Hidden Physics Models: Machine Learning of Nonlinear Partial Differential Equations

Maziar Raissi and George Em Karniadakis

Division of Applied Mathematics, Brown University,
Providence, RI, 02912, USA

Abstract

While there is currently a lot of enthusiasm about "big data", useful data is usually "small" and expensive to acquire. In this paper, we present a new paradigm of learning partial differential equations from *small* data. In particular, we introduce *hidden physics models*, which are essentially data-efficient learning machines capable of leveraging the underlying laws of physics, expressed by time dependent and nonlinear partial differential equations, to extract patterns from high-dimensional data generated from experiments. The proposed methodology may be applied to the problem of learning, system identification, or data-driven discovery of partial differential equations. Our framework relies on Gaussian processes, a powerful tool for probabilistic inference over functions, that enables us to strike a balance between model complexity and data fitting. The effectiveness of the proposed approach is demonstrated through a variety of canonical problems, spanning a number of scientific domains, including the Navier-Stokes, Schrödinger, Kuramoto-Sivashinsky, and time dependent linear fractional equations. The methodology provides a promising new direction for harnessing the long-standing developments of classical methods in applied mathematics and mathematical physics to design learning machines with the ability to operate in complex domains without requiring large quantities of data.

Keywords: probabilistic machine learning, system identification, Bayesian modeling, uncertainty quantification, fractional equations, small data

1. Introduction

There are more than a trillion sensors in the world today and according to some estimates there will be about 50 trillion cameras worldwide within the next five years, all collecting data either sporadically or around the clock. However, in scientific experiments, quality and error-free data is not easy to obtain – e.g., for system dynamics characterized by bifurcations and instabilities, hysteresis, and often irreversible responses. Admittedly, as in all everyday applications, in scientific experiments too, the volume of data has increased substantially compared to even a decade ago but analyzing big data is expensive and time-consuming. Data-driven methods, which have been enabled in the past decade by the availability of sensors, data storage, and computational resources, are taking center stage across many disciplines of science. We now have highly scalable solutions for problems in object detection and recognition, machine translation, text-to-speech conversion, recommender systems, and information retrieval. All of these solutions attain state-of-the-art performance when trained with large amounts of data. However, purely data driven approaches for machine learning present difficulties when the data is scarce relative to the complexity of the system. Hence, the ability to learn in a sample-efficient manner is a necessity in these data-limited domains. Less well understood is how to leverage the underlying physical laws and/or governing equations to extract patterns from small data generated from highly complex systems. In this work, we propose a modeling framework that enables blending conservation laws, physical principles, and/or phenomenological behaviors expressed by partial differential equations with the datasets available in many fields of engineering, science, and technology. This paper should be considered a direct continuation of a preceding one [1] in which we addressed the problem of inferring solutions of time dependent and nonlinear partial differential equations using noisy observations. Here, a similar methodology is employed to deal with the problem of learning, system identification, or data-driven discovery of partial differential equations [2].

2. Problem Setup

Let us consider parametrized and nonlinear partial differential equations of the general form

$$h_t + \mathcal{N}_x^{\lambda} h = 0,\ x \in \Omega,\ t \in [0, T], \tag{1}$$

where $h(t,x)$ denotes the latent (hidden) solution, $\mathcal{N}_x^\lambda$ is a nonlinear operator parametrized by λ, and Ω is a subset of $\mathbb{R}^D$. As an example, the one dimensional Burgers' equation corresponds to the case where $\mathcal{N}_x^\lambda h = \lambda_1 h h_x - \lambda_2 h_{xx}$ and $\lambda = (\lambda_1, \lambda_2)$. Here, the subscripts denote partial differentiation in either time or space. Given noisy measurements of the system, one is typically interested in the solution of two distinct problems. The first problem is that of inference or filtering and smoothing, which states: given fixed model parameters λ what can be said about the unknown hidden state $h(t,x)$ of the system? This question is the topic of a preceding paper [1] of the authors in which we introduce the concept of *numerical Gaussian processes* and address the problem of inferring solutions of time dependent and nonlinear partial differential equations using noisy observations. The second problem is that of learning, system identification, or data driven discovery of partial differential equations [2] stating: what are the parameters λ that best describe the observed data? Here we assume that all we observe are two snapshots $\{\boldsymbol{x}^{n-1}, \boldsymbol{h}^{n-1}\}$ and $\{\boldsymbol{x}^{n}, \boldsymbol{h}^{n}\}$ of the system at times t^{n-1} and t^n, respectively, which are $\Delta t = t^n - t^{n-1}$ apart. The main assumption is that Δt is small enough so that we can apply the backward Euler time stepping scheme[1] to equation (1) and obtain the discretized equation

$$h^n + \Delta t \mathcal{N}_x^\lambda h^n = h^{n-1}. \tag{2}$$

Here, $h^n(x) = h(t^n, x)$ is the hidden state of the system at time t^n. Approximating the nonlinear operator on the left-hand-side of equation (2) by a linear one we obtain

$$\mathcal{L}_x^\lambda h^n = h^{n-1}. \tag{3}$$

For instance, the nonlinear operator

$$h^n + \Delta t \mathcal{N}_x^\lambda h^n = h^n + \Delta t(\lambda_1 h^n h_x^n - \lambda_2 h_{xx}^n),$$

involved in the Burgers' equation can be approximated by the linear operator

$$\mathcal{L}_x^\lambda h^n = h^n + \Delta t(\lambda_1 h^{n-1} h_x^n - \lambda_2 h_{xx}^n),$$

where $h^{n-1}(x)$ is the state of the system at the previous time t^{n-1}.

[1] For a general treatment of arbitrary linear multi-step methods as well as Runge-Kutta time stepping schemes we would like to refer the readers to [1].

3. The Basic Model

Similar to Raissi et al. [3, 4], we build upon the analytical property of Gaussian processes that the output of a linear system whose input is Gaussian distributed is again Gaussian. Specifically, we proceed by placing a Gaussian process[2] prior over the latent function $h^n(x)$; i.e.,

$$h^n(x) \sim \mathcal{GP}(0, k(x, x', \theta)). \tag{4}$$

Here, θ denotes the hyper-parameters of the covariance function k. Without loss of generality, all Gaussian process priors used in this work are assumed to have a squared exponential[3] covariance function, i.e.,

$$k(x, x'; \theta) = \gamma^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D} w_d^2 (x_d - x'_d)^2\right),$$

where $\theta = (\gamma, w_1, \cdots, w_D)$ are the hyper-parameters and x is a D-dimensional vector. The Gaussian process prior assumption (4) along with equation (3) enable us to capture the entire structure of the operator $\mathcal{L}_x^\lambda$ in the resulting multi-output Gaussian process

$$\begin{bmatrix} h^n \\ h^{n-1} \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k^{n,n} & k^{n,n-1} \\ k^{n-1,n} & k^{n-1,n-1} \end{bmatrix}\right). \tag{5}$$

It is worth highlighting that the parameters λ of the operators $\mathcal{L}_x^\lambda$ and $\mathcal{N}_x^\lambda$ turn into hyper-parameters of the resulting covariance functions. The specific

[2] Gaussian processes (see [5, 6]) provide a flexible prior distribution over functions and enjoy analytical tractability. They can be viewed as a prior on one-layer feed-forward Bayesian neural networks with an infinite number of hidden units [7]. Gaussian processes are among a class of methods known as kernel machines (see [8, 9, 10]) and are analogous to regularization approaches (see [11, 12, 13]).

[3] From a theoretical point of view, each kernel (i.e., covariance function) gives rise to a Reproducing Kernel Hilbert Space (RKHS) [14, 15, 16] that defines a class of functions that can be represented by this kernel. In particular, the squared exponential covariance function implies smooth approximations. For a more systematic treatment of the kernel-selection problem we would like to refer the readers to [17, 18, 19]. Furthermore, more complex function classes can be accommodated by employing nonlinear warping of the input space to capture discontinuities [20, 21].

forms of the kernels[4]

$$\begin{aligned} &k^{n,n}(x,x';\theta), && k^{n,n-1}(x,x';\theta,\lambda),\\ &k^{n-1,n}(x,x';\theta,\lambda), && k^{n-1,n-1}(x,x';\theta,\lambda), \end{aligned}$$

are direct functions of equation (3) as well as the prior assumption (4); i.e.,

$$\begin{aligned} k^{n,n} &= k, & k^{n,n-1} &= \mathcal{L}_{x'}^{\lambda} k,\\ k^{n-1,n} &= \mathcal{L}_{x}^{\lambda} k, & k^{n-1,n-1} &= \mathcal{L}_{x}^{\lambda}\mathcal{L}_{x'}^{\lambda} k, \end{aligned}$$

We call the multi-output Gaussian process (5) a *hidden physics model*, because its matrix of covariance functions explicitly encodes the underlying laws of physics expressed by equations (1) and (3).

4. Learning

Given the noisy data $\{\boldsymbol{x}^{n-1}, \boldsymbol{h}^{n-1}\}$ and $\{\boldsymbol{x}^{n}, \boldsymbol{h}^{n}\}$ on the latent solution at times t^{n-1} and t^n, respectively, the hyper-parameters θ of the covariance functions and more importantly the parameters λ of the operators $\mathcal{L}_x^\lambda$ and $\mathcal{N}_x^\lambda$ can be learned by employing a Quasi-Newton optimizer L-BFGS [22] to minimize the negative log marginal likelihood [5]

$$-\log p(\boldsymbol{h}|\theta,\lambda,\sigma^2) = \frac{1}{2}\boldsymbol{h}^T\boldsymbol{K}^{-1}\boldsymbol{h} + \frac{1}{2}\log|\boldsymbol{K}| + \frac{N}{2}\log(2\pi), \tag{6}$$

where $\boldsymbol{h} = \begin{bmatrix}\boldsymbol{h}^n\\ \boldsymbol{h}^{n-1}\end{bmatrix}$, $p(\boldsymbol{h}|\theta,\lambda,\sigma^2) = \mathcal{N}(\boldsymbol{0},\boldsymbol{K})$, and $\boldsymbol{K}$ is given by

$$\boldsymbol{K} = \begin{bmatrix} k^{n,n}(\boldsymbol{x}^n,\boldsymbol{x}^n) & k^{n,n-1}(\boldsymbol{x}^n,\boldsymbol{x}^{n-1})\\ k^{n-1,n}(\boldsymbol{x}^{n-1},\boldsymbol{x}^n) & k^{n-1,n-1}(\boldsymbol{x}^{n-1},\boldsymbol{x}^{n-1}) \end{bmatrix} + \sigma^2\boldsymbol{I}.$$

Here, N is the total number of data points in $\boldsymbol{h}$. Moreover, σ^2 is included to capture the noise in the data and is also learned by minimizing the negative log marginal likelihood. The implicit underlying assumption is that $\boldsymbol{h}^n = h^n(\boldsymbol{x}^n) + \boldsymbol{\epsilon}^n$ and $\boldsymbol{h}^{n-1} = h^{n-1}(\boldsymbol{x}^{n-1}) + \boldsymbol{\epsilon}^{n-1}$ with $\boldsymbol{\epsilon}^n \sim \mathcal{N}(0,\sigma^2 I)$ and

[4] It should be noted that for all examples studied in this work the kernels are generated at the push of a button using Wolfram Mathematica, a mathematical symbolic computation program.

$\boldsymbol{\epsilon}^{n-1} \sim \mathcal{N}(0, \sigma^2 I)$ being independent. The negative log marginal likelihood (6) does not simply favor the models that fit the training data best. In fact, it induces an automatic trade-off between data-fit and model complexity. Specifically, minimizing the term $\boldsymbol{h}^T \boldsymbol{K}^{-1} \boldsymbol{h}$ in equation (6) targets fitting the training data, while the log-determinant term $\log |\boldsymbol{K}|$ penalizes model complexity. This regularization mechanism automatically meets the Occam's razor principle [23] which encourages simplicity in explanations. The aforementioned regularization mechanism of the negative log marginal likelihood (6) effectively guards against overfitting and enables learning the unknown model parameters from very few[5] noisy observations. However, there is no theoretical guarantee that the negative log marginal likelihood does not suffer from multiple local minima. Our practical experience so far with the negative log marginal likelihood seems to indicate that local minima are not a devastating problem, but certainly they do exist. Moreover, it should be highlighted that, although not pursued here, a fully Bayesian [25] and more robust estimate of the linear operator parameters λ can be obtained by assigning priors on $\{\theta, \lambda, \sigma^2\}$. However, this would require more costly sampling procedures such as Markov Chain Monte Carlo (see [5], chapter 5) to train the model. Furthermore, the most computationally intensive part of learning using the negative log marginal likelihood (6) is associated with inverting dense covariance matrices $\boldsymbol{K}$. This scales cubically with the number N of training data in $\boldsymbol{h}$. While it has been effectively addressed by the recent works of [26, 27, 28], this cubic scaling is still a well-known limitation of Gaussian process regression.

5. Results

The proposed framework provides a general treatment of time-dependent and nonlinear partial differential equations, which can be of fundamentally different nature. This generality will be demonstrated by applying the algorithm to a dataset originally proposed in [2], where sparse regression techniques are used to discover partial differential equations from time series measurements in the spatial domain. This dataset covers a wide range of canonical problems spanning a number of scientific domains including the

[5] Regularization is important even in data abundant regimes as witnessed by the recently growing literature on discovering ordinary and partial differential equations from data using sparse regression techniques [24, 2].

Navier-Stokes, Schrödinger, and Kuramoto-Sivashinsky equations. Moreover, all data and codes used in this manuscript are publicly available on GitHub at https://github.com/maziarraissi/HPM.

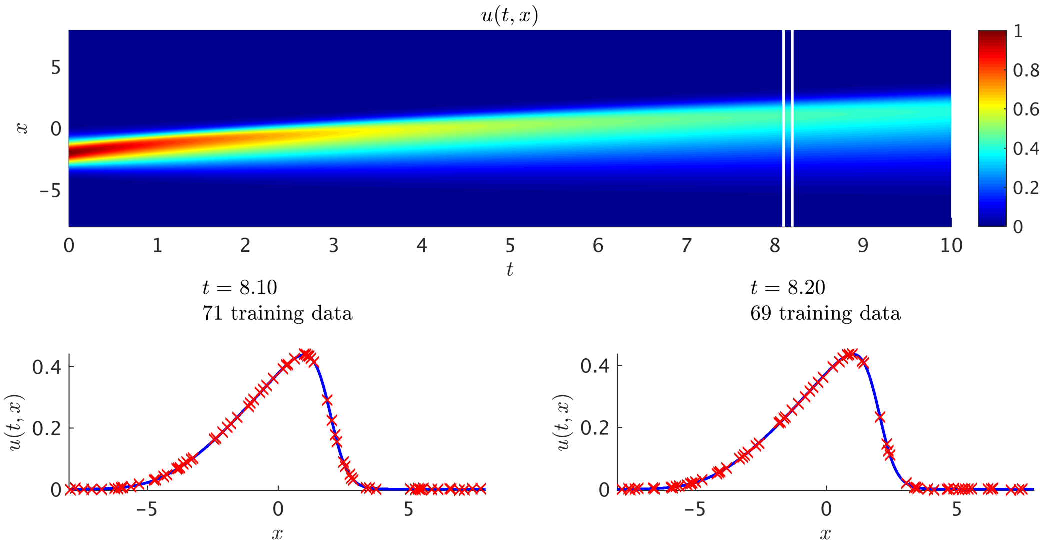
5.1. Burgers' Equation

Burgers' equation arises in various areas of applied mathematics, including fluid mechanics, nonlinear acoustics, gas dynamics, and traffic flow [29]. It is a fundamental partial differential equation and can be derived from the Navier-Stokes equations for the velocity field by dropping the pressure gradient term. Burgers' equation, despite its relation to the much more complicated Navier-Stokes equations, does not exhibit turbulent behavior. However, for small values of the viscosity parameters, Burgers' equation can lead to shock formation that is notoriously hard to resolve by classical numerical methods. In one space dimension the equation reads as

$$u_t + \lambda_1 u u_x - \lambda_2 u_{xx} = 0, \tag{7}$$

with (λ_1, λ_2) being the unknown parameters. The original data-set proposed in [2] contains 101 time snapshots of a solution to the Burgers' equation with a Gaussian initial condition, propagating into a traveling wave. The snapshots are $\Delta t = 0.1$ apart. The spatial discretization of each snapshot involves a uniform grid with 256 cells. As depicted in figure 1 using *only two of these snapshots* (randomly selected) with 71 and 69 data points, respectively, the algorithm is capable of identifying the correct parameter values up to a relatively good accuracy. It should be noted that we are using only $140 = 71 + 69$ data points out of a total of $25856 = 101 \times 256$ in the original data set. This surprising performance is achieved at the cost of explicitly encoding the underlying physical laws expressed by the Burgers' equation in the covariance functions of the *hidden physics model* (5). For a systematic study of the performance of the method, let us carry out the same experiment as the one illustrated in figure 1 for every pair of consecutive snapshots in the original dataset. We are still using the same number of data points (i.e., 71 and 69) for each pair of snapshots, albeit in different locations. The resulting statistics for the learned parameter values are reported in table 1. As is clearly demonstrated in this table, more noise in the data leads to less confidence in the estimated values for the parameters. Moreover, let us recall the main assumption of this work that the gap Δt between the pair of snapshots should be small enough so that we can employ the backward

Euler scheme (see equation (2)). To test the importance of this assumption, let us use the exact same setup as the one explained in figure 1, but increase Δt. The results are reported in table 2. Therefore, the most important facts about the proposed methodology are that more data, less noise, and a smaller gap Δt between the two snapshots enhance the performance of the algorithm.



Correct PDE	$u_t + uu_x - 0.1u_{xx} = 0$
Identified PDE (clean data)	$u_t + 1.028uu_x - 0.101u_{xx} = 0$
Identified PDE (1% noise)	$u_t + 1.017uu_x - 0.094u_{xx} = 0$

Figure 1: *Burgers' equation:* A solution to the Burgers' equation is depicted in the top panel. The two white vertical lines in this panel specify the locations of the two randomly selected snapshots. These two snapshots are $\Delta t = 0.1$ apart and are plotted in the middle panel. The red crosses denote the locations of the training data points. The correct partial differential equation along with the identified ones are reported in the lower panel.

	Clean Data		1% Noise		5% Noise	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
First Quartile	1.0247	0.0942	0.9168	0.0784	0.3135	0.0027
Median	1.0379	0.0976	1.0274	0.0919	0.8294	0.0981
Third Quartile	1.0555	0.0987	1.1161	0.1166	1.2488	0.1543

Table 1: *Burgers' equation:* Resulting statistics for the learned parameter values.

		$\Delta t = 0.1$	$\Delta t = 0.5$	$\Delta t = 1.0$	$\Delta t = 1.5$
Clean Data	λ_1	1.0283	1.1438	1.2500	1.2960
	λ_2	0.1009	0.0934	0.0694	0.0431
1% Noise	λ_1	1.0170	1.1470	1.2584	1.3063
	λ_2	0.0935	0.0939	0.0711	0.0428

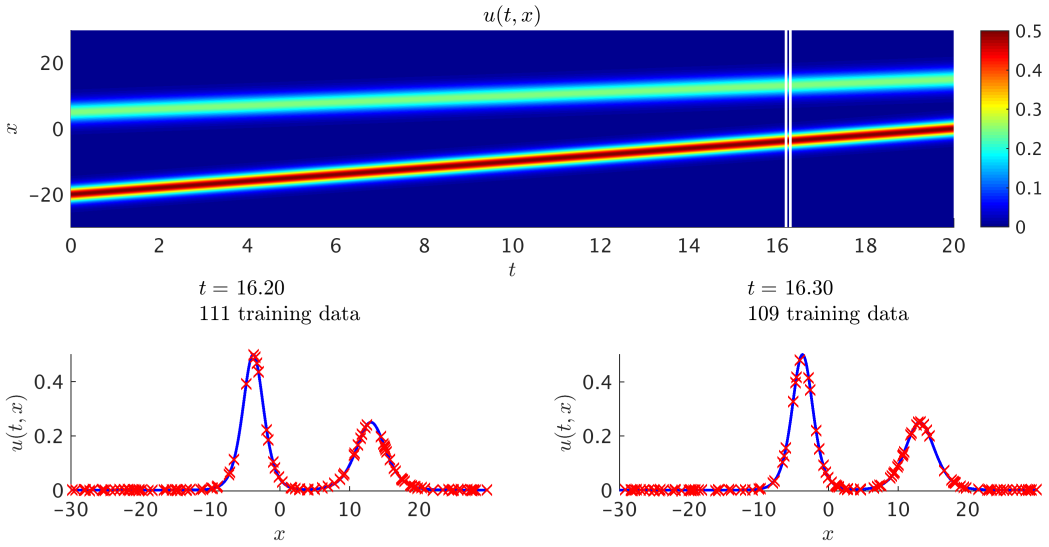
Table 2: *Burgers' equation:* Effect of increasing the gap Δt between the pair of snapshots.

5.2. The KdV Equation

As a mathematical model of waves on shallow water surfaces one could consider the Korteweg-de Vries (KdV) equation. This equation can also be viewed as Burgers' equation with an added dispersive term. The KdV equation has several connections to physical problems. It describes the evolution of long one-dimensional waves in many physical settings. Such physical settings include shallow-water waves with weakly non-linear restoring forces, long internal waves in a density-stratified ocean, ion acoustic waves in a plasma, and acoustic waves on a crystal lattice. Moreover, the KdV equation is the governing equation of the string in the Fermi-Pasta-Ulam problem [30] in the continuum limit. The KdV equation reads as

$$u_t + \lambda_1 u u_x + \lambda_2 u_{xxx} = 0, \tag{8}$$

with (λ_1, λ_2) being the unknown parameters. The original dataset proposed in [2] contains a two soliton solution to the KdV equation with 512 spatial points and 201 time-steps. The snapshots are $\Delta t = 0.1$ apart. As depicted in figure 2 using only two of these snapshots (randomly selected) with 111 and 109 data points, respectively, the algorithm is capable of identifying the correct parameter values up to a relatively good accuracy. In particular, we are using $220 = 111 + 109$ out of a total of $102912 = 201 \times 512$ data points in the original data set. This level of efficiency is a direct consequence of equation (5) where the covariance functions explicitly encode the underlying physical laws expressed by the KdV equation. As a sensitivity analysis of the reported results, let us perform the same experiment as the one illustrated in figure 2 for every pair of consecutive snapshots in the original dataset. We are still using the same number of data points (i.e., 111 and 109) for each pair of snapshots, albeit in different locations. The resulting statistics for the learned parameter values are reported in table 3. As is clearly demonstrated in this table, more noise in the data leads to less confidence in the estimated values for the parameters. Moreover, to test the sensitivity of the results with respect to the gap between the two time snapshots, let us use the exact same setup as the one explained in figure 2, but increase Δt. The results are reported in table 4. These results verify the most important facts about the proposed methodology that more data, less noise, and a smaller gap Δt between the two snapshots enhance the performance of the algorithm.



Correct PDE	$u_t + 6uu_x + u_{xxx} = 0$
Identified PDE (clean data)	$u_t + 6.115uu_x + 1.047u_{xxx} = 0$
Identified PDE (1% noise)	$u_t + 5.722uu_x + 0.958u_{xxx} = 0$

Figure 2: *The KdV equation:* A solution to the KdV equation is depicted in the top panel. The two white vertical lines in this panel specify the locations of the two randomly selected snapshots. These two snapshots are $\Delta t = 0.1$ apart and are plotted in the middle panel. The red crosses denote the locations of the training data points. The correct partial differential equation along with the identified ones are reported in the lower panel.

	Clean Data		1% Noise		5% Noise	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
First Quartile	5.7783	0.9299	5.3358	0.7885	3.7435	0.2280
Median	5.8920	0.9656	5.5757	0.8777	4.5911	0.6060
Third Quartile	6.0358	1.0083	5.7840	0.9491	5.5106	0.8407

Table 3: *The KdV equation:* Resulting statistics for the learned parameter values.

		$\Delta t = 0.1$	$\Delta t = 0.2$	$\Delta t = 0.3$	$\Delta t = 0.4$	$\Delta t = 0.5$
Clean Data	λ_1	6.1145	5.8948	5.4014	4.1779	3.5058
	λ_2	1.0470	0.9943	0.8535	0.4475	0.1816
1% Noise	λ_1	5.7224	5.8288	5.4054	4.1479	3.4747
	λ_2	0.9578	0.9801	0.8563	0.4351	0.1622

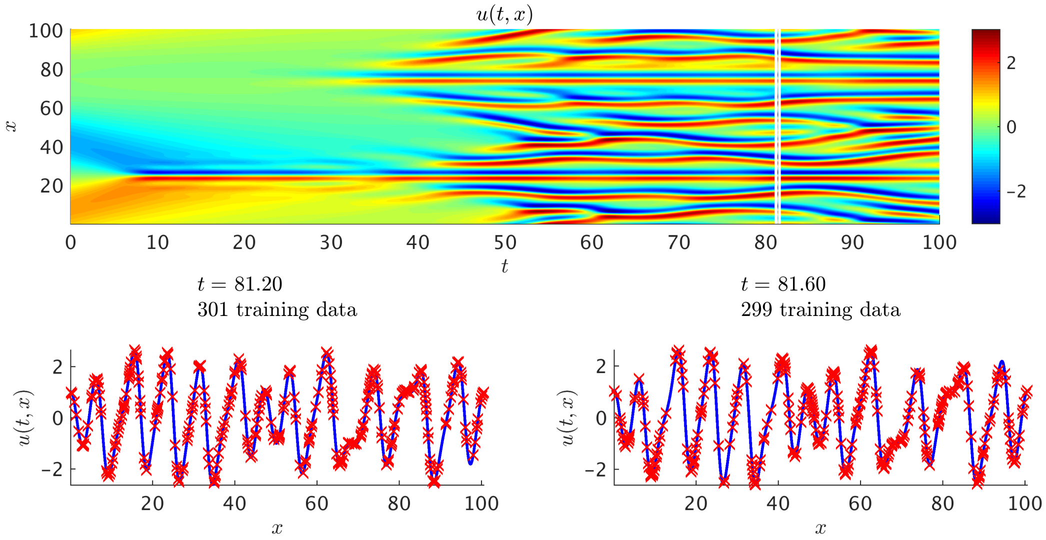
Table 4: *The KdV equation:* Effect of increasing the gap Δt between the pair of snapshots.

5.3. Kuramoto-Sivashinsky Equation

The Kuramoto-Sivashinsky equation [31, 32, 33] has similarities with Burgers' equation. However, because of the presence of both second and fourth order spatial derivatives, its behavior is far more complicated and interesting. The Kuramoto-Sivashinsky is a canonical model of a pattern forming system with spatio-temporal chaotic behavior. The sign of the second derivative term is such that it acts as an energy source and thus has a destabilizing effect. The nonlinear term, however, transfers energy from low to high wave numbers where the stabilizing fourth derivative term dominates. The first derivation of this equation was by Kuramoto in the study of reaction-diffusion equations modeling the Belousov-Zabotinskii reaction. The equation was also developed by Sivashinsky in higher space dimensions in modeling small thermal diffusive instabilities in laminar flame fronts and in small perturbations from a reference Poiseuille flow of a film layer on an inclined plane. In one space dimension it has also been used as a model for the problem of Bénard convection in an elongated box, and it may be used to describe long waves on the interface between two viscous fluids and unstable drift waves in plasmas. In one space dimension the Kuramoto-Sivashinsky equation reads as

$$u_t + \lambda_1 u u_x + \lambda_2 u_{xx} + \lambda_3 u_{xxxx} = 0, \tag{9}$$

where $(\lambda_1, \lambda_2, \lambda_3)$ are the unknown parameters. The original dataset proposed in [2] contains a direct numerical solution of the Kuramoto-Sivashinsky equation with 1024 spatial points and 251 time-steps. The snapshots are $\Delta t = 0.4$ apart. As depicted in figure 3 using only two of these snapshots (randomly selected) with 301 and 299 data points, respectively, the algorithm is capable of identifying the correct parameter values up to a relatively good accuracy. In particular, we are using $600 = 301 + 299$ out of a total of $257024 = 251 \times 1024$ data points in the original data set. This is possible because of equation (5) where the covariance functions explicitly encode the underlying physical laws expressed by the Kuramoto-Sivashinsky equation. For a sensitivity analysis of the reported results, let us perform the same experiment as the one illustrated in figure 3 for every pair of consecutive snapshots in the original dataset. We are still using the same number of data points (i.e., 301 and 299) for each pair of snapshots, albeit in different locations. The resulting statistics for the learned parameter values are reported in table 5. As shown in this table, more noise in the data leads to less confidence in the estimated parameter values. Moreover, to test the sensitivity of the results with respect to the gap between the two time snapshots, let us use the exact same setup as the one explained in figure 3, but increase Δt. The results are reported in table 6. These results indicate that more data, less noise, and a smaller gap Δt between the two snapshots enhance the performance of the algorithm.



Correct PDE	$u_t + uu_x + u_{xx} + u_{xxxx} = 0$
Identified PDE (clean data)	$u_t + 0.952uu_x + 1.005u_{xx} + 0.980u_{xxxx} = 0$
Identified PDE (1% noise)	$u_t + 0.908uu_x + 0.951u_{xx} + 0.927u_{xxxx} = 0$

Figure 3: *Kuramoto-Sivashinsky equation:* A solution to the Kuramoto-Sivashinsky equation is depicted in the top panel. The two white vertical lines in this panel specify the locations of the two randomly selected snapshots. These two snapshots are $\Delta t = 0.4$ apart and are plotted in the middle panel. The red crosses denote the locations of the training data points. The correct partial differential equation along with the identified ones are reported in the lower panel.

	Clean Data			1% Noise			5% Noise		
	λ_1	λ_2	λ_3	λ_1	λ_2	λ_3	λ_1	λ_2	λ_3
First Quartile	0.9603	0.9829	0.9711	0.7871	0.8095	0.5891	-0.0768	0.0834	-0.0887
Median	0.9885	1.0157	0.9970	0.8746	0.9124	0.8798	0.4758	0.5539	0.4086
Third Quartile	1.0187	1.0550	1.0314	0.9565	0.9948	0.9553	0.6991	0.7644	0.7009

Table 5: *Kuramoto-Sivashinsky equation:* Resulting statistics for the learned parameter values.

		$\Delta t = 0.4$	$\Delta t = 0.8$	$\Delta t = 1.2$
Clean Data	λ_1	0.9515	0.5299	0.1757
	λ_2	1.0052	0.5614	0.1609
	λ_3	0.9803	0.5438	0.1647
1% Noise	λ_1	0.9081	0.5124	0.1616
	λ_2	0.9511	0.5387	0.1436
	λ_3	0.9266	0.5213	0.1483

Table 6: *Kuramoto-Sivashinsky equation:* Effect of increasing the gap Δt between the pair of snapshots.

5.4. Nonlinear Schrödinger Equation

The one-dimensional nonlinear Schrödinger equation is a classical field equation that is used to study nonlinear wave propagation in optical fibers and/or waveguides, Bose-Einstein condensates, and plasma waves. In optics, the nonlinear term arises from the intensity dependent index of refraction of a given material. Similarly, the nonlinear term for Bose-Einstein condensates is a result of the mean-field interactions of an interacting, N-body system. The nonlinear Schrödinger equation is given by

$$ih_t + \lambda_1 h_{xx} + \lambda_2 |h|^2 h = 0, \tag{10}$$

where (λ_1, λ_2) are the unknown parameters. Let u denote the real part of h and v the imaginary part. Then, the nonlinear Schrödinger equation can be equivalently written as

$$\begin{aligned} u_t + \lambda_1 v_{xx} + \lambda_2 (u^2 + v^2) v = 0, \\ v_t - \lambda_1 u_{xx} - \lambda_2 (u^2 + v^2) u = 0. \end{aligned} \tag{11}$$

Employing the backward Euler time stepping scheme, we obtain

$$\begin{aligned} u^n + \Delta t \lambda_1 v^n_{xx} + \Delta t \lambda_2 [(u^n)^2 + (v^n)^2] v^n &= u^{n-1}, \\ v^n - \Delta t \lambda_1 u^n_{xx} - \Delta t \lambda_2 [(u^n)^2 + (v^n)^2] u^n &= v^{n-1}. \end{aligned} \tag{12}$$

The above equations can be approximated by

$$\begin{aligned} u^n + \Delta t \lambda_1 v^n_{xx} + \Delta t \lambda_2 [(u^{n-1})^2 + (v^{n-1})^2] v^n &= u^{n-1}, \\ v^n - \Delta t \lambda_1 u^n_{xx} - \Delta t \lambda_2 [(u^{n-1})^2 + (v^{n-1})^2] u^n &= v^{n-1}, \end{aligned} \tag{13}$$

which involves only linear operations. Here, $u^{n-1}(x)$ and $v^{n-1}(x)$ are the real and imaginary parts of the state of the system at the previous time step, respectively. We proceed by placing two independent Gaussian processes on $u^n(x)$ and $v^n(x)$; i.e.,

$$\begin{aligned} u^n(x) &\sim \mathcal{GP}(0, k_u(x, x'; \theta_u)), \\ v^n(x) &\sim \mathcal{GP}(0, k_v(x, x'; \theta_v)). \end{aligned} \tag{14}$$

Here, θ_u and θ_v are the hyper-parameters of the kernels k_u and k_v, respectively. The prior assumptions (14) along with equations (13) enable us to encode the underlying laws of physics expressed by the nonlinear Schrödinger equation in the resulting *hidden physics model*

$$\begin{bmatrix} u^n \\ v^n \\ u^{n-1} \\ v^{n-1} \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k^{n,n}_{u,u} & k^{n,n}_{u,v} & k^{n,n-1}_{u,u} & k^{n,n-1}_{u,v} \\ k^{n,n}_{v,u} & k^{n,n}_{v,v} & k^{n,n-1}_{v,u} & k^{n,n-1}_{v,v} \\ k^{n-1,n}_{u,u} & k^{n-1,n}_{u,v} & k^{n-1,n-1}_{u,u} & k^{n-1,n-1}_{u,v} \\ k^{n-1,n}_{v,u} & k^{n-1,n}_{v,v} & k^{n-1,n-1}_{v,u} & k^{n-1,n-1}_{v,v} \end{bmatrix}\right). \tag{15}$$

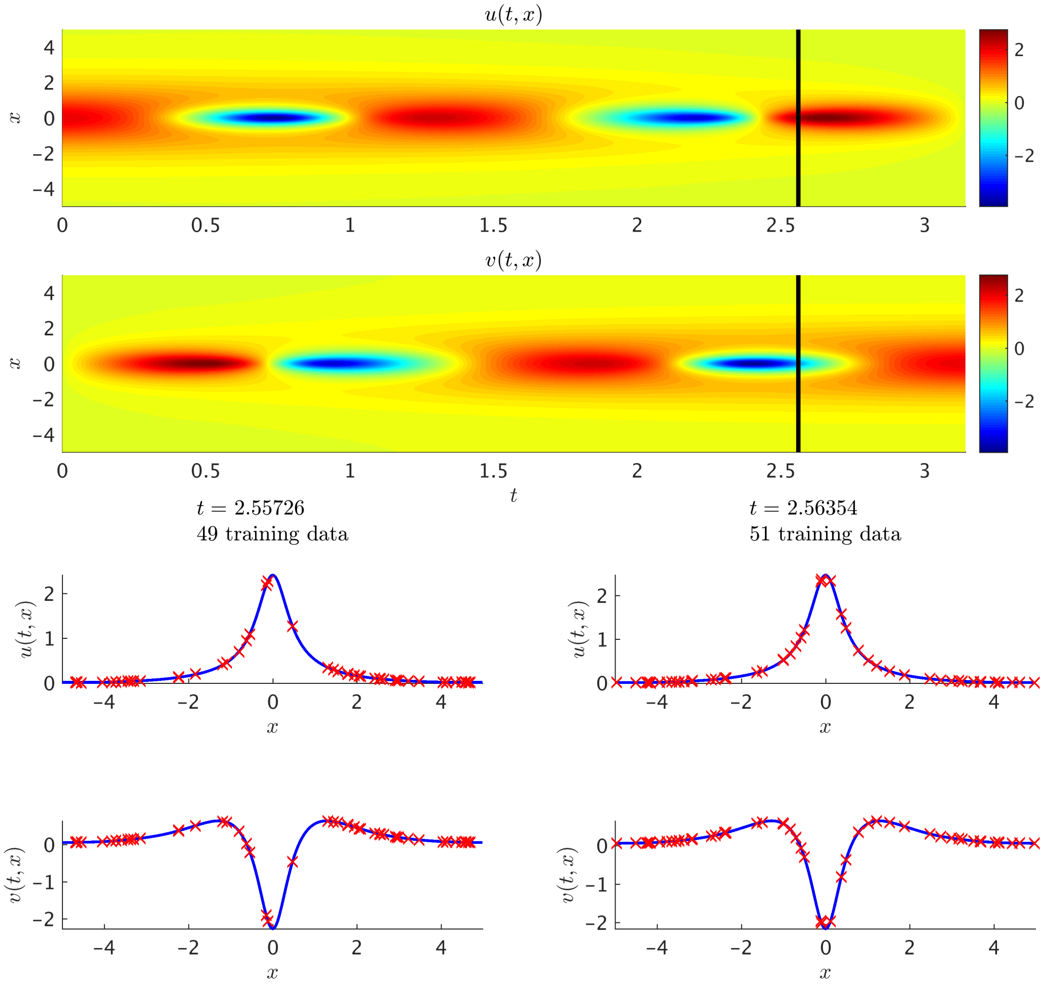
The specific forms of the covariance functions involved in model (15) is a direct function of the prior assumptions (14) as well as equations (13). The hyper-parameters θ_u and θ_v along with the parameters λ_1 and λ_2 are learned by minimizing the negative log marginal likelihood as outlined in section 4. The original data-set proposed in [2] contains 501 time snapshots of a solution to the nonlinear Schrödinger equation with a Gaussian initial condition. The snapshots are $\Delta t = 0.0063$ apart. The spatial discretization of each snapshot involves a uniform grid with 512 elements. As depicted in figure 4 using only two of these snapshots (randomly selected) with 49 and 51 data points, respectively, the algorithm is capable of identifying the correct parameter values up to a relatively good accuracy. It should be noted that we are using only $100 = 49 + 51$ data points out of a total of $256512 = 501 \times 512$ in the original data set. Such a performance is achieved at the cost of explicitly encoding the underlying physical laws expressed by the nonlinear Schrödinger equation in the covariance functions of the *hidden physics model* (15). For a systematic study of the performance of the method, let us carry out the same experiment as the one illustrated in figure 4 for every pair of consecutive snapshots in the original dataset. We are still using the same number of data points (i.e., 49 and 51) for each pair of snapshots. The resulting statistics for the learned parameter values are reported in table 7. As is clearly demonstrated in this table, more noise in the data leads to less confidence in the estimated values for the parameters. Moreover, let us recall the main assumption of this work that the gap Δt between the pair of snapshots should be small enough so that we can employ the backward Euler scheme (see equation (12)). To test the importance of this assumption, let us use the exact same setup as the one explained in figure 4, but increase Δt. The results are reported in table 8. Therefore, the most important facts about the proposed methodology are that more data, less noise, and a smaller gap Δt between the two snapshots enhance the performance of the algorithm.

	Clean Data		1% Noise		5% Noise	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
First Quartile	0.4950	0.9960	0.3714	0.9250	-0.1186	0.6993
Median	0.5009	1.0001	0.4713	0.9946	0.4259	0.9651
Third Quartile	0.5072	1.0039	0.5918	1.0670	0.9730	1.2730

Table 7: *Nonlinear Schrödinger equation:* Resulting statistics for the learned parameter values.

		$\Delta t = 0.0063$	$\Delta t = 0.0628$	$\Delta t = 0.1257$	$\Delta t = 0.1885$
Clean Data	λ_1	0.5062	0.4981	0.3887	0.3097
	λ_2	0.9949	0.8987	0.7936	0.7221
1% Noise	λ_1	0.4758	0.4976	0.3928	0.3128
	λ_2	0.9992	0.9011	0.7975	0.7255

Table 8: *Nonlinear Schrödinger equation:* Effect of increasing the gap Δt between the pair of snapshots.



Correct PDE	$ih_t + 0.5h_{xx} + \|h\|^2 h = 0$
Identified PDE (clean data)	$ih_t + 0.506h_{xx} + 0.995\|h\|^2 h = 0$
Identified PDE (1% noise)	$ih_t + 0.476h_{xx} + 0.999\|h\|^2 h = 0$

Figure 4: *Nonlinear Schrödinger equation:* A solution to the nonlinear Schrödinger equation is depicted in the top two panels. The two black vertical lines in these two panels specify the locations of the two randomly selected snapshots. These two snapshots are $\Delta t = 0.0063$ apart and are plotted in the two middle panels. The red crosses denote the locations of the training data points. The correct partial differential equation along with the identified ones are reported in the lower panel. Here, u is the real part of h and v is the imaginary part.

5.5. Navier-Stokes Equations

Navier-Stokes equations describe the physics of many phenomena of scientific and engineering interest. They may be used to model the weather, ocean currents, water flow in a pipe and air flow around a wing. The Navier-Stokes equations in their full and simplified forms help with the design of aircraft and cars, the study of blood flow, the design of power stations, the analysis of the dispersion of pollutants, and many other applications. Let us consider the Navier-Stokes equations in two dimensions[6] (2D) given explicitly by

$$\begin{aligned} u_t + \lambda_1(uu_x + vu_y) &= -p_x + \lambda_2(u_{xx} + u_{yy}), \\ v_t + \lambda_1(uv_x + vv_y) &= -p_y + \lambda_2(v_{xx} + v_{yy}), \end{aligned} \tag{16}$$

where $u(t,x,y)$ denotes the x-component of the velocity field, $v(t,x,y)$ the y-component, and $p(t,x,y)$ the pressure. Here, $\lambda = (\lambda_1, \lambda_2)$ are the unknown parameters. Solutions to the Navier-Stokes equations are searched in the set of divergence-free functions; i.e.,

$$u_x + v_y = 0. \tag{17}$$

This extra equation is the continuity equation for incompressible fluids that describes the conservation of mass of the fluid. Applying the backward Euler time stepping scheme to the Navier-Stokes equations (16) we obtain

$$\begin{aligned} u^n + \Delta t\lambda_1(u^n u^n_x + v^n u^n_y) + \Delta t p^n_x - \Delta t\lambda_2(u^n_{xx} + u^n_{yy}) &= u^{n-1}, \\ v^n + \Delta t\lambda_1(u^n v^n_x + v^n v^n_y) + \Delta t p^n_y - \Delta t\lambda_2(v^n_{xx} + v^n_{yy}) &= v^{n-1}, \end{aligned} \tag{18}$$

where $u^n(x,y) = u(t^n,x,y)$ and $v^n(x,y) = v(t^n,x,y)$. We make the assumption that

$$u^n = \psi^n_y, \quad v^n = -\psi^n_x, \tag{19}$$

for some latent function $\psi^n(x,y)$. Under this assumption, the continuity equation (17) will be automatically satisfied. We proceed by placing a Gaussian process prior on

$$\psi^n(x,y) \sim \mathcal{GP}\left(0, k((x,y),(x',y');\theta)\right), \tag{20}$$

[6] It is straightforward to generalize the proposed framework to the Navier-Stokes equations in three dimensions (3D).

where θ are the hyper-parameters of the kernel $k((x,y),(x',y');\theta)$. This will result in the following multi-output Gaussian process

$$\begin{bmatrix} u^n \\ v^n \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k_{u,u}^{n,n} & k_{u,v}^{n,n} \\ k_{v,u}^{n,n} & k_{v,v}^{n,n} \end{bmatrix}\right), \tag{21}$$

where

$$\begin{aligned} k_{u,u}^{n,n} &= \frac{\partial}{\partial y}\frac{\partial}{\partial y'}k, & k_{u,v}^{n,n} &= -\frac{\partial}{\partial y}\frac{\partial}{\partial x'}k, \\ k_{v,u}^{n,n} &= -\frac{\partial}{\partial x}\frac{\partial}{\partial y'}k, & k_{v,v}^{n,n} &= \frac{\partial}{\partial x}\frac{\partial}{\partial x'}k. \end{aligned}$$

By construction (see equation (19)), any samples generated from this multi-output Gaussian process will satisfy the continuity equation (17). Moreover, independent from $\psi^n(x,y)$, we will place a Gaussian process prior on $p^n(x,y)$; i.e.,

$$p^n(x,y) \sim \mathcal{GP}(0, k_{p,p}^{n,n}((x,y),(x',y');\theta_p)). \tag{22}$$

We linearize the backward Euler time stepping scheme by employing the states $u^{n-1}(x,y)$ and $v^{n-1}(x,y)$ of the system at the previous time step and writing

$$\begin{aligned} u^n + \Delta t\lambda_1(u^{n-1}u_x^n + v^{n-1}u_y^n) + \Delta t p_x^n - \Delta t\lambda_2(u_{xx}^n + u_{yy}^n) = u^{n-1}, \\ v^n + \Delta t\lambda_2(u^{n-1}v_x^n + v^{n-1}v_y^n) + \Delta t p_y^n - \Delta t\lambda_2(v_{xx}^n + v_{yy}^n) = v^{n-1}. \end{aligned} \tag{23}$$

The above equations (23) can be rewritten as

$$\begin{aligned} \mathcal{L}_{(x,y)}^{\lambda} u^n + \Delta t p_x^n = u^{n-1}, \\ \mathcal{L}_{(x,y)}^{\lambda} v^n + \Delta t p_y^n = v^{n-1}, \end{aligned} \tag{24}$$

by defining the linear operator $\mathcal{L}_{(x,y)}^{\lambda}$ to be given by

$$\mathcal{L}_{(x,y)}^{\lambda} h := h + \Delta t\lambda_1(u^{n-1}h_x + v^{n-1}h_y) - \Delta t\lambda_2(h_{xx} + h_{yy}). \tag{25}$$

This will allow us to obtain the following *hidden physics model* encoding the structure of the Navier-Stokes equations and the backward Euler time stepping scheme in its kernels; i.e.,

$$\begin{bmatrix} u^n \\ v^n \\ p^n \\ u^{n-1} \\ v^{n-1} \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k_{u,u}^{n,n} & k_{u,v}^{n,n} & 0 & k_{u,u}^{n,n-1} & k_{u,v}^{n,n-1} \\ & k_{v,v}^{n,n} & 0 & k_{v,u}^{n,n-1} & k_{v,v}^{n,n-1} \\ & & k_{p,p}^{n,n} & k_{p,u}^{n,n-1} & k_{p,v}^{n,n-1} \\ & & & k_{u,u}^{n-1,n-1} & k_{u,v}^{n-1,n-1} \\ & & & & k_{v,v}^{n-1,n-1} \end{bmatrix}\right), \tag{26}$$

where

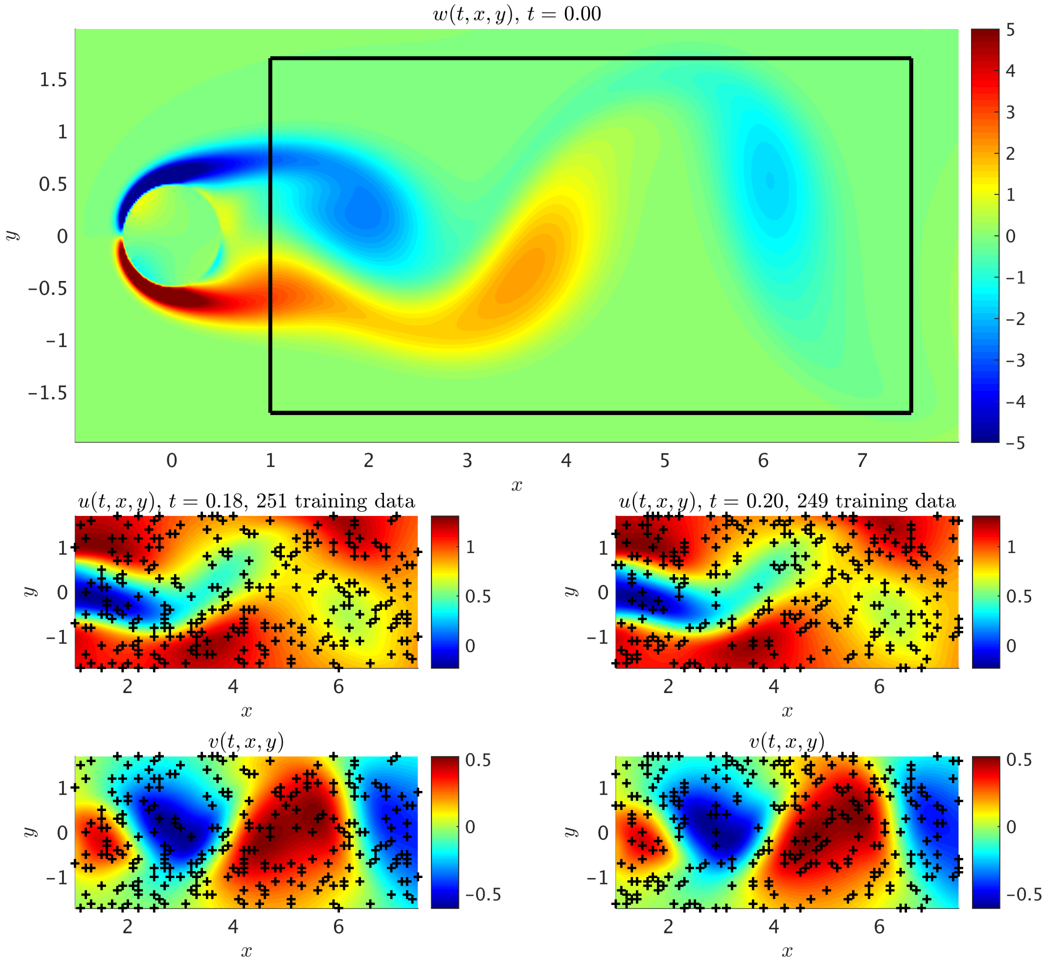
$$\begin{aligned} k_{u,u}^{n,n-1} &= \mathcal{L}_{(x',y')}^{\lambda} k_{u,u}^{n,n}, & k_{u,v}^{n,n-1} &= \mathcal{L}_{(x',y')}^{\lambda} k_{u,v}^{n,n}, \\ k_{v,u}^{n,n-1} &= \mathcal{L}_{(x',y')}^{\lambda} k_{v,u}^{n,n}, & k_{v,v}^{n,n-1} &= \mathcal{L}_{(x',y')}^{\lambda} k_{v,v}^{n,n}, \\ k_{p,u}^{n,n-1} &= \Delta t \frac{\partial}{\partial x'} k_{p,p}^{n,n}, & k_{p,v}^{n,n-1} &= \Delta t \frac{\partial}{\partial y'} k_{p,p}^{n,n}, \end{aligned}$$

and

$$\begin{aligned} k_{u,u}^{n-1,n-1} &= \mathcal{L}_{(x,y)}^{\lambda} k_{u,u}^{n,n-1} + \Delta t \frac{\partial}{\partial x} k_{p,u}^{n,n-1}, \\ k_{u,v}^{n-1,n-1} &= \mathcal{L}_{(x,y)}^{\lambda} k_{u,v}^{n,n-1} + \Delta t \frac{\partial}{\partial x} k_{p,v}^{n,n-1}, \\ k_{v,v}^{n-1,n-1} &= \mathcal{L}_{(x,y)}^{\lambda} k_{v,v}^{n,n-1} + \Delta t \frac{\partial}{\partial y} k_{p,v}^{n,n-1}. \end{aligned}$$

The lower triangular portion of the matrix of covariance functions (26) is not shown due to symmetry. The hyper-parameters θ and θ_p along with the parameters $\lambda = (\lambda_1, \lambda_2)$ are learned by minimizing the negative log marginal likelihood as outlined in section 4. As for the data, following the exact same instructions as the ones provided in [34] and [2], we simulate the Navier-Stokes equations describing the two-dimensional fluid flow past a circular cylinder at Reynolds number 100 using the Immersed Boundary Projection Method [35, 36]. This approach utilizes a multi-domain scheme with four nested domains, each successive grid being twice as large as the previous one. Length and time are nondimensionalized so that the cylinder has unit diameter and the flow has unit velocity. Data is collected on the finest domain with dimensions 9×4 at a grid resolution of 449×199. The flow solver uses a 3rd-order Runge Kutta integration scheme with a time step of t = 0.02, which has been verified to yield well-resolved and converged flow fields. After simulations converge to steady periodic vortex shedding, flow snapshots are saved every $\Delta t = 0.02$. As depicted in figure 5 using only two snapshots of the velocity[7] field with 251 and 249 data points, respectively, the algorithm is capable of identifying the correct parameter values up to a relatively good accuracy. It should be noted that we are using only two snapshots with a total of $500 = 251 + 249$ data points. This surprising performance is achieved at the cost of explicitly encoding the underlying physical laws expressed by the Navier-Stokes equations in the covariance functions of the *hidden physics model* (26). For a sensitivity analysis of the reported results, let us perform the same experiment as the one illustrated in figure 5 for 501 pairs of consecutive snapshots. We are still using the same number of data points (i.e., 251 and 249) for each pair of snapshots. The resulting statistics for the learned parameter values are reported in table 9. As is clearly demonstrated in this table, more noise in the data leads to less confidence in the estimated values for the parameters. Moreover, to test the sensitivity of the results with respect to the gap between two time snapshots, let us use the exact same setup as the one explained in figure 5, but increase Δt. The results are reported in table 10. These results verify the most important facts about the proposed methodology that more data, less noise, and a smaller gap Δt between the two snapshots enhance the performance of the algorithm. In particular, the results reported in table 10 indicate that to obtain more accurate estimates of the Reynolds number $1/\lambda_2$ one needs to utilize a smaller gap Δt between the pair of snapshots. To verify the validity of this conjecture let us decrease the gap Δt between the pair of time snapshots while employing the exact same setup as the one explained in figure 5. The results are reported in table 11. As is clearly demonstrated in this table, a smaller Δt leads to more accurate estimates of the Reynolds number $1/\lambda_2$ in the absence of noise in the data. However, a smaller Δt seems to make the algorithm more susceptible to noise in the data.

[7] It is worth emphasizing that we are not making use of any data on the pressure or vorticity fields. In practice, unlike velocity (e.g., Particle Image Velocimetry (PIV) data), obtaining direct measurements of the pressure or vorticity fields are more demanding if not impossible. Our method circumvents the need for having data on the pressure simply because of the prior assumption (21) where any samples generated from this multi-output Gaussian process satisfy the continuity equation (17).



Correct PDE	$u_t + (uu_x + vu_y) = -p_x + 0.01(u_{xx} + u_{yy})$ $v_t + (uv_x + vv_y) = -p_y + 0.01(v_{xx} + v_{yy})$
Identified PDE (clean data)	$u_t + 0.983(uu_x + vu_y) = -p_x + 0.00826(u_{xx} + u_{yy})$ $v_t + 0.983(uv_x + vv_y) = -p_y + 0.00826(v_{xx} + v_{yy})$
Identified PDE (1% noise)	$u_t + 0.849(uu_x + vu_y) = -p_x + 0.01399(u_{xx} + u_{yy})$ $v_t + 0.849(uv_x + vv_y) = -p_y + 0.01399(v_{xx} + v_{yy})$

Figure 5: *Navier-Stokes equations:* A single snapshot of the vorticity field of a solution to the Navier-Stokes equations for the fluid flow past a cylinder is depicted in the top panel. The black box in this panel specifies the sampling region. Two snapshots of the velocity field being $\Delta t = 0.02$ apart are plotted in the two middle panels. The black crosses denote the locations of the training data points. The correct partial differential equation along with the identified ones are reported in the lower panel. Here, u denotes the x-component of the velocity field, v the y-component, p the pressure, and w the vorticity field.

	Clean Data		1% Noise		5% Noise	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
First Quartile	0.9854	0.0069	0.8323	0.0057	0.5373	0.0026
Median	0.9928	0.0077	0.8717	0.0063	0.6498	0.0030
Third Quartile	1.0001	0.0086	0.9102	0.0070	0.7619	0.0046

Table 9: *Navier-Stokes equations:* Resulting statistics for the learned parameter values.

		$\Delta t = 0.02$	$\Delta t = 0.04$	$\Delta t = 0.06$	$\Delta t = 0.08$	$\Delta t = 1.0$
Clean Data	λ_1	0.9834	0.9925	0.9955	0.9976	1.0021
	λ_2	0.0083	0.0072	0.0058	0.0040	0.0027
1% Noise	λ_1	0.8488	0.9298	0.9597	0.9726	0.9791
	λ_2	0.0140	0.0110	0.0088	0.0069	0.0053

Table 10: *Navier-Stokes equations:* Effect of increasing the gap Δt between the pair of snapshots.

		$\Delta t = 0.02$	$\Delta t = 0.01$	$\Delta t = 0.005$
Clean Data	λ_1	0.9834	0.9688	0.9406
	λ_2	0.0083	0.0091	0.0104
1% Noise	λ_1	0.8488	0.7384	0.6107
	λ_2	0.0140	0.0159	0.0217

Table 11: *Navier-Stokes equations:* Effect of decreasing the gap Δt between the pair of snapshots.

5.6. Fractional Equations

Let us consider the one dimensional fractional equation

$$u_t - \lambda_1 \mathcal{D}^{\lambda_2}_{-\infty,x} u = 0, \tag{27}$$

where (λ_1, λ_2) are the unknown parameters. In particular, λ_2 is the fractional order of the operator $\mathcal{D}^{\lambda_2}_{-\infty,x}$ that is defined in the Riemann-Liouville sense [37]. Fractional operators often arise in modeling anomalous diffusion processes and other non-local interactions. Integer values such as $\lambda_2 = 1$ and $\lambda_2 = 2$ can model classical advection and diffusion phenomena, respectively. However, under the fractional calculus setting, λ_2 can assume real values and thus continuously interpolate between inherently different model behaviors. The proposed framework allows λ_2 to be directly inferred from noisy data, and opens the path to a flexible formalism for model discovery and calibration. Applying the backward Euler time stepping scheme to equation (27) we obtain

$$u^n - \Delta t \lambda_1 \mathcal{D}^{\lambda_2}_{-\infty,x} u^n = u^{n-1}. \tag{28}$$

Here, $u^n(x) = u(t^n, x)$ is the hidden state of the system at time t^n. We make the prior assumption that

$$u^n(x) \sim \mathcal{GP}(0, k(x, x'; \theta)). \tag{29}$$

The prior assumption (29) along with the backward Euler scheme (28) allow us to obtain the following *hidden physics model* corresponding to the fractional equation (27); i.e.,

$$\begin{bmatrix} u^n \\ u^{n-1} \end{bmatrix} \sim \mathcal{GP}\left(0, \begin{bmatrix} k^{n,n} & k^{n,n-1} \\ k^{n-1,n} & k^{n-1,n-1} \end{bmatrix}\right). \tag{30}$$

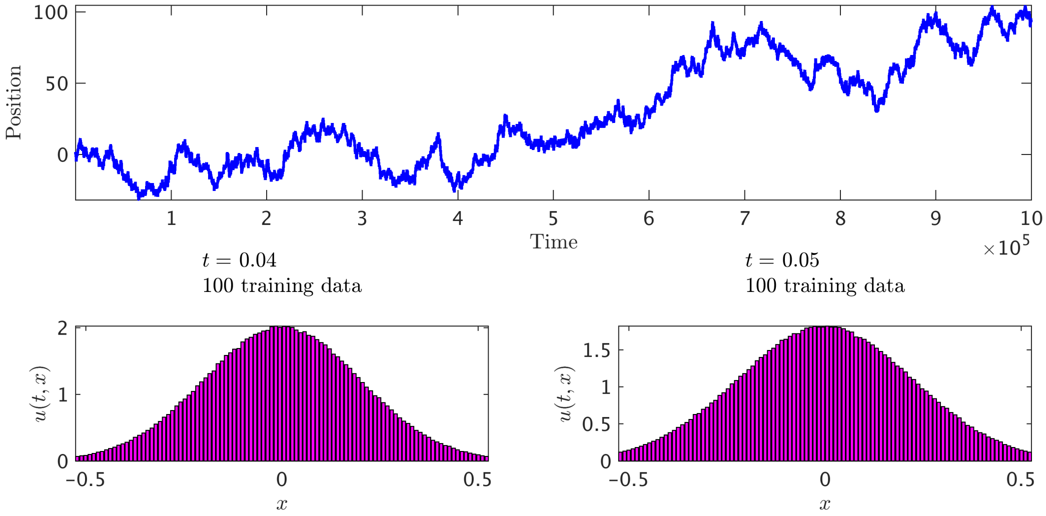
The only technicality induced by fractional operators has to do with deriving the kernels $k^{n,n-1}$, $k^{n-1,n}$, and $k^{n-1,n-1}$. Here, $k^{n,n-1}(x, x'; \theta, \lambda_1, \lambda_2)$ was obtained by taking the inverse Fourier transform [37] of

$$[1 - \Delta t \lambda_1 (-iw')^{\lambda_2}] \widehat{k}(w, w'; \theta),$$

where $\widehat{k}(w, w'; \theta)$ is the Fourier transform of the kernel $k(x, x'; \theta)$. Similarly, one can obtain $k^{n-1,n}$ and $k^{n-1,n-1}$. The hyper-parameters θ along with the parameters λ_1 and λ_2 are learned by minimizing the negative log marginal likelihood as outlined in section 4. We use the hidden physics model (30) to identify the long celebrated relation between Brownian motion and the diffusion equation [2]. The Fokker-Planck equation for a Brownian motion with $x(t + \Delta t) \sim \mathcal{N}(x(t), dt)$, associated with a particle's position, is $u_t = 0.5u_{xx}$. We simulated a Brownian motion at evenly spaced time points and generated two histograms of the particle's displacement. These two histograms are $\Delta t = 0.01$ apart. As depicted in figure 6 using only two histograms with 100 bins for each one, the algorithm is capable of identifying the correct fractional order and parameter values up to a relatively good accuracy. Moreover, let us now consider the one dimensional fractional equation

$$u_t + (-\nabla^\alpha_x) u = 0, \tag{31}$$

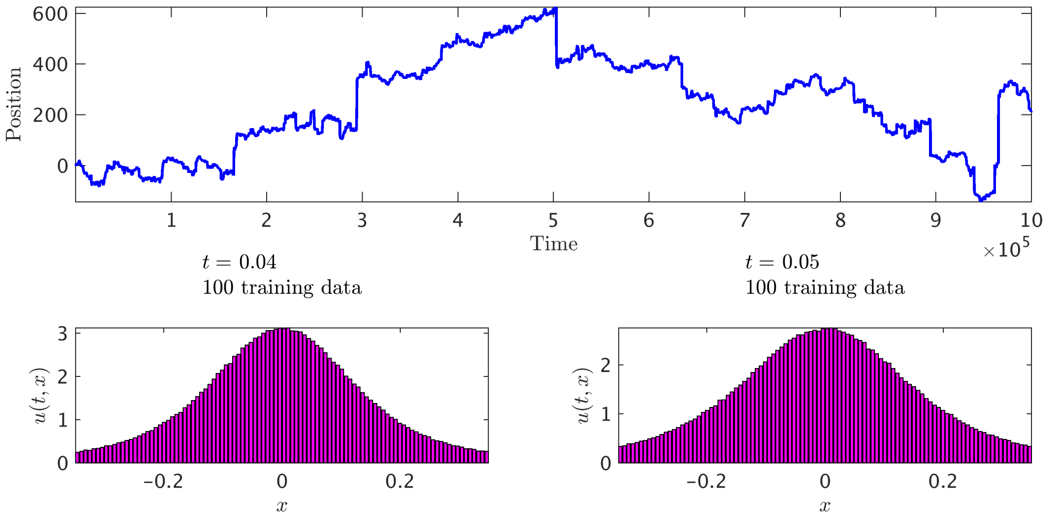
where α is the unknown parameter and $(-\nabla_x^\alpha)$ is the fractional Laplacian operator [37]. The fractional Laplacian is the operator with symbol $|w|^\alpha$. In other words, the Fourier transform of $(-\nabla_x^\alpha)u(x)$ is given by $|w|^\alpha \widehat{u}(w)$. The fractional Laplacian operator can also be defined as the generator of α-stable[8] Lévy processes. Motivated by this observation, we simulated an α-stable Lévy process [39, 40] and employed the hidden physics model resulting from equation (31) to identify the fractional order α. As depicted in figure 7 using only two histograms with 100 bins for each one, the algorithm is capable of identifying the correct fractional order up to a relatively good accuracy.



Correct PDE	$u_t - 0.5u_{xx} = 0$
Identified PDE	$u_t - 0.513\mathcal{D}^{2.016}_{-\infty,x}u = 0$

Figure 6: *Fractional Equation – Brownian Motion:* A single realization of a Brownian motion is depicted in the top panel. Two histograms of the particle's displacement, being $\Delta t = 0.01$ apart, are plotted in the middle panel. The correct partial differential equation along with the identified ones are reported in the lower panel.

[8] Stable distributions [38] are a rich class of probability distributions that allow skewness and heavy tails. Stable distributions have been proposed as a model for many types of physical and economic systems. In particular, it is argued that some observed quantities are the sum of many small terms – the price of a stock, the noise in a communication system, etc. – and hence a stable model should be used to describe such systems.



Correct PDE	$u_t + (-\nabla_x^{\sqrt{2}})u = 0$
Identified PDE	$u_t + (-\nabla_x^{1.413})u = 0$

Figure 7: *Fractional Equation – α-stable Lévy process:* A single realization of an α-stable Lévy process is depicted in the top panel. Two histograms of the particle's displacement, being $\Delta t = 0.01$ apart, are plotted in the middle panel. The correct partial differential equation along with the identified ones are reported in the lower panel.

6. Summary and Discussion

We have introduced a structured learning machine which is explicitly informed by the underlying physics that possibly generated the observed data. Exploiting this structure is critical for constructing data-efficient learning algorithms that can effectively distill information in the data-scarce scenarios appearing routinely when we study complex physical systems. We applied the proposed framework to the problem of identifying general parametric nonlinear partial differential equations from noisy data. This generality was demonstrated using various benchmark problems with different attributes. This work should be considered a direct follow up on [1] in which a similar methodology was employed to infer solutions to time-dependent and nonlinear partial differential equations, and effectively quantify and propagate uncertainty due to noisy initial or boundary data. The ideas introduced in these two papers provide a natural platform for learning from noisy data and computing under uncertainty. Perhaps the most pressing limitation of this work in its present form stems from the cubic scaling with respect to the total number of training data points. However, ideas such as recursive Kalman updates [41], variational inference [27], and parametric Gaussian processes [28] can be used to address this limitation.

Moreover, the examples studied in the current work were inspired by the pioneering work recently presented in [2]. The authors of [2] followed a sparse regression approach and a full set of spatio-temporal time series measurements consisting of thousands of data points. In contrast, here we used much smaller datasets with only hundreds of points and two snapshots of the systems. However, unlike the work in [2], here we did not use a dictionary of all possible terms involved in the partial differential equation. We could possibly include such a dictionary in our formulation but that would make our kernel evaluations more expensive. Moreover, in some systems, e.g., in an advection-diffusion-reaction system we know most of the terms of the equation, i.e., advection and diffusion but typically the reaction term is unknown. In this case, we would seek to obtain the parameters in front of the advection-diffusion and discover the functional form of the reaction term along with any parameters using the methodology outline in this paper. In comparison to [2], our method does not require numerical differentiation as the kernels are obtained analytically. Moreover, we do not require a regular lattice as in [2] and can work with scattered data. An additional advantage of our approach is that it can estimate parameters appearing anywhere in the formulation of the partial differential equation while the method of [2] is only suitable for parameters appearing as coefficients. For example, they cannot estimate the fractional order in the last example we presented in our paper or the parameters of partial differential equations (e.g., the sine-Gordon equation) involving a term like $\sin(\lambda u(x))$ with λ being the parameter. Also, the treatment of the noise is somewhat complex in the method of [2] as it involves some sort of filtering via e.g., singular value decomposition whereas our method can filter arbitrarily noisy data automatically via the Gaussian process prior assumptions. We believe that both methods can be used in different contexts effectively and we anticipate that this is only the beginning of a new way of thinking and formulating new and possibly simpler equations, e.g., by employing fractional operators that are naturally captured in our framework.

Acknowledgements

This work received support by the DARPA EQUiPS grant N66001-15-2-4055, the MURI/ARO grant W911NF-15-1-0562, and the AFOSR grant FA9550-17-1-0013. All data and codes used in this manuscript are publicly available on GitHub at https://github.com/maziarraissi/HPM.

References

[1] M. Raissi, P. Perdikaris, G. E. Karniadakis, Numerical gaussian processes for time-dependent and non-linear partial differential equations, arXiv preprint arXiv:1703.10230 (2017).

[2] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Science Advances 3 (2017).

[3] M. Raissi, P. Perdikaris, G. E. Karniadakis, Inferring solutions of differential equations using noisy multi-fidelity data, Journal of Computational Physics 335 (2017) 736 – 746.

[4] M. Raissi, P. Perdikaris, G. E. Karniadakis, Machine learning of linear differential equations using gaussian processes, Journal of Computational Physics 348 (2017) 683 – 693.

[5] C. E. Rasmussen, C. K. Williams, Gaussian processes for machine learning, volume 1, MIT press Cambridge, 2006.

[6] K. P. Murphy, Machine learning: a probabilistic perspective, MIT press, 2012.

[7] R. M. Neal, Bayesian learning for neural networks, volume 118, Springer Science & Business Media, 2012.

[8] V. Vapnik, The nature of statistical learning theory, Springer Science & Business Media, 2013.

[9] B. Schölkopf, A. J. Smola, Learning with kernels: support vector machines, regularization, optimization, and beyond, MIT press, 2002.

[10] M. E. Tipping, Sparse Bayesian learning and the relevance vector machine, The journal of machine learning research 1 (2001) 211–244.

[11] A. Tikhonov, Solution of incorrectly formulated problems and the regularization method, in: Soviet Math. Dokl., volume 5, pp. 1035–1038.

[12] A. N. Tikhonov, V. Y. Arsenin, Solutions of Ill-posed problems, W.H. Winston, 1977.

[13] T. Poggio, F. Girosi, Networks for approximation and learning, Proceedings of the IEEE 78 (1990) 1481–1497.

[14] N. Aronszajn, Theory of reproducing kernels, Transactions of the American Mathematical Society 68 (1950) 337–404.

[15] S. Saitoh, Theory of reproducing kernels and its applications, volume 189, Longman, 1988.

[16] A. Berlinet, C. Thomas-Agnan, Reproducing kernel Hilbert spaces in probability and statistics, Springer Science & Business Media, 2011.

[17] D. Duvenaud, J. R. Lloyd, R. Grosse, J. B. Tenenbaum, Z. Ghahramani, Structure discovery in nonparametric regression through compositional kernel search, arXiv preprint arXiv:1302.4922 (2013).

[18] R. Grosse, R. R. Salakhutdinov, W. T. Freeman, J. B. Tenenbaum, Exploiting compositionality to explore a large space of model structures, arXiv preprint arXiv:1210.4856 (2012).

[19] G. Malkomes, C. Schaff, R. Garnett, Bayesian optimization for automated model selection, in: Advances in Neural Information Processing Systems, pp. 2900–2908.

[20] R. Calandra, J. Peters, C. E. Rasmussen, M. P. Deisenroth, Manifold Gaussian processes for regression, in: Neural Networks (IJCNN), 2016 International Joint Conference on, IEEE, pp. 3338–3345.

[21] M. Raissi, G. Karniadakis, Deep multi-fidelity Gaussian processes, arXiv preprint arXiv:1604.07484 (2016).

[22] D. C. Liu, J. Nocedal, On the limited memory BFGS method for large scale optimization, Mathematical programming 45 (1989) 503–528.

[23] C. E. Rasmussen, Z. Ghahramani, Occam's razor, Advances in neural information processing systems (2001) 294–300.

[24] S. L. Brunton, J. L. Proctor, J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proceedings of the National Academy of Sciences 113 (2016) 3932–3937.

[25] A. M. Stuart, Inverse problems: a Bayesian perspective, Acta Numerica 19 (2010) 451–559.

[26] E. Snelson, Z. Ghahramani, Sparse Gaussian processes using pseudo-inputs, in: Advances in neural information processing systems, pp. 1257–1264.

[27] J. Hensman, N. Fusi, N. D. Lawrence, Gaussian processes for big data, in: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, UAI 2013, Bellevue, WA, USA, August 11-15, 2013.

[28] M. Raissi, Parametric gaussian process regression for big data, arXiv preprint arXiv:1704.03144 (2017).

[29] C. Basdevant, M. Deville, P. Haldenwang, J. Lacroix, J. Ouazzani, R. Peyret, P. Orlandi, A. Patera, Spectral and finite difference solutions of the Burgers equation, Computers & fluids 14 (1986) 23–41.

[30] T. Dauxois, Fermi, Pasta, Ulam and a mysterious lady, arXiv preprint arXiv:0801.1590 (2008).

[31] J. M. Hyman, B. Nicolaenko, The Kuramoto-Sivashinsky equation: a bridge between pde's and dynamical systems, Physica D: Nonlinear Phenomena 18 (1986) 113–126.

[32] B. I. Shraiman, Order, disorder, and phase turbulence, Physical review letters 57 (1986) 325.

[33] B. Nicolaenko, B. Scheurer, R. Temam, Some global dynamical properties of the Kuramoto-Sivashinsky equations: nonlinear stability and attractors, Physica D: Nonlinear Phenomena 16 (1985) 155–183.

[34] J. N. Kutz, S. L. Brunton, B. W. Brunton, J. L. Proctor, Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems, volume 149, SIAM, 2016.

[35] K. Taira, T. Colonius, The immersed boundary method: a projection approach, Journal of Computational Physics 225 (2007) 2118–2137.

[36] T. Colonius, K. Taira, A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions, Computer Methods in Applied Mechanics and Engineering 197 (2008) 2131–2146.

[37] I. Podlubny, Fractional differential equations: an introduction to fractional derivatives, fractional differential equations, to methods of their solution and some of their applications, volume 198, Academic press, 1998.

[38] J. Nolan, Stable distributions: models for heavy-tailed data, Birkhauser New York, 2003.

[39] J. M. Chambers, C. L. Mallows, B. Stuck, A method for simulating stable random variables, Journal of the american statistical association 71 (1976) 340–344.

[40] A. Weron, R. Weron, Computer simulation of lévy α-stable variables and processes, ChaosThe Interplay Between Stochastic and Deterministic Behaviour (1995) 379–392.

[41] J. Hartikainen, S. Särkkä, Kalman filtering and smoothing solutions to temporal Gaussian process regression models, in: Machine Learning for Signal Processing (MLSP), 2010 IEEE International Workshop on, IEEE, pp. 379–384.