpha=0}^{\infty} c_{\alpha} e_{\alpha}(x), \quad c_{\alpha} = \int_0^1 \phi(x) e_{\alpha}(x) dx.$$

For the numerical experiments we restrict to the finite-dimensional subspace

$$V_P = \left\{ \phi \in H_{\text{per}}^1([0, 1]) : \phi(x) = \sum_{\alpha=0}^{2P} c_{\alpha} e_{\alpha}(x) \right\}, \quad (\text{D.6})$$

of dimension $2P + 1$.

Our Gaussian Process Method

Based on the physical prior, we build our GP prior for Γ for solving (D.3) based on the LPA ansatz (see Appendix D.5 for an introduction of the LPA method)

$$\Gamma_k[\phi] = \int_0^1 dx \left[\frac{1}{2} (\partial_x \phi(x))^2 + U_k(\phi(x)^2) \right], \quad (\text{D.7})$$

More precisely, we place a fully nonparametric prior on the local potential by assuming

$$U_k(u) \sim \mathcal{GP}(0, K_U),$$

a zero-mean GP in the scalar argument $u = \phi^2$. Because $\Gamma_k[\phi]$ depends linearly on U_k , this induces a GP prior on the entire functional $\phi \mapsto \Gamma_k[\phi]$. Under these assumptions,

$$\Gamma_k[\cdot] \sim \mathcal{GP}(m_{\Gamma}, C_{\Gamma}), \quad m_{\Gamma}(\phi) = \frac{1}{2} \int_0^1 (\partial_x \phi)^2 dx, \quad C_{\Gamma}(\phi, \phi') = \int_0^1 \int_0^1 K_U(\phi(x)^2, \phi'(y)^2) dx dy.$$

This construction thus carries the nonparametric GP prior on U_k all the way through to the full functional $\Gamma_k[\phi]$. Moreover, the choice of the scalar kernel K_U is inspired by the form of the bare action in (D.4). In particular, for the ϕ^4 model we adopt the additive kernel

$$K_U(u, u') = 1 + uu' + (uu')^2 + \exp(-|u - u'|^2/(2\sigma^2)). \quad (\text{D.8})$$

Numerical experiment setup

We restrict the fields to the finite-dimensional subspace $V_P \subset H_{\text{per}}^1([0, 1])$ defined in (D.6), spanned by the first $2P + 1$ real Fourier modes. To construct the collocation ensemble, we draw $N_{\text{col}} = 1,000$ random coefficient vectors $c^{(i)} = (c_0^{(i)}, \dots, c_{2P}^{(i)}) \in$

$\mathbb{R}^{2P+1}$ from a correlated Gaussian prior with covariance $\Sigma = A^\top A$ (for a random Gaussian matrix A), and then apply a mild high-frequency decay

$$c_\alpha^{(i)} \leftarrow \frac{c_\alpha^{(i)}}{(1 + |p(\alpha)|)^{3/2}},$$

thereby biasing the sampled fields toward regular configurations while still retaining nontrivial high-frequency modes and improving the conditioning of the computation of the functional Hessian, where $p(\alpha)$ is the Fourier mode index. Each coefficient vector defines a real periodic field

$$\phi^{(i)}(x) = \sum_{\alpha=0}^{2P} c_\alpha^{(i)} e_\alpha(x), \quad i = 1, \dots, N_{\text{col}}.$$

The Gram matrix $K(\Phi, \Phi)$ induced by the scalar kernel K_U in (D.8) is assembled once and stabilized with a small nugget.

For each RG scale κ , the GP posterior defines a surrogate functional $\Gamma_\kappa^\dagger[\phi]$ which approximates $\Gamma_\kappa[\phi]$ for all $\phi \in V_P$. Inserting $\Gamma_\kappa^\dagger$ into the right-hand side of the Wetterich equation (D.3), and projecting onto the collocation ensemble, yields a closed system of ODEs in κ for these coefficients. At the ultraviolet scale κ_{UV} we initialize the flow with the bare action (D.4). Concretely, for each collocation field $\phi^{(i)}$, we compute its bare functional value $\Gamma_{\kappa_{\text{UV}}}[\phi^{(i)}]$, using the real Fourier representation to compute the kinetic term exactly and the potential term by averaging the integrand over the grid. Collecting these values into the vector

$$\Gamma_{\kappa_{\text{UV}}}(\Phi) := (\Gamma_{\kappa_{\text{UV}}}[\phi^{(1)}], \dots, \Gamma_{\kappa_{\text{UV}}}[\phi^{(N_{\text{col}})}])^\top,$$

we treat $\Gamma_{\kappa_{\text{UV}}}(\Phi)$ as the initial condition of the ODE system. We integrate the resulting ODE system backward in scale from $\kappa_{\text{UV}} = 10$ down to $\kappa_{\text{IR}} = 2$. The solver outputs $(\Gamma_\kappa^\dagger[\phi^{(i)}])_{i=1}^{N_{\text{col}}}$ along a uniform grid of RG scales $\{\kappa_\ell\}_{\ell=0}^{N_T} \subset [2, 10]$.

Since no explicit solution is available in this case, we compare the GP surrogate $\Gamma_\kappa^\dagger(\phi)$ with an independently computed LPA solution $U_\kappa(\phi^2)$. The latter is obtained by solving the flow for the local potential on a grid of ϕ^2 values using finite differences and the same Litim regulator (D.5). Concretely, we sample a uniform grid of values $\{\tilde{\phi}^{(j)}\}_{j=1}^{N_{\text{test}}} \subset [0, 4]$ and regard each $\tilde{\phi}^{(j)}$ as a constant test field. In the real Fourier basis, these fields are represented by activating only the constant mode. For the LPA reference, we independently solve the standard LPA flow equation for the local potential $U_\kappa(u)$ (with $u = \phi^2$) using the same bare parameters as in (D.4).

The variable u is discretized on a uniform grid, and the resulting ODE system for $\{U_\kappa(u_j)\}$ is integrated in the RG scale from the ultraviolet cutoff $\kappa_{\text{UV}} = 10$ down to an infrared scale $\kappa_{\text{IR}} = 2$. We compare the LPA potential $U_\kappa(\phi^2)$ with the GP surrogate by evaluating both on the same (κ, ϕ) grid. The comparison between the GP method and LPA in this section is not intended to demonstrate that we can reproduce LPA's results which, of course, are only approximate. Rather, it illustrates the ability of our method to solve the functional on non-constant fields, which lie outside the scope of the LPA approximation, and to generalize the resulting functional approximation to previously unseen fields.

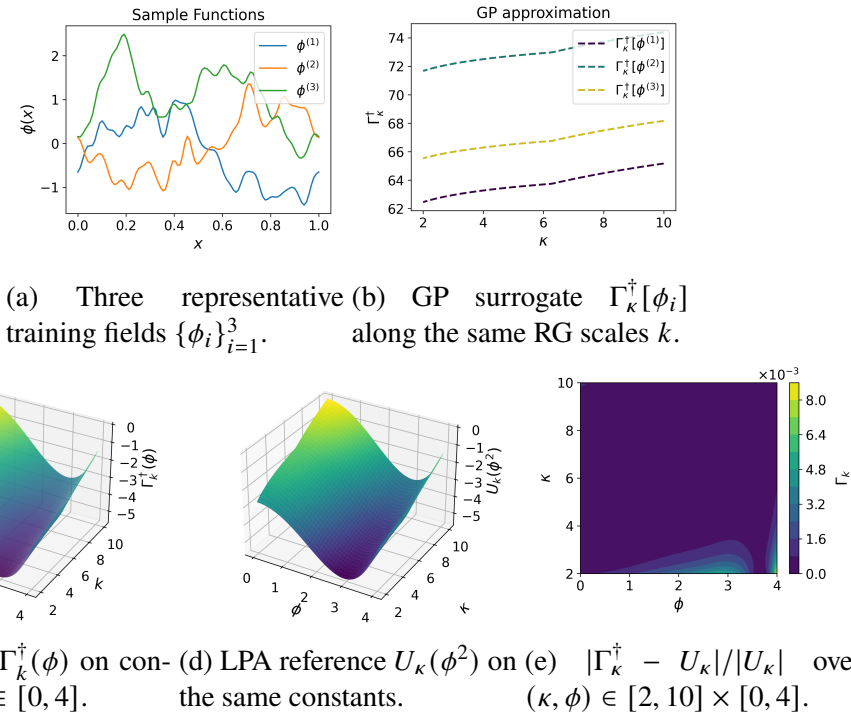


Figure D.2: Numerical results for the Wetterich flow with the ϕ^4 bare action. Panels (A)–(B) illustrate three of the $N_{\text{col}} = 1000$ training fields and the corresponding GP predictions, while panels (C)–(D) plot the surrogate and the LPA potential on constant-field inputs in $[0, 4]$. Panel (E) maps the pointwise relative difference across RG scales $\kappa \in [2, 10]$ and field amplitudes $\phi \in [0, 4]$.

Numerical results and discussion

Figure D.2 summarizes the performance of our GP-based solver for the interacting ϕ^4 Wetterich flow. Panel (A) displays three representative training fields drawn from the collocation ensemble Φ . Panel (B) plots the corresponding GP surrogate evaluations along the RG trajectory.

Panel (C) shows the surface $\Gamma_\kappa^\dagger(\phi)$ for $\kappa \in [2, 10]$ and $\phi \in [0, 4]$, while panel (D) presents the LPA reference potential $U_\kappa(\phi^2)$ evaluated on the same (κ, ϕ) grid. The two surfaces are visually indistinguishable at the scale of the plots, indicating that the functional GP, which is trained on general nonconstant fields, closely tracks the local potential extracted from a dedicated LPA computation when restricted to homogeneous configurations.

Panel (E) quantifies this agreement by plotting the pointwise relative difference between the LPA and GP results on constant fields,

$$\varepsilon(\kappa, \phi) = \frac{|\Gamma_\kappa^\dagger(\phi) - U_\kappa(\phi^2)|}{|U_\kappa(\phi^2)|}, \quad (\kappa, \phi) \in [2, 10] \times [0, 4].$$

Across this entire grid the errors remain in the range 10^{-3} – 10^{-2} .

Taken together, these diagnostics show that the GP prior for Γ , engineered from the LPA ansatz and the ϕ^4 structure of the bare action, yields a surrogate that accurately captures the RG evolution of the effective action while retaining excellent quantitative agreement with the LPA potential on constant fields. The plots in panels (A)–(B) demonstrate accurate interpolation on the training ensemble, while the constant-field comparison in panels (C)–(E) plays the role of an out-of-sample test and confirms generalization of the surrogate across the entire RG flow.

D.3 Mathematical presentation of the FRG flow problem

For a mathematical audience, it is convenient to recall a probabilistic interpretation of the functional renormalization group (FRG), following Hudes and Eyink, 2025. Let ψ be an underlying “true” field and consider a noisy observation

$$\phi = \psi + \epsilon,$$

where ϵ is a Gaussian error with zero mean and covariance matrix R_κ^{-1} or equivalently precision matrix R_κ . We assume that ϵ is independent of ψ . By Bayes’ rule, the conditional law of ψ given the observation ϕ is

$$P[\psi \mid \phi] = \frac{P[\psi] P[\epsilon = \phi - \psi]}{P[\phi]}. \quad (\text{D.9})$$

To encode this conditional distribution, one introduces an external source (or test function) J , which should be thought of as an element of the dual of the field space, and defines the conditional generating functional

$$Z_\kappa[J \mid \phi] := \mathbb{E} \left[\exp \left(\int_x J(x) \psi(x) \, dx \right) \mid \phi \right].$$

The associated conditional cumulant-generating functional is then

$$W_\kappa[J \mid \phi] := \ln Z_\kappa[J \mid \phi].$$

From $W_\kappa[J \mid \phi]$ one defines a scale-dependent functional $\Gamma_\kappa[\phi]$ via a generalized Legendre transform in the source variable J :

$$\Gamma_\kappa[\phi] = \sup_J \left\{ \int_x J(x) \phi(x) dx - W_\kappa[J \mid \phi] \right\}.$$

In the FRG literature, Γ_κ is referred to as the *effective average action*. As the scale parameter κ varies, Γ_κ interpolates between the bare (microscopic) action in the ultraviolet limit $\kappa \rightarrow \infty$, $R_\kappa \rightarrow \infty$ (perfect measurements) and the full effective action in the infrared limit $\kappa \rightarrow 0$, $R_\kappa \rightarrow 0$ (no measurements), which governs the macroscopic behavior. Starting from the probabilistic setup above, one can derive, as shown in Hudes and Eyink, 2025, the Wetterich–Morris flow equation (5.1) for Γ_κ .

D.4 Derivative and Hessian of kernel interpolants

We now discuss the computation of functional derivatives and Hessians of the kernel interpolant (5.12). For a fixed scale κ we suppress the κ -dependence and write the surrogate in the form

$$\Gamma^\dagger[\phi] = \sum_{i=1}^N c_i K(w(\phi), w(\phi^{(i)})), \quad (\text{D.10})$$

where $\phi \in \mathcal{U}$ is a generic field, $\phi^{(i)}$ are the collocation fields, and $w : \mathcal{U} \rightarrow \mathbb{R}^m$ is the feature map introduced in Section 5.2. The vector $c = (c_1, \dots, c_N)^\top$ collects the coefficients appearing in the representer formula, for example $c = (K(Z, Z) + \varepsilon I)^{-1} g_\kappa$ in (5.12). In this appendix we treat c as fixed and focus on the dependence on the field ϕ .

Throughout, we consider Gâteaux derivatives of $\Gamma^\dagger$ along directions $\eta, v \in \mathcal{U}$. We assume that the feature map w is linear, so that $w(\phi + \varepsilon\eta) = w(\phi) + \varepsilon w(\eta)$ for all $\phi, \eta \in \mathcal{U}$ and $\varepsilon \in \mathbb{R}$; this covers the cases of pointwise sampling and Fourier coefficients considered below.

First variation

Fix a collocation field $\phi^{(i)}$ and abbreviate $z = w(\phi) \in \mathbb{R}^m$, $z^{(i)} = w(\phi^{(i)}) \in \mathbb{R}^m$. Consider the directional derivative of a single kernel term,

$$\left. \frac{d}{d\varepsilon} K(w(\phi + \varepsilon\eta), w(\phi^{(i)})) \right|_{\varepsilon=0}.$$

Using the chain rule and linearity of w we obtain

$$\left. \frac{d}{d\varepsilon} K(w(\phi + \varepsilon\eta), w(\phi^{(i)})) \right|_{\varepsilon=0} = \nabla_1 K(z, z^{(i)}) \cdot w(\eta), \quad (\text{D.11})$$

where ∇_1 denotes the gradient of $K(\cdot, z^{(i)})$ with respect to its first argument, and the dot denotes the Euclidean inner product in $\mathbb{R}^m$.

By linearity of the sum in (D.10), the first variation of $\Gamma^\dagger$ at ϕ in the direction η is

$$\left\langle \frac{\delta \Gamma^\dagger}{\delta \phi}, \eta \right\rangle = \left. \frac{d}{d\varepsilon} \Gamma^\dagger[\phi + \varepsilon\eta] \right|_{\varepsilon=0} = \sum_{i=1}^N c_i \nabla_1 K(w(\phi), w(\phi^{(i)})) \cdot w(\eta).$$

This expression for the functional derivative $\delta \Gamma^\dagger / \delta \phi$ will take a concrete form once a specific form of the feature map w is chosen.

Second variation

A similar calculation yields the second variation. Differentiating (D.11) once more in the direction v gives

$$\left. \frac{d}{d\varepsilon} \left[\nabla_1 K(w(\phi + \varepsilon v), w(\phi^{(i)})) \cdot w(\eta) \right] \right|_{\varepsilon=0} = (w(\eta))^\top \nabla_1^2 K(w(\phi), w(\phi^{(i)})) w(v),$$

where $\nabla_1^2 K$ denotes the Hessian of $K(\cdot, z^{(i)})$ with respect to its first argument. Summing over i with weights c_i then yields the bilinear form $\langle \eta, \delta^2 \Gamma^\dagger / (\delta \phi \delta \phi) v \rangle$ for the second variation. As in the first-variation case, this pairing provides an explicit expression for the second-variation kernel $\delta^2 \Gamma^\dagger / (\delta \phi \delta \phi)$ once the feature map w is specified. We now record the resulting formulas for the two choices considered in this paper.

Examples of different measurements

In this subsection, we detail the functional derivative calculations for kernel interpolants in two examples: pointwise measurements and measurements given by Fourier coefficients.

Suppose the feature map is given by pointwise samples at physical collocation points $x_1, \dots, x_m$, so that

$$w(\phi) = (\phi(x_1), \dots, \phi(x_m)) \in \mathbb{R}^m.$$

In this case

$$w(\eta) = (\eta(x_1), \dots, \eta(x_m)),$$

and (D.11) becomes

$$\begin{aligned}\nabla_1 K(w(\phi), w(\phi^{(i)})) \cdot w(\eta) &= \sum_{j=1}^m \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \eta(x_j) \\ &= \int \left[\sum_{j=1}^m \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \delta_{x_j}(x) \right] \eta(x) dx,\end{aligned}$$

where ∂_{z_j} denotes differentiation with respect to the j th component of the first kernel argument and δ_{x_j} is the Dirac measure at x_j . We may therefore identify the functional derivative of a single kernel term as

$$\frac{\delta}{\delta \phi(x)} K(w(\phi), w(\phi^{(i)})) = \sum_{j=1}^m \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \delta_{x_j}(x).$$

Integrals involving $\delta \Gamma^\dagger / \delta \phi$ are thus reduced to evaluations of the kernel gradient at the finite set of spatial points $\{x_j\}_{j=1}^m$.

For the second variation we use the Hessian $\nabla_1^2 K$ and the representation $w(\eta) = (\eta(x_j))_{j=1}^m$. A calculation analogous to the one above yields

$$\begin{aligned}w(\eta)^\top \nabla_1^2 K(w(\phi), w(\phi^{(i)})) w(v) &= \sum_{j,k=1}^m \eta(x_k) \partial_{z_k} \partial_{z_j} K(w(\phi), w(\phi^{(i)})) v(x_j) \\ &= \int \int \eta(y) \left[\sum_{j,k=1}^m \partial_{z_k} \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \delta_{x_k}(y) \delta_{x_j}(x) \right] v(x) dx dy.\end{aligned}$$

Hence the second functional derivative of a single kernel term is the distribution

$$\frac{\delta^2}{\delta \phi(x) \delta \phi(y)} K(w(\phi), w(\phi^{(i)})) = \sum_{j,k=1}^m \partial_{z_k} \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \delta_{x_k}(y) \delta_{x_j}(x).$$

Again, any integral involving the Hessian of $\Gamma^\dagger$ only depends on the physical collocation points $\{x_j\}$ and on the kernel Hessian evaluated at the feature vector $w(\phi)$.

Next, we consider the case where w records Fourier coefficients. For simplicity, let $\{e_j\}_{j=1}^m$ be a finite family of (complex or real) Fourier modes on the spatial domain and set

$$w(\phi) = (\hat{\phi}_j)_{j=1}^m, \quad \hat{\phi}_j := \int \phi(x) e_j(x) dx.$$

Then

$$w(\eta) = (\hat{\eta}_j)_{j=1}^m, \quad \hat{\eta}_j := \int \eta(x) e_j(x) dx.$$

Proceeding as in the pointwise case, we obtain from (D.11)

$$\begin{aligned}\nabla_1 K(w(\phi), w(\phi^{(i)})) \cdot w(\eta) &= \sum_{j=1}^m \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \hat{\eta}_j \\ &= \int \left[\sum_{j=1}^m \partial_{z_j} K(w(\phi), w(\phi^{(i)})) e_j(x) \right] \eta(x) dx.\end{aligned}$$

Thus

$$\frac{\delta}{\delta\phi(x)} K(w(\phi), w(\phi^{(i)})) = \sum_{j=1}^m \partial_{z_j} K(w(\phi), w(\phi^{(i)})) e_j(x).$$

For the second derivative we obtain, by the same argument,

$$\begin{aligned}w(\eta)^\top \nabla_1^2 K(w(\phi), w(\phi^{(i)})) w(v) &= \sum_{j,k=1}^m \hat{\eta}_k \partial_{z_k} \partial_{z_j} K(w(\phi), w(\phi^{(i)})) \hat{v}_j \\ &= \int \int \eta(y) \left[\sum_{j,k=1}^m \partial_{z_k} \partial_{z_j} K(w(\phi), w(\phi^{(i)})) e_k(y) e_j(x) \right] v(x) dx dy,\end{aligned}$$

so that

$$\frac{\delta^2}{\delta\phi(x) \delta\phi(y)} K(w(\phi), w(\phi^{(i)})) = \sum_{j,k=1}^m \partial_{z_k} \partial_{z_j} K(w(\phi), w(\phi^{(i)})) e_k(y) e_j(x).$$

In both the pointwise and Fourier settings, the functional derivatives and Hessians of $\Gamma^\dagger$ are obtained by summing the corresponding expressions over $i = 1, \dots, N$ with weights c_i , as in (D.10).

D.5 Local Potential Approximation (LPA)

We consider the effective average action $\Gamma_\kappa[\phi]$ satisfying the Wetterich flow

$$\partial_\kappa \Gamma_\kappa[\phi] = \frac{1}{2} \text{Tr} \left[(\Gamma_\kappa^{(2)}[\phi] + R_\kappa)^{-1} \partial_\kappa R_\kappa \right], \quad \Gamma_\kappa^{(2)}[\phi](x, y) = \frac{\delta^2 \Gamma_\kappa[\phi]}{\delta\phi(x) \delta\phi(y)},$$

where the renormalization scale κ flows backward from a large terminal value κ_{UV} to an infrared value κ_{IR} , and R_κ is a symmetric, translation-invariant regulator.

In the local potential approximation (LPA), one assumes an ansatz with a standard quadratic gradient term encoding all scale dependence in a spatially local potential Delamotte, 2012. Concretely, on a periodic continuum domain $\Omega \subset \mathbb{R}$ (for example $[0, 1]$ with periodic boundary conditions)

$$\Gamma_\kappa[\phi] = \int_\Omega \left(\frac{1}{2} |\partial_x \phi(x)|^2 + U_\kappa(\phi(x)) \right) dx,$$

and on a periodic lattice with N sites, spacing h , and index set $\mathbb{T}_N$, one sets

$$\Gamma_\kappa[\phi] = \sum_{i \in \mathbb{T}_N} \left(\frac{1}{2} |(\nabla_h \phi)_i|^2 + U_\kappa(\phi_i) \right) h, \quad (\nabla_h \phi)_i := \frac{\phi_{i+1} - \phi_i}{h}.$$

The scalar function U_κ is the scale-dependent effective potential to be determined. At a constant configuration $\phi(x) \equiv \varphi \in \mathbb{R}$, the second variational derivative takes the form

$$\Gamma_\kappa^{(2)}[\varphi] = -\Delta + U_\kappa''(\varphi) I \quad (\text{continuum}), \quad \Gamma_\kappa^{(2)}[\varphi] = -\Delta_h + U_\kappa''(\varphi) I \quad (\text{lattice}),$$

where $-\Delta$ is the Laplacian on Ω with periodic boundary conditions and $-\Delta_h$ is the standard second-order discrete Laplacian.

Because the regulator is translation-invariant, both $\Gamma_\kappa^{(2)}[\varphi]$ and R_κ are diagonal in a Fourier basis. We therefore choose a real orthonormal eigenbasis of the (continuous or discrete) Laplacian,

$$-\Delta e_{\alpha,j} = \lambda_\alpha e_{\alpha,j} \quad \text{or} \quad -\Delta_h e_{\alpha,j} = \lambda_\alpha e_{\alpha,j},$$

where $\alpha \in \mathcal{I}$ labels the distinct eigenvalues $\lambda_\alpha \geq 0$, and $j = 1, \dots, g_\alpha$ runs over an orthonormal basis of the corresponding eigenspace of multiplicity $g_\alpha \in \mathbb{N}$.

By translation invariance, the regulator acts in mode:

$$R_\kappa e_{\alpha,j} = r_\kappa(\lambda_\alpha) e_{\alpha,j}, \quad (\partial_\kappa R_\kappa) e_{\alpha,j} = \partial_\kappa r_\kappa(\lambda_\alpha) e_{\alpha,j},$$

for some scalar profile $\lambda \mapsto r_\kappa(\lambda)$. At a constant field φ , the operator inside the trace in the Wetterich equation is diagonal in the same basis:

$$(\Gamma_\kappa^{(2)}[\varphi] + R_\kappa) e_{\alpha,j} = (\lambda_\alpha + U_\kappa''(\varphi) + r_\kappa(\lambda_\alpha)) e_{\alpha,j},$$

and hence

$$(\Gamma_\kappa^{(2)}[\varphi] + R_\kappa)^{-1} e_{\alpha,j} = \frac{1}{\lambda_\alpha + r_\kappa(\lambda_\alpha) + U_\kappa''(\varphi)} e_{\alpha,j}.$$

Therefore, the trace reduces to a sum over distinct eigenvalues,

$$\text{Tr} \left[(\Gamma_\kappa^{(2)}[\varphi] + R_\kappa)^{-1} \partial_\kappa R_\kappa \right] = \sum_{\alpha \in \mathcal{I}} g_\alpha \frac{\partial_\kappa r_\kappa(\lambda_\alpha)}{\lambda_\alpha + r_\kappa(\lambda_\alpha) + U_\kappa''(\varphi)}.$$

To treat the continuum and lattice cases uniformly, we work with the potential per unit volume (continuum) or per site (lattice). For a constant field φ , define

$$U_\kappa(\varphi) := \frac{1}{\text{vol}(\Omega)} \Gamma_\kappa[\phi \equiv \varphi], \quad \text{vol}(\Omega) := \begin{cases} |\Omega|, & \text{continuum,} \\ N, & \text{lattice,} \end{cases}$$

so that U_κ always denotes the effective potential normalized by the size of the domain. With this convention,

$$\partial_\kappa U_\kappa(\varphi) = \frac{c_\Omega}{2} \sum_{\alpha \in \mathcal{I}} g_\alpha \frac{\partial_\kappa r_\kappa(\lambda_\alpha)}{\lambda_\alpha + r_\kappa(\lambda_\alpha) + U''_\kappa(\varphi)}, \quad c_\Omega := \frac{1}{\text{vol}(\Omega)}.$$

This is the LPA equation. It applies to both settings; in particular $c_\Omega = 1$ on $[0, 1]$ and $c_\Omega = 1/N$ on a lattice when U_κ is defined per site.

A commonly used choice is the Litim profile, defined modewise by

$$r_\kappa(\lambda) = (\kappa^2 - \lambda)_+, \quad \partial_\kappa r_\kappa(\lambda) = 2\kappa \mathbf{1}_{\{\lambda < \kappa^2\}}.$$

On the *active* set $\{\lambda_\alpha < \kappa^2\}$ the denominator simplifies to $\kappa^2 + U''_\kappa(\varphi)$. Introducing the active-mode count

$$S(\kappa) := \sum_{\lambda_\alpha < \kappa^2} g_\alpha,$$

the equation collapses to

$$\partial_\kappa U_\kappa(\varphi) = c_\Omega \kappa \frac{S(\kappa)}{\kappa^2 + U''_\kappa(\varphi)}.$$

Integrating this equation backward from κ_{UV} to κ_{IR} yields the infrared potential $U_{\kappa_{\text{IR}}}$, which can then be used to compute physical observables.

D.6 Transfer Matrix Approach

We outline the transfer-matrix computation used to obtain reference solutions for the one-dimensional ϕ^4 model studied in Section 5.3. We consider a one-dimensional periodic lattice with N_x sites and lattice spacing $a > 0$. A field configuration is $\phi = (\phi_1, \dots, \phi_{N_x}) \in \mathbb{R}^{N_x}$, and the bare action in the presence of an external source $c = (c_x)_{x=1}^{N_x}$ is

$$S[\phi; c] = a \sum_{x=1}^{N_x} \left[\frac{1}{2} \left(\frac{\phi_{x+1} - \phi_x}{a} \right)^2 + U(\phi_x; c_x) \right], \quad \phi_{N_x+1} = \phi_1,$$

with local potential

$$U(\phi; c_x) := \frac{1}{2} m^2 \phi^2 + \frac{\lambda}{24} \phi^4 - c_x \phi.$$

The partition function and cumulant-generating functional with $\mathcal{D}\phi := \prod_{x=1}^{N-x} d\phi_x$ are

$$Z[c] := \int e^{-S[\phi; c]} \mathcal{D}\phi, \quad W[c] := \ln Z[c].$$

To obtain a finite-dimensional representation, we discretise the range of field values. We choose an interval $[-\phi_{\max}, \phi_{\max}]$ and a grid

$$\{\phi^{(1)}, \dots, \phi^{(N_\phi)}\} \subset [-\phi_{\max}, \phi_{\max}],$$

and restrict the field at each site to this grid,

$$\phi_x \in \{\phi^{(1)}, \dots, \phi^{(N_\phi)}\}, \quad x = 1, \dots, N_x.$$

For this discretized model the path integral becomes a finite sum and can be written exactly in terms of a transfer matrix. We comment that this discretization may introduce inaccuracies when $|\phi| > \phi_{\max}$ but is chosen here for consistency with the kernel approximation.

The transfer matrix T_x encodes the Boltzmann weight associated with the nearest-neighbour pair between sites x and $x + 1$. It is an $N_\phi \times N_\phi$ matrix with entries

$$(T_x)_{ij} := \exp \left[-\frac{a}{2} \left(\left(\frac{\phi^{(j)} - \phi^{(i)}}{a} \right)^2 + U(\phi^{(i)}; c_x) + U(\phi^{(j)}; c_{x+1}) \right) \right], \quad i, j = 1, \dots, N_\phi. \quad (\text{D.12})$$

The kinetic term $(\phi_{x+1} - \phi_x)^2$ couples neighbouring field values $\phi^{(i)}$ and $\phi^{(j)}$, while the local potentials at sites x and $x + 1$ are split symmetrically between the two ends of the bond. The partition function of the discretised model is

$$Z[c] = \text{Tr}(T_1 T_2 \cdots T_{N_x}), \quad W[c] = \ln \text{Tr}(T_1 T_2 \cdots T_{N_x}).$$

Let

$$\phi_x[c] := \langle \phi_x \rangle_c$$

denote the expectation value of the field at site x in the Gibbs measure with weight $e^{-S[\phi; c]}$. Since the source couples as $-c_x \phi_x$, we have

$$\frac{\delta W[c]}{\delta c_x} = \langle \phi_x \rangle_c = \phi_x[c].$$

In the transfer-matrix representation, introduce the diagonal matrix

$$\Phi_{ij} := \phi^{(i)} \delta_{ij}, \quad i, j = 1, \dots, N_\phi,$$

which represents multiplication by the local field value on field-space indices. Differentiating (D.12) gives

$$\frac{\partial T_x}{\partial c_x} = a \Phi T_x.$$

Using $Z[c] = \text{Tr}(T_1 \cdots T_{N_x})$ and cyclicity of the trace, we obtain

$$\phi_x[c] = \frac{\delta W[c]}{\delta c_x} = \frac{\text{Tr}(\Phi T_1 \cdots T_{N_x})}{\text{Tr}(T_1 \cdots T_{N_x})}. \quad (\text{D.13})$$

Here Φ is an operator (a diagonal matrix); the scalar $\phi_x[c]$ is obtained after taking the trace.

For the observables used in this work (magnetization and susceptibility), it suffices to consider homogeneous sources $c_x \equiv c$. Then all transfer matrices coincide, $T_x = T(c)$, and translation invariance implies that $\phi_x[c]$ is independent of x . We denote this common value by $\bar{\phi}(c)$ and identify it with the magnetisation per site,

$$m(c) := \frac{1}{N_x} \sum_{x=1}^{N_x} \langle \phi_x \rangle_c = \bar{\phi}(c) = \frac{\text{Tr}(\Phi T(c)^{N_x})}{\text{Tr}(T(c)^{N_x})}.$$

Thus, in our notation the magnetisation per site is precisely the expectation value of the local field.

The effective action $\Gamma[\phi]$ is the Legendre transform of $W[c]$,

$$\Gamma[\phi] = \sum_{x=1}^{N_x} c_x \phi_x[c] - W[c], \quad \phi_x[c] = \frac{\delta W[c]}{\delta c_x}.$$

For a homogeneous configuration $\phi_x \equiv \phi$ and $c_x \equiv c$, the effective potential

$$U(\phi) := \frac{\Gamma[\phi]}{N_x}$$

is the central quantity used to compute the susceptibility. In the double limit $N_\phi \rightarrow \infty$ and $\phi_{\max} \rightarrow \infty$, with a correspondingly refined field-space grid, $U(\phi)$ converges to the exact effective potential of the continuum model.

If (D.12) is used directly in numerical computations, overflow can occur for large homogeneous sources c . To avoid this, we use an equivalent transfer matrix based on a shifted local potential. Fix a reference field value ϕ_0 and introduce

$$U_2(\phi) := \frac{1}{2} m^2 \phi^2 + \frac{\lambda}{24} \phi^4 - c(\phi - \phi_0). \quad (\text{D.14})$$

The corresponding transfer matrix is

$$(T_2)_{ij} := \exp \left[-\frac{a}{2} \left(\left(\frac{\phi^{(j)} - \phi^{(i)}}{a} \right)^2 + U_2(\phi^{(i)}) + U_2(\phi^{(j)}) \right) \right]. \quad (\text{D.15})$$

This shifted formulation is algebraically equivalent to the original one but is numerically more stable for large $|c|$.

D.7 Details of the GP representation for the Wetterich flow

In this section, we present the implementation details for solving the Wetterich equation on the lattice in Section 5.3.

GP parametrization of the local potential

In our numerical implementation, the scale-dependent local potential U_κ on a field window $[-\phi_{\max}, \phi_{\max}]$ is modeled on the continuum as

$$U_\kappa(\phi) = m_0(\phi) + v_\kappa(\phi) + g_\kappa(\phi), \quad \phi \in [-\phi_{\max}, \phi_{\max}], \quad (\text{D.16})$$

with three distinct contributions. The baseline

$$m_0(\phi) := \frac{1}{2}m^2\phi^2 + \frac{\lambda}{24}\phi^4$$

is the bare lattice ϕ^4 potential, fixed for all κ . The parametric correction

$$v_\kappa(\phi) = \text{softplus}(\theta_{2,\kappa}) \phi^2/2 + \text{softplus}(\theta_{4,\kappa}) \phi^4/12$$

tracks the running of the quadratic and quartic couplings through a latent parameter vector $\theta_\kappa = (\theta_{2,\kappa}, \theta_{4,\kappa})^\top \in \mathbb{R}^2$. The map $\text{softplus}(x) = \log(1 + e^x)$ is smooth and strictly positive, so that the effective couplings in v_κ are positive by construction. The remainder

$$g_\kappa \sim \mathcal{GP}(0, K_U)$$

is a zero-mean GP with a fixed covariance kernel K_U on $[-\phi_{\max}, \phi_{\max}]$, given concretely below. The latter term captures non-polynomial and higher-order structure that is not represented by the quadratic-quartic component.

A direct use of (D.16) in the Wetterich equation (5.20) would proceed as follows. At each RG scale κ , the flow provides the values of U_κ at a set of field points, and from these data one would infer the latent parameters θ_κ and the GP remainder g_κ that best explain the current profile. The curvature U_κ'' entering (5.20) is then evaluated from the reconstructed U_κ . In practice, this procedure requires solving, at each scale, a nonlinear constrained estimation problem for θ_κ coupled to GP interpolation conditions for g_κ . This is too expensive for long RG trajectories.

To obtain a stable and efficient solver, we use an interleaved predictor-projector scheme: on each recorded scale interval we (i) advance the flow using a linear-Gaussian surrogate that yields an explicit curvature closure, and then (ii) project the predicted profile onto the nonlinear constrained GP model. The projection replaces the predicted state and is used as the initial condition for the next predictor step.

We discretize the field window $[-\phi_{\max}, \phi_{\max}]$ at N_ϕ points $\{\phi_j\}_{j=1}^{N_\phi}$ and collect the sampled potentials in the vector

$$Y_\kappa := (U_\kappa(\phi_j))_{j=1}^{N_\phi} \in \mathbb{R}^{N_\phi}.$$

We solve the Wetterich equation (5.20) using the local ansatz (5.22) for Γ_κ . The resulting ODE is integrated from κ_{UV} down to κ_{IR} and recorded on a discrete scale grid,

$$\{(\kappa_s, Y_s)\}_{s=0}^S, \quad \kappa_0 = \kappa_{\text{UV}} > \cdots > \kappa_S = \kappa_{\text{IR}},$$

with $Y_s = (U_{\kappa_s}(\phi_j))_{j=1}^{N_\phi}$.

Prediction: linear-Gaussian surrogate

On each interval $[\kappa_s, \kappa_{s+1}]$ we first advance the flow with a linear-Gaussian surrogate. Concretely, we relax the softplus parametrization and write

$$v_\kappa(\phi) = a_{2,\kappa} \phi^2/2 + a_{4,\kappa} \phi^4/12, \quad a_\kappa := \begin{bmatrix} a_{2,\kappa} \\ a_{4,\kappa} \end{bmatrix} \in \mathbb{R}^2,$$

and place a Gaussian prior on the coefficients, $a_\kappa \sim \mathcal{N}(0, \Sigma_a)$, where Σ_a is a fixed 2×2 covariance matrix (diagonal and small in the experiments). The remainder is modeled as

$$g_\kappa \sim \mathcal{GP}(0, K_U),$$

independent of a_κ , where K_U denotes the kernel used in the surrogate. Introducing the feature map

$$f(\phi) := (\phi^2/2, \phi^4/12)^\top,$$

the surrogate GP model reads

$$U_\kappa(\phi) = m_0(\phi) + f(\phi)^\top a_\kappa + g_\kappa(\phi). \quad (\text{D.17})$$

Under these assumptions, the induced prior on U_κ in the prediction step is

$$U_\kappa \sim \mathcal{GP}(m_0, K^{\text{I}}), \quad K^{\text{I}}(\phi, \phi') := f(\phi)^\top \Sigma_a f(\phi') + K_U(\phi, \phi'). \quad (\text{D.18})$$

This model is Gaussian in the latent variables (a_κ, g_κ) and leads to a linear dependence of the curvature U''_κ on the potential samples Y_κ . Integrating the resulting closed ODE from κ_s to κ_{s+1} produces a predicted profile, which we denote by $\tilde{Y}_{s+1}$.

Correction: constrained softplus GP reconstruction

At the correction step, we return to the nonlinear GP model (D.16) and use the predictor output $\tilde{Y}_{s+1}$ as data to reconstruct a constrained representation of $U_{\kappa_{s+1}}$. Recall that the discrete profile collects the sampled values

$$(Y_{s+1})_j = U_{\kappa_{s+1}}(\phi_j), \quad j = 1, \dots, N_\phi,$$

on the fixed field grid $X := \{\phi_j\}_{j=1}^{N_\phi} \subset [-\phi_{\max}, \phi_{\max}]$. At each scale κ_{s+1} we adopt the nonlinear GP model

$$U_{\kappa_{s+1}}(\phi) = m_0(\phi) + \text{softplus}(\theta_{2,\kappa_{s+1}}) \phi^2/2 + \text{softplus}(\theta_{4,\kappa_{s+1}}) \phi^4/12 + g_{\kappa_{s+1}}(\phi),$$

where $(\theta_{2,\kappa_{s+1}}, \theta_{4,\kappa_{s+1}})^\top \in \mathbb{R}^2$ are latent parameters and

$$g_{\kappa_{s+1}} \sim \mathcal{GP}(0, K_U).$$

The GP remainder $g_{\kappa_{s+1}}$ captures the nonpolynomial structure that cannot be represented by a quartic polynomial. For a fixed $\theta_{\kappa_{s+1}}$, the polynomial part is deterministic, and we interpret the residual at the grid points,

$$r_{\theta_{\kappa_{s+1}}}(\phi_j) := (\tilde{Y}_{s+1})_j - m_0(\phi_j) - \text{softplus}(\theta_{2,\kappa_{s+1}}) \phi_j^2/2 - \text{softplus}(\theta_{4,\kappa_{s+1}}) \phi_j^4/12,$$

as observations of $g_{\kappa_{s+1}}$ at ϕ_j . Standard GP regression with kernel K_U then yields a posterior mean $g_{\kappa_{s+1},\text{post}}(\cdot; \theta_{\kappa_{s+1}})$, and the size of the implied correction is quantified by the squared RKHS norm $\|g_{\kappa_{s+1},\text{post}}(\cdot; \theta_{\kappa_{s+1}})\|_{K_U}^2$.

We determine $\theta_{\kappa_{s+1}}$ by solving the two-dimensional optimization problem

$$\min_{\theta_{\kappa_{s+1}} \in \mathbb{R}^2} \frac{1}{2} \|g_{\kappa_{s+1},\text{post}}(\cdot; \theta_{\kappa_{s+1}})\|_{K_U}^2 + \frac{\gamma}{2} \|\theta_{\kappa_{s+1}} - \theta_{\kappa_s}\|_2^2, \quad (\text{D.19})$$

where $\gamma > 0$ is a small regularization parameter and θ_{κ_s} denotes the parameter obtained at the previous scale. In practice, (D.19) is minimized at each scale by a damped Gauss–Newton iteration. The resulting minimizer $\theta_{\kappa_{s+1}}$ and posterior mean $g_{\kappa_{s+1},\text{post}}$ define the corrected profile Y_{s+1} , which is then used as the initial condition for the next prediction step.

In particular, the infrared parameters $\theta_{\text{IR}} := \theta_{\kappa_s}$ and remainder $g_{\text{IR}} := g_{\kappa_s,\text{post}}$ define the reconstructed infrared potential

$$U_{\text{IR}}(\phi) = m_0(\phi) + \text{softplus}(\theta_{\text{IR},2}) \phi^2/2 + \text{softplus}(\theta_{\text{IR},4}) \phi^4/12 + g_{\text{IR}}(\phi),$$

from which all physical observables, including the magnetization and susceptibility used in Section 5.3, are subsequently computed.

D.8 Magnetization and Susceptibility

In the infrared, the scalar effective potential $U_{\kappa_{\text{IR}}}$ encodes equilibrium thermodynamics for spatially homogeneous configurations. Introducing a homogeneous external source c that couples linearly to the field selects a preferred orientation and probes the system's linear response. The *magnetization* $m(c)$ is the thermodynamic order parameter conjugate to c ; it captures symmetry breaking and phase behavior and is defined as the equilibrium field that minimizes the tilted potential

$$\mathcal{F}_c(\phi) := U_{\kappa_{\text{IR}}}(\phi) - c\phi, \quad m(c) := \arg \min_{\phi \in \mathbb{R}} \mathcal{F}_c(\phi).$$

Whenever $U_{\kappa_{\text{IR}}}$ is twice differentiable and strictly convex at $m(c)$, first-order optimality yields the equivalent scalar equation

$$U'_{\kappa_{\text{IR}}}(m(c)) = c, \quad U''_{\kappa_{\text{IR}}}(m(c)) > 0.$$

The *susceptibility* $\chi(c)$ quantifies the linear response of the order parameter to the source and is given by

$$\chi(c) := \frac{dm(c)}{dc} = \frac{1}{U''_{\kappa_{\text{IR}}}(m(c))}.$$

These two quantities are standard diagnostics in statistical physics: $m(c)$ tracks the emergence of spontaneous order, while $\chi(c)$ measures fluctuations and exhibits enhancement near criticality (through small curvature $U''_{\kappa_{\text{IR}}}$). They are also natural validation targets for our reconstruction of $U_{\kappa_{\text{IR}}}$, because they depend on both its shape and its curvature and admit high-quality lattice Monte Carlo and transfer-matrix benchmarks for direct comparison.

Appendix E

CHAPTER 6

E.1 Gaussian process benchmarks

The Gaussian process (GP) benchmarks in Table 6.1 are based on Batlle, Matthieu Darcy, Hosseini, et al., 2024c. The GP measurements are given by either pointwise measurements or projection onto PCA coefficients, and the intermediate vector valued map is learned using a Gaussian process with a linear combination of a Matérn and dot product kernels. All hyper-parameters (number of PCA coefficients, length-scale, regularization...) are learned via five-fold cross validation on the training set and optimized using Optuna Akiba et al., 2019b.

E.2 Nested Kriging

Nested Kriging is a hierarchical extension of Gaussian process regression designed to handle very large datasets that cannot be represented efficiently by a single GP. The idea is to train separate specialized GP models (experts) on subsets of the data or related problems, and then combine (aggregate) their predictions using another GP or a statistically optimal weighting scheme. This aggregation step, often based on minimizing the overall prediction variance, produces a global predictor that retains much of the accuracy of a full GP model while drastically reducing computational and memory costs.

More formally, let $M_1(x), M_2(x), \dots, M_p(x) \in \mathbb{R}^m, x \in \mathbb{R}^n$ be a set of $p \in \mathbb{N}$ experts that are described by the same underlying Gaussian process $Y(x)$ with kernel k , i.e., there holds

$$M_i(x) = k(x, X_i)k(X_i, X_i)^{-1}Y(X_i), \quad i = 1, 2, \dots, p.$$

Furthermore, define $M(x) = (M_1(x), M_2(x), \dots, M_p(x))^T$ as well as the covariance matrix $K_M(X) = \text{Cov}(M(x), M(x))$ and the vector $k_M(x) = \text{Cov}(Y(x), M(x))$. Then, the aim is to minimize the variance

$$E[(Y(x) - \alpha^T M(x))^2] = k(x, x) - 2\alpha^T k_M(x) + \alpha^T K_M(x)\alpha$$

with respect to the aggregation vector α . The solution to this mean squared problem is given by the

$$\alpha^* = K_M(x)^{-1}k_M(x)$$

leading to the unconstrained best linear unbiased predictor (BLUP) given by

$$M_{\mathcal{A}}(x) = K_M(x)^{-1}k_M(x)M(x).$$

However, we aim to aggregate our models by weighting the results of the individual experts by incorporating the constraint $\sum_{i \in A} \alpha_i(x) = \mathbf{1}^T \alpha(x) = 1$ for all $x \in \mathbb{R}^n$, where $\mathbf{1}$ is a vector of ones with the same dimension as α . The constrained optimization problem is then solved via Lagrange multipliers. The corresponding Lagrange function is given by

$$\begin{aligned} \mathcal{L}(\alpha, \lambda) &= E[(Y(x) - \alpha^T M(x))^2] + \lambda(\mathbf{1}^T \alpha - 1) \\ &= k(x, x) - 2\alpha^T k_M(x) + \alpha^T K_M(x) \alpha + \lambda(\mathbf{1}^T \alpha - 1), \end{aligned} \quad (\text{E.1})$$

where λ is the Lagrange multiplier associated with the constraint $\mathbf{1}^T \alpha = 1$. This gives the following result proved in Section E.8

Proposition 18. *The Lagrange function given by (E.1) is minimized by*

$$\alpha = K_M(x)^{-1} \left(k_M(x) - \frac{\mathbf{1} (\mathbf{1}^T K_M(x)^{-1} k_M(x) - 1)}{\mathbf{1}^T K_M(x)^{-1} \mathbf{1}} \right). \quad (\text{E.2})$$

Proof of Proposition 18. To minimize $\mathcal{L}(\alpha, \lambda)$, we take the derivative of $\mathcal{L}$ with respect to α and λ , and set them equal to zero. This gives the equations:

$$K_M(x) \alpha = k_M(x) - \frac{\lambda}{2} \mathbf{1}, \quad \mathbf{1}^T \alpha = 1.$$

Substituting the α in the first equation into the second equation, solving for λ , and substituting λ back into the first equation gives the new values of α that satisfy the constraint $\sum_{i \in A} \alpha_i(x) = 1$. This finally yields the desired formula (E.2). $\square$

By enforcing the constraint $\sum_{i \in A} \alpha_i(x) = 1$, the coefficients $\alpha_i(x)$ are adjusted from the unconstrained BLUP solution, and the Lagrange multiplier method gives us the corrected formula. This ensures that the coefficients sum to 1 while still minimizing the mean squared error in the best linear unbiased way.

Computational complexity

The computational cost of the Nested Kriging can be divided into two parts: the offline training of the individual experts and the online prediction (aggregation). During training, each expert M_i requires the inversion (or Cholesky factorization) of its covariance matrix $K_i \in \mathbb{R}^{n_i \times n_i}$, resulting in a computational cost of order $\mathcal{O}(n_i^3)$ and a memory requirement of $\mathcal{O}(n_i^2)$. Training all p experts therefore costs $\mathcal{O}\left(\sum_{i=1}^p n_i^3\right)$, which is significantly cheaper than training a single global Gaussian

process on all $N = \sum_{i=1}^p n_i$ data points, whose cost would be $O(N^3)$. For experts of equal size ($n_i = n$), this corresponds to a reduction by a factor of approximately p^2 .

At prediction time, computing the prediction $M_{\mathcal{A}}(x)$ for a new input x involves two main steps. First, for each expert, we compute the predictive weights $v_i(x) = K_i^{-1}k(X_i, x)$, which requires $O(n_i^2)$ operations per expert, yielding a total cost of $O\left(\sum_{i=1}^p n_i^2\right)$. These vectors are then used to construct the aggregated covariance terms: $k_M(x) \in \mathbb{R}^p$ and $K_M(x) \in \mathbb{R}^{p \times p}$, where forming all pairwise covariances typically costs $O(N^2)$ if cross-covariances between experts are precomputed. Finally, the aggregation weights $\alpha(x)$ are obtained by inverting $K_M(x)$, which adds an additional $O(p^3)$ cost. Consequently, the total online prediction complexity per test point is of order $O(N^2 + p^3)$, compared to $O(N^2)$ for a single global Gaussian process.

Hence, Nested Kriging achieves a substantial reduction in offline training cost—from cubic in the total dataset size to the sum of the individual cubic costs—while maintaining comparable prediction complexity and offering improved scalability with respect to both memory and computation.

E.3 Choice of ordering in the Cholesky Factorization

The Cholesky factors of Q depend on the ordering of its rows and columns, which corresponds to the degrees of freedom in $\mathbb{R}^N$. While this ordering can be inherited from the discretization of the space $\mathcal{V}$, the accuracy of the approximation in (6.9) can be significantly improved by selecting an ordering that induces a hierarchical or multiresolution structure in the operator problem Houman Owhadi, 2017; Houman Owhadi and Scovel, 2019c. To describe this, consider first the case where $\mathcal{F}$ is a local differential operator (e.g., a PDE) acting on functions defined over a domain $\Omega \subset \mathbb{R}^d$. Here, $\frac{\delta \mathcal{F}}{\delta v}^* \frac{\delta \mathcal{F}}{\delta v} + \lambda I$ can be interpreted as a (discretized) elliptic PDE, and its inverse, Θ , represents a (discretized) Green’s function. Suppose the discretization of $\mathcal{F}$ is obtained using a numerical method (e.g., finite element, finite difference, or collocation), with elements centered around points $x_1, \dots, x_N \in \Omega$. Although the Cholesky factors of Θ are generally dense when using a lexicographic ordering, they exhibit significant sparsity when a *max-min ordering* is used. As presented in Schafer, Sullivan, and Houman Owhadi, 2021, a max-min ordering π of $\{1, \dots, N\}$ is defined such that

$$\pi(1) = \operatorname{argmax}_i \operatorname{dist}(x_i, \partial\Omega),$$

and for $i \geq 1$:

$$\pi(i+1) = \operatorname{argmax}_j \operatorname{dist}(x_j, \partial\Omega \cup \{x_{\pi(1)}, \dots, x_{\pi(i)}\}).$$

Writing Π for the permutation matrix associated with π and

$$\Pi^T L^\dagger L^{\dagger,T} \Pi = \Theta$$

for the exact Cholesky factorization of Θ in the maxmin ordering, those Cholesky factors have two desirable properties Schafer, Sullivan, and Houman Owahdi, 2021; Schafer and Houman Owahdi, 2021; Y. Chen, Houman Owahdi, and Schafer, 2024b: (1) they are rank revealing, and (2) they are sparse.

Rank-Revealing Property

The matrix $L^{\dagger,(k)}$, defined as the truncation of $L^\dagger$ to its first k columns (with all remaining columns set to zero), provides a near-optimal rank- k approximation (measured in operator norm) within a constant factor Schafer, Sullivan, and Houman Owahdi, 2021, Thm. 2.3. Specifically, it satisfies

$$\|\Theta - \Pi^T L^{\dagger,(k)} L^{\dagger,(k),T} \Pi\|_{\text{Fro}} \lesssim \inf_{M \text{ of rank } k} \|\Theta - M\|_{\text{Fro}},$$

where $\|\cdot\|_{\text{Fro}}$ denotes the Frobenius matrix norm. This rank-revealing property is (a) analogous to the properties achieved with gamblets or operator-adapted wavelets Houman Owahdi, 2017; Houman Owahdi and Scovel, 2019c for the compression of elliptic PDEs. (b) more desirable for operator-learning than mesh invariance because it induces, within a constant factor, an optimal approximation of the underlying operator at any level of truncation/discretization which has been the core objective of numerical homogenization Altmann, Henning, and Peterseim, 2021.

Sparsity

The Cholesky factors $L^\dagger$ exhibit exponential decay in their entries, as shown in Schafer, Sullivan, and Houman Owahdi, 2021, Thm. 5.23. Specifically:

$$|L_{i,j}^\dagger| \lesssim \exp\left(-C \frac{\operatorname{dist}(x_{\pi(i)}, x_{\pi(j)})}{h}\right),$$

where h is the mesh norm associated with $\{x_1, \dots, x_N\}$, and C is some positive constant. To describe the sparsity structure, let $l_1 := \max_i \operatorname{dist}(x_i, \partial\Omega)$ and, for $i \geq 1$,

$$l_{i+1} := \max_j \operatorname{dist}(x_j, \partial\Omega \cup \{x_{\pi(1)}, \dots, x_{\pi(i)}\}),$$

represent the sequence of decreasing distances associated with the max-min ordering. Given $\rho \in \mathbb{N}$, the sparsity set is defined as

$$S_\rho := \{(i, j) \in \{1, \dots, N\}^2 \mid i \geq j \text{ and } \text{dist}(x_{\pi(i)}, x_{\pi(j)}) \leq \rho l_i\}.$$

The set S_ρ contains approximately $\mathcal{O}(N\rho^d)$ elements. Using an incomplete Cholesky factorization with sparsity pattern S_ρ , one obtains a lower triangular matrix L_ρ with non-zero entries restricted to S_ρ . This approach achieves the approximation:

$$\|\Pi^T L_\rho L_\rho^T \Pi - \Theta\|_{\text{Fro}} \lesssim e^{-C\rho}.$$

Consequently, it is possible to achieve an accuracy ϵ by accessing only $\mathcal{O}(N \log^d \frac{1}{\epsilon})$ entries of Θ . The resulting Cholesky factors L_ρ will have just $\mathcal{O}(N \log^d \frac{1}{\epsilon})$ non-zero entries.

Exploiting the maxmin ordering for operator learning

The rank revealing and sparsity properties of the maxmin ordering have been leveraged in Schafer and Houman Owhadi, 2024 for achieving SOTA in terms of both complexity and data efficiency for learning the solution operator of arbitrary elliptic PDEs (Schafer and Houman Owhadi, 2024 shows that those solution operators can be rigorously approximated to accuracy ϵ from only $\mathcal{O}(\log N \log^d \frac{N}{\epsilon})$ source-solution pairs). Here we employ the maxmin ordering to improve the accuracy of the map $v \rightarrow L(v)$ such that

$$\Pi^T L L^T \Pi \approx \Theta. \quad (\text{E.3})$$

Exploiting Random Ordering for Operator Learning

The rank-revealing and sparsity properties of the max-min ordering can also be achieved using a random ordering, as it implicitly induces a max-min structure (see Houman Owhadi, 2017, Thm. 2.9, derived from Halko, Martinsson, and Tropp, 2011). In cases where $\mathcal{F}$ is high-dimensional or represents an arbitrary operator not necessarily associated with a PDE, the rank-revealing property is still preserved by the random ordering Y. Chen, Epperly, et al., 2023, therefore in those settings we can simply replace Π in (E.3) by random permutation matrix.

E.4 Discretizations and Cholesky factorization

In practice, we work with an arbitrary discretization of the operator $\mathcal{G}$, setting $\mathcal{U} = \mathbb{R}^J$ and $\mathcal{V} = \mathbb{R}^K$. This discretization can be derived from various numerical

methods such as finite-element, or finite-difference methods or any other discretization method such as the optimal recovery approach in Batlle, Matthieu Darcy, Hosseini, et al., 2024c. For simplicity, we retain the original notation for the discretized operators, continuing to use $\mathcal{G}$ and $\mathcal{F}$ to denote their discretized counterparts. The spaces $\mathcal{U}$ and $\mathcal{V}$ are equipped with the Euclidean inner product. Consequently, the derivative $\frac{\delta \mathcal{F}}{\delta v}$ becomes a function mapping $\mathbb{R}^J$ to the space of $K \times K$ matrices. The elliptic operator

$$\frac{\delta \mathcal{F}^*}{\delta v} \frac{\delta \mathcal{F}}{\delta v} + \lambda I : \mathcal{V} \rightarrow \mathcal{L}_+(\mathcal{V})$$

then simplifies to a map from $\mathbb{R}^J$ to the space of $J \times J$ symmetric positive definite matrices. Then learning the operator Q reduces to learning a mapping from $\mathbb{R}^J$ to the space of $J \times J$ symmetric positive definite matrices. To facilitate this, we represent Q through its Cholesky factorization:

$$Q = \mathcal{R}\mathcal{R}^T,$$

and seek to learn a lower-triangular-matrix-valued operator:

$$\mathcal{R} : \mathcal{V} \rightarrow \{R \in \mathbb{R}^{J \times J} \mid R \text{ is lower triangular}\}.$$

Then a neural network or matrix valued RKHS can be used to parameterize and learn $\mathcal{R}$. In practice we find that this parameterization of the DNO leads to higher accuracy as the composed operators will enforce positivity and symmetry of Q ; however, one can design intermediate algorithms where Q or $\mathcal{N}$ are directly learned.

E.5 Choice of the Learning rate α and Relaxation λ

We often also found it necessary to adapt the relaxation λ across iterations for convergence to machine precision in a reasonable number of steps. Generally speaking, λ is decreased as our NK/CHONKNORIS approximation approaches the true solution in order to achieve the quadratic convergence of Gauss–Newton steps. In such scenarios, the learned Cholesky factor $\hat{\mathcal{R}}$ has an additional dependence on the relaxation λ . To simultaneously tune the learning rate and relaxation, at any given iteration we choose to inflate, deflate, or keep both parameters from the previous iteration. Specifically, if (λ_n, α_n) are the values at iteration n , then we choose $\lambda_{n+1} \in \{\beta_\lambda \lambda_n, \lambda_n, \kappa_\lambda \lambda_n\}$ and $\alpha_{n+1} \in \{\beta_\alpha \alpha_n, \alpha_n, \kappa_\alpha \alpha_n\}$ among all nine possible combinations where $\kappa_\lambda, \kappa_\alpha \in (0, 1)$ are decay factors and $\beta_\lambda, \beta_\alpha$ are inflation factors. The choices are visualized in Figure E.1.

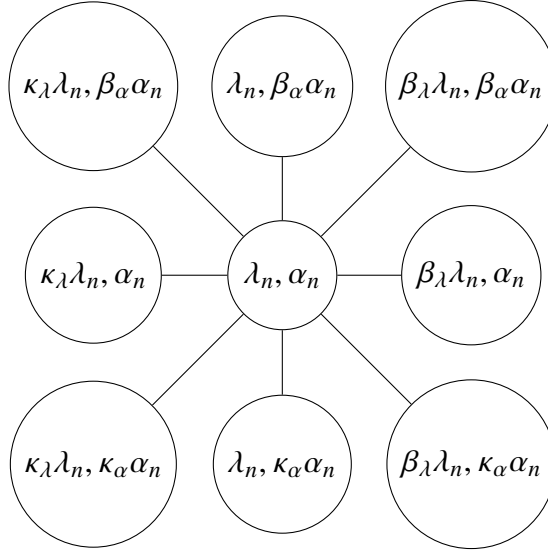


Figure E.1: Options for jointly adaptive relaxation and learning rate.

E.6 Approximation of the Fréchet derivative

In this section, we discuss some numerical approximations of the Jacobian in cases when the explicit computation and storage of the Jacobian becomes infeasible, e.g., for high-dimensional problems such as problems in seismic imaging cf. Section 6.3. The idea is to approximate the Fréchet derivative of $\mathcal{F}$ by the finite difference

$$\frac{\delta \mathcal{F}}{\delta v}(u, v)[h] \approx \frac{\mathcal{F}(u, v + th) - \mathcal{F}(u, v)}{t},$$

for sufficiently small $t > 0$ justified by the limit

$$\frac{\delta \mathcal{F}}{\delta v}(u, v)[h] = \lim_{t \searrow 0} \frac{\mathcal{F}(u, v + th) - \mathcal{F}(u, v)}{t}.$$

We obtain for all $h \in \mathcal{V}$

$$\left\langle \left(\frac{\delta \mathcal{F}}{\delta v}(u, v) \right)^* \mathcal{F}(u, v), h \right\rangle_{\mathcal{V}} = \left\langle \mathcal{F}(u, v), \frac{\delta \mathcal{F}}{\delta v}(u, v)h \right\rangle_{\mathcal{W}} \approx \frac{1}{t} \langle \mathcal{F}(u, v), \mathcal{F}(u, v + th) - \mathcal{F}(u, v) \rangle_{\mathcal{W}}. \quad (\text{E.4})$$

Denoting again with $\mathbb{R}^{N_{\mathcal{V}}}$ the finite-dimensional approximation of the space $\mathcal{V}$, we can find the values $\frac{\delta \mathcal{F}}{\delta v}^* \mathcal{F}(u, v)$ by choosing h to be the values of a basis $\mathcal{B}$ of $\mathbb{R}^{N_{\mathcal{V}}}$. In order to further reduce the complexity, we can perform a singular value decomposition of $\frac{\delta \mathcal{F}}{\delta v}^* \mathcal{F}(u, v) : \mathcal{U} \times \mathcal{V} \rightarrow \mathcal{V}$ on a given dataset $\{(u_i, v_i)\}_{i=1}^m \subset \mathcal{U} \times \mathcal{V}$: Denoting with $F \in \mathbb{R}^{m \times N_{\mathcal{V}}}$ the matrix collocating the point evaluations $\frac{\delta \mathcal{F}}{\delta v}^* \mathcal{F}(u_i, v_i)$, we obtain $F = U \Sigma V^T$ with $U \in \mathbb{R}^{m \times m}$, $\Sigma \in \mathbb{R}^{m \times N_{\mathcal{V}}}$, and $V \in \mathbb{R}^{N_{\mathcal{V}} \times N_{\mathcal{V}}}$. Given a low-rank approximation $r \ll N_{\mathcal{V}}$, we have $F \approx U_r \Sigma_r V_r^T$. Then, an

approximation of $\left(\frac{\delta \mathcal{F}}{\delta v}(u, v)\right)^* \mathcal{F}(u, v)$ can be achieved by expanding it in terms of the reduced basis $\mathcal{B}_r = \{v_1, \dots, v_r\}$ where the coefficients of this can be found by testing with $h = v_i, i = 1, \dots, r$ in (E.4).

E.7 Connection to Attention Mechanism

The *operator-valued operator* $\mathcal{N}(v_n) = \left(\left(\frac{\delta \mathcal{F}}{\delta v}(u, v_n)\right)^* \frac{\delta \mathcal{F}}{\delta v}(u, v_n) + \lambda I\right)^{-1}$ depends nonlinearly on its input (the current iterate v_n), and its output is a nonlocal, linear operator. This observation leads to the connection with the attention mechanism proposed in Vaswani et al., 2017, which defines itself an operator of this nature. The attention mechanism is at the heart of transformers, a neural network architecture widely used for language (Touvron et al., 2023) and vision (Dosovitskiy et al., 2021) tasks and recently employed in the context of operator learning, for example as in Cao, 2021; Z. Hao, Z. Wang, et al., 2023; Zijie Li, Meidani, and Farimani, 2023; Rahman et al., 2024; Calvello et al., 2024. Indeed, self-attention and cross-attention are extended to the function space setting in Calvello et al., 2024. Following this framework, we write the extension to function space of the attention mechanism from Vaswani et al., 2017 in its most general form. We define the attention operator $\mathcal{A} : \mathcal{V} \times \mathcal{W} \times \mathcal{U} \rightarrow \mathcal{U}$ by

$$(\mathcal{A}(v, w, u))(x) = \mathbb{E}_{y \sim p(\cdot; v, w, x)} V u(y), \quad (\text{E.5})$$

where

$$p(y; v, w, x) = \frac{\exp\left(\langle Qv(x), Kw(y) \rangle\right)}{\int_{\Omega} \exp\left(\langle Qv(x), Kw(s) \rangle\right) ds},$$

where the learnable $Q \in \mathbb{R}^{d \times d_v}, K \in \mathbb{R}^{d \times d_w}, V \in \mathbb{R}^{d \times d_u}$ parametrize the attention operator $\mathcal{A}$. Given this definition, we may approximate

$$\mathcal{N}(v_n) \approx \mathcal{A}(v_n, v_n, \cdot) \in \mathcal{L}(\mathcal{U}, \mathcal{V}). \quad (\text{E.6})$$

The above is unlike the standard way attention is implemented in practice, as the second and third input functions of $\mathcal{A}$ differ. An alternative approximation of $\mathcal{N}$ based on the cross-attention operator is given by $\mathcal{A}(v_n, \mathcal{F}(u, v_n), \cdot)$; note that, here, the second input is the residual itself. Using an approximation of $\mathcal{N}$ given by (E.6) leads to the reformulation of the iteration (6.7) as a neural network block

$$\begin{cases} r_{n+1} &= \mathcal{F}(u, v_n), \\ \delta v_{n+1} &= \mathcal{A}(v_n, v_n, r_{n+1}) \approx \mathcal{N}(v_n)(r_{n+1}), \\ v_{n+1} &= v_n + \delta v_{n+1}. \end{cases} \quad (\text{E.7})$$

We highlight that as the transformer from Vaswani et al., 2017 consists of the application of the attention mechanism (a nonlocal operator) along with residual connections and pointwise linear transformations, the block in (E.7) is defined itself by a pointwise transformation, attention and a residual connection. This perspective may be used as the basis for an end-to-end learning approach. In particular, we may unroll CHONKNORIS, as outlined in Monga, Y. Li, and Eldar, 2021. This entails choosing a fixed number of iterations for our iterative method and interpreting this iterative method as a composition of blocks of the form (E.7). We leave further investigation of this end-to-end approach to future work.

Remark 18. *Note that while common operator learning models fail to achieve high accuracy with increasing depth, the proposed iterative structure (6.7) is specifically designed to converge toward the true operator $\mathcal{G}$ as depth increases as the compositions emulate a quasi-Newton algorithm. This connection aligns with observations in the ANN literature Monga, Y. Li, and Eldar, 2021; Houman Owhadi, 2023a, where successful algorithms are effective not solely due to their depth or complexity, but because they possess: (1) Sufficient expressivity to emulate convergent numerical approximation methods; and (2) Enough structure in their computational graphs Houman Owhadi, 2022b to leverage the same principles that make such numerical methods effective. Without (2) increasing depth in an ANN algorithm may only increase expressivity and complexity without improving accuracy.*

E.8 Proofs

Theoretical results

Proof of Theorem 11. Write $A(v) := \mathcal{F}'(v)$, $A_k := A(v_k)$, and $F_k := \mathcal{F}(v_k)$. For $\lambda > 0$, $R_\lambda(v) := (\lambda I_{\mathcal{V}} + A(v)^* A(v))^{-1}$.

Step 1. On $\mathcal{W}$,

$$I_{\mathcal{W}} - A(v) R_\lambda(v) A(v)^* = \lambda (\lambda I_{\mathcal{W}} + A(v) A(v)^*)^{-1}. \quad (\text{E.8})$$

Indeed, $(\lambda I_{\mathcal{V}} + A^* A) R_\lambda = I_{\mathcal{V}}$; multiplying on the left by A and on the right by A^* gives $A(\lambda I + A^* A) R_\lambda A^* = A A^*$, i.e., $\lambda A R_\lambda A^* + A A^* R_\lambda A^* = A A^*$, and rearranging yields (E.8) since $(\lambda I_{\mathcal{W}} + A A^*)$ is invertible by (A3).

Step 2. With $\delta v_k = -\widehat{R}_{\lambda_k}(v_k) A_k^* F_k$,

$$A_k \delta v_k + F_k = (I - A_k \widehat{R}_{\lambda_k}(v_k) A_k^*) F_k = \underbrace{(I - A_k R_{\lambda_k}(v_k) A_k^*)}_{(\text{I})} F_k + \underbrace{A_k (R_{\lambda_k}(v_k) - \widehat{R}_{\lambda_k}(v_k)) A_k^*}_{(\text{II})} F_k.$$

By (E.8) at v_k and (A3),

$$\|(I)\| = \|\lambda_k(\lambda_k I_{\mathcal{W}} + A_k A_k^*)^{-1}\| \leq \frac{\lambda_k}{\lambda_k + \sigma_*^2} =: \theta_k^{\text{tik}}.$$

For (II), using submultiplicativity and Theorem 10(A3),

$$\|(II)\| \leq \|A_k\| \|R_{\lambda_k}(v_k) - \widehat{R}_{\lambda_k}(v_k)\| \|A_k^*\| \leq M^2 \varepsilon_{\lambda_k} =: \theta_k^{\text{des}}.$$

Therefore

$$\|A_k \delta v_k + F_k\| \leq (\theta_k^{\text{tik}} + \theta_k^{\text{des}}) \|F_k\| =: \theta_k \|F_k\|,$$

which is the inexact Newton forcing inequality (6.23).

Step 3. By (A2), for any v and s with $v, v + s \in \overline{B}(v_0, R)$,

$$\|\mathcal{F}(v + s) - \mathcal{F}(v) - \mathcal{F}'(v)s\| \leq \frac{1}{2}L\|s\|^2.$$

The standard (Dembo–Eisenstat–Steihaug) majorant analysis applies with

$$\bar{\theta} := \sup_k \theta_k < 1, \quad \widetilde{L} := \frac{L}{1 - \bar{\theta}},$$

yielding existence/uniqueness in the ball, the majorant bounds, and convergence provided $\widetilde{h} = \beta \widetilde{L} \eta \leq \frac{1}{2}$ and $t_* \leq R$. If additionally $\lambda_k \rightarrow 0$ and $\varepsilon_{\lambda_k} \rightarrow 0$, then $\theta_k \rightarrow 0$ and the local rate approaches quadratic; if $\theta_k = \mathcal{O}(\|F_k\|)$, the rate is quadratic. $\square$

Proof of Corollary 2. We sketch the standard derivation; see also DES Dembo, Eisenstat, and Steihaug, 1982. By the mean-value theorem for Gâteaux differentiable functions and (A2), we obtain

$$\mathcal{F}(v_k + \delta v_k) - \mathcal{F}(v_k) - A(v_k)\delta v_k = \int_0^1 (A(v_k + \tau \delta v_k) - A(v_k)) \delta v_k d\tau,$$

so $\|\mathcal{F}(v_{k+1})\| \leq \|A(v_k)\delta v_k + \mathcal{F}(v_k)\| + \frac{L}{2}\|\delta v_k\|^2$. Left-multiplying by $A(v_k)^{-1}$ and using $\|A(v_k)^{-1}\| \leq \beta$ on $\overline{B}(v_0, R)$ yields

$$\|v_{k+1} - v_k - A(v_k)^{-1}\mathcal{F}(v_k)\| \leq \beta \|A(v_k)\delta v_k + \mathcal{F}(v_k)\| + \frac{\beta L}{2} \|\delta v_k\|^2.$$

By the forcing condition with $\bar{\theta}$ and the stability of the step $\|\delta v_k\| \leq \|A(v_k)^{-1}\| \|\mathcal{F}(v_k)\| + o(\|\mathcal{F}(v_k)\|) \leq \beta \|\mathcal{F}(v_k)\| + o(\|\mathcal{F}(v_k)\|)$, one arrives at

$$\|v_{k+1} - v^*\| \leq \frac{\bar{\theta}}{1 - \bar{\theta}} \|v_k - v^*\| + \frac{\beta L}{2(1 - \bar{\theta})} \|v_k - v^*\|^2$$

which is (6.24). Assertions (i)–(iv) follow immediately: (i) by dropping the quadratic term and applying limes superior; (ii) because $\theta_k \rightarrow 0$ eliminates the linear term; (iii) and (iv) use the local equivalence $\|\mathcal{F}(v_k)\| \asymp \|v_k - v_*\|$ (since $A(v_*)$ is continuous and invertible for large k) to replace $\|\mathcal{F}(v_k)\|^\alpha$ by e_k^α . $\square$

Proof of Remark 15. (a) is immediate from $\theta_{\text{tik},k} \rightarrow 0$ and $\theta_{\text{des},k} \rightarrow 0$. For (b), $\theta_{\text{tik},k} \leq \lambda_k / \sigma_*^2 = O(\|\mathcal{F}(v_k)\|)$ and $\theta_{\text{des},k} = M^2 \varepsilon_{\lambda_k} = O(\|\mathcal{F}(v_k)\|)$, hence $\theta_k = O(\|\mathcal{F}(v_k)\|)$; apply Corollary 2 (iv). For (c), differentiate ϕ on $(0, \infty)$:

$$\phi'(\lambda) = \frac{\sigma_*^2}{(\lambda + \sigma_*^2)^2} - 2C\lambda^{-3}.$$

Setting $\phi'(\lambda) = 0$ yields $(\lambda + \sigma_*^2)^{-2} \sim 2C\sigma_*^{-2}\lambda^{-3}$; for the coarse scaling one may drop the $+\sigma_*^2$ inside parentheses (or solve exactly), giving $\lambda^3 \sim 3C\sigma_*^2$ and the stated λ_* . This balances the two terms in $\bar{\theta}$; an annealing $\lambda_k \downarrow 0$ then ensures $\theta_k \rightarrow 0$ and, by Corollary 2, superlinear/quadratic rates. $\square$

Proof of Theorem 12. We verify the assumptions of Theorem 11 for the PDE setting and identify the constants appearing there.

Step 1: Setting and choice of base point. For a fixed $f \in \mathcal{W}$, we suppress the dependence on f and write $\mathcal{F}(u) = \mathcal{F}(u, f) = -\Delta u + \kappa u^3 - f : \mathcal{V} \rightarrow \mathcal{W}$. For simplicity, we choose the base point $v_0 = 0$ and $\kappa = 1$.

By definition of the residual map,

$$\mathcal{F}(0) = -f \in \mathcal{W}, \quad \mathcal{F}'(0) = A := -\Delta : \mathcal{V} \rightarrow \mathcal{W}.$$

Recall that the H^{-1} -norm is defined by duality with respect to the $\mathcal{V}$ -inner product induced by A , so that for all $g \in \mathcal{W} = H^{-1}$,

$$\|g\|_{\mathcal{W}} = \|A^{-1}g\|_{\mathcal{V}}.$$

Therefore

$$\eta := \|\mathcal{F}'(0)^{-1}\mathcal{F}(0)\|_{\mathcal{V}} = \|A^{-1}(-f)\|_{\mathcal{V}} = \|f\|_{\mathcal{W}},$$

which is the first identity in the statement.

Step 2: Verification of (A1) and explicit β . Assumption (A1) of Theorem 10 requires that $\mathcal{F}'(v_0)$ is invertible and $\|\mathcal{F}'(v_0)^{-1}\| \leq \beta$. As just noted, $\mathcal{F}'(0) = A$ is an isomorphism $\mathcal{V} \rightarrow \mathcal{V}^* = \mathcal{W}$, so (A1) holds with

$$\beta = \|\mathcal{F}'(0)^{-1}\| = \|A^{-1}\| = 1,$$

where the last equality follows from the way the H^{-1} -norm is induced by the $\mathcal{V}$ -inner product via A .

Step 3: Lipschitz bound (A2) and choice of L . Let $B_r := \{v \in \mathcal{V} : \|v\|_{\mathcal{V}} \leq r\}$. Then, for each $v \in X$, we have

$$\mathcal{F}'(v)h = Ah + 3v^2h.$$

Hence, for $u, v \in B_r$,

$$(\mathcal{F}'(u) - \mathcal{F}'(v))h = 3(u^2 - v^2)h = 3(u + v)(u - v)h.$$

Using the one-dimensional Sobolev embedding and the Poincaré inequality on $\mathbb{T}$, we obtain

$$\|w\|_{L^\infty} \leq \frac{1}{\pi} \|w\|_{\mathcal{V}}, \quad \|w\|_{L^2} \leq \frac{1}{\pi} \|w\|_{\mathcal{V}}.$$

Together with the Cauchy-Schwarz inequality, we obtain the bound

$$\|(u^2 - v^2)h\|_{H^{-1}} \leq \frac{1}{2\pi^2} (\|u\|_{\mathcal{V}} + \|v\|_{\mathcal{V}}) \|u - v\|_{\mathcal{V}} \|h\|_{\mathcal{V}}.$$

Since $u, v \in B_r$, this yields

$$\|\mathcal{F}'(u) - \mathcal{F}'(v)\| \leq \frac{3}{2\pi^2} r \|u - v\|_{\mathcal{V}},$$

so (A2) holds with

$$L := \frac{3}{2\pi^2} r.$$

Step 4: Uniform bound (A3) and choice of M and σ_ .* For any $v \in B_r$ and $h \in \mathcal{V}$ with $\|h\|_{\mathcal{V}} = 1$,

$$\|\mathcal{F}'(v)h\|_{H^{-1}} \leq \|Ah\|_{H^{-1}} + \|3v^2h\|_{H^{-1}}.$$

By definition of the norms, $\|Ah\|_{H^{-1}} = \|h\|_{\mathcal{V}} = 1$. Arguing as in Step 3, we obtain

$$\|3v^2h\|_{H^{-1}} \leq \frac{3}{4\pi^2} \|v\|_{\mathcal{V}}^2 \|h\|_{\mathcal{V}} \leq \frac{3}{4\pi^2} r^2.$$

Hence,

$$\|\mathcal{F}'(v)\| = \sup_{\|h\|_{\mathcal{V}}=1} \|\mathcal{F}'(v)h\|_{H^{-1}} \leq 1 + \frac{3}{4\pi^2} r^2 =: M$$

for all $v \in B_r$, which is the M stated in Theorem 12.

Moreover, by coercivity of A and the positivity of the nonlinearity, we obtain

$$\langle \mathcal{F}'(v)h, h \rangle_{\mathcal{W}, \mathcal{V}} = \langle Ah, h \rangle_{\mathcal{W}, \mathcal{V}} + \langle 3v^2h, h \rangle_{\mathcal{W}, \mathcal{V}} \geq \|h\|_{\mathcal{V}}^2,$$

so that the smallest singular value of $\mathcal{F}'(v)$ is bounded below by 1. Therefore

$$\sigma_* := \inf_{v \in B} \sigma_{\min}(\mathcal{F}'(v)) \geq 1.$$

Thus (A3) holds with the above M and $\sigma_* \geq 1$.

Step 5: Forcing term and choice of $\bar{\theta}$. Let $\lambda > 0$ be fixed and assume that we use a constant Tikhonov parameter, i.e. $\lambda_k \equiv \lambda$. For the learned resolvent $\widehat{R}_\lambda$, we define the design error

$$\varepsilon_\lambda := \sup_{v \in B_r} \|\widehat{R}_\lambda(v) - R_\lambda(v)\|_{\mathcal{L}(\mathcal{V}, \mathcal{V})}.$$

Then, Theorem 11 (Tikhonov–inexact NK) gives, for any iterate $v_k \in B_r$,

$$\frac{\|\mathcal{F}'(v_k)\delta v_k + \mathcal{F}(v_k)\|_{\mathcal{W}}}{\|\mathcal{F}(v_k)\|_{\mathcal{W}}} \leq \frac{\lambda}{\lambda + \sigma_*^2} + M^2 \varepsilon_\lambda =: \theta_k.$$

Using $\sigma_* \geq 1$, we further obtain

$$\theta_k \leq \frac{\lambda}{1 + \lambda} + M^2 \varepsilon_\lambda =: \bar{\theta}.$$

If $\bar{\theta} < 1$, then the hypothesis $\sup_k \theta_k \leq \bar{\theta} < 1$ of Theorem 11 is satisfied.

Step 6: Application of Theorem 11 and explicit $\tilde{L}$. Theorem 11 then yields that all the inexact Newton–Kantorovich conclusions hold with modified Lipschitz constant

$$\tilde{L} = \frac{L}{1 - \bar{\theta}}.$$

Plugging in the PDE-specific values

$$L = \frac{3}{2\pi^2} r, \quad \bar{\theta} = \frac{\lambda}{1 + \lambda} + M^2 \varepsilon_\lambda, \quad M = 1 + \frac{3}{4\pi^2} r^2,$$

we obtain the expression stated in Theorem 12,

$$\tilde{L} = \frac{L}{1 - \bar{\theta}} = \frac{\frac{3}{2\pi^2} r}{1 - \frac{\lambda}{1 + \lambda} - M^2 \varepsilon_\lambda}.$$

Together with the identification $\eta = \|\mathcal{F}'(0)^{-1}\mathcal{F}(0)\|_{\mathcal{V}} = \|f\|_{H^{-1}}$ from Step 1, this shows that all conclusions of Theorem 11 hold in the PDE setting with the explicit constants given in Theorem 12. $\square$

Appendix F

CHAPTER 7

F.1 Additional Plots.

In this section, we provide additional plots of the results of the numerical experiments.

Exponential volatility process

The following figures illustrate the results on the second trajectory of the exponential volatility process.

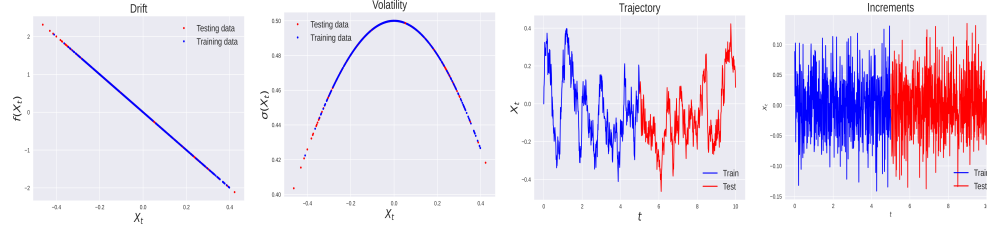


Figure F.1: From left to right: drift function, volatility function, sample trajectory, and sample increments of the exponential volatility process.

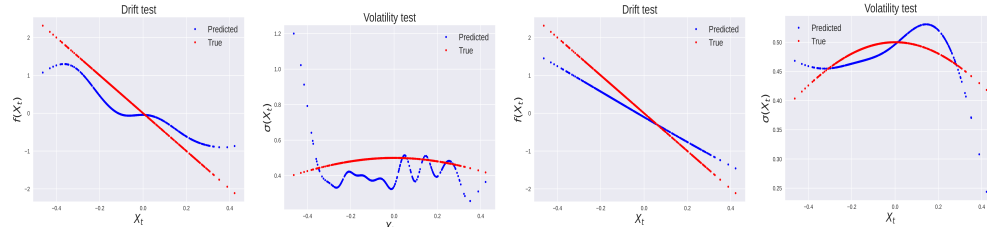


Figure F.2: Predicted drift and volatility on the testing set for trajectory 1 of the exponential volatility process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel).

Trigonometric process

The following figures illustrate the results of the first trajectory of the trigonometric process.

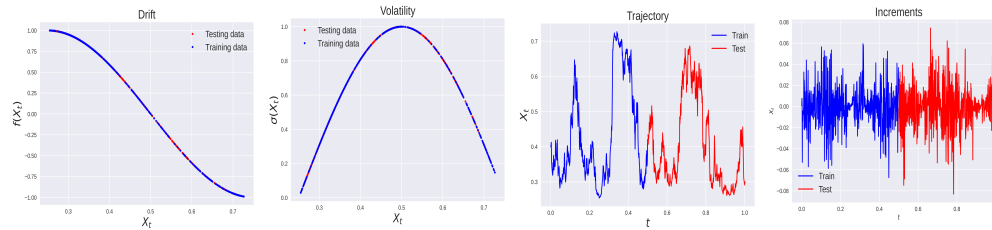


Figure F.3: From left to right: drift function, volatility function, sample trajectory, and sample increments of the trigonometric process.

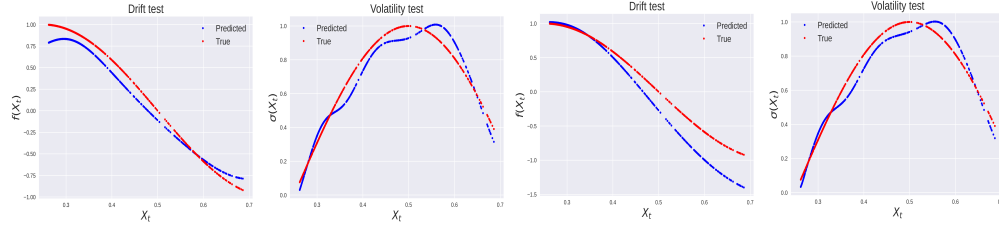


Figure F.4: Predicted drift and volatility on the testing set for trajectory 1 of the trigonometric process. From left to right: drift (non-learned kernel), volatility (non-learned kernel), drift (learned kernel), volatility (learned kernel).

F.2 Gaussian Process Regression and Extension of Bishop, 2006, Page 306

In this section, we give a very brief overview of Gaussian processes for regression. We suppose that the values of $Y(X)$ are distributed according to a Gaussian process, namely $Y(X) \stackrel{d}{\sim} \mathcal{GP}(\mathbf{O}, \mathbf{K})$. In particular, in the case of SDEs, given the data (X, Y) , the predicted drift and diffusion for a new point x_* are given by

$$\begin{aligned}\bar{f}(x_*) &= \mathbf{K}(x_*, X) \mathbf{K}(X, X)^{-1} Y \\ \bar{\sigma}(x_*) &= \mathbf{K}(x_*, x_*) - \mathbf{K}(x_*, X) \mathbf{K}(X, X)^{-1} \mathbf{K}(X, x_*).\end{aligned}$$

The above expressions are valid in noisy observation with independent and identical Gaussian noise. We derive this distribution in the case where the observations are noisy with independent, but not necessarily identical, Gaussian noise; the proof is generalized from the one presented in Bishop, 2006, p. 306.

Formally, suppose that we have at our disposal the noisy observations $(X_n, Y_n)_{1 \leq n \leq N}$, where $Y_i = f(X_i) + W_i$ and the W_i are independent but not necessarily identically distributed $\mathcal{N}(0, \sigma^2(X_i))$ random variables. The problem is the identification of the unknown function f given these noisy observations. Set $f_i = f(X_i)$, so that $\mathbf{f} := (f_i)_{1 \leq i \leq N}$. In addition, set $\mathbf{Y} := (Y_i)_{1 \leq i \leq N}$. Observe that $\mathbf{Y} | \mathbf{f} \stackrel{d}{\sim} \mathcal{N}(\mathbf{f}, \mathbf{\Sigma})$ where $(\mathbf{\Sigma})_{ij} = \delta_{ij} \sigma^2(X_i)$, and assume that $\mathbf{f} \stackrel{d}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{K})$. Then, Bishop, 2006, Page 93,

$$p(\mathbf{Y}) = \int p(\mathbf{Y} | \mathbf{f}) p(\mathbf{f}) d\mathbf{f} = \mathcal{N}(\mathbf{0}, \mathbf{K} + \mathbf{\Sigma}).$$

Now, let $\mathbf{C} = \mathbf{K} + \mathbf{\Sigma}$ so that $C_{ij} = k(X_i, X_j) + \sigma_i^2 \delta_{ij}$. Denote $\mathbf{Y}_{N+1} = (Y_1, \dots, Y_{N+1})$, $\mathbf{Y}_N = (Y_1, \dots, Y_N)$. We wish to derive the conditional distribution $p(Y_{N+1} | \mathbf{Y}_N)$. First, observe that

$$p(\mathbf{Y}_{N+1}) = \mathcal{N}(\mathbf{0}, \mathbf{C}_{N+1}),$$

where $\mathbf{C}_{N+1}$ is the $(N+1) \times (N+1)$ with entries defined as previously for the vector $\mathbf{Y}_{N+1}$. We may partition the covariance matrix as

$$\mathbf{C}_{N+1} = \begin{pmatrix} \mathbf{C}_N & \mathbf{K} \\ \mathbf{K}^T & c \end{pmatrix}$$

where $c = K(X_{N+1}, X_{N+1}) + \sigma_{N+1}^2$ and $\bar{\mathbf{K}}$ is the vector with entries $\bar{K}_i = K(X_{N+1}, X_i)$. Then $p(Y_{N+1}|\mathbf{Y}_N) = \mathcal{N}(m(X_{N+1}), \sigma^2(X_{N+1}))$ and the conditional mean and covariance are given by

$$\begin{aligned} m(X_{N+1}) &= \bar{\mathbf{K}}^T \mathbf{C}^{-1} \mathbf{y}_N = \bar{\mathbf{K}}^T (\mathbf{K} + \Sigma)^{-1} \mathbf{y}_N \\ \sigma^2(X_{N+1}) &= c - \bar{\mathbf{K}}^T \mathbf{C}^{-1} \bar{\mathbf{K}} = c - \bar{\mathbf{K}}^T (\mathbf{K} + \Sigma)^{-1} \bar{\mathbf{K}}. \end{aligned}$$

Note that in our case, the observations actually of the form $Y_i = \Delta t_i f(X_i) + W_i$. The same proof holds with the slight modification that the entries of the covariance matrix now become

$$\begin{aligned} C_{ij} &= \Delta t_i \Delta t_j k(X_i, X_j) + \sigma_i^2 \delta_{ij} \\ \bar{K}_i &= \Delta t_i K(X_{N+1}, X_i) \end{aligned}$$

which yields equations (7.7) and (7.8).

F.3 Log-marginal likelihood for hyper-parameter optimization

In this section, we briefly present the log-marginal likelihood method for hyper-parameter optimization. The marginal log-likelihood over the kernel parameters can be expressed as

$$-\log(p(Y|\boldsymbol{\theta}, X)) \propto \frac{1}{2} \mathbf{Y}^T \mathbf{K}(X, X)^{-1} \mathbf{Y} + \log(|\mathbf{K}(X, X)|).$$

Therefore, the optimal parameters $\boldsymbol{\theta}$ are obtained via the minimization of the function in the previous equation (with respect to $\boldsymbol{\theta}$). In the case of noisy observations, the kernel K can be defined as

$$K(X_1, X_2) = K'(X_1, X_2) + \delta(X_1, X_2),$$

where K' is a standard kernel and δ is the white noise kernel defined as

$$\delta(X_1, X_2) := \begin{cases} c & \text{if } X_1 = X_2, \\ 0 & \text{otherwise,} \end{cases}$$

where c is the noise level, a kernel parameter that must be optimized. It is important to note that this kernel can only account for a constant level of noise, which is not the case for many SDEs.

F.4 Randomized cross-validation for hyper-parameter learning

In this section, we describe the general framework of *Randomized cross-validation* we use in our optimization.

The general problem is the following: we have a set of training data $(X, Y) := \{(X_i, Y_i)\}_{1 \leq i \leq N}$ and we are trying to recover a function $f : X \rightarrow Y$. In particular, we assume that we have a class of functions $\mathcal{S}$ indexed by a set of parameters $\mathbf{w}_p \in \mathcal{W}$, i.e. $\mathcal{S} := \{f(\cdot, \mathbf{w}_p) : \mathbf{w}_p \in \mathcal{W} \text{ and } f(\cdot, \mathbf{w}_p) : X \rightarrow Y\}$. Notice that in practice, this could be approximated via reproducing kernels as in this paper or via neural networks. We assume that we have a *method* to find the optimal parameter $\mathbf{w}_p^* \in \mathcal{W}$ for a given training set of data (X, Y) . In our case, we assume that such a method involves the minimization of some loss function $\mathcal{L}(f(X, \mathbf{w}_p), Y)$ with respect to $\mathbf{w}_p$, where $\mathcal{L} : Y \times Y \rightarrow \mathbb{R}$. Moreover, like in many Machine Learning (ML) algorithms, we assume that we have a set of hyper-parameters $\theta \in \Theta$ that affect the recovery of the optimal $\hat{f} \in \mathcal{S}$. Such hyper-parameters θ can parametrize the function $\hat{f} := \hat{f}_\theta := f(\cdot; \mathbf{w}_p, \theta)$, regularize the loss function $\mathcal{L}$ (often through some prior distribution on the parameters $\mathbf{w}_p$) or affect the minimization of the loss $\mathcal{L}$ (such as the learning rate of a gradient descent). We summarize this point by saying that $\hat{f}$ is recovered by minimizing $\mathcal{L}_\theta$.

In the present work, in order to recover the function $f : X \rightarrow Y$, we apply the Robust Learning Algorithm, whose rationale is explained in the following paragraph.

Robust Learning Algorithm via Randomized cross-validation. Many ML algorithms are built upon minimizing a loss function $\mathcal{L} : Y \times Y \rightarrow \mathbb{R}$ over the set of parameters $(\mathbf{w}_p, \theta)$ of a class of models $\hat{f}(\cdot, \mathbf{w}_p; \theta)$. In other words, *ideally*, we can define the risk function R as the expected value of the loss function $\mathcal{L}$ with respect to the data probability density function $p(\mathbf{x}, y)$:

$$R(\mathbf{w}_p, \theta) := \mathbb{E}_{(X, Y) \sim p(\mathbf{x}, y)} \left\{ \mathcal{L}(\hat{f}(X, \mathbf{w}_p; \theta), Y) \right\} \quad (\text{F.1})$$

and then find the set of optimal parameters according to

$$(\mathbf{w}_p^*, \theta^*) := \arg \min_{\mathbf{w}_p, \theta} R(\mathbf{w}_p, \theta). \quad (\text{F.2})$$

Notice that, in this setup, by the assumption that one has access to $p(\mathbf{x}, y)$, there is no theoretical distinction between parameters and hyper-parameters.

However, in practice, one only sees a realization of (X, Y) , namely $\mathcal{D} = \{(X_i, Y_i)\}_{i \in \mathcal{I}}$. Therefore, it is impossible to use Equations (F.1) and (F.2). Moreover, in order to achieve generalization, in practice, $\mathbf{w}_p$ is optimized by minimizing a loss function dependent on θ :

$$\mathbf{w}_p^* = \arg \min_{\mathbf{w}_p} \mathcal{L}_\theta \left(\hat{f} (X; \mathbf{w}_p, \theta), Y \right). \quad (\text{F.3})$$

At this point, the best parameters θ can be chosen via a cross-validation approach where the function $\hat{f}_\theta$ is evaluated on an unseen test set (X_u, Y_u) :

$$R(\theta) = \mathcal{L} \left(\hat{f} \left(X_u; \mathbf{w}_p^*, \theta \right), Y_u \right). \quad (\text{F.4})$$

Generally, the optimization of hyper-parameters Bergstra, Bardenet, et al., 2011; Bergstra and Bengio, 2012; F. Hutter, Hoos, and Leyton-Brown, 2011; Snoek, Larochelle, and Adams, 2012; Maclaurin, Duvenaud, and Adams, 2015; Franceschi et al., 2017 is usually done on a prefixed number of cross validation sets. In this work, we propose an approach that is not bound to a fixed number of cross-validation sets. Our algorithm can be summarized as follows:

- (1) Partition the available data $\mathcal{D} = \{(X_i, Y_i)\}_{i \in \mathcal{I}}$ into a training subset $\mathcal{D}_\mathcal{T} = \{(X_i, Y_i)\}_{i \in \mathcal{T}}$ and a test subsets $\mathcal{D}_\mathcal{U} = \{(X_i, Y_i)\}_{i \in \mathcal{U}}$; the two subsets are mutually exclusive.
- (2) Randomly partition the training set $\mathcal{D}_\mathcal{T}$ into two mutually exclusive subsets of almost equal size: $\mathcal{D}_\Pi = \{(X_i, Y_i)\}_{i \in \Pi(\mathcal{T})}$ and $\mathcal{D}_{\Pi^c} = \{(X_i, Y_i)\}_{i \in \Pi^c(\mathcal{T})}$. Here, $\Pi(\mathcal{T})$ returns the first half of the random permutation of indices in $\mathcal{T}$ and $\Pi^c(\mathcal{T})$ returns the second half of the same permutation.
- (3) Train a ML model on $\mathcal{D}_\Pi$ and evaluate the random loss on $\mathcal{D}_{\Pi^c}$ representing a realization of the generalization error.
- (4) Repeat steps (2) and (3) to optimize the expected loss over the random sets $\mathcal{D}_{\Pi^c}$ with respect to $\mathbf{w}_p$.
- (5) Check the goodness of fit by evaluating the loss over $\mathcal{D}_\mathcal{U}$.

More precisely, the optimization problem is the following one:

$$\begin{aligned} \theta^* &= \arg \min_{\theta} \mathbb{E}_\Pi \left\{ R_{\Pi-\Pi}(\bar{\mathbf{w}}_p, \theta) \right\} \\ \text{s.t. } \bar{\mathbf{w}}_p &= \arg \min_{\mathbf{w}_p} R_\Pi(\mathbf{w}_p, \theta), \end{aligned} \quad (\text{F.5})$$

where

$$R_{\Pi}(\mathbf{w}_p, \boldsymbol{\theta}) = \mathbb{E}_{X, Y \sim \mathcal{D}_{\Pi}} \left\{ \mathcal{L}_{\boldsymbol{\theta}} \left(\hat{f}(X; \mathbf{w}_p, \boldsymbol{\theta}), Y \right) \right\}, \quad (\text{F.6})$$

and

$$R_{\mathbb{I}-\Pi}(\mathbf{w}_p, \boldsymbol{\theta}) = \mathbb{E}_{X, Y \sim \mathcal{D}_{\Pi^c}} \left\{ \mathcal{L} \left(\hat{f}(X; \mathbf{w}_p, \boldsymbol{\theta}), Y \right) \right\}. \quad (\text{F.7})$$

In the previous Equations, $\mathbb{E}_{X, Y \sim \mathcal{D}_{\Pi}} \{ \cdot \}$ means that the expected value is taken over the empirical distribution defined by $\mathcal{D}_{\Pi}$. A similar notion applies to $\mathbb{E}_{X, Y \sim \mathcal{D}_{\Pi^c}} \{ \cdot \}$. Furthermore, $\mathbb{E}_{\Pi} \{ \cdot \}$ means that this expected value is taken over all permutations of the train set. Note that the recovery of $\bar{\mathbf{w}}_p$ is done through the minimization of $\mathcal{L}_{\boldsymbol{\theta}}$ which we refer to as the train loss function, whereas the evaluation of $\hat{f}(X; \mathbf{w}_p, \boldsymbol{\theta})$ is done through $\mathcal{L}$ which we refer to as the test loss function. The Robust Learning Algorithm is presented here below in (3):

In particular, notice that $R_{\mathbb{I}-\Pi_n} = \frac{1}{K} \sum_{j=1}^K e_j$ is a (noisy) approximation of

$$\mathbb{E}_{X, Y \sim \mathcal{D}_{\Pi^c}} \left\{ \mathcal{L} \left(\hat{f}(X; \mathbf{w}_p, \boldsymbol{\theta}), Y \right) \right\}.$$

F.5 Extension to the multivariate case

We now examine the case where the SDE is multi-dimensional. In such a case

$$dX_t = f(X_t)dt + \sigma(X_t)dW_t, \quad (\text{F.8})$$

where $X_t \in \mathbb{R}^d$, $f: \mathbb{R}^d \mapsto \mathbb{R}^d$, $\sigma: \mathbb{R}^d \mapsto \mathbb{R}^{d \times m}$ and $W_t = (W_t^i)_{i \leq m}$ is a *standard* multivariate Wiener process.

For simplicity, we place independent GP priors on each dimension, which is equivalent to using a matrix-valued kernel with diagonal entries:

$$K(x, y) = \begin{pmatrix} K_1(x, y) & 0 \\ 0 & K_2(x, y) \end{pmatrix}. \quad (\text{F.9})$$

More generally, we may consider matrix-valued kernels with non-zero off-diagonals, but the use of diagonal matrices is practical because it makes all optimization problems independent for each dimension. Then the problem leads to d equations:

$$dX_t^i = f(X_t)^i dt + \sum_{j=1}^d \sigma_{i,j}(X_t) dW_t^j \quad (\text{F.10})$$

Algorithm 3 Robust Learning Algorithm

Require:

- (1) Pick a model class $\hat{f}_\theta$
- (2) Pick an initial guess $\theta^{(0)}$.
- (3) Pick an active learning algorithm Al .
- (4) Set $\mathcal{S} = \phi$, $n = 0$ and $C = 0$.

Ensure: Partition the data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i \in \mathcal{I}}$ into two mutually exclusive training $\mathcal{D}_{\mathcal{T}} = \{(\mathbf{x}_i, y_i)\}_{i \in \mathcal{T}}$ and test subsets $\mathcal{D}_{\mathcal{U}} = \{(\mathbf{x}_i, y_i)\}_{i \in \mathcal{U}}$.

while $C = 0$ **do**

$n \leftarrow n + 1$

while $1 \leq j \leq K$ **do**

$j \leftarrow j + 1$

 Pick a random index permutation $\Pi_n(\mathcal{T})$ of $\mathcal{T}$.

 Divide the train set $\mathcal{D}_{\mathcal{T}}$ into $\mathcal{D}_{\Pi_n}$ and $\mathcal{D}_{\Pi_n^c}$.

 Train $\hat{f}(\mathbf{X}; \mathbf{w}_p, \theta^{(n-1)})$ on $\mathcal{D}_{\Pi_n}$ with respect to $R_{\Pi_n}(\mathbf{w}_p, \theta^{(n-1)})$ to obtain $\bar{\mathbf{w}}_p$.

 Evaluate $e_j = R_{\mathbb{I}-\Pi_n}(\bar{\mathbf{w}}_p, \theta^{(n-1)})$ on $\mathcal{D}_{\Pi_n^c}$.

end while

 Set $R_{\mathbb{I}-\Pi_n} = \frac{1}{K} \sum_{j=1}^K e_j$.

 Set $\mathcal{S} \leftarrow \mathcal{S} \cup \left\{ \left(\theta^{(n-1)}, R_{\mathbb{I}-\Pi_n}(\bar{\mathbf{w}}_p, \theta^{(n-1)}) \right) \right\}$.

if $Al(\mathcal{S})$ has converged **then**

$C = 1$
 $(\mathbf{w}_p^*, \theta^*) = (\bar{\mathbf{w}}_p, \theta^{(n-1)})$.

else if $Al(\mathcal{S})$ has not converged **then**

$\theta^{(n)} = Al(\mathcal{S})$

end if

end while

Check the loss over $\mathcal{D}_{\mathcal{U}}$.

which can all be optimized independently. The optimal value of $\bar{f}(\mathbf{X}_t)$ conditioned on $\bar{\sigma}$ is given by the posterior mean equation:

$$\bar{f}(\mathbf{X}_n)^i = \mathbb{E}[\mathbf{f}(\mathbf{X}_n)^i] = K^i(\backslash \mathbf{X}_n, \backslash \mathbf{X})\Lambda(\Lambda K^i(\backslash \mathbf{X}, \backslash \mathbf{X})\Lambda + \Sigma^i + \lambda I)^{-1}\mathbf{Y}^i, \quad (\text{F.11})$$

where $\Sigma^i = \sum_j \Sigma^j$ is the sum of the (diagonal) matrices with entries $\Sigma_{k,k}^j = (\bar{\sigma}(\mathbf{X}_k)^j)^2 \Delta t_k$, $1 \leq k \leq N$, $1 \leq j \leq d$. On the other hand, the MAP estimate for can be computed jointly for all entries of the matrix σ^i with entries σ_k^i as

$$\begin{aligned}
& \arg \min_{\sigma^i} (\backslash Y^i - \Lambda \backslash f(X)^i)^T (\Sigma^i + \lambda I)^{-1} (\backslash Y^i - \Lambda \backslash f(X)^i) + \sum_{k=1}^N \ln(\Delta t_k \sum_{i=1}^d (\sigma_k^i)^2 + \lambda) \\
& + \sum_{i=1}^d \backslash \sigma^{iT} G^i(\backslash X, \backslash X)^{-1} \backslash \sigma^i.
\end{aligned}
\tag{F.12}$$

Additional assumptions can be made regarding the structure of the noise values, such as assuming that the noise is separate across all dimensions (i.e., $\backslash \sigma$ is a diagonal matrix) or that the diffusion function $\backslash \sigma$ is identical across all dimensions (i.e., $\backslash \sigma$ is a diagonal matrix with identical values along the diagonals).

Acknowledgments

MD, BH, and HO acknowledge partial support by the Air Force Office of Scientific Research under MURI award number FA9550-20-1-0358 (Machine Learning and Physics-Based Modeling and Simulation). MD, PT, and HO acknowledge support from Beyond Limits (Learning Optimal Models) through CAST (The Caltech Center for Autonomous Systems and Technologies).

Appendix G

CHAPTER 8

G.1 Additional figures

Bernoulli time series

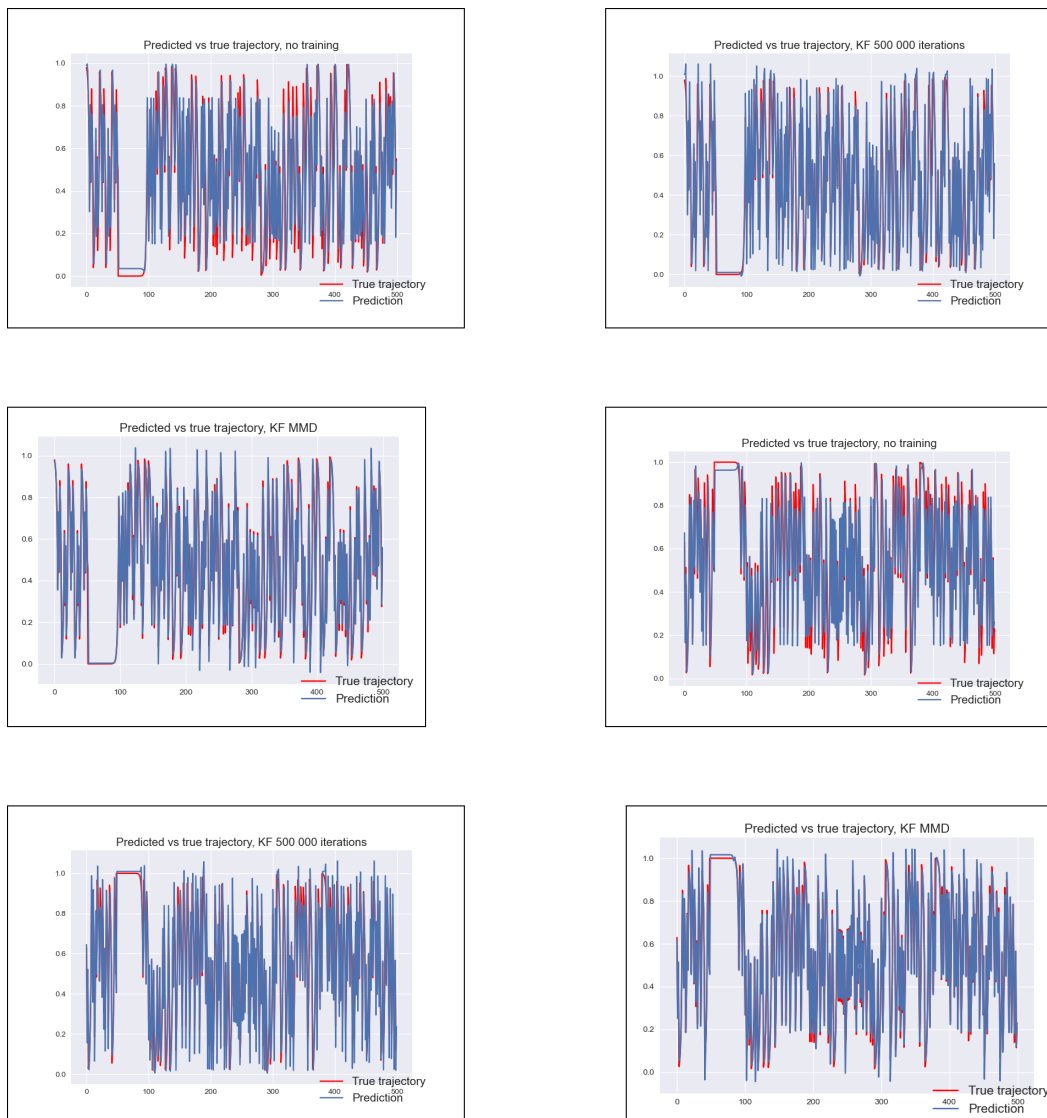


Figure G.1: Prediction results for the Bernoulli map for different initial conditions.

Lorenz time series

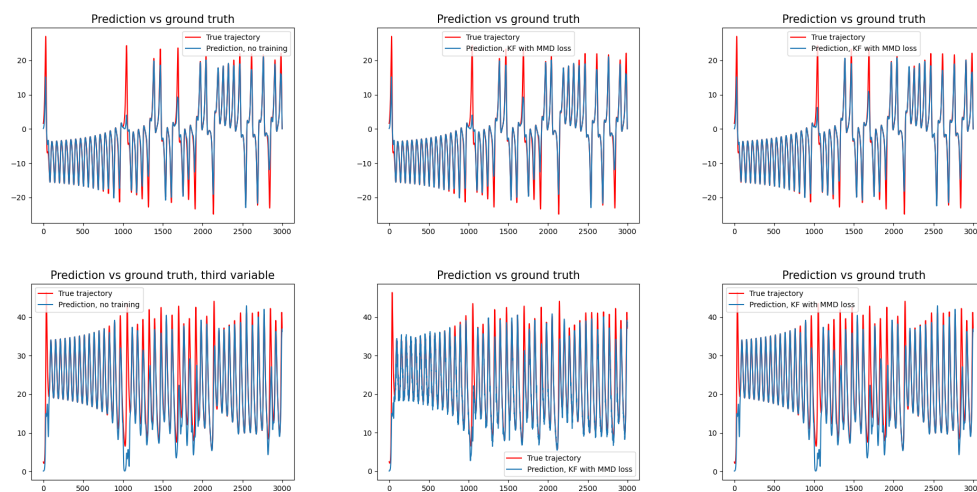


Figure G.2: Lorenz system: comparison between the different predictions of the second (y) and third variables (z) (top, bottom) for the baseline, ρ_{MMD} training and ρ training (left, middle, right).

Convergence of the loss

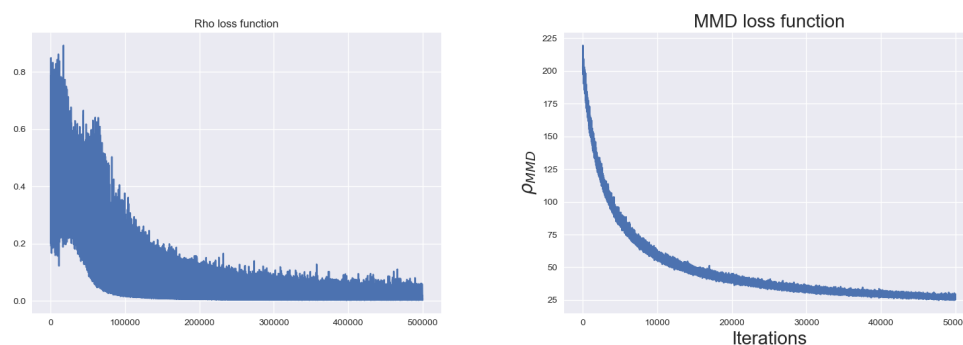


Figure G.3: Convergence of the ρ (left) and ρ_{MMD} (right) for higher learning rates on the Bernoulli time series.

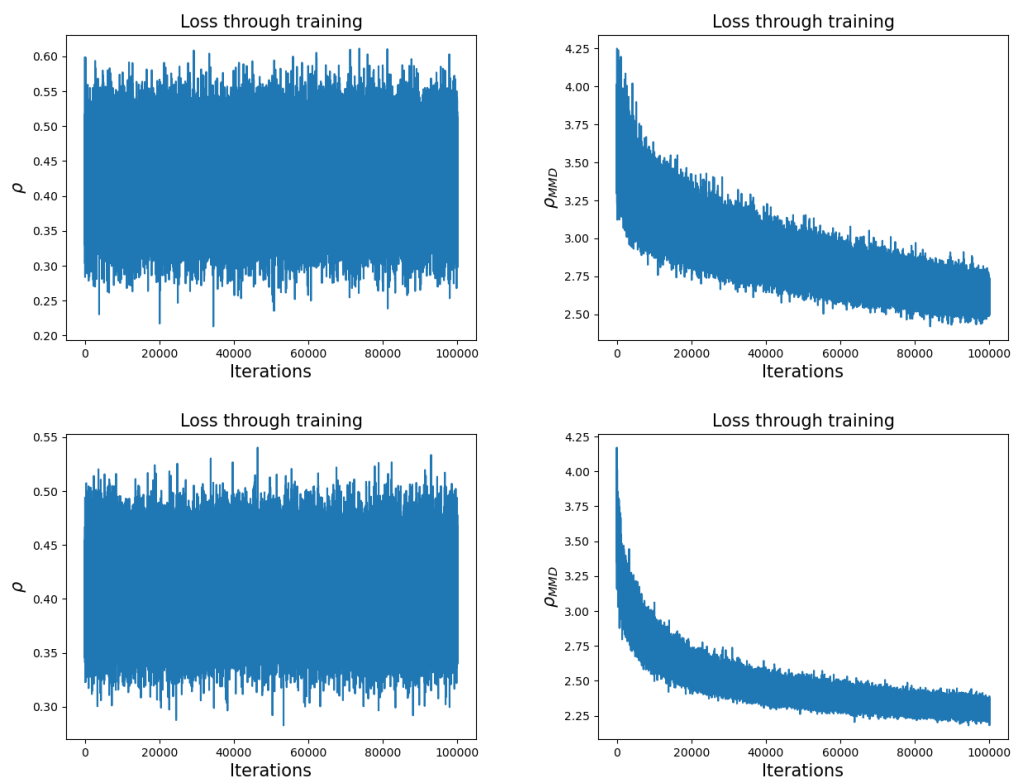


Figure G.4: Loss functions through training for the second (y) and third variables (z) of the Lorenz times series (top, bottom) for ρ (left) and ρ_{MMD} (right).