Noise-aware physics-informed machine learning for robust PDE discovery

IOP Publishing Mach. Learn.: Sci. Technol. 4 (2023) 015009 https://doi.org/10.1088/2632-2153/acb1f0

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
29 July 2022

REVISED
5 January 2023

ACCEPTED FOR PUBLICATION
10 January 2023

PUBLISHED
1 February 2023

PAPER

Noise-aware physics-informed machine learning for robust PDE discovery

Pongpisit Thanasutives[1,*], **Takashi Morita**[2], **Masayuki Numao**[2] and **Ken-ichi Fukui**[2]

[1] Graduate School of Information Science and Technology, Osaka University, Suita, Osaka 565-0871, Japan
[2] SANKEN (The Institute of Scientific and Industrial Research), Osaka University, Ibaraki, Osaka 567-0047, Japan
* Author to whom any correspondence should be addressed.

E-mail: thanasutives@ai.sanken.osaka-u.ac.jp

Keywords: denoising method, machine learning, multi-task learning, partial differential equation, physics-informed neural network

Abstract

This work is concerned with discovering the governing partial differential equation (PDE) of a physical system. Existing methods have demonstrated the PDE identification from finite observations but failed to maintain satisfying results against noisy data, partly owing to suboptimal estimated derivatives and found PDE coefficients. We address the issues by introducing a noise-aware physics-informed machine learning framework to discover the governing PDE from data following arbitrary distributions. We propose training a couple of neural networks, namely solver and preselector, in a multi-task learning paradigm, which yields important scores of basis candidates that constitute the hidden physical constraint. After they are jointly trained, the solver network estimates potential candidates, e.g. partial derivatives, for the sparse regression to initially unveil the most likely parsimonious PDE, decided according to information criterion. Denoising physics-informed neural networks, based on discrete Fourier transform, is proposed to deliver the optimal PDE coefficients respecting the noise-reduced variables. Extensive experiments on five canonical PDEs affirm that the proposed framework presents a robust and interpretable approach for PDE discovery, leading to a new automatic PDE selection algorithm established on minimization of the information criterion decay rate.

1. Introduction

Data-driven discovery has recently gained popularity due to its flexibility and satisfactory accuracy in uncovering the hidden underlying partial differential equation (PDE) of a dynamical system with less required domain knowledge. Applying sparse regression-based approaches to a library of the target variable and its partial derivative candidates is a promising method for discovering a parsimonious model purely out of observational data. A few of such previous attempts were, for instance, sequential threshold ridge regression (STRidge) [1], L_1-regularized sparse optimization [2] based on the least absolute shrinkage and selection operator (LASSO) [3], and sparse Bayesian regression [4].

Since partial derivatives are the vital input features, their inaccurate estimations can poorly affect the discovered results. For example, numerical differentiation, such as the finite difference method, may not accurately approximate high-order derivatives when facing sparse corrupted data. Prior works tackled the problem by formulating indirect but smoother representations of derivatives, such as weak formulation (WF) [5] and convolutional weak formulation (CWF) [6], which were demonstrated to be tolerant to noise and computationally efficient. However, the formulations are usually restricted to spatial data in mesh form. Particularly for identifying ordinary differential equations, RK4-SINDy [7] deals with noise by internal simulation of the equation using the 4th-order Runge–Kutta-inspired method. This paper utilizes automatic differentiation (AD) [8] on a neural network that we refer to as the solver to be an alternative approach, as formerly suggested by [9]. AD allows derivative computation given a mere implementation of the hypothesis function; therefore, the method does not suffer from truncation error, mitigating the numerical imprecision of computing high-order derivatives.

Although the utilization of neural networks is not restricted by the assumption of particular input distributions, the solver that learns just by correcting prediction errors may be prone to overfitting the finite observations and inadequate for capturing the proper PDE solution, especially when encountering sparse measurements. This trouble motivates us to formulate the solver network with weakly physics-informed regularization, maintaining the prediction performance while respecting an implicit form of the governing physical law. Specifically new in this work, we propose multi-task training with a preselector neural network that promotes sparsity based on an interpretable self-gated mechanism to alleviate the issue. The preselector learns the estimated system evolution from the spatial derivatives (both are produced by the solver) and other features to represent the hidden parsimonious PDE. Furthermore, we present a workaround of using the trained preselector's feature importance to encourage selecting the expressive candidates that derive non-overfitted PDEs.

Once both the networks are converged using a multi-task learning procedure and the library of potential (nonlinear) terms is prepared, we then apply a form of sparse linear regression algorithms, e.g. STRidge [1] and weak sparse identification of nonlinear dynamics (WSINDy) [6], to the (discretized) domain of interest. Nonetheless, the proper selection of the regularization hyperparameters regarding the sparse linear model can be problematic. While the underlying PDE remains unknown, solely cross-validating equations together with the Pareto analysis based on fixed-valued regularization hyperparameters may still yield insignificant, probably wrong results, especially in a small data regime. Thus, as an additional consideration, the initial discovered PDE is encouraged to include the candidates whose importance is greater than a threshold defined within the proposed preselector network's L_0-regularized self-gated mechanism. After the cooperative learning, among the expected PDEs formable by the threshold-passing basis candidates, the parsimonious and informative PDEs are preferred, i.e. having sufficiently low Bayesian information criterion (BIC) [10] or Akaike information criterion (AIC) [11]. The aforementioned heuristics and experimental analysis of BIC decay rate per an increase in the complexity (number of included/support candidates) led us to invent an algorithm for selecting the sparse PDE automatically.

We here clarify the major distinction between our information criterion calculation and the metrics utilized in the prior studies [12–14]. We choose to calculate BIC on the model that predicts the estimated system evolution given by the solver network, hence the unnecessity of simulating any found PDEs, unlike [12, 14], which measured AIC between their model prediction and the ground time series (in the power spectral density form for [14]). Thereby, our choice is computationally cheaper, but the BIC may be treated as pseudo-BIC because it does not connect with the direct comparison. In [13], the pruning after STRidge is executed as follows: if deselecting a candidate results in a ratio between the residual loss and starting loss that is less than a threshold, the candidate is removed. Obviously, finding a suitable threshold value is challenging, and the problem of putting the right regularization to STRidge still needs to be understood.

At this point, the sparse learning algorithm has yielded a guess of the hidden governing PDE, referred to as the initial discovered PDE. However, propagating error is woefully inevitable since the sparse regression is separated from the candidate library preparation step. This consequently causes the initial PDE to not be at its optimum concerning the given input data. To achieve the most-favorable PDE, we parameterize all the discovered coefficients as the gradient-based learnable parameters of the physics-informed solver network that is finetuned such that its output approximates the target variable while concurrently respecting the most-relevant underlying PDE as per the core proposal of physics-informed learning [9, 15]. Remark that if the candidate library is incomplete (we miss including some necessary candidates), the benefit of finetuning the discovered coefficients becomes less because of the faulty form of governing PDE used. Also, without an appropriate initialization of targeting PDE coefficients, training a physics-informed neural network (PINN) [9] may be a task that could be developed further by, for example, multi-task learning [16] or sinusoidal feature mapping [17], even though the actual governing function is presumably known beforehand.

In a practical scenario where noise may disturb both the independent and dependent variables, the optimization process of PINN is perturbed; thus, attaining a local optimum set of coefficients is spaced out from the ground truth. The previous work, abbreviated as DLrSR [18], tackled the difficulty via low-rank matrix factorization solved by robust principal component analysis (PCA) [19], neglecting the assumably sparse noise and utilizing the low-rank data. Nevertheless, if the sparse noise presumption does not hold, the method can be impotent for various situations. To mitigate the issue, we introduce denoising layers based on precomputed discrete Fourier transform (DFT) to the vanilla PINN, optimizing the solver-founded PDE. The denoising layers filter out the frequency components of the input signal, whose power is less than a predefined threshold, then obtain contaminated noise by taking the difference between the original and reconstructed signal. The extracted noises are projected using projection neural networks to perturb backwardly or denoise the noisy measurements with the appropriate intensities, then reconstruct the noise-reduced dataset. This paper coins a PINN attached to the proposed denoising mechanism as denoising

PINNs (dPINNs). Ultimately, after the dPINNs' learning, the converged parameters regarding all effective coefficients are treated as the end results.

Experimental results from five canonical models, including three ordinary PDEs and two complex-valued PDEs, reveal the robustness and accuracy of the proposed framework in noiseless and noisy datasets. As a proof of concept distinct from prior works, we conduct investigations on learning from noisy independent variables, e.g. polluted spatial and temporal variables, which are relevant to Global Positioning System (GPS) coordinate measurements [20] and manual timing in physical experiments [21].

We summarize our main contributions as follows:

- We introduce the multi-task learning with the preselector network to impose the weakly physics-informed constraint, which is calculable without labeled supervision.
- We introduce the preselector's perceived feature importance scores to bring an auxiliary view to the candidate selection, later inspiring the new automatic approach of finding the right complexity for the initial discovered PDE.
- We propose dPINNs based on DFT and the projection networks to handle noisy independent and dependent variables.

2. Related work

The foundation of this work is built upon a combination of the SINDy [22] specifically for PDEs, namely STRidge [1], and PINN [9]. Previously taking on the similar idea, Berg and Nyström [23] trained a feed-forward neural network with L_2-regularization on its weights to learn the dependent function from the standardized domain. AD was applied to the converged network to generate derivatives as ingredients for LASSO, unveiling the latent PDEs formulated on the scaled domain. The found PDE was then transformed to align with the original domain. Unlike our work, absent a finetuning process, the propagating error through the pipeline was never taken care of. Nominating a compact physics-informed solution, called DeepMoD, Both *et al* [24] added LASSO regression loss (including L_1-regularization) to the neural network's cost function. Consequently, the constructed candidate library was possibly large and memory-inefficient. After the sparsity pattern was obtained by the normalization and thresholding, the network was finetuned without the L_1-penalty to find the optimal unbiased coefficients. Yet, given that one of the true coefficients is small and the data is noisy, the robust model selection by thresholding/pruning may be inaccurate. Chen *et al* [25] leveraged STRidge regression loss as the physics cost function instead and let STRidge select the parsimonious model during the alternating direction optimization (ADO), where not only the coefficients but the whole PDE was changed or optimized. How much of L_0-penalty for STRidge evaluation was decided by a Pareto analysis, but the regularization hyperparameter on solving a ridge regression problem, possibly influential to the model selection during ADO, was kept underexplored and fixed to a specific value for each discovery. More recently, a two-network PDE discovery framework called PDE-READ, where one network regularizes the other network in the form of physical constraint, was introduced by Stephany and Earls [26]. Different from this work, the interpretation of learned physical constraint (PDE ranking in [26]) relied on recursive feature elimination [27], not from the regularizing network.

Xu *et al* were interested in an alternative view of the PDE discovery problem, using genetic algorithms to deal with the incompleteness of the candidate library [28] and enable the robust algorithm in noisy conditions [29]. Each genome's module comprised possible functions and derivatives produced by a neural network, representing a form of the unknown PDE. Although the proposal discouraged searching the PDE against giant sets of features by design, the incompleteness problem was boiled down to which functions to consider when building a genome.

To the best of our knowledge, little to no mentioned previous works have suggested or explored ideas of the constituted agreement to ensure the found Pareto-optimal PDE. Particularly for those that utilize the training or finetuning using PINN, e.g. [9, 24–26, 28, 29], none of them have integrated a denoising mechanism with the process yet.

3. Method: noise-aware physics-informed machine learning (nPIML) framework

3.1. Problem formulation and overview

We consider the following general form of nonlinear PDE in the dynamical system perspective:

$$u_t = \mathcal{N}_\xi[\Theta]; \quad \Theta = [u \quad u_x \quad u_{xx} \quad \cdots \quad x]. \tag{1}$$

$\mathcal{N}_\xi$ is the governing function parameterized by the vector of coefficients ξ. The function depends on Θ, which may consist of the spatial variable x, the derivatives and any indispensable features. In regards to $\mathcal{N}_\xi$,

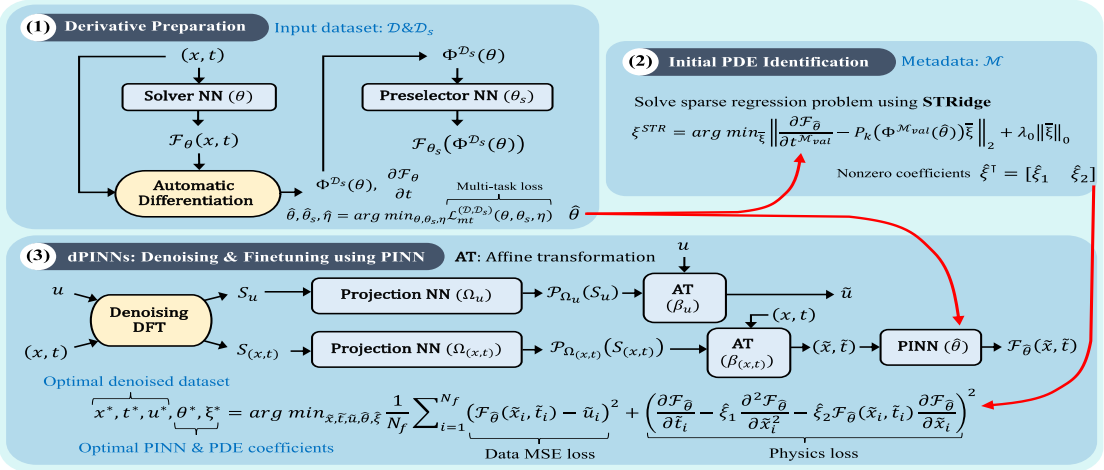


Figure 1. Exemplary discovery scheme of the proposed noise-aware physics-informed machine learning (nPIML) framework: (1) physics-regularized derivative preparation by multi-task learning of the solver and preselector. (2) Initial identification of the hidden PDE by STRidge. (3) Applying the denoising DFT to $(x,t)\&u$ then finetuning the initial PDE coefficients on the denoised variables using PINN. Here, we suppose that the discovered governing equation is Burgers' PDE.

Θ is the smallest possible library, merely composed of the necessary terms. u is the dependent PDE solution, observed with the space-time matrix (x,t).

Figure 1 conceptualizes the three principal procedures for uncovering ξ preferably in a low-dimensional space by walking through an exemplar of discovering Burgers' PDE [30]. Step ⑪, we numerically equivalizes u and $\mathcal{N}_\xi[\Theta]$ to the solver and preselector neural network's output $\mathcal{F}_\theta(x,t)$ and $\mathcal{F}_{\theta_s}(\Phi^{\mathcal{D}_s}(\theta))$. $\Phi^{\mathcal{D}_s}(\theta) \in \mathbb{C}^{(N_f+N_r)\times C}$ is the library of C linearly independent atomic/basis candidates from which the preselector learns to embed physics by inferring the system evolution. The candidates are evaluated on a multiset $\mathcal{D}_s = \{(x_i,t_i)_{i=1}^{N_f+N_r}\}$. Step ⑫, the well-fitted networks, $\hat{\theta}$ and $\hat{\theta}_s$, put together a larger library of potential k-degree polynomial features $P_k(\Phi^{\mathcal{M}_{val}}(\hat{\theta}))$ of which an initial analytical expression of Burgers' PDE, worked out approximately by STRidge [1], is made. Step ⑬, $\hat{\theta}$ is henceforth transferred to the PINN that is optimally finetuned with the PDE, initialized by nonzero coefficients $\hat{\xi}$, on the denoised variables $\tilde{x}$, $\tilde{t}$ and $\tilde{u}$, offered by the projection networks $\mathcal{P}_{\Omega_{(x,t)}}$ and $\mathcal{P}_{\Omega_u}$. The noise-reduction mechanism functions as a series of affine transformations, controlled by $\beta_{(x,t)}$ and β_u, of the dataset with the projected noises $\mathcal{P}_{\Omega_{(x,t)}}(S_{(x,t)})$ and $\mathcal{P}_{\Omega_u}(S_u)$, after applying the frequency-based denoising DFT. The mathematical derivation of the relevant variables are elaborated in sections 3.2–3.4. The summary of the main notations and their description is given in table 11.

3.2. Derivative preparation

Concerning ⑪ of figure 1, we utilize the solver $(\mathcal{F}_\theta)$ and preselector $(\mathcal{F}_{\theta_s})$ networks, which are jointly trained for the solver network to be weakly physics-constrained. Facilitating the co-training, the solver network is pretrained on the dataset $\mathcal{D} = \{(x_i,t_i,u_i)_{i=1}^{N_f}\}$ to approximate the mapping function. Therefore, the partial derivative candidate values are assured of becoming close to the valid values. At the pretraining stage, the solver network minimizes the mean square error (MSE)

$$\mathcal{L}_{\text{sup}}^{\mathcal{D}}(\theta) = \frac{1}{N_f}\sum_{i=1}^{N_f}(\mathcal{F}_\theta(x_i,t_i) - u_i)^2;\ (x_i,t_i,u_i) \in \mathcal{D}, \tag{2}$$

where N_f is the number of labeled subsamples. If u is complex-valued, the sum of the MSEs from the real and imaginary parts is taken as the supervised loss function. Since a futile search over infinitely feasible $\Phi^{\mathcal{D}_s}(\theta)$ setups would be intractable, we presume the ability to build an overcomplete basis candidate library given to the preselector network for deciding the predictive set of features by minimizing

$$\begin{aligned} \mathcal{L}_{\text{unsup}}^{\mathcal{D}_s}(\theta,\theta_s) &= \frac{1}{N_f+N_r}\sum_{i=1}^{N_f+N_r}\left(\frac{\partial \mathcal{F}_\theta}{\partial t_i} - \mathcal{F}_{\theta_s}(\Phi_i^{\mathcal{D}_s}(\theta))\right)^2; \\ \Phi_i^{\mathcal{D}_s}(\theta) &= \left[\mathcal{F}_\theta(x_i,t_i) \quad \frac{\partial \mathcal{F}_\theta}{\partial x_i} \quad \frac{\partial^2 \mathcal{F}_\theta}{\partial x_i^2} \quad \cdots \quad x_i\right], \end{aligned} \tag{3}$$

where N_r is the number of random unsupervised subsamples within the domain that may disjoint the supervised set $\mathcal{D}$. We attain $\mathcal{D}_s$ by fusing up the spatio-temporal measurements without supervision. Each

derivative term's input is usually omitted for notational convenience. Inspired by the assumption that low-order partial derivatives are commonly included more than the higher ones, we embed the thresholded self-gated mechanism, parameterized by W^b, to the preselector forward pass, emphasizing the priority of simple models as follows:

$$\begin{aligned}\mathcal{F}_{\theta_s}(\Phi^{\mathcal{D}_s}(\theta)) &= \mathcal{F}_{\theta_s^r}(\mathcal{F}_{W^b}(\Phi^{\mathcal{D}_s}(\theta))),\\ \mathcal{F}_{W^b}(\Phi^{\mathcal{D}_s}(\theta)) &= \Phi^{\mathcal{D}_s}(\theta)\odot\mathcal{A}^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b),\\ \mathcal{A}_j^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b) &= \max(\mathcal{A}_j(\Phi^{\mathcal{D}_s}(\theta),W^b)-\mathcal{T},0),\\ \mathcal{A}_j(\Phi^{\mathcal{D}_s}(\theta),W^b) &= \frac{\sum_{i=1}^{N_f+N_r}\sigma(\sum_{k=1}^{C}\Phi_{ik}^{\mathcal{D}_s}(\theta)W_{kj}+b_j)}{N_f+N_r}.\end{aligned} \tag{4}$$

$\odot$ refers to Hadamard product (broadcast multiplication). $\mathcal{A}^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b)$ is interpreted as the thresholded vector-valued feature importance the preselector perceive. The self-gated mechanism utilizes the activation function σ to compute the expected importance of each candidate in terms of (unnormalized) probability across N_f+N_r samples. Note that we only consider the real part of $\Phi^{\mathcal{D}_s}(\theta)W+b$ in the case of complex-valued PDEs. $\mathcal{T}$ is a threshold for allowing the effective basis candidates. The threshold is initialized to be surely less than the minimal candidate importance, specifically we set $\mathcal{T}=\kappa\min_j\mathcal{A}_j^{(1)}(\Phi^{\mathcal{D}_s}(\theta),W^b)$, where $0<\kappa<1$, before the joint gradient update at the first epoch (superscript (1)). The parameter W^b consists of $W\in\mathbb{C}^{C\times C}$ and $b\in\mathbb{C}^{1\times C}$ (weights and biases of the linear layer), serving as the share of the preselector's parameters:

$$\theta_s=(W^b,\theta_s^r);\ \mathcal{F}_{\theta_s}=\mathcal{F}_{\theta_s^r}\circ\mathcal{F}_{W^b}. \tag{5}$$

Excluding W^b, the rest of the preselector network's parameters get referred to as θ_s^r. We devise $R^{\mathcal{D}_s}(\theta,W^b)$ as a L_0-regularization on $\mathcal{A}^{\mathcal{T}}$ for selecting the expressive subset with priority to lower-order candidates in favor of Occam's razor principle. The regularization, encouraging the sparse and simple preselector learned representations, reads

$$R^{\mathcal{D}_s}(\theta,W^b)=\lambda_1\left(\left\|\mathcal{A}^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b)\right\|_0+\lambda_2\sum_{j=1}^{C}w_j\mathcal{A}_j^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b)\right). \tag{6}$$

w is the weighting by derivative orders directly applied to the feature importance. For instance, suppose that jth basis candidate associates to the second-order derivative u_{xx}. Then we have $w_j=2$. For nonderivative terms, we assign $w_j=1$. λ_1 is the parameter that controls the regularization intensity. λ_2 closes the gap between the derivative orders such that the high-order derivatives are not always deselected. To practically minimize $R^{\mathcal{D}_s}(\theta,W^b)$ with $\mathcal{L}_{\text{unsup}}^{\mathcal{D}_s}(\theta,\theta_s)$ by a gradient-based optimizer, we have to overcome the obstacle that the L_0-norm is not yet readily differentiable with respect to its input vector. Unlike how [31] mask candidates (with 0 or 1) before AD, we require the smooth approximation of L_0 for achieving the thresholded feature importance. Adapted from SL0 algorithm [32], we estimates

$$\left\|\mathcal{A}^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b)\right\|_0\approx C-\sum_{j=1}^{C}\exp\left(\frac{-(\mathcal{A}_j^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b))^2}{2(\eta\mathbb{V}(\mathcal{A}^{\mathcal{T}}(\Phi^{\mathcal{D}_s}(\theta),W^b)))^2}\right), \tag{7}$$

where $\mathbb{V}$ is the unbiased variance estimator over the C basis candidates. η determines the trade-off between the accuracy and smoothness: the smaller η gives the closer approximation, and the larger η gives the smoother approximation. η is initialized at 1.0 and learned with the gradients. We now denote the differentiable regularization function as $R_\eta^{\mathcal{D}_s}(\theta,W^b)$. Combining equations (2), (3), (6) and (7), we view the multi-task learning of the weakly physics-informed solver and the coordinating simplicity-guided preselector inherently as the semi-supervised multi-objective optimization formulated as follows:

$$\begin{aligned}\hat{\theta},\hat{\theta}_s,\hat{\eta} &= \arg\min_{\theta,\theta_s,\eta}\mathcal{L}_{\text{mt}}^{(\mathcal{D},\mathcal{D}_s)}(\theta,\theta_s,\eta);\\ \mathcal{L}_{\text{mt}}^{(\mathcal{D},\mathcal{D}_s)}(\theta,\theta_s,\eta) &= \text{MT}(\mathcal{L}_{\text{sup}}^{\mathcal{D}}(\theta),\mathcal{L}_{\text{unsup}}^{\mathcal{D}_s}(\theta,\theta_s)+R_\eta^{\mathcal{D}_s}(\theta,W^b)).\end{aligned} \tag{8}$$

The parameters of both networks are concurrently updated with the expectancy that the preselector network distills the hidden PDE function $\mathcal{N}_\xi$, and informs physics back to the solver. MT is a function that reasonably manipulates learning by multiple losses, such as Uncert [33] and PCGrad [34], which are shown to accelerate

Algorithm 1. Multi-task learning for derivative preparation and initial PDE identification.

Goal: Discover the governing function $\hat{\mathcal{N}}_{\hat{\xi}}$ based on the solver and preselector parameters $\hat{\theta}, \hat{\theta}_s$
Require: Pretrained θ by (2), initialized θ_s
1: Joint train[a] $\theta, \hat{\theta}_s, \hat{\eta} \leftarrow \arg\min_{\theta,\theta_s,\eta} \mathcal{L}_{mt}^{(\mathcal{D},\mathcal{D}_s)}(\theta, \theta_s, \eta)$
2: Assign $I_j \leftarrow \mathcal{A}_j(\Phi^{\mathcal{D}_s}(\theta, \hat{W}^{\hat{b}})) - \mathcal{T} + \frac{1}{C}$ as the feature importance for each jth basis candidate
3: Converge the solver $\hat{\theta} \leftarrow \arg\min_{\theta} \mathcal{L}_{\text{sup}}^{\mathcal{D}}(\theta)$
4: Build the candidate library on the metadata $\mathcal{M}$ by $\Phi^{\mathcal{M}}(\hat{\theta}) \leftarrow \left[\mathcal{F}_{\hat{\theta}}(x^{\mathcal{M}}, t^{\mathcal{M}}) \quad \frac{\partial \mathcal{F}_{\hat{\theta}}}{\partial x^{\mathcal{M}}} \quad \frac{\partial^2 \mathcal{F}_{\hat{\theta}}}{\partial (x^{\mathcal{M}})^2} \quad \cdots \quad x^{\mathcal{M}}\right]$
5: Find $\hat{\mathcal{N}}_{\hat{\xi}}$ on $P_k(\Phi^{\mathcal{M}_{\text{val}}}(\hat{\theta}))$ using λ_{STR}-varied STRidge
6: **return** $\hat{\theta}, \hat{\theta}_s = (\hat{W}^{\hat{b}}, \hat{\theta}_s^r)$, $\hat{\eta}$ and $\hat{\mathcal{N}}_{\hat{\xi}}$

[a] After the joint training until empirical plateau, the learned preselector's parameters are regarded as $\hat{\theta}_s$. Converging the preselector could have been done, i.e. $\min_{\hat{\theta}_s} \mathcal{L}_{\text{unsup}}^{\mathcal{D}_s}(\theta, \hat{\theta}_s)$, but did not to reduce the run time.

the PINN generalized performance [16]. Algorithm 1 describes a relaxed approach that numerically minimizes the loss in equation (8) until detected plateau; then, converging the solver network independently.

3.3. Initial PDE identification

Depicted by ② of figure 1, we train STRidge [1] on top of the candidates and their polynomial features up to k degree: $P_k(\Phi^{\mathcal{M}}(\hat{\theta}))$, which is evaluated on metadata $\mathcal{M}$. For example, assume that $k = 2$, the nonconstant interaction-only polynomial features of u, u_x and u_{xx} are formed as

$$P_2([u \quad u_x \quad u_{xx}]) = [u \quad u_x \quad u_{xx} \quad uu_x \quad uu_{xx} \quad u_x u_{xx}]. \tag{9}$$

The metadata $\mathcal{M} = \{(x_i^{\mathcal{M}}, t_i^{\mathcal{M}})_{i=1}^{N_{\mathcal{M}}}\}$ can be samples from a desired domain of interest, e.g. linearly discretized samples within a bounded rectangle domain are generated with the equal spaces as follows: $\Delta x = \min_{i,j;(i\neq j)} |x_i - x_j|$ and $\Delta t = \min_{i,j;(i\neq j)} |t_i - t_j|$. If x is in a higher dimension, the equal space is computed separately for each spatial direction. In fact, naively equating $\forall i \in \{1, 2, \ldots, N_f\}, (x_i^{\mathcal{M}}, t_i^{\mathcal{M}}) = (x_i, t_i)$ is also viable for identifying the governing PDE as $\hat{\mathcal{N}}_{\hat{\xi}}[P_k(\Phi^{\mathcal{M}_{\text{val}}}(\hat{\theta}))\mathcal{E}]$, where $\hat{\xi}$ and $\mathcal{E}$ are found by the following selection criterion:

$$\begin{aligned} \xi^{\text{STR}} &= \arg\min_{\bar{\xi}} \left\| \frac{\partial \mathcal{F}_{\hat{\theta}}}{\partial t^{\mathcal{M}_{\text{val}}}} - P_k(\Phi^{\mathcal{M}_{\text{val}}}(\hat{\theta}))\bar{\xi} \right\|_2 + \lambda_0 \|\bar{\xi}\|_0; \\ \lambda_0 &= \mu \lambda_{\text{STR}} \varepsilon, E = \{f_{i+1} \mid i \in \mathbb{N}_{\|\xi^{\text{STR}}\|_0} \wedge \xi_{f_{i+1}}^{\text{STR}} \neq 0\}, \\ \hat{\xi} &= \left[\xi_{f_1}^{\text{STR}} \quad \cdots \quad \xi_{f_{|E|}}^{\text{STR}}\right]^{\mathsf{T}}, \mathcal{E} = \left[\boldsymbol{e}_{f_1} \quad \cdots \quad \boldsymbol{e}_{f_{|E|}}\right]. \end{aligned} \tag{10}$$

$\varepsilon = \varepsilon(P_k(\Phi^{\mathcal{M}}(\hat{\theta})))$ is the significand of the condition number (written in the scientific notation) of the polynomial candidate library. $\mathcal{M}_{\text{val}}$ is a 20% split of the full $\mathcal{M}$. For a tolerance tol, $\bar{\xi}$ is estimated by solving a relaxed λ_{STR}-regularized ridge regression problem on $P_k(\Phi^{\mathcal{M}}(\hat{\theta}))$, whose column is normalized by its L_2-norm unless noted otherwise, with hard thresholding. To attain ξ^{STR}, tol is iteratively refined with respect to different values of $\lambda_0 \propto \lambda_{\text{STR}}$ using a variable d_{tol} that initializes tol. $\mu > 0$ is assigned data-dependently. After applying STRidge, $\hat{\mathcal{N}}_{\hat{\xi}}$ is the linear combination of the effective polynomial candidates chosen by $\mathcal{E}$. $\mathbb{N}_{\|\xi^{\text{STR}}\|_0}$ denotes $\{0, 1, \ldots, \|\xi^{\text{STR}}\|_0 - 1\}$. E is an indexed set, and $\boldsymbol{e}_j$ is an elementary column vector whose entries are all zero except for the nonzero jth polynomial candidate. The matrix $\mathcal{E}$ reduces the dimensionality such that we focus solely on the effective candidates, which hopefully capture the ideal Θ in equation (1). $\hat{\xi}$ successively stores the nonzero coefficients in ξ^{STR}. If the library is overcomplete under the evaluation on $\mathcal{M}$, there exists $\mathcal{E}$ such that $\Theta^{\mathcal{M}} \approx P_k(\Phi^{\mathcal{M}}(\hat{\theta}))\mathcal{E}$.

The pair values of $(\lambda_1, \lambda_{\text{STR}})$ are grid searched with BIC [10] as the guidance score. The pairs whose PDEs are in agreement with the corresponding preselectors, according to definition 1, are expected.

Definition 1 (Agreement). If P_k is regarded as the candidate building function and every nonzero $f\text{th}_{i+1}$ term can be written as a polynomial of certain jth candidates whose jth is taken from the set of threshold-passing basis candidate indices $\{j \mid I_j > \frac{1}{C}\}$ (see algorithm 1), we determine that the initial discovered PDE of a particular pair of $(\lambda_1, \lambda_{\text{STR}})$ is in the 'agreement' with the λ_1-trained preselector network.

The likely models, from which we can actually choose a reasonable one as the initial discovered PDE voluntarily, are conceived of being in their agreements and relatively sparse (small $\|\xi^{\text{STR}}\|_0 = \|\hat{\xi}\|_0 = |E|$) while conveying sufficiently low BIC scores defined as follows:

$$\begin{aligned}
\mathrm{BIC}(\xi^{\mathrm{STR}},\hat{\theta}) &= \left\|\xi^{\mathrm{STR}}\right\|_0 \log N_{\mathcal{M}} - 2\log\hat{L}(\xi^{\mathrm{STR}},\hat{\theta});\\
\log\hat{L}(\xi^{\mathrm{STR}},\hat{\theta}) &= \frac{-N_{\mathcal{M}}}{2}\left(1+\log 2\pi+\log\frac{\mathrm{RSS}(\xi^{\mathrm{STR}},\hat{\theta})}{N_{\mathcal{M}}}\right),\\
\mathrm{RSS}(\xi^{\mathrm{STR}},\hat{\theta}) &= \sum_{i=1}^{N_{\mathcal{M}}}\left|\frac{\partial\mathcal{F}_{\hat{\theta}}}{\partial t_i^{\mathcal{M}}} - P_k(\Phi_i^{\mathcal{M}}(\hat{\theta}))\xi^{\mathrm{STR}}\right|^2.
\end{aligned} \tag{11}$$

$\log\hat{L}(\xi^{\mathrm{STR}},\hat{\theta})$ is the maximized (natural) log-likelihood of the $\hat{\theta}$-produced model parameterized by ξ^{STR}. RSS denotes the real-valued residual sum of squares because the absolute value of each (complex-valued) residual term is considered. The BIC formulation is primarily used in Statsmodels [35], which follows [36]. The pseudocode for sections 3.2 and 3.3 is detailed in algorithm 1. Let us mention that our BIC terminology can be treated as pseudo-BIC in the sense that the calculation compares to the estimated system evolution $\frac{\partial\mathcal{F}_{\hat{\theta}}}{\partial t^{\mathcal{M}}} \approx u_t$ not u, which the simulated solution of the initial discovered PDE should closely approximate. With that said, we can optionally calculate a real-valued RSS for a complex-valued PDE by the comparison to $\left\|\frac{\partial\mathcal{F}_{\hat{\theta}}}{\partial t^{\mathcal{M}}}\right\|_2$ instead.

Furthermore, assume the spatial metadata (maybe in a higher dimension) is in mesh form. In that case, we can exploit the other noise-tolerant library representation or PDE discovery method, such as WF [5] or CWF [6]. One extension of our generic initial PDE identification is by passing reshapable $\mathcal{F}_{\hat{\theta}}(x^{\mathcal{M}},t^{\mathcal{M}})$ as input to the mentioned algorithms.

Later in section 4.3, we shall see that the heuristics search for the agreed PDEs with the minimal BIC score is questionable in terms of future applications, where there may be no expert to supervise an acceptable range of $(\lambda_1,\lambda_{\mathrm{STR}})$. Nevertheless, we observe that the BIC decay rate (per one increasing candidate), e.g. $\frac{\Delta\mathrm{BIC}}{\Delta\|\xi^{\mathrm{STR}}\|_0}$ between different estimated PDEs from λ_{STR}-varying STRidge with a fixed λ_1, can assist as an explicit information metric, which inspires us to design the automatic complexity selection (ACS) algorithm of the optimal number of effective candidates (complexity) detailed in algorithm 3. Additionally, the corresponding PDE is ensured to be traceable.

Pedagogically, suppose that the preferred initial PDE exemplifies Burgers' PDE; we write the effective candidate matrix concerning the training set of labeled subsamples $\mathcal{D}$ as

$$\Phi_{\mathcal{E}}^{\mathcal{D}}(\hat{\theta}) = P_k(\Phi^{\mathcal{D}}(\hat{\theta}))\mathcal{E} = \begin{bmatrix}\frac{\partial^2\mathcal{F}_{\hat{\theta}}}{\partial x^2} & \mathcal{F}_{\hat{\theta}}(x,t)\frac{\partial\mathcal{F}_{\hat{\theta}}}{\partial x}\end{bmatrix}. \tag{12}$$

3.4. dPINNs: denoising and finetuning using PINN

As illustrated by ③ of figure 1, we introduce the dPINNs for achieving the precise recovery of PDE coefficients ξ^* under uncertainties. After algorithm 1 is performed, we take the weakly physics-constrained solver $\mathcal{F}_{\hat{\theta}}$ and the initial PDE $\hat{\mathcal{N}}_{\hat{\xi}}$ to build the dPINNs, minimizing the vigorous physics-informed loss $\mathcal{L}_{\mathrm{sup}}^{\tilde{\mathcal{D}}}(\hat{\theta}) + \mathcal{L}_{\mathrm{unsup}}^{\tilde{\mathcal{D}}'}(\hat{\theta},\hat{\mathcal{N}}_{\hat{\xi}})$ on the denoised dataset $\tilde{\mathcal{D}} = \{(\tilde{x}_i,\tilde{t}_i,\tilde{u}_i)_{i=1}^{N_f}\}$. The physics loss is generally given by

$$\mathcal{L}_{\mathrm{unsup}}^{\tilde{\mathcal{D}}'}(\hat{\theta},\hat{\mathcal{N}}_{\hat{\xi}}) = \frac{1}{N_f}\sum_{i=1}^{N_f}\left(\frac{\partial\mathcal{F}_{\hat{\theta}}}{\partial\tilde{t}_i} - \hat{\mathcal{N}}_{\hat{\xi}}[(\Phi_{\mathcal{E}}^{\tilde{\mathcal{D}}'}(\hat{\theta}))_i]\right)^2, \tag{13}$$

where the unsupervised set $\tilde{\mathcal{D}}' = \{(\tilde{x}_i,\tilde{t}_i)_{i=1}^{N_f}\}$ is viewed simply as the slice of $\tilde{\mathcal{D}}$ without the supervision. Let us now continue the Burgers' example, we can derive the physics-constraint as

$$\hat{\mathcal{N}}_{\hat{\xi}}[(\Phi_{\mathcal{E}}^{\tilde{\mathcal{D}}'}(\hat{\theta}))_i] = P_k(\Phi_i^{\tilde{\mathcal{D}}'}(\hat{\theta}))\mathcal{E}\hat{\xi} = \hat{\xi}_1\frac{\partial^2\mathcal{F}_{\hat{\theta}}}{\partial\tilde{x}_i^2} + \hat{\xi}_2\mathcal{F}_{\hat{\theta}}(\tilde{x}_i,\tilde{t}_i)\frac{\partial\mathcal{F}_{\hat{\theta}}}{\partial\tilde{x}_i}. \tag{14}$$

To continually denoise $\mathcal{D}$ during the dPINNs' learning, we subtract the transformed noises, initially precomputed by the DFT algorithm, from both (x,t) and u. The denoising mechanism is formulated as the double affine transformations of the entire training dataset given by

$$\begin{aligned}
(\tilde{x},\tilde{t}) &= (x,t) - \beta_{(x,t)}\odot\mathcal{P}_{\Omega_{(x,t)}}(S_{(x,t)}); \quad S_{(x,t)} = (S_x,S_t),\\
\tilde{u} &= u - \beta_u\odot\mathcal{P}_{\Omega_u}(S_u),
\end{aligned} \tag{15}$$

Algorithm 2. Denoising physics-informed neural networks' (dPINNs) learning.

Goal: Achieve the optimal solver θ^* and PDE coefficients ξ^*
Require: (x,t), u, $\hat{\theta}$[a], $\hat{\mathcal{N}}_{\hat{\xi}}$, initialized $\Omega_{(x,t)}$, $\beta'_{(x,t)}$, Ω_u, β'_u
1: Compute $S_{(x,t)}$, and S_u using denoising DFT defined in equation (16)
2: Assign $\beta_{(x,t)} \leftarrow (\sqrt{\mathbb{V}(x)}\beta'_{(x,t)}, \sqrt{\mathbb{V}(t)}\beta'_{(x,t)})$ ▷ row vector
3: Assign $\beta_u \leftarrow \sqrt{\mathbb{V}(u)}\beta'_u$ ▷ single parameter
4: **While** not converge **do**
5: Denoise $(\tilde{x},\tilde{t}) \leftarrow (x,t) - \beta_{(x,t)} \odot \mathcal{P}_{\Omega_{(x,t)}}(S_{(x,t)})$
6: Denoise $\tilde{u} \leftarrow u - \beta_u \odot \mathcal{P}_{\Omega_u}(S_u)$
7: Build $\tilde{\mathcal{D}}' \leftarrow \{(\tilde{x}_i,\tilde{t}_i)_{i=1}^{N_f}\}$ and $\tilde{\mathcal{D}} \leftarrow \{(\tilde{x}_i,\tilde{t}_i,\tilde{u}_i)_{i=1}^{N_f}\}$
8: Compute loss $\mathcal{L}^{\tilde{\mathcal{D}}}_{\text{sup}}(\hat{\theta}) + \mathcal{L}^{\tilde{\mathcal{D}}'}_{\text{unsup}}(\hat{\theta},\hat{\mathcal{N}}_{\hat{\xi}})$ on $\tilde{\mathcal{D}}$ and $\tilde{\mathcal{D}}'$
9: Gradient-based update $\hat{\theta}$, $\hat{\xi}$, $\Omega_{(x,t)}$, $\beta'_{(x,t)}$, Ω_u and β'_u
10: **end while**
11: Minimize $\mathcal{L}^{\tilde{\mathcal{D}}}_{\text{sup}}(\hat{\theta}) + \mathcal{L}^{\tilde{\mathcal{D}}'}_{\text{unsup}}(\hat{\theta},\hat{\mathcal{N}}_{\xi^*})$; $\hat{\mathcal{N}}_{\xi^*}$ is represented by $\xi^* \leftarrow ((\Phi^{\tilde{\mathcal{D}}'}_{\mathcal{E}}(\hat{\theta}))^\intercal \Phi^{\tilde{\mathcal{D}}'}_{\mathcal{E}}(\hat{\theta}))^{-1}(\Phi^{\tilde{\mathcal{D}}'}_{\mathcal{E}}(\hat{\theta}))^\intercal \frac{\partial \mathcal{F}_{\hat{\theta}}}{\partial \hat{t}}$
▷ Redo line 4–10 with ξ^* iteratively resolved between line 7 and 8 by LS instead of its gradient-based update at line 9.
12: **return** (x^*,t^*)[b], u^*, θ^*, ξ^*, $\Omega^*_{(x,t)}$, $\beta^*_{(x,t)}$, Ω^*_u and β^*_u

[a] $\hat{\theta}$ and $\hat{\mathcal{N}}_{\hat{\xi}}$ are attained from algorithm 1.
[b] The learned outputs are assigned as the optimal parameters superscripted with the asterisk $(*)$ notation.

where $\mathcal{P}_{\Omega_{(x,t)}}$ and $\mathcal{P}_{\Omega_u}$ are the projecting functions parameterized by $\Omega_{(x,t)}$ and Ω_u, capturing the unknown noise distributions. $\beta_{(x,t)}$ and β_u are updated proportional to the unbiased standard deviations $(\sqrt{\mathbb{V}(x)}, \sqrt{\mathbb{V}(t)})$ and $\sqrt{\mathbb{V}(u)}$, controlling the relevant comparable intensity of the noise corrections. The denoising DFT algorithm, which considers power spectrum density (PSD), is meant to deduct small power frequencies components. The starting noises S_u and $S_{(x,t)}$ are obtained by limiting frequencies whose power is less than the threshold ζ. To attain the low-PSD noise for the signal $\psi \in \{x,t,u\}$, we compute the following quantities:

$$
\begin{aligned}
S_\psi &= \psi - \text{DFT}^{-1}(\text{DFT}^{\zeta}(\psi));\\
\text{DFT}^{\zeta}_k(\psi) &= \begin{cases} \text{DFT}_k(\psi); & \text{if} \quad \text{PSD}_k(\psi) > \zeta \\ 0; & \text{otherwise,} \end{cases}\\
\text{PSD}_k(\psi) &= \frac{1}{N_f}\|\text{DFT}_k(\psi)\|_2^2,\\
\widetilde{\text{PSD}}_k(\psi) &= \frac{\text{PSD}_k(\psi) - \mathbb{E}(\text{PSD}(\psi))}{\sqrt{\mathbb{V}(\text{PSD}(\psi))}},\\
\zeta &= \mathbb{E}(\text{PSD}(\psi)) + \alpha \max_k(\widetilde{\text{PSD}}_k(\psi))\sqrt{\mathbb{V}(\text{PSD}(\psi))}.
\end{aligned}
\tag{16}
$$

Here, k denotes an index in the frequency domain. ζ is defined according to the α portion of the maximal normalized PSD. $\mathbb{E}$ and $\mathbb{V}$ calculates the sample mean and variance over k. We precompute $S_{(x,t)}$ and S_u since the gradients cannot flow to α. The denoising physics-informed learning is described in algorithm 2. Succeeding the first optimization loop, to compensate the numerical error, least squares (LS) regression (see line 11) is repeatedly employed on the denoised dataset $\tilde{\mathcal{D}}'$ until the convergence, i.e. no changes of the optimal unbiased ξ^* are detected between the learning epochs.

4. Experiments and results

We experimented with five canonical PDEs, including three ordinary PDEs and two complex-valued PDEs, to investigate the accuracy and robustness of our proposed method. We present the results of ① derivative preparation and ② initial PDE discovery and discuss the regularization hyperparameter effects on finding the appropriate initial PDE. Later, we show the tolerance of ③ dPINNs against noise in both $(x,t)\&u$ for each PDE as well as against the decreasing number of training samples (scarce data). Beyond the numerical results, we visualize how the projection networks handle the increasing noise intensity in the exemplar of discovering Burgers' PDE.

4.1. Canonical PDEs

4.1.1. Burgers' PDE

The equation arises in sub-areas of applied mathematics, such as fluid mechanics and traffic flow. We consider the following Burgers' equation dataset simulated with Dirichlet boundary conditions, studied in [9]

$$u_t + uu_x - \nu u_{xx} = 0; \quad \nu = \frac{0.01}{\pi}, x \in [-1, 1], t \in [0, 1]. \tag{17}$$

Different from the previous works such as [1, 37], where the viscosity of fluid ν was set to 0.1, thus the smooth fluid speed without a shock wave. Here, $\nu = \frac{0.01}{\pi}$ is so small that the shock wave emerges. Spectral methods and standard finite differences [30] are used to accurately simulate the PDE with this small viscosity.

4.1.2. Korteweg–de Vries (KdV) PDE

The KdV equation [38] is a nonlinear dispersive PDE for describing the motion of unidirectional shallow water surfaces. For a function $u(x, t)$ the actual form of KdV we consider is expressed as

$$u_t + 6uu_x + u_{xxx} = 0; \quad x \in [0, 50], t \in [0, 50]. \tag{18}$$

KdV was known to have soliton solutions, representing two one-way moving waves with different amplitudes. Such characteristics challenge discovery methods to distinguish and yield the sparsest governing PDE that generalizes the situation. The PDE is also an excellent prototypical example to test discovering the relatively high-order spatial derivative u_{xxx}. For the PDE simulation, Scipy's odeint function [39] is applied to derivatives computed by the pseudo-spectral method.

4.1.3. Kuramoto–Sivashinsky (KS) PDE

The KS or flame equation is a chaotic nonlinear PDE with a spatial fourth-order derivative term, primarily to model the diffusive instabilities in a laminar flow. The PDE reads

$$u_t + uu_x + u_{xx} + u_{xxxx} = 0; \quad x \in [0, 100], t \in [0, 100]. \tag{19}$$

The solution was generated with an initial condition $u(x, 0) = \cos(\frac{x}{16})(1 + \sin(\frac{x}{16}))$, integrated up to the wide temporal bound of $[0, 100]$ [1] using a spectral method in MATLAB [40]. Consequently, we got a chaotic and complicated PDE solution. Raissi [37] very first noticed that it was challenging to fit a vanilla neural network to the entire chaotic solution while minimizing the residual physics loss; for example, $\min_{\theta,\theta_s}(\mathcal{L}^{\mathcal{D}}_{\text{sup}}(\theta) + \mathcal{L}^{\mathcal{D}_s}_{\text{unsup}}(\theta, \theta_s))$. A similar problem was independently found by Rudy *et al* [1]. When encountering the whole chaotic domain of KS, the PDEs produced by STRidge could be inaccurate and unstable with the complication of noise.

4.1.4. Quantum harmonic oscillator (QHO) PDE

The QHO is the Schrodinger equation with a parabolic potential $0.5x^2$. The PDE is given by

$$iu_t + \frac{1}{2}u_{xx} - \frac{x^2}{2}u = 0; \quad x \in [-7.5, 7.5], t \in [0, 4]. \tag{20}$$

We construct the basis candidate matrix that includes the parabolic potential. The dataset is temporally sliced from [1].

4.1.5. Nonlinear Schrodinger (NLS) PDE

The NLS equation is used to study nonlinear wave propagation. The PDE and its true discretization read as

$$iu_t + \frac{1}{2}u_{xx} + u\|u\|_2^2 = 0; \quad x \in [-5, 5], t \in \left[0, \frac{\pi}{2}\right]. \tag{21}$$

We include candidate terms depending on the magnitude of the solution, e.g. $\|u\|_2^2$, which may appear in the correct identification of the dynamics of the complex-valued function. We experiment with the exact dataset from [9].

4.2. Experimental settings

The training data points $(x, t)\&u$ are randomly subsampled from all the generated discretized points in the domain according to the size N_f specified in table 6. All the discretized (noisy) data points are exploited as the validation set for early stopping after detecting the rise(s) in the validation MSE during the pretraining and converging of the solver network. $N_r = (1, 1, 0.5, 0.5, 1)N_f$ for Burgers', KdV, KS, QHO and NLS PDE, respectively. The solver architecture comprises 6 hidden layers with 50 neurons each and Tanh activation functions in the between. For the preselector, W^b are devised as a single hidden layer. At the same time, the rest parameters θ_s^r are implemented as a sequence of 3 hidden layers, each with 50 neurons whose outputs are layer normalized [41], randomly dropped out [42] and Tanh activated, excluding Tanh from the final layer. The dropout probability is 0.1 for KdV and KS, otherwise is 0. Hidden weights are initialized by uniform Xavier [43] and biases are initialized to 0.01. $\sigma(\cdot) = \frac{1}{2}(\tanh(\cdot) + 1)$ is defined for all the canonical models except for Burgers' PDE, $\sigma(\cdot) = \frac{1}{1+\exp(-1(\cdot))}$, Sigmoid is employed to convey the flexibility in the design. λ_1 is varied for accomplishing the suitable value while λ_2 is set to 0.1. The projection networks $\Omega_{(x,t)}$ and Ω_u are 2 hidden layers, each having 32 neurons with Tanh; hence, the final layer's raw outputs of the networks $\mathcal{P}_{\Omega_{(x,t)}}$ and $\mathcal{P}_{\Omega_u}$ are activated by Tanh. $\beta'_{(x,t)}$ and β'_u are initialized at 10^{-3} for the ordinary PDEs and 10^{-5} for the complex-valued PDEs (QHO and NLS).

For algorithm 1, full-batch stochastic LBFGS [44] and vanilla LBFGS [45], with 0.1 step sizes and the strong Wolfe line search, are leveraged separately, to pretrain and converge the solver network. The pretraining (second-order optimization) epoch is limited to 1 to prevent overfitting in the noisy $(x, t)\&u$ case. MADGRAD [46] with gradient-deconflicting PCGrad [34] is applied to joint learn (line 1) for 1000 epochs in Burgers' and KdV cases. The weighted average with the ratios $\mathcal{L}_{\text{sup}}^{\mathcal{D}}(\theta) : (\mathcal{L}_{\text{unsup}}^{\mathcal{D}_s}(\theta, \theta_s) + R_\eta^{\mathcal{D}_s}(\theta, W^b)) = 1 : 1$ and $1 : 10^{-3}$ are put to optimize for 300 and 1500 epochs in KS and the complex-valued PDEs. The learning rate for updating the pretrained θ is assigned with a low value of 10^{-7}, while the higher rates from $(10^{-2}, 10^{-2}, 10^{-3}, 10^{-1}, 10^{-1})$ are set for updating untrained θ_s. κ is set, in the same dataset order, to $(0.75, 0.7, 0.8, 0.9, 0.9)$ before the first gradient updates of the joint training. Then, LBFGS [45] is mainly used for the dPINNs' learning (algorithm 2). For every noisy KdV and KS experimental case, the denoising-related parameters $\Omega_{(x,t)}$, $\beta'_{(x,t)}$, Ω_u and β'_u are reinitialized with the conceivably closer estimate $\hat{\theta}$ prior to executing the subroutine at line 11.

As for the input of STRidge, the candidate library is $P_2(\cdot)$, collecting nonconstant interaction-only real-valued polynomial features up to the 2nd degree of the estimated PDE solution and its partial derivatives $P_2(\cdot)$, computed with respect to $\mathcal{M}$.

The precomputed denoising DFT is configured with $\alpha = 0.1$ for all the canonical PDEs. DFT and DFT^{-1} (the inverse transform) are the one-dimensional `fft` and `ifft` functions from PyTorch [47] package. Our nPIML framework is as well implemented dominantly using PyTorch package.

Step ① and ③ of the proposed framework are run on a single Quadro RTX graphics processing unit (GPU) with 49 152 MiB memory in less than an hour. Step ② is run on a CPU with the following specification: 2.6 GHz 6-Core Intel Core i7 with 32 GB ram 2667 MHz DDR4.

In the noisy experiments, we presume that a matrix, say z, gets perturbed, right after the time of its subsampling, by the p% biased (no Bessel's correction) standard deviation (std) of Gaussian noise Z simulated as follows:

$$\text{noise}(z, p) = \frac{p \cdot \text{std}(z)}{100} \times Z; \forall i, j(Z_{ij} \sim \mathcal{N}(0, 1)). \tag{22}$$

Suppose that 1% noise is exerted, subsampled u and (x, t) get polluted in turn with $\text{noise}(u, 1)$ and $(\frac{\text{noise}(x,1)}{\sqrt{2}}, \frac{\text{noise}(t,1)}{\sqrt{2}})$.

The metric to measure how far an estimate ξ^{est} from the ground truth ξ is $\text{mean}(\delta) \pm \text{std}(\delta)$ over all j effective coefficients in ξ^{est}. If only the correct candidates are identified, $\delta_j = \delta_j(\xi^{\text{est}}, \xi)$ is the %coefficient error (%CE) defined as

$$\delta_j = \left|\xi_j^{\text{est}} - \xi_j\right| \Big/ \left|\xi_j\right| \times 100\%; j \in \{1, \ldots, \text{cols}(\Theta)\}. \tag{23}$$

In tables 6–8, $\xi^{\text{est}} \in \{\hat{\xi}, \xi^*\}$. $\text{cols}(\Theta)$ represents the number of columns of Θ.

Specific treatments for complex-valued PDEs: our complex neural networks are initialized based on the prior work called deep complex networks [48]. Since the spatio-temporal points lay on a real two-dimensional (2D) plane, the model starts from 1 (real) hidden layer with 200 neurons, followed by 5 complex linear layers, each consisting of 200 neurons that account for 100 real parameters and 100 imaginary parameters. Note that the complex forward pass is essentially iteratively performing naive complex-valued matrix multiplication and bias addition. The differentiation of complex-valued $\mathcal{F}_\theta(x, t)$, respecting a

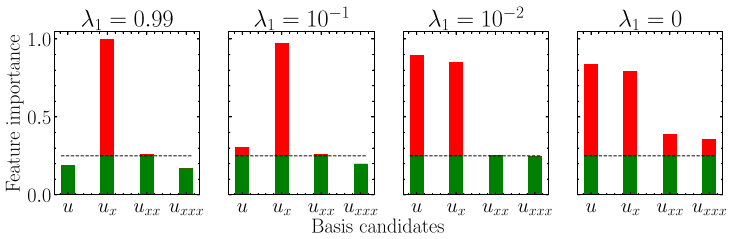


Figure 2. Burgers: learned feature importance with varied λ_1.

Table 1. Burgers regularization hyperparameter selection: concerning the coefficient selection criteria, STRidge's λ_0, controlling the L_0-penalty, is set to $10^4\lambda_{\text{STR}}\varepsilon$, and d_{tol} equals 2 for the three noise conditions. The assignment of $(\mu, \lambda_{\text{STR}}, d_{\text{tol}})$ is purely for gathering the likely different PDEs. Each PDE is accompanied by the '(BIC)' score. Blue indicates the agreement. **Bold** means the lowest BIC score, compared to the scores acquired by the same λ_1. Among the agreed models, we check (✓) the sparse PDEs with $\left\|\xi^{\text{STR}}\right\|_0 \leqslant 4$, which demonstrate sufficiently low BIC score. Out of those, the PDE with ★ is regarded as the initial guess.

$\lambda_1/\lambda_{\text{STR}}$	10^{-6}	10^{-3}	10^{0}
0.99	$[u_{xx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xx}, uu_x]$	$[uu_x]$
	(**−8723.69**)	(−7636.39)	(15 823.14)
10^{-1}	$[u_{xx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xx}, uu_x]$	$[uu_x]$
	(**−8456.28**)	(−7154.65) ✓	(15 824.98)
10^{-2}	$[u_{xx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xx}, uu_x]$	$[uu_x]$
	(**−8294.55**)	(−7178.84) ★	(15 824.29)
0 (supplement)	$[u_{xx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xx}, uu_x]$	$[uu_x]$
	(**−8437.81**) ✓	(−7243.32) ✓	(15 827.68)

real-valued vector, e.g. x, can be computed distributively. Concretely, we apply AD to the real and imaginary parts (denoted by $\Re(\cdot)$ and $\Im(\cdot)$) with respect to x separately; then, we form the output complex-valued matrix as

$$\frac{\partial \mathcal{F}_\theta(x,t)}{\partial x} = \frac{\partial \Re(\mathcal{F}_\theta(x,t))}{\partial x} + \frac{\partial \Im(\mathcal{F}_\theta(x,t))}{\partial x} i;\ i^2 = -1. \tag{24}$$

Likewise, W^b of the preselector is treated as a single complex linear layer, including the bias, with 50 neurons. θ_s^r is modeled by 3 complex linear layers, each with total 50 neurons that are batch normalized [49] and component-wise Relu activated.

Because the estimated PDE solution is complex-valued, we may include norm-based candidates, e.g. $\left\|\mathcal{F}_{\hat{\theta}}(x^{\mathcal{M}}, t^{\mathcal{M}})\right\|_2^2$, as one of the bases that build all the nonconstant polynomial features (not interaction-only) up to the 2nd degree. Once prepared, the candidate library can be directly input to STRidge.

4.3. Effect of regularization hyperparameters on initial PDE identification

For each canonical PDE, we present the domain of interest from which the metadata $\mathcal{M}$ is generated for the initial PDE extraction. We then concentrate on the multi-perspective assessment of the different discovered PDEs by STRidge while varying the two major regularization hyperparameters: λ_1 of the preselector network and λ_{STR} of STRidge algorithm. Before the finetuning process, we present how accurate the initial discovered PDEs concerning the following three cases distinguished by the noise conditions: noiseless dataset, noiseless (x, t) but noisy u, and noisy $(x, t)\&u$ in which the spatial-temporal (x, t) becomes mesh-free.

4.3.1. Initial discovered Burgers' PDE

We trained the preselector network with varying λ_1 to perceive the significance of each candidate. The distributed feature importance values (I_j for each jth basis candidate) are presented in figure 2. Although several choices of the expressive subset of threshold-passing candidates are contributed, identifying the optimal set is still not obvious by merely adjusting λ_1. Hence, STRidge was subsequently employed multiple times with diverse levels of regularization intensity λ_{STR}. For convenience, we simply set $\forall i \leqslant N_f + N_r, (x_i^{\mathcal{M}}, t_i^{\mathcal{M}}) = (x_i, t_i)$ for all Burgers' experimental cases that differed in the noise conditions. The cross results, table 1, are assessed for obtaining the initial discovered PDE that agrees with the corresponding preselector and sparse with a sufficiently low BIC score.

Assigning the $\lambda_1 = 0.99$ is so high that the true candidate, i.e. u, is lacking from the threshold-passing candidates. Accordingly, the resulting PDEs cannot match the particular importance scores. The preselector

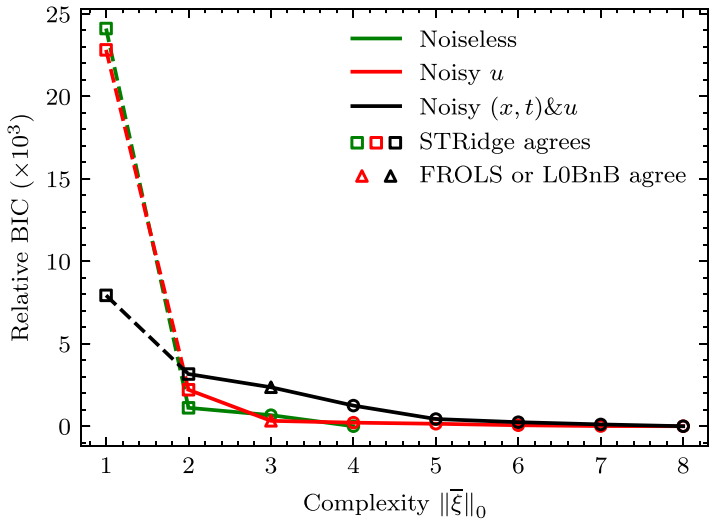
properly focuses on the true candidates when λ_1 is set to 10^{-1} and 10^{-2}. Notice that u_{xx} consistently passes the threshold with marginal values, conveying the small viscosity estimates. As seen in table 1, for $\lambda_1 > 0$, 10^{-2} gave the best initial result, covering the sparse PDE with the lowest BIC among the agreed models. We shall examine more to find out soon what BIC level herein should be considered sufficiently low.

Deciding on the value of λ_{STR} requires an akin principle: the values that are too low or high are likely to yield incorrect forms. For example, $\lambda_{\text{STR}} = 10^0$ is immensely high, outputting the too sparse and noninformative PDE with the single effective uu_x, delivering the high BIC scores. $\lambda_{\text{STR}} = 10^{-3}$ is more suitable, suggesting the sparse models, which conform with the preselectors and offer the low BIC scores that vastly improve from those given by $\lambda_{\text{STR}} = 10^0$. Conditioned by $\lambda_1 > 0$, $u_t = 0.003\,063u_{xx} - 0.986\,174uu_x$ contains the few terms and offers the minimal BIC among the acceptable PDEs; thus, taken as our initial guess (★) to be finetuned. Remind that, when considering the models from diverse values of λ_1, although their functions differ solely in the PDE coefficient values, they cannot be compared by BIC because the change in $\hat{\theta}$ affects $\mathcal{F}_{\hat{\theta}}(\cdot)$, i.e. $\frac{\partial \mathcal{F}_{\hat{\theta}}}{\partial t^{\mathcal{M}}}$ varies (see equation (11)), hence the incomparable RSS without an explicit static referenced time derivative. Nevertheless, we straightforwardly prefer the one with the lower BIC score. By the disagreements, the sparsity-promoting preselectors, trained with $\lambda_1 > 0$, all entail that $\lambda_{\text{STR}} = 10^{-6}$ gives overly parameterized models with the minor improvements per the increased independent candidates. If we were to independently have the mere consideration on $\lambda_1 = 0$ or technically diminutive to a certain value, none of the basis candidates would probably get deselected, and the resulting PDEs would be all in their agreements. The justification, whether including uu_{xxx} and $u_x u_{xx}$ worth the reduction in BIC, would turn ambiguous, though the PDE outcome by $(\lambda_1, \lambda_{\text{STR}}) = (0, 10^{-3})$: $u_t = 0.003\,063u_{xx} - 0.985\,882uu_x$ captures the ground on a par with our PDE guess (★). If the preselector were not at all constructed, the concern would still persist. For the noisy cases, the %CE (see equation (23)) of the initial PDE estimates are listed in the nPIML:IPI row of table 6.

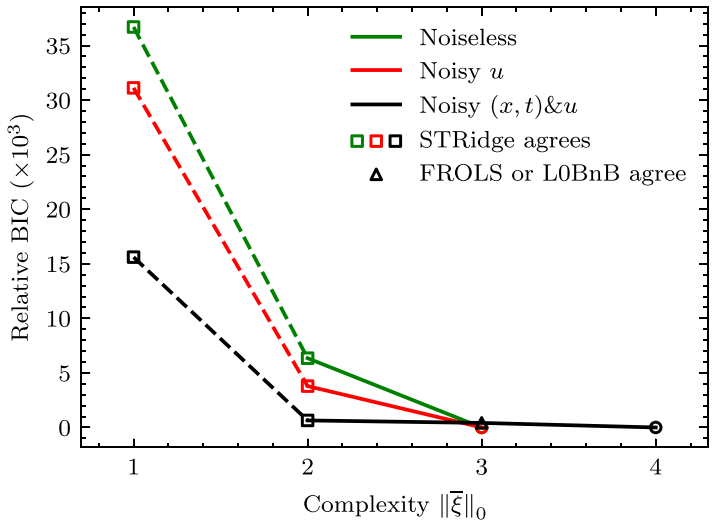
We have comprehensively decided on the initial PDE solely based on the agreement and BIC derived from the simulation result. However, the experiment on $\lambda_1 = 0$ gives a sense that the agreed PDE that minimize BIC respecting a small $\lambda_1 < 10^{-2}$ can have an arbitrary complexity, especially if there is no human in the loop to supervise the choices of $(\lambda_1, \lambda_{\text{STR}})$, which influences the perception of sufficiently low BIC. Since the conducted heuristics need to be more extensive, and we have yet to impose an explicit selection metric, let us seek a useful general metric to tackle real-world problems by analyzing the 'BIC decay rate' with respect to the increasing number of effective candidates, $\frac{\Delta \text{BIC}}{\Delta \|\bar{\xi}\|_0}$. Unlike the previous work [12], which underwent expensive numerical integration to a found PDE to obtain AIC based upon comparison to the PDE solution on full domain grid, our BIC is of the model that predicts the system evolution $\frac{\partial \mathcal{F}_{\hat{\theta}}}{\partial t}$ estimated by the solver network instead, avoiding simulation of various false PDE forms which arise ubiquitously.

Towards such a pursuit, STRidge alone is inadequate to create the Pareto front (see figure 3) because, for a λ_1, not every complexity less than a $\max_{\lambda_{\text{STR}}}(\|\xi^{\text{STR}}\|_0)$ is considered. Also, we are not guaranteed to obtain the optimal PDE for a unique output complexity. Therefore, we employ additional algorithms to estimate the best subset of candidates naturally for each complexity, e.g. FROLS (forward regression with orthogonal least squares) [50, 51] and L0BnB (sparse regression where the maximum number of nonzero candidates are determined beforehand) [52]. Specifically, FROLS and L0BnB take $P_2(\Phi^{\mathcal{M}}(\hat{\theta}))$ to predict $\frac{\partial \mathcal{F}_{\hat{\theta}}}{\partial t}$ with a constraint that the number of effective candidates is less than $\max_{\lambda_{\text{STR}}}(\|\xi^{\text{STR}}\|_0)$. The ridge hyperparameter of L0BnB is set according to table 10. We union the solutions produced by λ_{STR}-varying STRidge, FROLS and L0BnB. Next, backward elimination [27] coupled with calculating the most predictive candidates (both consider based on MSE score) for every decreased complexity are applied iteratively to each unique solution until we are left with a single effective candidate while tracking all the BIC scores to update the best subset for each complexity. In doing so, we extend `RFE (with refitted ordinary linear regression to assign the MSE score)` and `SelectKBest` functions from `sklearn.feature_selection` [53]. As illustrated in figure 3, the PDE that results in the transition minimizing the BIC decay rate is within the group of PDEs on which the preselector and regressors (STRidge, FROLS or L0BnB) can agree. This is observable for both the polynomial library and WF [5] in the three noise conditions. The %CEs of the coefficients recovered by extending the initial PDE identification with WF or CWF [6] (nPIML:IPI + WF/CWF) are given in table 6. The evidence convinces us to investigate and devise an algorithm for PDE selection based on the BIC decay rate.

To study the Pareto front in high noise situations, we try adding noise directly to the estimated full-domain PDE solution by the solver $\mathcal{F}_{\hat{\theta}}$ attained by algorithm 1 without joint training (line 1) on the noiseless dataset, prior to computing polynomial library and WF. We keep increasing the noise intensity until before the falsely discovered PDE at the correct complexity occurs. Compared to figure 3, similar trends are seen on the left of figure 4, but the %CEs are quite high: 61.08 ± 3.62 (1%/Poly.) and 18.07 ± 17.65 (6%/Weak.). With an ensemble of three WFs with different 10 000 domain centers, algorithm 3 yields the



(a) Polynomial library



(b) Weak formulation: an ensemble of 30 instances, each having $N_{\mathcal{M}} = 3,000$ integration domain centers. Finite differences calculate spatial derivatives on the full-domain PDE solution estimated by the solver.

Figure 3. Burgers: Pareto front between relative BIC and complexity. A relative BIC measures the increased value from the minimum BIC across every considered complexity. Dashed lines indicate the transition detected by algorithm 3. The PDEs formable using the threshold-passing candidates of the preselector network trained with $\lambda_1 = 10^{-2}$ has two types: ones that STRidge agrees (denoted by □) in the first place and the others on which FROLS or L0BnB agree (△).

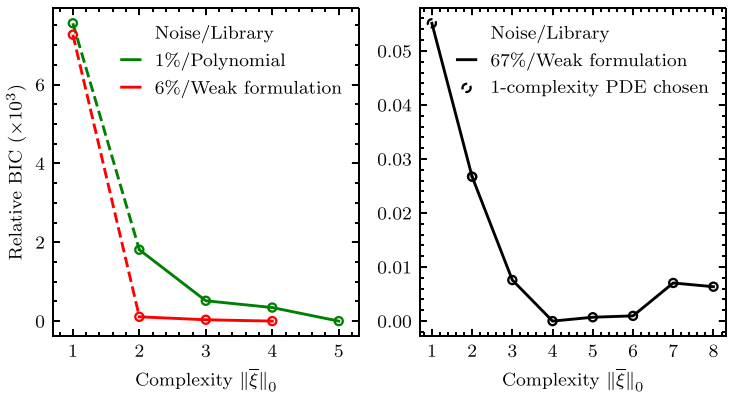


Figure 4. Burgers: Pareto front in high noise situations.

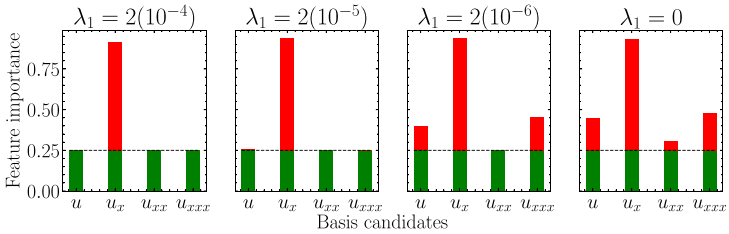


Figure 5. KdV: learned feature importance with varied λ_1.

Table 2. KdV regularization hyperparameter selection: STRidge's λ_0 is set to $10^2\lambda_{\text{STR}}\varepsilon$, and d_{tol} equals 1 for the three noise conditions. Blue indicates the agreement. **Bold** means the lowest BIC score, compared to the scores acquired by the same λ_1.

$\lambda_1/\lambda_{\text{STR}}$	10^{-5}	10^{-3}	10^{-1}
$2(10^{-4})$	$[u_x, u_{xxx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xxx}, uu_x]$	$[u_x]$
	(**−651 496.23**)	(−593 260.84)	(−493 869.28)
$2(10^{-5})$	$[u_x, u_{xxx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xxx}, uu_x]$	$[u_x]$
	(**−651 650.73**)	(−593 259.27) ✓	(−493 885.29)
$2(10^{-6})$	$[u_x, u_{xxx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xxx}, uu_x]$	$[u_x]$
	(**−651 782.07**)	(−593 389.01) ★	(−493 868.73)
0 (supplement)	$[u_x, u_{xxx}, uu_x, uu_{xxx}, u_xu_{xx}]$	$[u_{xxx}, uu_x]$	$[u_x]$
	(**−651 733.37**)	(−593 275.19) ✓	(−493 851.71)

true PDE form of 2-complexity up to 6% noise level, instructing that the metric is indeed standalone and operable although the preselector is ablated or trained with an improper λ_1. On the right of figure 4, since the improvement is underserved due to the high noise, it turns unworthy of stepping to the 2-complexity PDE. This notably tells that detecting the extravagance of increasing complexity is crucial for selecting 1-complexity PDE, which we elaborate in appendix B.1. In appendix B, we confirm the usability of the BIC decay rate in detecting the true complexity of various PDEs.

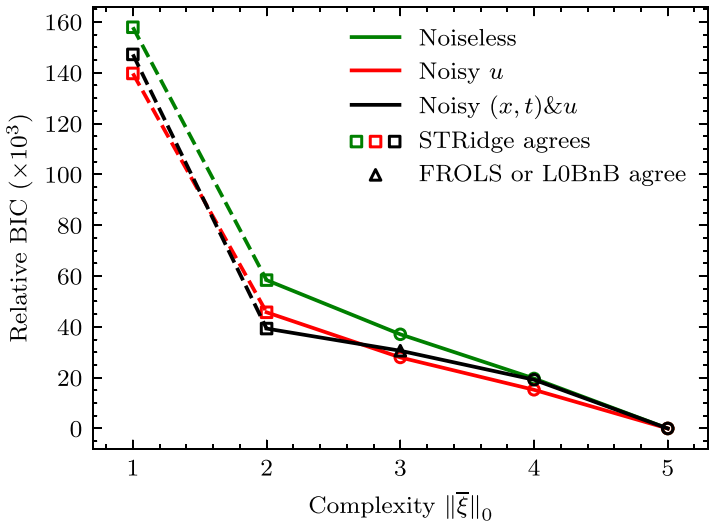
4.3.2. Initial discovered KdV PDE

We inspect how the preselector weights each basis candidate in figure 5. Trained with $\lambda_1 = 2(10^{-5})$ or $2(10^{-6})$, the preselector can capture the true candidates while the relatively high value of $\lambda_1 = 2(10^{-4})$ solely let u_x pass the threshold. u and u_{xxx} barely pass the threshold if $\lambda_1 = 2(10^{-5})$, nonetheless their effectivenesses become vivid when $\lambda_1 \leqslant 2(10^{-6})$.

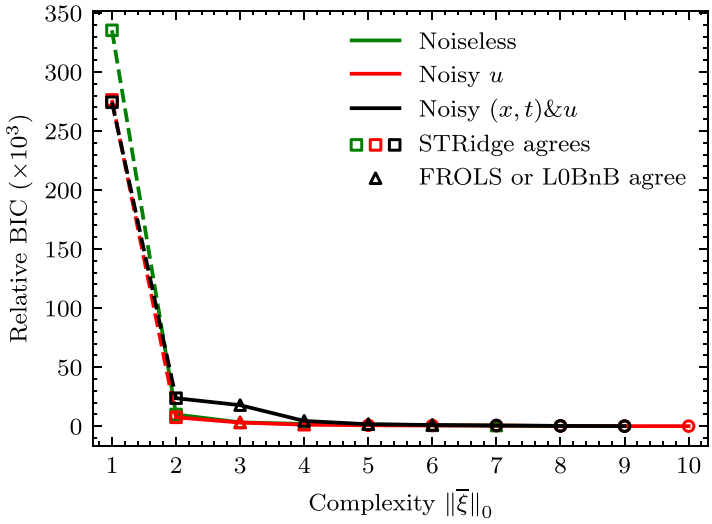
STRidge was leveraged multiple times on the candidate library built on $\mathcal{M}$. For KdV, we regarded the metadata as the linear discretization of the entire spatio-temporal domain; $N_{\mathcal{M}} = 64\,128$, facilitating the disambiguation of the different wave amplitudes. The found PDEs for the several pairs of $(\lambda_1, \lambda_{\text{STR}})$ are listed in table 2. By pondering the PDEs that harmonize with $\lambda_1 > 0$, we neglect the selection of the PDEs with the minimal BIC (for a particular λ_1) because they neither agree with the L_0-penalized feature importance nor be sparse as expected. The reduced BIC per an increasing effective term of transition from $\lambda_{\text{STR}} = 10^{-3}$ to $\lambda_{\text{STR}} = 10^{-5}$ is much less when compared with moving from $\lambda_{\text{STR}} = 10^{-1}$ to $\lambda_{\text{STR}} = 10^{-3}$, signifying the inefficiency of including the unnecessary terms. Remark that setting $\lambda_{\text{STR}} = 10^{-1}$ gives the PDEs, each describing a one-way traveling wave which can be considered as the relaxed form of KdV PDE, still not well fit the overall character of the dataset. Based on the mentioned justification, we thus prefer $\lambda_{\text{STR}} = 10^{-3}$, and choose the agreed PDE with the better BIC, taking the form of $u_t = -0.989\,065u_{xxx} - 5.961\,087uu_x$ as our initial guess (★). The selected PDE is noticed as a more precise to the ground truth than the PDE based $\lambda_1 = 0$, which is $u_t = -0.988\,350u_{xxx} - 5.959\,614uu_x$. Also, just naively, the BIC cannot elucidate the overfitting hurdle without the auxiliary knowledge gained by varying $\lambda_1 > 0$. For the noisy KdV cases, the initial results %CE of the algorithm 1 are as well shown in the nPIML:IPI row of in table 6.

In figure 6, we plot the KdV Pareto front created via the aid of the best-subset regressors: FROLS and L0BnB analogous to what is explained in section 4.3.1. Visually, we are inclined to stop after the transition to the agreed PDE between the preselector and STRidge at an elbow, where the BIC decay rate is minimized. Especially in the WF case, the reduced BICs become relatively small after having the actual candidates whose %CE is reported in the nPIML: IPI+WF row of table 6. In the case of the extension with CWF, the %CE is as well calculated. Algorithm 3 numerically verifies our intuition, not including extra candidates to the 2-complexity PDE.

In a realistic circumstance where basis candidates rendering a library for steps ⑴ and ⑵ are incomplete, we urge an exploration at step ⑵, growing from a small set of basis candidates to larger sets. For example, we



(a) Polynomial library



(b) Weak formulation: an ensemble of 3 instances, each having $N_{\mathcal{M}} = 64, 128$ integration domain centers.

Figure 6. KdV: Pareto front between relative BIC and complexity. The preselector network is trained with either $\lambda_1 = 2(10^{-5})$ (noisy cases) or $2(10^{-6})$ (noiseless case).

may start with the set without u_{xxx} and then gradually add the higher-order derivatives as demonstrated in figure 7. The area between the incomplete $C = 3$ and overcomplete $C = 4$ plots is big, like the considerable drop in BIC depicted in figure 6. If we enlarge the bases up to $[u, u_x, u_{xx}, u_{xxx}, u_{xxxx}]$, we still attain the identical (optimal, guaranteed by brute-force searches) effective candidates. However, the naive exploration path of solely adding the spatial derivative order may not be enough (e.g. lacking other necessary operations) to reach the complete library in other problems. Such incompleteness impacts the subsequent dPINNs' learning ③ in a bad manner. For example, the final physics loss would increase likewise to what happens in figure 7 owing to the wrong PDE finetuned. Training merely the PINN component ascertains the statement. In the overcomplete cases, the final losses for learning the PDE solution (supervised from data) and the discovered physics are around 10^{-6} and $5(10^{-7})$, respectively. But in the incomplete case, the losses escalate to $2(10^{-5})$ and 10^{-6}. To prevent the deficiency in future studies, we see a prospect in employing best-subset regression on a collection of basis candidates whose complexity grows by an evolutionary algorithm as formerly advocated by [28, 29].

4.3.3. Initial discovered KS PDE

Our early attempt was performing algorithm 1 with train/validation sets. The training samples were abundant as $N_f = 80,000$. $N_r = 0$ was chosen to avert the GPU memory overflow because of the computation up to the fifth-order u_{xxxxx}. Unfortunately, suggested by the plots in figure 8, we have quickly

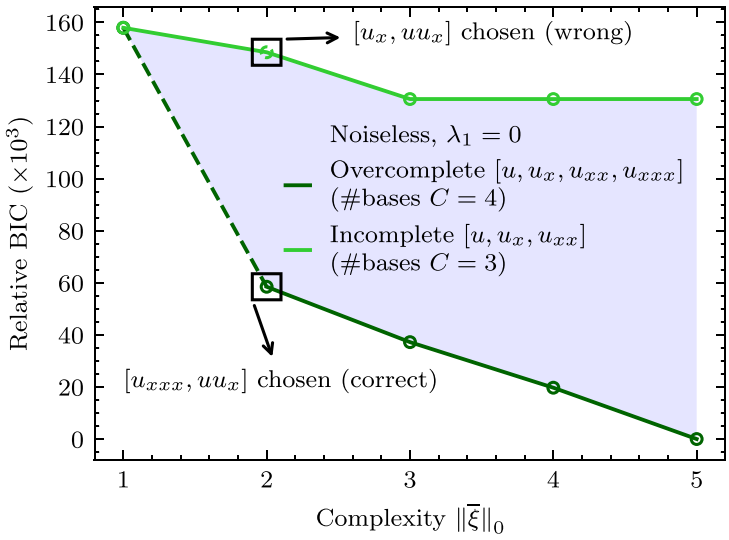


Figure 7. KdV: Pareto front when the candidate library $\Phi^{\mathcal{M}}(\hat{\theta})$ is intentionally turned incomplete. The preselector is kept fixed with $\lambda_1 = 0$, the same network as what is listed in table 2, to disable the guiding agreement.

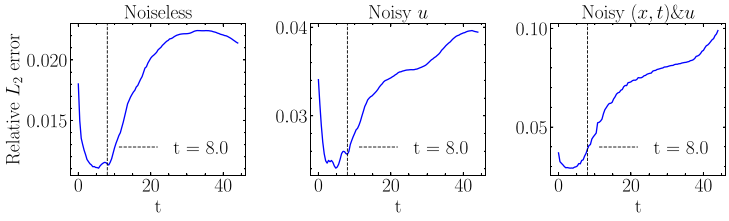


Figure 8. KS: training relative L_2 error of the learned (from $N_f = 80\,000$) solver $\hat{\theta}$ against temporally varying sub-regions of the KS training set bounded by $[0, 100] \times [0, 44]$, revealing a local optimum around the stability domain at the beginning of the evolution.

realized that the relative L_2 error of the solver network starts diverging, especially if noise exists when entering the highly chaotic region of KS, admonishing the evidential burden of training PINN upon the full-field domain [37]. The issue leads to unreliable derivation estimation, hence non-sparse and cluttered discoveries of the governing function by STRidge.

We bypass the complication by selectively focusing on the samples from a more stable sub-region at the beginning of the evolution, where the solver can accurately approximate as indicated by the relative L_2 error plots in figure 8. We assumed that the unknown PDE governs persistently throughout the evolution; nevertheless, the presumption does not universally hold, since specific coefficients of the chaotic behavior can be distinct over time [54]. Based on the encountered evidences, as a result, the first 21 504 (1024×21) discretized points within $[0, 100] \times [0, 8]$, were instead used with randomly generated nonoverlapping 10 752 unsupervised points for the (re)training in the noiseless experiment. The temporally-wise increased number of training samples to be the first 30 000 polluted discretized points, where $t \leqslant 11.6$, were used with randomly generated disjoint 15 000 unsupervised points for both the noisy experiments. The validation sets were commonly left unaffected. Before the initial PDE identification, we retrained the networks using algorithm 1 once from scratch on these altered training sets for better stability.

We investigate the learned feature importance of the preselector for ranking each potential atomic candidate, helping us choose the right PDE as presented in figure 9. It is intriguing to discern that u_{xxxx} is one of the essential terms for every choice of λ_1, despite its order being 4, implying the possibility of including the high-order derivative.

We list the possible PDEs provided by STRidge for the various set of regularization hyperparameters in table 3. The metadata was specified as the 21 000 samples ($N_{\mathcal{M}}$) within the $[0, 100] \times [0, 8]$ boundary generated by a Latin hypercube strategy [55]. It alludes to us that the $\lambda_{\text{STR}} = 10^{-5}$ founded PDEs cannot correspond to any specified $\lambda_1 > 0$ feature importance because of the inclusion of u_{xxx}, which may be inessential. Conversely, if we were to solely contemplate on the resulting PDEs associated with $\lambda_1 = 0$, we would suspect that some terms are missing from $[u_{xx}, u_{xxxx}, uu_x]$, as the big PDE model comprising $[uu_{xxx}, uu_{xxxxx}, \ldots, u_{xxx}u_{xxxx}]$ whose coefficient magnitudes were all comparable in size, e.g. of order $> 10^{-1}$,

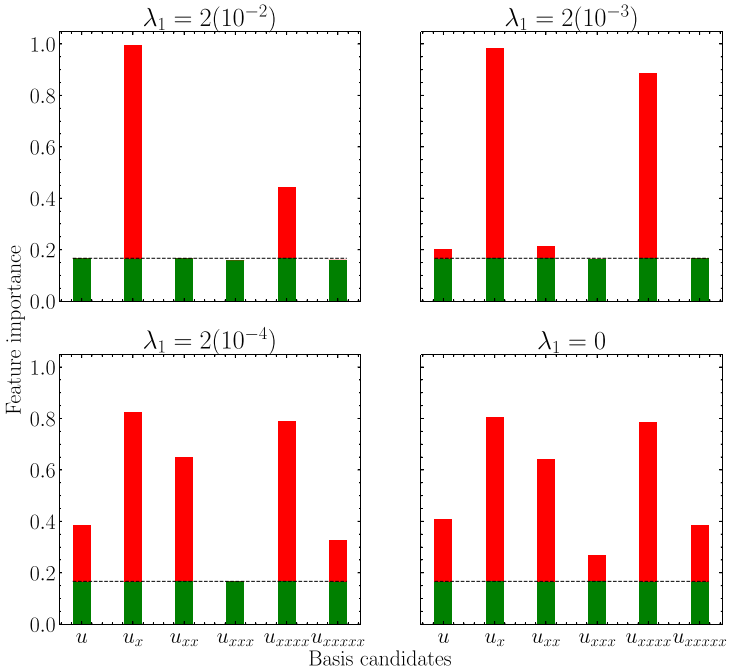


Figure 9. KS: learned feature importance with varied λ_1.

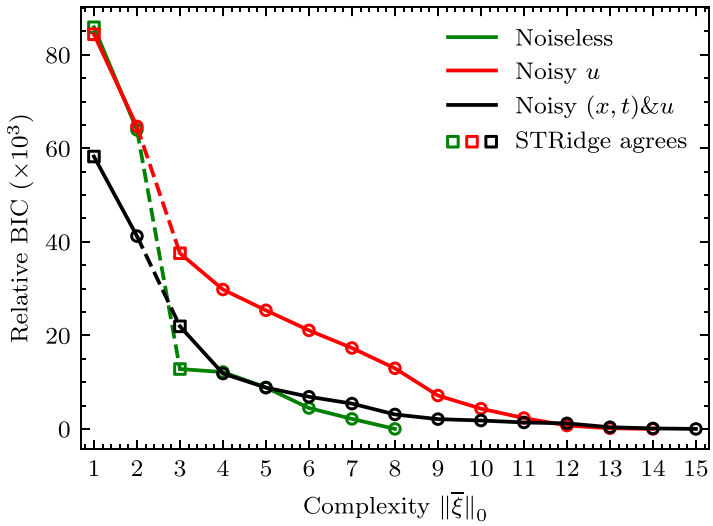
Table 3. KS regularization hyperparameter selection: STRidge's μ is set to $(2(10^2), 5(10^3), 5(10^3))$, and d_{tol} equals $(1, 1, 50)$ for the three noise conditions. For the noisy (x, t)&u case, each polynomial candidate is normalized by its L_1-norm to get the better three-term PDE in terms of the BIC score. Blue indicates the agreement. **Bold** means the lowest BIC score, compared to the scores acquired by the same λ_1.

$\lambda_1/\lambda_{\text{STR}}$	10^{-5}	10^{-3}	10^{-1}
$2(10^{-2})$	$[u_{xx}, u_{xxxx}, uu_x, uu_{xxx}, uu_{xxxxx}, u_x u_{xx}, u_{xx}u_{xxx}, u_{xx}u_{xxxxx}]$	$[u_{xx}, u_{xxxx}, uu_x]$	$[uu_x]$
	(**−153 326.24**)	(−141 117.36)	(−67 989.30)
$2(10^{-3})$	$[u_{xx}, u_{xxxx}, uu_x, uu_{xxx}, uu_{xxxxx}, u_x u_{xx}, u_{xx}u_{xxx}, u_{xx}u_{xxxxx}]$	$[u_{xx}, u_{xxxx}, uu_x]$	$[uu_x]$
	(**−153 661.00**)	(−141 032.21) ★	(−67 956.02)
$2(10^{-4})$	$[u_{xx}, u_{xxxx}, uu_x, uu_{xxx}, uu_{xxxxx}, u_x u_{xx}, u_{xx}u_{xxx}, u_{xx}u_{xxxxx}]$	$[u_{xx}, u_{xxxx}, uu_x]$	$[uu_x]$
	(**−151 328.93**)	(−138 842.33) ✓	(−68 022.47)
0 (supplement)	[a] $[u_{xx}, u_{xxxx}, uu_x, uu_{xxx}, uu_{xxxxx}, u_x u_{xxxx}, u_{xx}u_{xxx}, u_{xx}u_{xxxxx}, u_{xxx}u_{xxxx}]$	$[u_{xx}, u_{xxxx}, uu_x]$	$[uu_x]$
	(**−146 610.75**)	(−135 102.20) ✓	(−67 942.84)

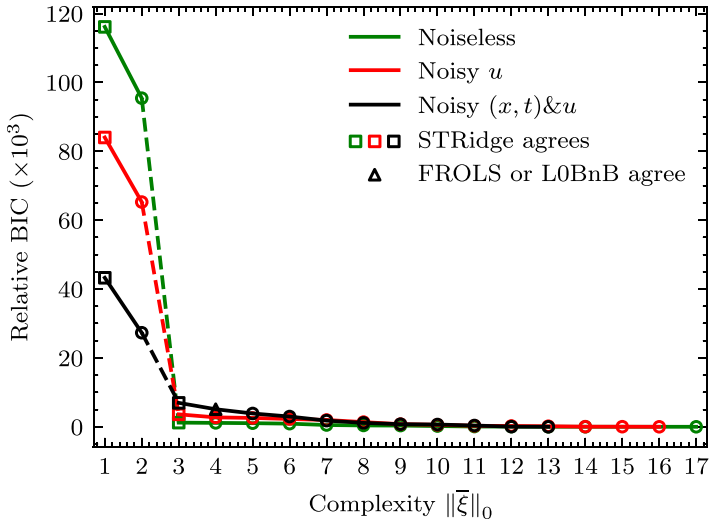
[a] To avoid minor details of the cluttered discoverie, STRidge gets recursively reiterated with small magnitude coefficient removal until $\forall j, \left|\hat{\xi}_j\right| > 10^{-1}$.

demonstrated the lowest BIC score. The dilemma signifies that the unaided BIC, whose value varies dominantly by the changing log-likelihood term, cannot righteously balance the model complexity and accuracy, partly because no parsimonious governing PDE is involved behind the criterion assumption. In fact, the well-matched BIC is achievable by the simpler model built on the three correct candidates in $\lambda_1 = 2(10^{-3})$. We mark the correct PDE expression $u_t = -0.989\,019u_{xx} - 0.962\,360u_{xxxx} - 0.966\,931uu_x$ found by $\lambda_1 = 0$ as inferior to the selected model (★) in terms of discovery precision. $\lambda_{\text{STR}} = 10^{-1}$ offers us the sparse PDEs, still, their BIC scores are much higher along with the clear BIC worthy enhancements observed when comparing against $\lambda_{\text{STR}} = 10^{-3}$, thus designated as the condition giving the underfitted models. We take the PDE with the lowest BIC $u_t = -0.989\,305u_{xx} - 0.970\,189u_{xxxx} - 0.978\,123uu_x$ as our starting PDE (★), after assessing the agreed models for each $\lambda_1 > 0$ row. For the noisy cases, the initial discovered KS %CE are listed in table 6 (see the nPIML:IPI row). On the subsequent learning ③ of figure 1, the first (repolluted, if noisy) 21 504 data points were employed to finetune dPINNs.

Generally, it is helpful to beware that including the higher-order derivatives in the basis candidates means enlarging the library size, which may have an ill effect on the discovery results. For the noisy (x, t)&u KS example, if we include up to u_{xxxxxx} then generate the 20-degree polynomials $(C = 7, k = 20)$, the PDE with three terms, produced by λ_{STR}-varied STRidge, is not Pareto-optimal:
$u_t = 0.524\,544u_{xx} - 1.120\,505uu_x - 0.626\,087uu_{xxx}$ (BIC $= -93\,841.66$) instead of the previously found $u_t = -0.845\,746u_{xx} - 0.818\,840u_{xxxx} - 0.913\,990uu_x$ (BIC $= -104\,867.55$). Nonetheless, this specific issue



(a) Polynomial library



(b) Weak formulation: an ensemble of 3 instances, each having $N_{\mathcal{M}} = 21,000$ integration domain centers.

Figure 10. KS: Pareto front between relative BIC and complexity. The preselector network is trained with $\lambda_1 = 2(10^{-3})$.

can be solved by brute-force search over all the possible PDEs with three terms to obtain the best PDE that shows the minimal BIC/MSE.

Figure 10 reveals that stepping to the discovered PDE of the actual form still makes a satisfactory (best in KS example) BIC improvement per one candidate over the optimal 2-complexity PDE built on $[uu_x, uu_{xxx}]$. After that, we immediately get stagnant progress (undoubtedly for the WF-based library), which algorithm 3 does not tolerate. The %CEs of the found coefficients by the extension with WF or CWF are listed in table 6.

Figure 11 visualizes the incomplete library's effect on the Pareto front, which implies the optimal complexity and also the associated candidates we can easily trace back. The area gap between the consecutive plots of the increasing number of bases gets smaller and saturates when all the necessary bases are presented. For example, the $C = 6$ (up to u_{xxxxx}) plot is slightly better than the $C = 5$ (up to u_{xxxx}) plot, but the optimal PDE pointed by algorithm 3 remains unchanged. If we were to finetune dPINNs with one of the incomplete libraries, we would have failed to converge dPINNs and eventually had a higher physics loss. Therefore, the initial PDE identification step is vital for a PDE discovery approach that operates sequentially like ours.

4.3.4. Initial discovered QHO PDE

As per the specific treatments for QHO mentioned in section 4.2, algorithm 1 turns applicable for the complex-valued PDEs. The preselector was trained with varied λ_1. Each basis candidate importance at the

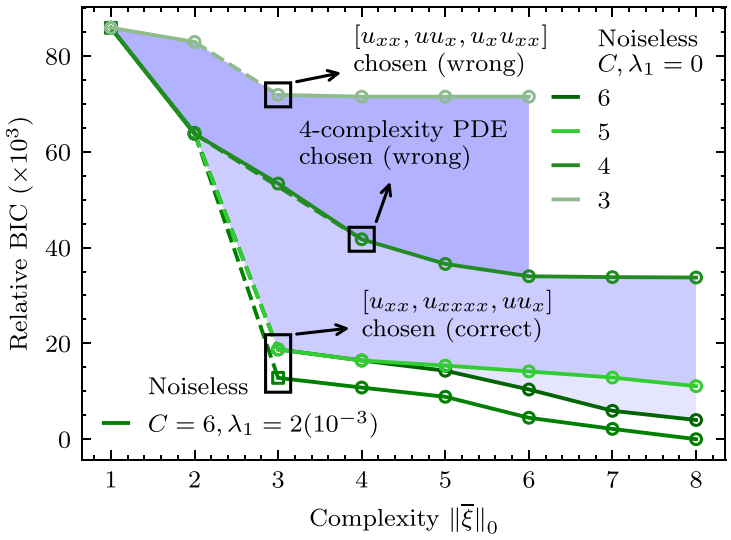


Figure 11. KS: Pareto front when the candidate library $\Phi^{\mathcal{M}}(\hat{\theta})$ is intentionally turned incomplete. The used preselectors are the same as those listed in table 3.

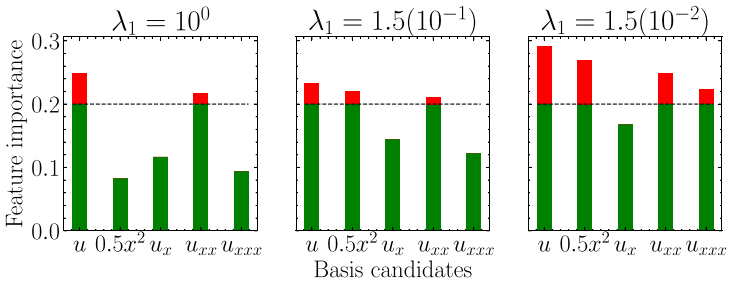


Figure 12. QHO: learned feature importance with varied λ_1.

different levels is shown in figure 12. All three correct candidates surpass the threshold when $\lambda_1 = 1.5(10^{-1})$ or $1.5(10^{-2})$ whereas $\lambda_1 = 10^0$ compels the too strong regularization.

For QHO, the metadata for STRidge was the linearly discretized points from the full-field spatio-temporal domain, i.e, $N_{\mathcal{M}} = 82,432$ and $\forall i, t_i^{\mathcal{M}} \leqslant 4$. The cross results for the regularization hyperparameter selection are listed in table 4. If the λ_{STR} intensity is loosen from 10^{-1} to 10^{-3} the considerable shoots in the BIC improvement are apparently gained. However, regularizing too mildly, e.g. $\lambda_{\text{STR}} = 10^{-5}$, does not provide any left necessary candidates, exhibiting the small BIC reductions with the more unsound terms that are unstable across varying λ_1. Ultimately, $u_t = (-0.000\,463 + 0.498\,906i)u_{xx} + (-0.002\,272 - 0.999\,284i)0.5x^2u$ (★) is accepted for the denoising and finetuning stage owing to its minimal BIC score among the agreed PDEs. In the cases where noise exists, the %CE of the initial discovered complex-valued PDEs are shown in table 6 (see the nPIML:IPI row).

To testify that the BIC decay rate is reliable for identifying (complex-valued) QHO, we show the Pareto front for the three noise conditions in figure 13(a). λ_{STR}-varying STRidge dictates the limit in complexity then, as a straightforward implementation, a brute-force (exhaustive) search solves the PDE for each complexity. Choosing the 2-complexity PDE is reasonable visually because we see no beneficial improvement in the more complicated PDEs. Algorithm 3 also numerically verify the claim.

4.3.5. Initial discovered NLS PDE

The feature importance measures are displayed in figure 14. The correct candidates are safely secured, passing the threshold and becoming effective for all the choices of $\lambda_1 = 10^0, 10^{-1}$ or 10^{-2}. Despite that, $\lambda_1 = 10^{-2}$ is relatively low such that the inclusion of u_x might have complicated the hyperparameter selection procedure.

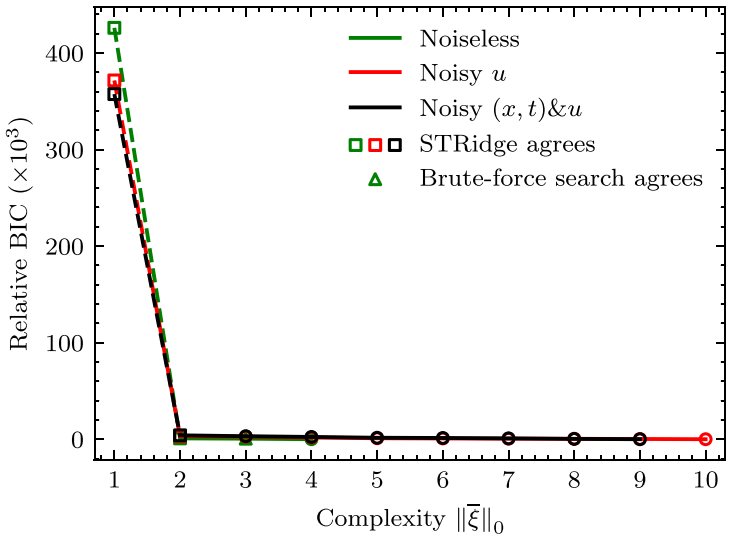
We limited the whole domain arbitrarily at $t < 1.25$ for bounding the interested region upon which the metadata was linearly discretized, i.e. in total $N_{\mathcal{M}} = 40\,960$. Still, we positively ensured that the essential dynamics were covered. The found PDEs are assimilated in table 5, indexing diverse set of $(\lambda_1, \lambda_{\text{STR}})$ for the regularization hyperparameter selection. The admittance of u_{xx} by decreasing λ_{STR} from 10^{-2} to 10^{-5}

Table 4. QHO regularization hyperparameter selection: STRidge's λ_0 is set to $10^2\lambda_{\text{STR}}\varepsilon$, and d_{tol} equals 10 for the three noise conditions. Blue indicates the agreement. **Bold** means the lowest BIC score, compared to the scores acquired by the same λ_1.

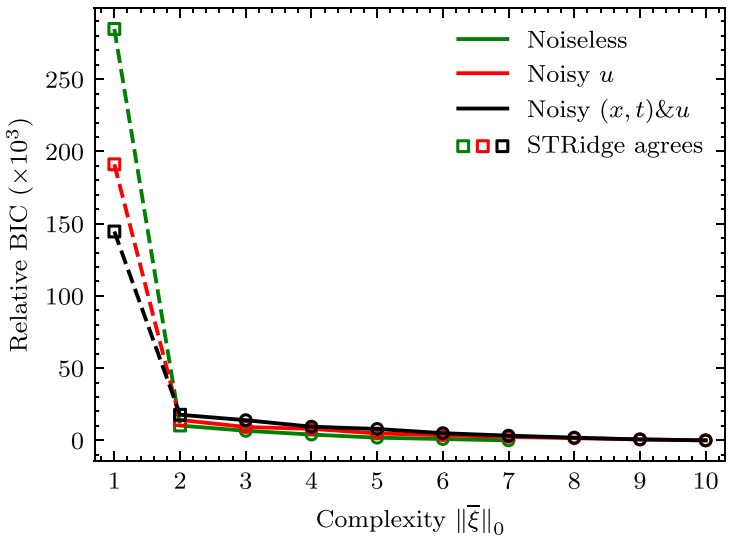
λ_1/[a] λ_{STR}	[b] 10^{-5}	10^{-3}	10^{-1}
10^0	$[u_x, u_{xx}, uu_x, uu_{xx}, 0.5x^2u]$	$[u_{xx}, 0.5x^2u]$	$[u]$
	(**−356 020.01**)	(−355 726.94)	(144 126.16)
$1.5(10^{-1})$	$[u_x, u_{xx}, uu_x, 0.5x^2u]$	$[u_{xx}, 0.5x^2u]$	$[u]$
	(**−325 329.72**)	(−325 110.45) ✓	(144 190.29)
$1.5(10^{-2})$	$[u_x, u_{xx}, uu_x, 0.5x^2u]$	$[u_{xx}, 0.5x^2u]$	$[u]$
	(**−325 526.78**)	(−325 307.53) ★	(144 195.08)

[a] We enumerate λ_{STR} from $(10^{-3}, 10^{-2}, 10^{-1})$ for the noisy (x,t)&u case.

[b] STRidge is refitted once to show only the term that $\left|\hat{\xi}_j\right| > 1.4(10^{-2})$.



(a) QHO: The preselector network is trained with $\lambda_1 = 1.5(10^{-2})$.



(b) NLS: The preselector network is trained with $\lambda_1 = 10^{-1}$.

Figure 13. Pareto front between relative BIC and complexity for identifying complex-valued PDEs.

apparently upgrades the BIC scores. Further dropping λ_{STR} down to 10^{-7} can push the BIC scores down slightly with the increased terms that eventually end up disagreeing with the preselectors. Like QHO example in section 4.3.4, the agreed sparse PDE that exhibits the minimal BIC gets accepted to be denoised and finetuned. For NLS, the initial discovered PDE reads $u_t = (-0.000\,863 + 0.499\,928i)u_{xx} + (-0.000\,973 + 0.999\,259i)u\|u\|_2^2$ (★). In the noisy experiments, the %CE of the initial discovered complex-valued PDEs are provided in table 6 (see the nPIML:IPI row).

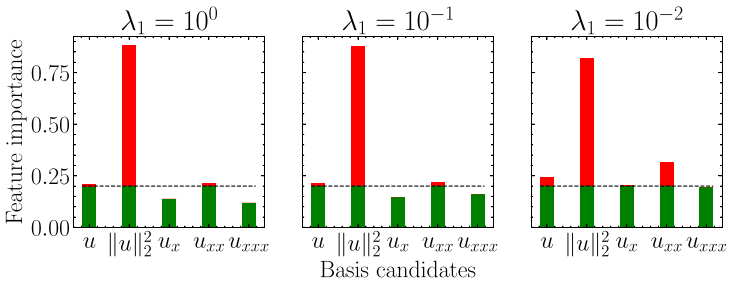


Figure 14. NLS: learned feature importance with varied λ_1.

Table 5. NLS regularization hyperparameter selection: STRidge's λ_0 is set to $10^5\lambda_{\text{STR}}\varepsilon$, and d_{tol} equals 100 for the three noise conditions. Blue indicates the agreement. **Bold** means the lowest BIC score, compared to the scores acquired by the same λ_1.

$\lambda_1/\lambda_{\text{STR}}$	[a]10^{-7}	10^{-5}	10^{-2}
10^0	$[u, \|u\|_2^2, u_x, u_{xx}, u^2, u\|u\|_2^2, uu_x, \|u\|_2^2 u_x]$	$[u_{xx}, u\|u\|_2^2]$	$[u\|u\|_2^2]$
	(**−108 572.98**)	(−107 799.25) ✓	(166 786.21)
10^{-1}	$[u_x, u_{xx}, u^2, u\|u\|_2^2, uu_x, uu_{xx}, u_x^2]$	$[u_{xx}, u\|u\|_2^2]$	$[u\|u\|_2^2]$
	(**−108 582.11**)	(−107 847.00) ★	(166 790.61)
10^{-2}	[b]$[u_x, u_{xx}, u^2, u\|u\|_2^2, uu_x, uu_{xx}, u_x^2]$	$[u_{xx}, u\|u\|_2^2]$	$[u\|u\|_2^2]$
	(**−108 508.42**)	(−107 766.91) ✓	(166 790.63)

[a] STRidge is refitted once to show only the term that $\left|\hat{\xi}_j\right| > 1.4(10^{-3})$.

[b] $(0.000\,294 - 0.000\,829i)u_{xxx}$ that partly causes the disagreement is withdrawn from the list since $|0.000\,294 - 0.000\,829i| \leqslant 1.4(10^{-3})$.

The Pareto fronts shown in figures 13(a) and (b) (for QHO and NLS) are very close to each other; hence the equal-complexity PDEs chosen. Howbeit, it is important to comment that the faultless identification of the PDE forms is made possible by the overcomplete assumption, which includes the basis candidate $0.5x^2$ in QHO and $\|u\|_2^2$ in NLS example. Otherwise, impacted by the incompleteness, we would have obtained a suboptimal PDE as demonstrated in figures 7 and 11.

4.4. Finetuning PDE coefficients by dPINNs

Based on the results in table 6, nPIML establishes superior results over nPIML without the denoising DFT and projection networks for the noisy cases, especially when both (x, t) and u are contaminated. For the clean dataset, the denoising mechanism seems to not over perturb backwardly through converging $\beta_{(x,t)}, \beta_u \to 0$, maintaining the effectiveness of the dPINNs' learning by algorithm 2, on a par to the nPIML without the denoising that exactly matches the noiseless hypothesis. Indeed, nPIML can outperform nPIML without the denoisers since the shifting to the more propitious finite set, e.g. $\{(x_i^*, t_i^*, u_i^*)_{i=1}^{N_f}\}$, is still technically probable. In Burgers' example, nPIML surpasses vanilla PINN for all experimental cases regardless of the denoising modules, implying the superiority and benefits of the precomputed initialization followed by finetuning $\hat{\theta}$ and $\hat{\xi}$. Moreover, if the genuine PDE is known beforehand, training PINN from scratch eventually leads to the good close-formed discovery accuracy on a par with CWF, better than PDE-FIND (STRidge), DLrSR and WF. The accuracy enhancement points out the usefulness of AD and physics-informed learning. Still, in KS example, IPI+CWF sets an impressive baseline error that is better than the dPINNs, which finetunes the 4th-order derivative directly and thus risks suffering from local optima of the coefficients. CWF offers the precise approximation of found coefficients, better than WF, for the mesh data with fine resolution but struggles to uncover the genuine governing PDE in the scarce subsampled mesh data. This observation naturally allows for incorporating our dPINNs' learning with the (convolutional) WF to get the best from both methods in future investigations.

4.5. Robustness against scarce data

Table 7 reveals the tolerance against the decreasing number of training samples in Burgers' example. The precise discovered PDEs are obtainable by finetuning the coefficients even though only the 500 training data points are available. However, it is challenging to recover Burgers' PDE if the noise is added or dPINNs are trained with just the 100 training samples, implied by the faulty discoveries by algorithm 1. Fortunately, the results show that the denoising affine transformation by the projection networks is feasible even under the noisy and moderately limited number of labeled samples, e.g. 1000. It is worth pointing out that data bias

Table 6. Summary of the robust discovery results by nPIML: the noise is 1% of standard deviation. Generally, the adopted λ_1s for the noisy experiments are identical to the noiseless condition unless noted otherwise. Underline and **bold** indicate the best error among the mesh-based and mesh-free methods.

Dataset	Method	# Train samples (N_f)	Noiseless	u + Noise$_u$	u + Noise$_u$ & (x,t) + Noise$_{(x,t)}$
Burgers	PDE-FIND (STRidge) [1]	256 × 100[a]	19.2070 ± 19.0686	Failed ($-0.0698uu_x$)	Not applicable[b]
	DLrSR [18]	256 × 100	19.2070 ± 19.0686	Failed ($-0.0698uu_x$)	Not applicable
	WF[c] [5]	256 × 100	18.7103 ± 17.7589	18.5517 ± 17.6983	Not applicable
	CWF [6]	256 × 100	0.3135 ± 0.2825	0.3316 ± 0.1370	Not applicable
	CWF (subsampled)	128 × 50[d]	0.5283 ± 0.4245	0.1476 ± 0.0220	Not applicable
	PINN[e] [9]	3000	0.3256 ± 0.1921	0.9212 ± 0.8589	4.0893 ± 2.9622
	nPIML:IPI[f]	3000	2.5730 ± 1.1904	7.0093 ± 2.6069	55.2051 ± 15.8919
	nPIML:IPI+WF [5]	3000	18.5244 ± 17.6550	19.2048 ± 18.1213	24.9787 ± 24.9340
	nPIML:IPI+CWF	3000	0.7741 ± 0.6189	0.7253 ± 0.7113	7.8921 ± 5.0705
	nPIML w/o denoise[g]	3000	0.1264 ± 0.0605	0.4271 ± 0.2451	2.9920 ± 2.2222
	nPIML	3000	**0.0557 ± 0.0170**	**0.3360 ± 0.1251**	**0.8546 ± 0.4806**
KdV	PDE-FIND (STRidge)	128 × 501	0.5194 ± 0.1733	Failed ($-5.4128uu_x$)	Not applicable
	DLrSR	128 × 501	0.5194 ± 0.1733	Failed ($-5.3521uu_x$)	Not applicable
	WF	128 × 501	1.5526 ± 0.9140	1.5529 ± 0.8999	Not applicable
	CWF	128 × 501	<(0.0001 ± 0.0001)	0.0203 ± 0.0027	Not applicable
	CWF (subsampled)	26 × 101	Failed ($-5.4090uu_x$)	Failed ($-5.4412uu_x$)	Not applicable
	nPIML:IPI	2000	0.8710 ± 0.2224	2.9887 ± 1.1612[h]	3.7460 ± 1.4158[h]
	nPIML:IPI+WF	2000	0.9660 ± 1.5963	1.8076 ± 2.0034	1.4701 ± 1.7390
	nPIML:IPI+CWF	2000	0.2347 ± 0.0693	**0.1177 ± 0.1143**	0.9487 ± 0.4943
	nPIML w/o denoise	2000	0.6413 ± 0.3904	1.2547 ± 0.8369	2.9378 ± 1.6140
	nPIML	2000	**0.0890 ± 0.0568**	0.2845 ± 0.2463	**0.4344 ± 0.2696**
KS	PDE-FIND (STRidge)	1024 × 251	0.7557 ± 0.5967	52.2843 ± 1.4005	Not applicable
	DLrSR	1024 × 251	0.7571 ± 0.5966	Failed[i]	Not applicable
	WF	1024 × 251	0.1521 ± 0.0598	0.1487 ± 0.0658	Not applicable
	CWF	1024 × 251	0.0004 ± 0.0004	0.0128 ± 0.0038	Not applicable
	CWF (subsampled)	1024 × 21	Failed ($-1.2218uu_x$)	Failed ($-0.9u_{xx} - uu_x$)	Not applicable
	nPIML:IPI	⩽30 000	2.0794 ± 0.7842	10.7558 ± 3.3449	14.0475 ± 4.0048
	nPIML:IPI+WF	⩽30 000	0.7265 ± 0.5199	**0.3557 ± 0.3068**	2.1058 ± 1.7677
	nPIML:IPI+CWF	⩽30 000	**0.1913 ± 0.1347**	0.5944 ± 0.5621	**1.2546 ± 1.488**
	nPIML w/o denoise	⩽30 000	1.7417 ± 1.1171	8.8925 ± 5.2704	9.2365 ± 6.5974
	nPIML	⩽30 000	0.4775 ± 0.2751	2.9320 ± 1.4401	3.6493 ± 3.968
QHO	PDE-FIND (STRidge)	512 × 161	0.2458 ± 0.0101	9.3850 ± 6.7242	Not applicable
	DLrSR	512 × 161	0.2850 ± 0.0090	9.3711 ± 6.7143	Not applicable
	nPIML:IPI	30 000	0.2379 ± 0.0003	0.3163 ± 0.0705	0.4197 ± 0.0121
	nPIML w/o denoise	30 000	0.0377 ± 0.0211	0.2380 ± 0.1463	0.3278 ± 0.1694
	nPIML	30 000	**0.0278 ± 0.0193**	**0.1235 ± 0.0580**	**0.2669 ± 0.1639**
NLS	PDE-FIND (STRidge)	256 × 201	0.3469 ± 0.2888	2.8485 ± 2.6764	Not applicable
	DLrSR	256 × 201	0.3294 ± 0.2801	2.8542 ± 2.6778	Not applicable

(Continued.)

Table 6. (Continued.)

Dataset	Method	# Train samples (N_f)	Noiseless	u + Noise$_u$	u + Noise$_u$ & (x,t) + Noise$_{(x,t)}$
	nPIML:IPI	2500	0.1478 ± 0.0255	0.5686 ± 0.2517	2.3726 ± 1.5939
	nPIML w/o denoise	2500	0.0491 ± 0.0060	0.0953 ± 0.0114	0.2205 ± 0.0877
	nPIML	2500	$\mathbf{0.0421 \pm 0.0172}$	$\mathbf{0.0571 \pm 0.01327}$	$\mathbf{0.1652 \pm 0.0532}$

[a] All the discretized points are shown in the mesh representation: # in $x \times t$.
[b] Because a mesh is required for taking polynomial derivatives used in PDE-FIND.
[c] An instance of WF, having 10 000 integration domain centers, is used with STRidge whose $(\lambda_0, \lambda_{\text{STR}}, d_{\text{tol}}) = (10^{-5}, 10^{-2}, 5)$.
[d] We subsample every 2nd point (5th point for KdV) in both x and t. For KS, we take the first 21 504 (1024×21) discretized points as described in section 4.3.3.
[e] $\hat{\xi}$ is initialized at $[\exp(-7.0), 1.0]^\intercal$ before training PINN.
[f] The results until ② of figure 1, initial PDE identification, with polynomial library.
[g] The results from ③ of figure 1, dPINNs, but without the denoising DFT module and projection networks.
[h] λ_1 is assigned to $2(10^{-5})$ instead of $2(10^{-6})$.
[i] DLrSR with the original and unvarying λ_0 discovers the following mismatched PDE:
$u_t = -0.60uu_x - 0.39u_{xx} - 0.10uu_{xxx} - 0.49u_{xxxx}$.

Table 7. Discovered Burgers' PDE on the scarce data. **Bold** indicates the best error.

# Train samples	Noise	nPIML:IPI %CE	Finetuned PDE %CE	
			w/o denoise	w/ denoise
3000[a]	No[b]	2.5730 ± 1.1904	0.1264 ± 0.0605	$\mathbf{0.0557 \pm 0.0170}$
	Yes[c]	55.2051 ± 15.8919	2.9920 ± 2.2222	$\mathbf{0.8546 \pm 0.4806}$
1000	No	3.8530 ± 1.6829	1.0953 ± 1.0526	$\mathbf{0.8105 \pm 0.7565}$
	Yes	26.5837 ± 2.6611	8.3633 ± 8.2076	$\mathbf{1.6114 \pm 1.1907}$
500	No	6.2883 ± 2.6029	1.9302 ± 1.7908	$\mathbf{1.4888 \pm 1.2651}$
	Yes	Failed[d]	Not applicable	Not applicable
100	No	Failed[e]	Not applicable	Not applicable
	Yes	Failed[f]	Not applicable	Not applicable

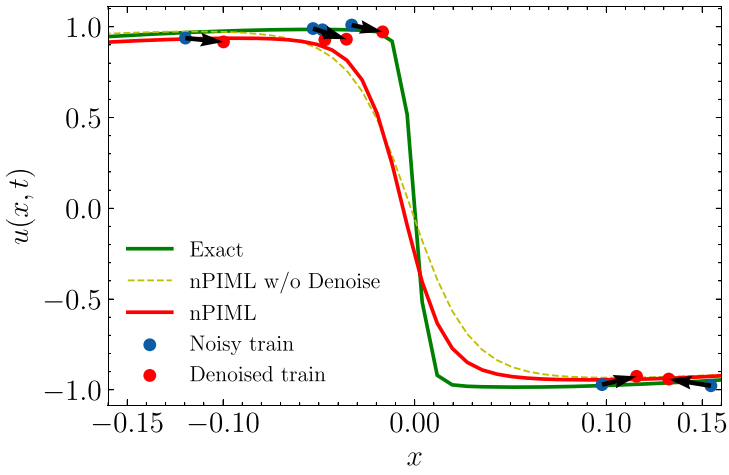
[a] Taken from table 6.
[b] Noiseless.
[c] u + Noise$_u$ & (x,t) + Noise$_{(x,t)}$.
[d] $u_t = -0.703\,435uu_x - 0.000\,041u_xu_{xx}$.
[e] $u_t = -0.509\,967uu_x$.
[f] $u_t = -0.621\,560uu_x$.

towards diverse training sets leads to diversity in initial discovery results when learning from a few samples. In addition, the involving parameter and model initializations affect PINN approximated outputs as discussed in [17] and consequently the PDEs derived from those outputs.

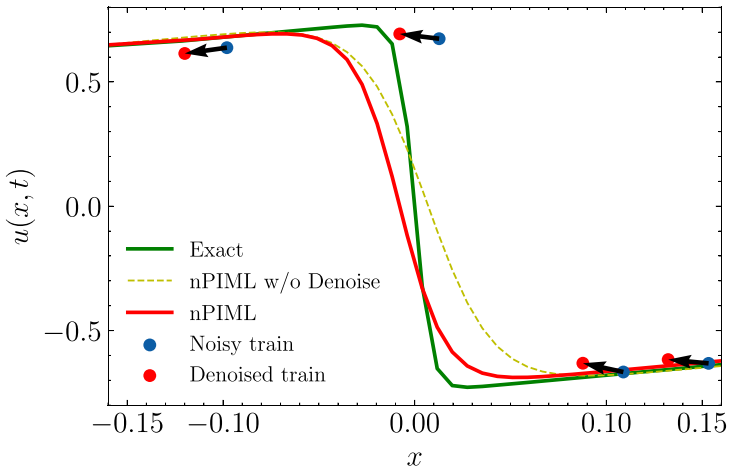
4.6. Denoising mechanism against high noise

4.6.1. Denoising visualization

We sought to apprehend how the projection networks respond to high noise visually by letting dPINNs expose the strongly contaminated dataset, where u and (x,t) are polluted with noise$(u,5)$ and noise$((x,t),5)$. Specifically, we finetuned the dPINNs pretrained by $\hat{\theta}$, taken from the 1%Noise+(x,t)&u case of Burgers' PDE. The initialized PDE was resolved by LS based on the intentionally uplifted 5% noisy (x,t), expressing the form as follows: $u_t = 0.000\,606u_{xx} - 0.403\,049uu_x$. For such high noise, we find it is useful that $\mathcal{P}_{\Omega_{(x,t)}}(x,t)$ and $\mathcal{P}_{\Omega_u}(u)$ should not be only activated by the final Tanh but also unbiased standardized and then scaled down to be 0.01 times the values to denoise gradually from small to larger noise magnitude since denoising the considerable amount at the beginning of the dPINNs' learning can ultimately cause the divergence. α and $(\beta'_{(x,t)}, \beta'_u)$ are initialized at 0.1 and $(10^{-3}, 10^{-3})$. We display how the projection networks denoise closely around $t = 0.46, 0.97$ in figure 15. By the proximate examination near the dynamically changing region, where there are only a few supervised samples, the naive PDE estimation: $u_t = 0.012\,378u_{xx} - 0.948\,156uu_x$ neglecting the noise effect is observed when the denoising components are ablated. In comparison, the projection networks can shift the polluted samples towards the direction that drives the approximated solution by dPINNs to better captures the exact characteristics of Burgers' PDE when the denoising components are utilized. For example, the noisy samples get redirected (mostly) to the



(a) Snapshot around $t = 0.46$



(b) Snapshot around $t = 0.97$

Figure 15. Close visualization of how the preselector networks react to the high noise at $x \in [-0.16, 0.16]$, around the abrupt transition caused by the shock waves.

Table 8. Numerical results of finetuning dPINNs on highly noisy KS data. **Bold** indicates the best error.

	Finetuned PDE %CE	
Noise level	w/o denoise	w/ denoise
1%[a]	9.2365 ± 6.5974	$\mathbf{3.6493 \pm 3.9688}$
3%	27.0814 ± 20.0158	$\mathbf{11.5509 \pm 9.7542}$
5%	45.8996 ± 31.3289	$\mathbf{20.5851 \pm 24.1741}$
10%	56.8900 ± 40.8643	$\mathbf{52.3366 \pm 38.7182}$

[a] Taken from table 6.

right in figure 15(a) and left in figure 15(b). With the denoising process, the optimized PDE carries the better form of $u_t = 0.008\,550 u_{xx} - 0.972\,390 u u_x$.

4.6.2. Finetuning against high noise

Since restoring a decent approximation of the hidden KS PDE from highly noisy data can be sensitive and challenging. We, therefore, set up more experiments, similar to section 4.6.1, finetuning the dPINNs initialized with $\hat{\theta}$ taken from the 1%Noise+(x,t)&u case, but single $\hat{\xi}$ uniformly generated such that $\forall i, \hat{\xi}_i \sim (-10^{-6})\boldsymbol{\mathcal{U}}(0,1)$. The intensity of the noise that contaminates (x,t)&u is explicitly increased to 3%, 5% and 10%. α and $(\beta'_{(x,t)}, \beta'_u)$ are initialized at 0.1 and $(10^{-2}, 10^{-2})$. $\beta'_{(x,t)}$ and β'_u are clamped within $[-1.0, 1.0]$ during the finetuning process. The quantitative results in table 8 emphasize the superiority of asserting the denoising mechanism to minimize the discovery error numerically under the much-corrupted datasets.

5. Conclusion

We have presented the interpretable and nPIML framework for distilling the nonlinear PDE governing a physical system in an analytical expression. The proposed method mainly tackles the problems with the suboptimal derivatives, sensitivity of regularization hyperparameters, and polluted datasets. The weakly physics-informed solver network is the primary building block for derivative computation. Giving rise to the automatic PDE selection algorithm, multi-perspective assessment of the diverse sets of regularization hyperparameters is feasible through the physics-learning preselector network and the sparse regression. Finally, dPINNs are introduced for finetuning the objective PDE coefficients on the affine-transformed noise-reduced dataset given by the projection networks. The numerical results show that the proposed method is accurate and robust to the scarcity of labeled samples and noise on five classic canonical PDEs.

Nonetheless, the proposed framework exhibits some limitations. For instance, there is no explicit denoising mechanism at the early derivative preparation and sparse regression stages; thus, particular noise of an unknown distribution may fake those initial processes and let the entire framework fail. The predicament that underlying physics remains mysterious initially causes the projection networks to be inoperable, as the affine transformation can yield the unwanted $\tilde{u} \approx \vec{0}$, and solely assigning an appropriate threshold for denoising DFT is not either trivial or readily beneficial. Towards future improvements, researchers may conduct extensive studies on grounded topics such as the effect of parameter initialization on the discovery stability or a border class of inferable PDEs that is not restricted by the linear assumption.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://github.com/nPIML-team/nPIML.

Appendix A. Choice of λ_2

Our experiments in section 4.3 have a fixed value of $\lambda_2 = 0.1$. The decision is primarily made to deal with the sensitivity with respect to λ_1 and λ_2, which raises due to how equation (6) is formulated. Since λ_1 controls both the sparsity of the feature importance and regularization on high-order derivatives, it is intuitive to vary λ_1 and keep λ_2 invariant with a value that lets the λ_1-varying preselectors deliver different distributions of learned feature importance.

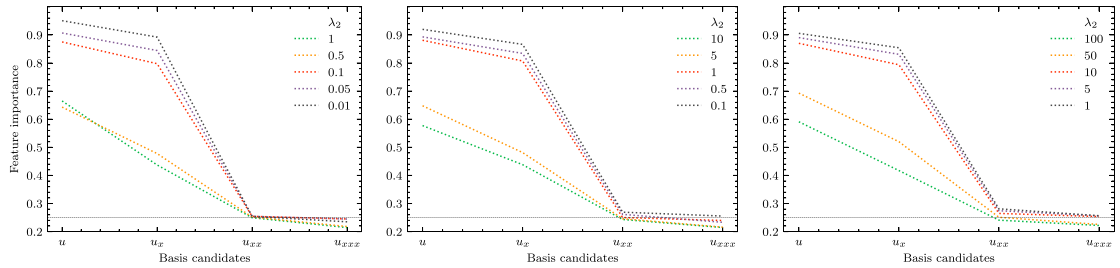
In fact, one can adopt the alternative opposite option. Here, we run the couple of solver and preselector networks trained with $\lambda_1 = 0$ (hence, no regularization has been enforced yet) through algorithm 1 again, but this time we vary λ_2 instead and fix $\lambda_1 = 0.99, 0.1, 0.01$ at a time. The results are plotted in figure 16, which reveals the pattern of the learned feature importance analogous to figure 2. For example, $\lambda_1 = 0.1, \lambda_2 = 5, 0.5, 0.1$ or $\lambda_1 = 0.01, \lambda_2 = 100, 50, 10$ represent the pattern of enforcing too mild, just right and too strong regularization respectively.

Appendix B. Automatic PDE selection

Given a decreasing sequence of raw (not relative) BIC scores $\mathcal{B}$ (containing $\overline{C}$ elements in total) and the corresponding increasing sequence of complexity Ξ, let us define the BIC decay rate from an index j to k as follows:

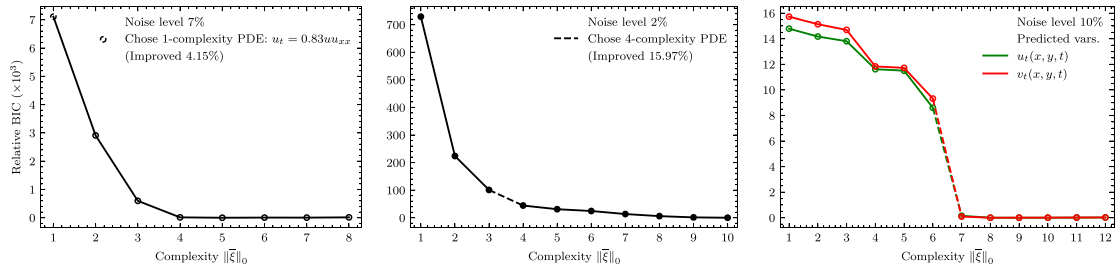
$$\mathcal{I}_{jk}^{\mathrm{BIC}}(\mathcal{B}, \Xi) = \frac{\Delta_{jk}(\mathcal{B})}{\Delta_{jk}(\Xi)} = \frac{\mathrm{BIC}_k - \mathrm{BIC}_j}{\left\|\bar{\xi}_k\right\|_0 - \left\|\bar{\xi}_j\right\|_0} < 0; k > j \geqslant 1. \tag{25}$$

The ACS in algorithm 3 is based on minimizing the BIC decay rate (line 5). Suppose we consider stepping from an index l to $\bar{k}$; we compute the magnitude of the decreased BIC per one increasing candidate and see whether it is, at least, greater than $\mathcal{I}^P$ (slightly increasing by ϵ) of the maximum between BIC magnitude at l and the latest improvement $\mathcal{I}^L$. If yes, we step to $\bar{k}$; otherwise we stop at l. In the detailed implementation, $\bar{k}$ is the index to which the transition that minimizes the current BIC decay point, and l is the latest index where we stay. At the first execution when l does not exist, we set $l = \bar{j} = \bar{k} - 1$. It is worth noting that we have not confirmed any theoretical guarantee that ACS will always determine the actual function of every physical system. ACS is simply one feasible algorithm inspired by the observation in our experiments that the minimization of the BIC decay rate gives the lower bound of the true complexity. Thus, future work on the theoretical (e.g. a design of the selection algorithm) or numerical results are encouraged.



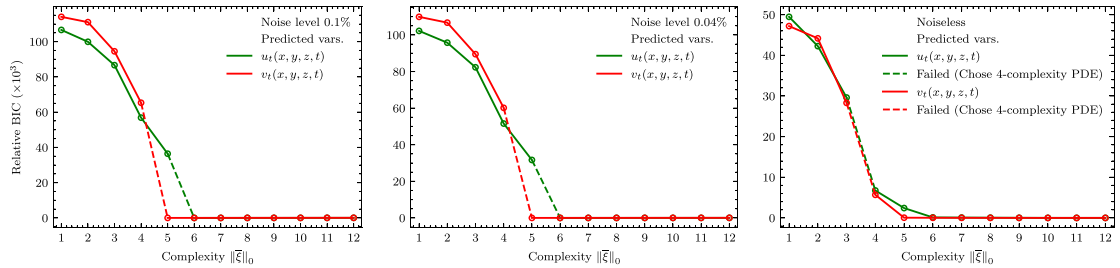
(a) $\lambda_1 = 0.99$ (b) $\lambda_1 = 0.1$ (c) $\lambda_1 = 0.01$

Figure 16. Burgers: varying λ_2 with fixed λ_1.



(a) Nonlinear diffusion (b) Navier-Stokes (c) 2D reaction-diffusion system

Figure 17. Pareto fronts plotted with the transition to the optimal complexity (dashed line style) detected by algorithm 3 for PDEs of different actual complexity.



(a) High resolution (b) Medium resolution (c) Low resolution

Figure 18. 3D reaction-diffusion system (Gray–Scott equation): Pareto fronts are affected by the different spatial resolutions. From left to right, the spatial points in each direction are 128, 64 and 32.

Table 9. Numerical results of the coefficients discovered by algorithm 3.

Dataset	Noise level	Selected PDE %CE
Nonlinear diffusion PDE (one true candidate)	7%	u_t: 17.1717
Navier–Stokes PDE	2%	ω_t: 15.5900 ± 15.5292
2D reaction-diffusion PDE	10%	u_t: 3.1177 ± 1.0276, v_t: 3.3251 ± 1.2490
3D reaction-diffusion PDE (high resolution)	0.10%	u_t: 0.0490 ± 0.0720, v_t: 0.0071 ± 0.0072
3D reaction-diffusion PDE (medium resolution)	0.04%	u_t: 0.0859 ± 0.0974, v_t: 0.0584 ± 0.0338
3D reaction-diffusion PDE (low resolution)	0.00%	Failed (see figure 18(c))

We conduct independent experiments, extending the prior results mostly exemplified in [56, 57], on the following four different PDEs and present the outcomes of ACS in detecting the actual complexity as shown in figures 17 and 18. Table 9 lists the corresponding %CE for each experiment.

Similar to section 4.3.1, we leverage the best-subset regressors (FROLS and L0BnB) with our backward elimination strategy to achieve the BIC associated with the best subset at each complexity. Table 10 lists the ridge hyperparameter of L0BnB for each example. Note that, for appendix B.2, since the library size is large (300 000 rows), we use FROLS and BESS (best-subset selection with a given complexity) algorithm [58, 59] instead of L0BnB because it works faster with the big data. Indeed, more diverse solvers are welcome to be

Algorithm 3. Automatic complexity selection (ACS) based on minimization of BIC decay rate.

Goal: Select the optimal complexity in Ξ_i (sliced)
Require: $\mathcal{B}_i = (\mathrm{BIC}_j)_{j=i}^{\overline{C}}$, $\Xi_i = \left(\left\|\overline{\xi}_j\right\|_0\right)_{j=i}^{\overline{C}}$, $\mathcal{I}^P > 0$, ϵ, $\mathcal{I}^L$

procedure ACS($\mathcal{B}_i, \Xi_i, \mathcal{I}^P, \epsilon, \mathcal{I}^L$)[a]
  if $i = \overline{C}$ **then**
    return $\left\|\overline{\xi}_i\right\|_0$
  end if
  $\overline{j}, \overline{k} \leftarrow \arg\min_{j,k} \mathcal{I}^{\mathrm{BIC}}_{jk}(\mathcal{B}_i, \Xi_i)$; $k = j+1, i \leqslant j < \overline{C}$
  $l \leftarrow \overline{j}$
  if $\mathcal{I}^L > 0$ **then**
    $l \leftarrow i$
  end if
  $\phi \leftarrow -\mathcal{I}^{\mathrm{BIC}}_{l\overline{k}}(\mathcal{B}_i, \Xi_i) = \frac{\Delta_{\overline{l}l}(\mathcal{B}_i)}{\Delta_{\overline{l}\overline{k}}(\Xi_i)} > 0$
  if $\phi > \mathcal{I}^P \max(|\mathrm{BIC}_l|, \mathcal{I}^L)$ **then**
    return ACS($\mathcal{B}_{\overline{k}}, \Xi_{\overline{k}}, \mathcal{I}^P + \epsilon, \epsilon, \phi$)
  end if
  return $\left\|\overline{\xi}_l\right\|_0$
end procedure

[a] For every experiment, we initialize $(\mathcal{I}^P, \epsilon, \mathcal{I}^L) = (0.09, 0.01, 0)$ and call ACS ($\mathcal{B}_1 = \mathcal{B}, \Xi_1 = \Xi, \mathcal{I}^P, \epsilon, \mathcal{I}^L$) to deliver the optimal complexity. We do not find problematic sensitivity to the initialization of $\mathcal{I}^P$ and ε. We achieve the same correct selections by initializing $(\mathcal{I}^P, \epsilon, \mathcal{I}^L) = (0.095, 0.005, 0)$.

Table 10. Ridge regularization hyperparameter of L0BnB [52] used to analyze each example. A smaller value achieves a shorter run-time limit. For QHO and NLS, the best-subset solver is brute-force search. For Navier–Stokes PDE, BESS without L_2-regularization [58, 59] is used instead of L0BnB, achieving faster recovery.

Burgers	KdV	KS	Nonlinear diffusion	2D reaction diffusion	3D reaction diffusion
10^{-3}	10^{-3}	10^{-2}	10^{-3}	10^{-1}	10^{-2}

aggregated. For example, we union the models given by library-ensemble [60] STRidge and SR3 [61] respectively in appendices B.3 and B.4. We determine the maximum complexity as a constant. If it is computationally possible, we can ultimately brute-force search for the best subset of any complexity upon all the candidates once effective by any solver. Gaussian noise, according to equation (22), is added solely to every independent generated PDE solution up to the limit. For appendices B.1, B.3 and B.4, the library representation is overcomplete and constructed as an instance by the WF [5] with 10 000 integration domain centers and the derivative (approximated by finite difference method) order not greater than 2. An overcomplete polynomial-based library is leveraged for appendix B.2.

B.1. Nonlinear diffusion PDE

Dissimilar to the other examples, the PDE raises the challenge of discovering just one mixed term between function and derivative. Scipy's initial value problem (ivp) solver with spectral differentiation is applied to simulate the PDE solution. The complete simulation details can be found at PySINDy URL: https://github.com/dynamicslab/pysindy

$$u_t = uu_{xx}; \quad x \in [0,5],\ t \in [0,1]. \tag{26}$$

As shown in figure 17(a), ACS stops at the 1-complexity model because we check $\phi^{\mathrm{chk}} \leftarrow \phi / \max(|\mathrm{BIC}_l|, \mathcal{I}^L) = 0.0415 < \mathcal{I}^P = 0.09$ immediately at the first procedure execution. The noise limit is 7% before the best 1-complexity model with respect to the perturbed data changes its form to a wrong one.

B.2. Navier–Stokes PDE

The PDE represents a 2D vorticity equation, whose solution simulates fluid flowing around a circular cylinder at Reynolds number 100, as follows:

$$\begin{gathered} \omega_t = -u\omega_x - v\omega_y + 0.01\omega_{xx} + 0.01\omega_{yy}; \\ (x,y) \in [0,9] \times [0,4],\ t \in [0,30], \end{gathered} \tag{27}$$

where ω denotes the vorticity. The coordinates of the velocity field, u and v, are treated as known terms. The PDE solution, which has a cylinder of unit diameter, is obtained by the immersed boundary projection method [62, 63] with 3rd-order Runge–Kutta time stepping. We retrieve the Pareto front from the

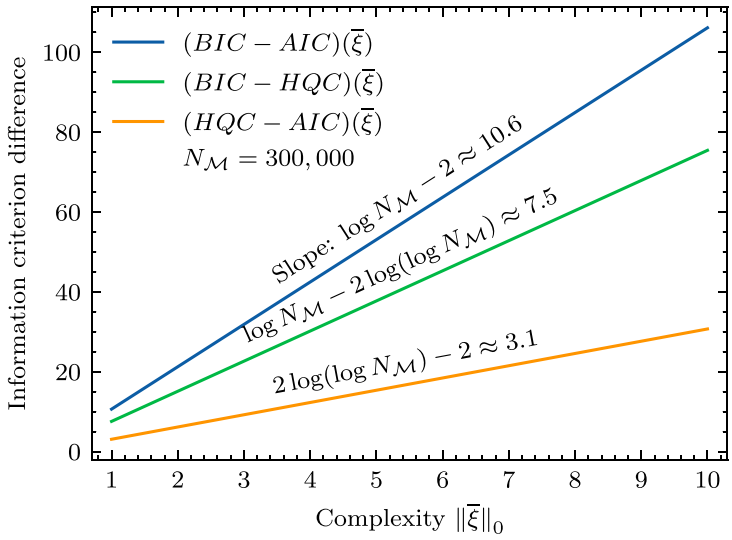


Figure 19. Navier–Stokes PDE: comparison between BIC, AIC and HQC.

polynomial library developed in [1]. Accordingly, to compute a derivative candidate, we take polynomial differentiation at each subsampled point and then aggregate the output value for the entire sub-domain, where the spatial region past the cylinder bounded by $(x, y) \in [2, 8.5] \times [0.3, 3.7]$ is kept.

ACS travels along the Pareto front drawn in figure 17(b) passing through 1→2, 2→3 and 3→4 complexity transitions with the latest valid $\phi^{\text{chk}} = 0.1597$ improvement. Remark that the subsequent transition to the 5-complexity model (4→5) gives only $\phi^{\text{chk}} = 0.0323$, less than $\mathcal{I}^P + 3\epsilon = 0.12$. Here, the noise limit is 2% higher than the 1% noise used in [1]. Thus, with the identical library, the best-subset solver is better than a single sparse regressor, like STRidge, at searching the actual governing PDE under the higher noise situation because every complexity within a broader range is taken into account.

Exhibiting regarding this Navier–Stokes example that the nearly identical Pareto fronts are obtained for the other information criteria closely formulated to BIC, e.g. AIC and HQC (Hannan–Quinn information criterion) [64], we plot their difference to BIC in raw values as shown in figure 19. Given that metadata $\mathcal{M}$ is unvaried and independent of solver network parameters $\hat{\theta}$, specifically in appendix B, we adopt the formulation of AIC and HQC following [35, 36]:

$$\begin{aligned} \text{AIC}(\bar{\xi}) &= 2\left\|\bar{\xi}\right\|_0 - 2\log\hat{L}(\bar{\xi}), \\ \text{HQC}(\bar{\xi}) &= 2\left\|\bar{\xi}\right\|_0 \log\left(\log N_{\mathcal{M}}\right) - 2\log\hat{L}(\bar{\xi}). \end{aligned} \tag{28}$$

Therefore, the constant slopes of their linear differences are annotated in figure 19. These differences are so relatively minor that algorithm 3 delivers precisely the same optimal PDE for the three information criteria. The ordering BIC>HQC>AIC remains true when $N_{\mathcal{M}}$ satisfies $2\log(\log N_{\mathcal{M}}) > 2$, or we have $N_{\mathcal{M}} \geqslant 16$ samples, which results in HQC>AIC. The condition BIC>HQC is satisfied if $\log N_{\mathcal{M}} > 2\log(\log N_{\mathcal{M}})$, which gives $\sqrt{N_{\mathcal{M}}} > \log N_{\mathcal{M}} > 0$, always true under $N_{\mathcal{M}} > 1$. Thereupon, the ordering is usual in the regime of big data. Concerning a future extension of the ACS algorithm to another criterion, we recommend setting a workable $(\mathcal{I}^P, \epsilon)$ that suits its raw value range and preferably generalizes to various (simulated) examples.

B.3. 2D reaction-diffusion PDE

The PDE governs a system simulating double spiral waves on a periodic domain and consists of seven terms, which is greater than other canonical PDEs

$$\begin{aligned} u_t &= u - u^3 + v^3 - uv^2 + u^2v + 0.1u_{xx} + 0.1u_{yy}, \\ v_t &= v - u^3 - v^3 - uv^2 - u^2v + 0.1v_{xx} + 0.1v_{yy}; \\ x, y &\in [-10, 10],\ t \in [0, 10]. \end{aligned} \tag{29}$$

We follow the dataset creation process from [1], which uses MATLAB's odeint to simulate the PDE solution with 256 points in each direction. The specification of WF ensures the completeness and follows the example given by [56, 57] at PySINDy URL.

ACS is performed twice to both u_t and v_t for studying the Pareto front in figure 17(c), where the subset at each complexity is guaranteed to be at its optimum over the set of every single unique effective candidate that

Table 11. Summary of the main notations and their description.

Notation	Description
$\mathcal{A}^{\mathcal{T}}$	(Thresholded) preselector network's feature importance
$\mathcal{D}, \mathcal{D}_s$	Labeled set and unlabeled multiset (repetitions allowed) containing all unlabeled spatio-temporal points (x, t)
$\tilde{\mathcal{D}}, \tilde{\mathcal{D}}'$	Denoised datasets with and without referenced $\tilde{u}$
$\mathcal{F}_\theta, \mathcal{F}_{\theta_s}$	$\mathcal{F}$ refers to a neural network's forward pass. θ, θ_s denote parameters of the solver and preselector networks.
$\mathcal{F}_{W^b}, \mathcal{F}_{\theta_s^r}$	θ_s consists of two sub-parameters: W^b and θ_s^r.
$\mathcal{F}_{\hat{\theta}}, \mathcal{F}_{\hat{\theta}_s}$	The hat notation indicates trained parameters resulted from algorithm 1.
$\mathcal{L}_{\text{sup}}^{\mathcal{D}}, \mathcal{L}_{\text{unsup}}^{\mathcal{D}_s}$	Supervised and unsupervised losses computed respectively on $\mathcal{D}$ and $\mathcal{D}_s$
$\mathcal{L}_{\text{mt}}^{(\mathcal{D}, \mathcal{D}_s)}$	Multi-task learning loss utilizes both $\mathcal{D}$ and $\mathcal{D}_s$.
$\mathcal{M}, \mathcal{M}_{\text{val}}$	Metadata input for STRidge and validation set split from $\mathcal{M}$
$\mathcal{N}_\xi, \hat{\mathcal{N}}_{\hat{\xi}}$	True and estimated governing function parameterized by ξ and $\hat{\xi}$ respectively
$N_f, N_f + N_r, N_{\mathcal{M}}$	Number of elements in $\mathcal{D}$, $\mathcal{D}_s$ and $\mathcal{M}$
p	Gaussian noise level
P_k	Function to create k-degree polynomial library
$\mathcal{P}_{\Omega_u}, \mathcal{P}_{\Omega_{(x,t)}}$	Projecting functions parameterized by Ω_u and $\Omega_{(x,t)}$ for denoising on both u and (x, t)
$R_\eta^{\mathcal{D}_s}$	Regularization function encouraging sparse preselector network's feature importance and lower-order derivatives
S_ψ	Low-PSD noise for $\psi \in \{x, t, u\}$
$u, u_t, u_x, u_{xx}, \ldots$	(Noisy) labeled PDE solution and its temporal and spatial derivatives
$\tilde{u}$	Denoised u
Z	Gaussian noise
$\beta_u, \beta_{(x,t)}$	Parameters controlling outputs of $\mathcal{P}_{\Omega_u}$ and $\mathcal{P}_{\Omega_{(x,t)}}$
$\Phi^{\mathcal{D}_s}, \Phi^{\mathcal{M}}$	Φ refers to library of candidates. Superscript indices dataset on which Φ is evaluated.
$\Phi_{\mathcal{E}}^{\mathcal{D}}, \Phi_{\mathcal{E}}^{\tilde{\mathcal{D}}'}$	$\mathcal{E}$ indicates that Φ includes only effective candidates.
Θ	Matrix whose columns are the true terms of governing PDE only
$\lambda_0, \lambda_{\text{STR}}$	Hyperparameters for L_0-norm penalty of STRidge algorithm
λ_1, λ_2	Hyperparameters control sparsity of preselector network's feature importance and number of effective derivative order
η	Trade-off parameter for smooth L_0-norm approximation
$\xi^{\text{STR}}, \hat{\xi}, \bar{\xi}$	ξ^{STR} is an estimated PDE coefficient (column) vector by STRidge algorithm. $\hat{\xi}$ only stores nonzero coefficients in ξ^{STR}. $\bar{\xi}$ usually means a nonspecific estimate of PDE coefficients.
ζ	Threshold hyperparameter for denoising DFT algorithm (used in algorithm 2)
δ	Absolute error function

exists along the Pareto front. When the actual PDE form is fully filled, the BIC scores drop considerably and stay steady with tiny improvement, even though several variables contributing to the predictions have been included before. As a result, one execution of ACS is enough to know the actual complexity, moving directly to the transition where the BIC decay rate is minimized.

B.4. Three-dimensional (3D) reaction-diffusion PDE

Following [65], we tackle the reaction-diffusion system, called Gray–Scott equation, in 3D spatial grid data

$$\begin{aligned} u_t &= 0.014 - 0.014u - uv^2 + 0.02u_{xx} + 0.02u_{yy} + 0.02u_{zz}, \\ v_t &= -0.067v + uv^2 + 0.01v_{xx} + 0.01v_{yy} + 0.01v_{zz}; \\ x, y, z &\in [-1.25, 1.25],\ t \in [0, 10]. \end{aligned} \tag{30}$$

The data creation process (mainly by Scipy's ivp solver in the Fourier space) and specification of the complete WF are also easily found at PySINDy URL. For this system, we are interested in whether the data resolution, defined by the number of points used in each spatial direction, affects the Pareto front, the essential input to ACS algorithm.

As noticed in figures 18(a) and (b), the inclusion of the last missing candidate corresponds to the significant decrease in BIC. Contrary to figure 18(c), the biggest drop, followed by the intolerable small decreases, happens earlier in the Pareto front, resulting in a falsely discovered PDE system even under the noiseless circumstance. The characteristics of the Pareto fronts are similar to each other if the data resolution is enough, e.g. having $128 \times 128 \times 128$ or $64 \times 64 \times 64$ discretized points. More noise tolerance is also observed in the higher resolution case. Conclusively, the sparse spatial-temporal data, indifferent from noise, poorly affect the discovery quality.

ORCID iDs

Pongpisit Thanasutives https://orcid.org/0000-0001-7991-315X
Takashi Morita https://orcid.org/0000-0002-3900-5410
Masayuki Numao https://orcid.org/0000-0002-4890-1970
Ken-ichi Fukui https://orcid.org/0000-0002-2451-1919

References

[1] Rudy S H, Brunton S L, Proctor J L and Kutz J N 2017 Data-driven discovery of partial differential equations *Sci. Adv.* **3** e1602614
[2] Schaeffer H 2017 Learning partial differential equations via data discovery and sparse optimization *Proc. R. Soc.* A **473** 20160446
[3] Tibshirani R 1996 Regression shrinkage and selection via the lasso *J. R. Stat. Soc.* B **58** 267–88
[4] Zhang S and Lin G 2018 Robust data-driven discovery of governing physical laws with error bars *Proc. R. Soc.* A **474** 20180305
[5] Reinbold A, Gurevich D R and Grigoriev R O 2020 Using noisy or incomplete data to discover models of spatiotemporal dynamics *Phys. Rev.* E **101** 010203
[6] Messenger D A and Bortz D M 2021 Weak SINDy for partial differential equations *J. Comput. Phys.* **443** 110525
[7] Goyal P and Benner P 2022 Discovery of nonlinear dynamical systems using a Runge–Kutta inspired dictionary-based sparse regression approach *Proc. R. Soc.* A **478** 20210883
[8] Baydin A G, Pearlmutter B A, Radul A A and Siskind J M 2018 Automatic differentiation in machine learning: a survey *J. Mach. Learn. Res.* **18** 1–43 (available at: http://jmlr.org/papers/v18/17-468.html)
[9] Raissi M, Perdikaris P and Karniadakis G E 2019 Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations *J. Comput. Phys.* **378** 686–707
[10] Schwarz G 1978 Estimating the dimension of a model *Ann. Stat.* **62** 461–4
[11] Akaike H 1998 Information theory and an extension of the maximum likelihood principle *Selected Papers of Hirotugu Akaike* (Berlin: Springer) pp 199–213
[12] Mangan N M, Kutz J N, Brunton S L and Proctor J L 2017 Model selection for dynamical systems via sparse regression and information criteria *Proc. R. Soc.* A **473** 20170009
[13] Lagergren J H, Nardini J T, Michael Lavigne G, Rutter E M and Flores K B 2020 Learning partial differential equations for biological transport models from noisy spatio-temporal data *Proc. R. Soc.* A **476** 20190800
[14] Horrocks J and Bauch C T 2020 Algorithmic discovery of dynamic models from infectious disease data *Sci. Rep.* **10** 1–18
[15] Karniadakis G E, Kevrekidis I G, Lu L, Perdikaris P, Wang S and Yang L 2021 Physics-informed machine learning *Nat. Rev. Phys.* **3** 422–40
[16] Thanasutives P, Numao M and Fukui K 2021 Adversarial multi-task learning enhanced physics-informed neural networks for solving partial differential equations *2021 Int. Joint Conf. on Neural Networks (IJCNN)* (IEEE) pp 1–9
[17] Wong J C, Ooi C, Gupta A and Ong Y-S 2022 Learning in sinusoidal spaces with physics-informed neural networks *IEEE Trans. Artif. Intell.* **1** 1–15
[18] Li J, Sun G, Zhao G and Li-wei H L 2020 Robust low-rank discovery of data-driven partial differential equations *Proc. AAAI Conf. on Artificial Intelligence* vol 34 pp 767–74
[19] Candès E J, Li X, Ma Y and Wright J 2011 Robust principal component analysis? *J. ACM* **58** 1–37
[20] Ranacher P, Brunauer R, Trutschnig W, Van der Spek S and Reich S 2016 Why GPS makes distances bigger than they are *Int. J. Geogr. Inf. Sci.* **30** 316–33
[21] Faux D A and Godolphin J 2019 Manual timing in physics experiments: error and uncertainty *Am. J. Phys.* **87** 110–15
[22] Brunton S L, Proctor J L and Kutz J N 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems *Proc. Natl Acad. Sci.* **113** 3932–7
[23] Berg J and Nyström K 2019 Data-driven discovery of PDEs in complex datasets *J. Comput. Phys.* **384** 239–52
[24] Both G-J, Choudhury S, Sens P and Kusters R 2021 DeepMoD: deep learning for model discovery in noisy data *J. Comput. Phys.* **428** 109985
[25] Chen Z, Liu Y and Sun H 2021 Physics-informed learning of governing equations from scarce data *Nat. Commun.* **12** 1–13
[26] Stephany R and Earls C 2022 PDE-READ: human-readable partial differential equation discovery using deep learning *Neural Netw.* **154** 360–82
[27] Guyon I, Weston J, Barnhill S and Vapnik V 2002 Gene selection for cancer classification using support vector machines *Mach. Learn.* **46** 389–422
[28] Xu H, Chang H and Zhang D 2020 DLGA-PDE: discovery of PDEs with incomplete candidate library via combination of deep learning and genetic algorithm *J. Comput. Phys.* **418** 109584
[29] Xu H and Zhang D 2021 Robust discovery of partial differential equations in complex situations *Phys. Rev. Res.* **3** 033270
[30] Basdevant C, Deville M, Haldenwang P, Lacroix J, Ouazzani J, Peyret R, Orlandi P and Patera A 1986 Spectral and finite difference solutions of the Burgers equation *Comput. Fluids* **14** 23–41
[31] Kaheman K, Brunton S L and Kutz J N 2022 Automatic differentiation to simultaneously identify nonlinear dynamics and extract noise probability distributions from data *Mach. Learn.: Sci. Technol.* **3** 015031
[32] Mohimani G H, Babaie-Zadeh M and Jutten C 2007 Fast sparse representation based on smoothed L0 norm *Int. Conf. on Independent Component Analysis and Signal Separation* (Springer) pp 389–96
[33] Kendall A, Gal Y and Cipolla R 2018 Multi-task learning using uncertainty to weigh losses for scene geometry and semantics *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 7482–91
[34] Yu T, Kumar S, Gupta A, Levine S, Hausman K and Finn C 2020 Gradient surgery for multi-task learning *Advances in Neural Information Processing Systems* vol 33
[35] Seabold S and Perktold J 2010 Statsmodels: econometric and statistical modeling with python *Proc. 9th Python in Science Conf. (Austin, TX)* vol 57 p 61
[36] Anderson D and Burnham K 2004 *Model Selection and Multi-Model Inference* vol 63 2nd edn (New York: Springer) p 10
[37] Raissi M 2018 Deep hidden physics models: deep learning of nonlinear partial differential equations *J. Mach. Learn. Res.* **19** 1–24 (available at: http://jmlr.org/papers/v19/18-046.html)

[38] Korteweg D J and De Vries G 1895 XLI. On the change of form of long waves advancing in a rectangular canal and on a new type of long stationary waves *London, Edinburgh Dublin Phil. Mag. J. Sci.* **39** 422–43
[39] Virtanen P *et al* SciPy 1.0 Contributors 2020 SciPy 1.0: fundamental algorithms for scientific computing in python *Nat. Methods* **17** 261–72
[40] Trefethen L N 2000 *Spectral Methods in Matlab* (Philadelphia, PA: SIAM)
[41] Ba J L, Kiros J R and Hinton G E 2016 Layer normalization (arXiv:1607.06450)
[42] Srivastava N, Hinton G, Krizhevsky A, Sutskever I and Salakhutdinov R 2014 Dropout: a simple way to prevent neural networks from overfitting *J. Mach. Learn. Res.* **15** 1929–58 (available at: http://jmlr.org/papers/v15/srivastava14a.html)
[43] Glorot X and Bengio Y 2010 Understanding the difficulty of training deep feedforward neural networks *Proc. 13th Int. Conf. on Artificial Intelligence and Statistics (JMLR Workshop and Conf. Proc.)* pp 249–56
[44] Yatawatta S, De Clercq L, Spreeuw H and Diblen F 2019 A stochastic LBFGS algorithm for radio interferometric calibration *2019 IEEE Data Science Workshop (DSW)* (IEEE) pp 208–12
[45] Liu D C and Nocedal J 1989 On the limited memory BFGS method for large scale optimization *Math. Program.* **45** 503–28
[46] Defazio A and Jelassi S 2021Adaptivity without compromise: a momentumized, adaptive, dual averaged gradient method for stochastic optimization (arXiv:2101.11075)
[47] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L *et al* 2019 Pytorch: an imperative style, high-performance deep learning library *Advances in Neural Information Processing Systems* vol 32
[48] Trabelsi C, Bilaniuk O, Zhang Y, Serdyuk D, Subramanian S, Santos J F, Mehri S, Rostamzadeh N, Bengio Y and Pal C J 2018 Deep complex networks *Int. Conf. on Learning Representations*
[49] Ioffe S and Szegedy C 2015 Batch normalization: accelerating deep network training by reducing internal covariate shift *Int. Conf. on Machine Learning* (PMLR) pp 448–56
[50] Chen S, Billings S A and Luo W 1989 Orthogonal least squares methods and their application to non-linear system identification *Int. J. Control* **50** 1873–96
[51] Billings S A 2013 *Nonlinear System Identification: NARMAX Methods in the Time, Frequency and Spatio-Temporal Domains* (New York: Wiley)
[52] Hazimeh H, Mazumder R and Saab A 2022 Sparse regression at scale: branch-and-bound rooted in first-order optimization *Math. Program.* **196** 347–88
[53] Pedregosa F *et al* 2011 Scikit-learn: machine learning in python *J. Mach. Learn. Res.* **12** 2825–30 (available at: http://jmlr.org/papers/v12/pedregosa11a.html)
[54] Quade M, Abel M, Nathan Kutz J and Brunton S L 2018 Sparse identification of nonlinear dynamics for rapid model recovery *Chaos* **28** 063116
[55] Stein M 1987 Large sample properties of simulations using Latin hypercube sampling *Technometrics* **29** 143–51
[56] de Silva B, Champion K, Quade M, Loiseau J-C, Kutz J and Brunton S 2020 PySINDy: a python package for the sparse identification of nonlinear dynamical systems from data *J. Open Source Softw.* **5** 2104
[57] Kaptanoglu A A *et al* 2022 PySINDy: a comprehensive python package for robust sparse system identification *J. Open Source Softw.* **7** 3994
[58] Zhu J, Wen C, Zhu J, Zhang H and Wang X 2020 A polynomial algorithm for best-subset selection problem *Proc. Natl Acad. Sci.* **117** 33117–23
[59] Zhu J, Wang X, Hu L, Huang J, Jiang K, Zhang Y, Lin S and Zhu J 2022 abess: a fast best-subset selection library in python and R *J. Mach. Learn. Res.* **23** 1–7 (available at: https://www.jmlr.org/papers/v23/21-1060.html)
[60] Fasel U, Kutz J N, Brunton B W and Brunton S L 2022 Ensemble-SINDy: robust sparse model discovery in the low-data, high-noise limit, with active learning and control *Proc. R. Soc.* A **478** 20210904
[61] Zheng P, Askham T, Brunton S L, Kutz J N and Aravkin A Y 2018 A unified framework for sparse relaxed regularized regression: SR3 *IEEE Access* **7** 1404–23
[62] Taira K and Colonius T 2007 The immersed boundary method: a projection approach *J. Comput. Phys.* **225** 2118–37
[63] Colonius T and Taira K 2008 A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions *Comput. Methods Appl. Mech. Eng.* **197** 2131–46
[64] Hannan E J and Quinn B G 1979 The determination of the order of an autoregression *J. R. Stat. Soc.* B **41** 190–5
[65] Maddu S, Cheeseman B L, Sbalzarini I F and Müller C L 2022 Stability selection enables robust learning of differential equations from limited noisy data *Proc. R. Soc.* A **478** 20210916