

Error-in-variables modelling for operator learning

Ravi G. Patel^{1*}, Indu Manickam¹, Myoungkyu Lee², and Mamikon Gulian¹

¹*Sandia National Laboratories*

²*University of Alabama*

**rgpatel@sandia.gov*

Abstract

Deep operator learning has emerged as a promising tool for reduced-order modelling and PDE model discovery. Leveraging the expressive power of deep neural networks, especially in high dimensions, such methods learn the mapping between functional state variables. While proposed methods have assumed noise only in the dependent variables, experimental and numerical data for operator learning typically exhibit noise in the independent variables as well, since both variables represent signals that are subject to measurement error. In regression on scalar data, failure to account for noisy independent variables can lead to biased parameter estimates. With noisy independent variables, linear models fitted via ordinary least squares (OLS) will show attenuation bias, wherein the slope will be underestimated. In this work, we derive an analogue of attenuation bias for linear operator regression with white noise in both the independent and dependent variables, showing that the norm upper bound of the operator learned via OLS decreases with increasing noise in the independent variable. In the nonlinear setting, we computationally demonstrate underprediction of the action of the Burgers operator in the presence of noise in the independent variable. We propose error-in-variables (EiV) models for two operator regression methods, MOR-Physics and DeepONet, and demonstrate that these new models reduce bias in the presence of noisy independent variables for a variety of operator learning problems. Considering the Burgers operator in 1D and 2D, we demonstrate that EiV operator learning robustly recovers operators in high-noise regimes that defeat OLS operator learning. We also introduce an EiV model for time-evolving PDE discovery and show that OLS and EiV perform similarly in learning the Kuramoto-Sivashinsky evolution operator from corrupted data, suggesting that the effect of bias in OLS operator learning depends on the regularity of the target operator.

1 Introduction

Operator regression, or operator learning, has emerged as an important field in scientific computing that focuses on the flexible and expressive discovery of partial differential equations (PDEs) and related models. These methods fit parameterizations of operators, i.e., mappings between functions, using observations of the input and output of the operators. Recently, operator regression methods have begun to incorporate tech-

niques from machine learning such as deep neural networks and Gaussian processes, opening the way for novel applications in scientific computing challenges, such as data-driven model discovery [1], surrogate models of data-to-solution maps [2, 3], and closure models in fluid flow [4]. However, current operator regression methods based on deep learning assume that the independent variable, i.e. the input to the operator, is free of noise. The objective of this work is to explore the bias in models learned from noisy independent variables and to propose a correction, applicable to a wide variety of operator regression architectures, based on the error-in-variables method in classical statistics.

Fitting parametrized operators to observations is a classical problem and has been approached, e.g., with Bayesian methods [5, 6, 7], PDE-constrained optimization [8, 9], and physics-informed Gaussian processes [10, 11]. Earlier works leveraged significant prior knowledge of physics and fitted small numbers of physically interpretable parameters. In contrast, recent methods based on high-dimensional deep neural network (DNN) parametrization of operators have established deep operator learning as a widely applicable and domain-agnostic area of scientific computing. Deep operator learning methods vary based on the discretizations of function spaces inherent in each method as well as the architecture and training of the DNN. They include modal methods [12, 13, 14, 15], graph based methods [16, 17], PCA based methods [18], meshless methods [1], trunk-branch based methods [19, 3], and time-stepping methods [20, 21, 14]. Such methods can be purely data-driven or incorporate knowledge from physics [22], and can be utilized as general operator surrogates or be specifically for PDE model discovery [13]. Similar techniques in the computer science and statistics literature have been described as function to function regression within the field of functional data analysis (FDA) [23] but applied outside the PDE context. These methods include reproducing kernel Hilbert space approaches [24, 25], additive models [26], wavelet-based approaches [27], and neural network models [28, 29]. See [30] for a broad overview of the topic.

The effect of noisy inputs in the training and test data for deep operator regression methods is relatively unexplored, despite being widely studied in the related context of adversarial examples in deep learning [31, 32]. In single and multivariate statistics, error in the independent variables leads to inconsistent parameter estimates without proper error modelling. In particular, linear models fit via ordinary least squares (OLS) to scalar data $\{(x + \epsilon, y)\}$ with error in the independent variable persistently underpredict the slope as

$$m^* = m \frac{\text{var}(x)}{\text{var}(x) + \text{var}(\epsilon)}, \quad (1)$$

where m^* is the predicted slope, m is the true slope, x is the independent variable, and ϵ is the error in x [33]. This phenomenon is known as attenuation bias and methods such as total least squares [34], and Deming regression [35] have been developed to counteract it. More generally, including in nonlinear contexts, an error-in-variable (EiV) model is required to correct for inconsistency arising from error in the independent variables. Typically, EiV models are highly specialized to specific problems. See [36, 37] for overviews of this topic.

In contrast to single and multivariate data, where one often has more control over

the error in the independent variables, e.g. the placement of sensors, functional data originate from signals which are generally noisy, particularly when measured in extreme environments or at high sampling rates. This makes it essential that operator regression methods be robust to noise in both the input (independent) and output (dependent) variables. To the best of the authors' knowledge, there are currently no error-in-variable models available for either function-to-function regression or operator regression methods.

In this work, we derive a generalization to (1) for OLS inference of linear operators and demonstrate computationally that the action of OLS learned operators underpredict the action of the true operators. Under the assumption that the underlying functional data is smooth but corrupted by white noise, we propose EiV models for the non-linear operator regression methods, MOR-Physics [12, 13] and DeepONet [19]. We demonstrate that these models can correct for the bias introduced by noisy independent variables for several PDE learning problems.

2 Operator regression for noisy input data

Many operator regression methods seek to infer an operator by solving an ordinary least squares problem,

$$\mathcal{L} = \underset{\hat{\mathcal{L}}}{\operatorname{argmin}} \mathbb{E}_{(u,v)} \left[\left\| \hat{\mathcal{L}}u - v \right\|_V^2 \right], \quad (2)$$

where \mathcal{L} is potentially nonlinear, $(u, v) \in U \times V$ represent the input and output of the target operator and the expectation \mathbb{E} is over a distribution of functions that is appropriate for a given application, including any stochastic variables meant to model measurement noise. The specification of the Banach spaces U and V and the distribution inherent to the expectation (2) of input functions is a major theoretical challenge. The distribution weights a subset within the function space U where the learned operator is expected to provide an accurate surrogate of the target operator, as well as measurement noise. In practice, it is assumed that a finite sample of training functions are available which represent this theoretical distribution, and that these functions are discretized in some way so that they can be represented as finite-dimensional vectors,

$$\mathbf{u} \in \mathbb{R}^{d_1}, \quad \mathbf{v} \in \mathbb{R}^{d_2}, \quad (3)$$

It is also assumed that the operator \mathcal{L} is discretized in a consistent way,

$$\mathcal{L} : \mathbb{R}^{d_1} \rightarrow \mathbb{R}^{d_2}, \quad (4)$$

and that $\|\cdot\|_V = \|\cdot\|_{\ell^2(\mathbb{R}^{d_2})}$ for ease of computation.

We consider the case when the observations (u, v) originate from pairs (\hat{u}, \hat{v}) corrupted by independent white noise. With the discretization specified by (3) and (4), and the analogous discretization of (\hat{u}, \hat{v}) , we assume that we are given N observations such that

$$\begin{aligned} \mathbf{u}^i &= \hat{\mathbf{u}}^i + \boldsymbol{\epsilon}_u^i, & \mathbf{v}^i &= \hat{\mathbf{v}}^i + \boldsymbol{\epsilon}_v^i, & i &= 1, 2, \dots, N, \\ \boldsymbol{\epsilon}_u^i &\sim \mathcal{N}(\mathbf{0}, \sigma_u^2 I), & \boldsymbol{\epsilon}_v^i &\sim \mathcal{N}(\mathbf{0}, \sigma_v^2 I), \end{aligned} \quad (5)$$

where $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ is the multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance Σ . We seek an operator \mathcal{L} such that

$$\mathcal{L}(\hat{\mathbf{u}}) \approx \hat{\mathbf{v}}$$

for pairs $(\hat{\mathbf{u}}, \hat{\mathbf{v}})$ pairs of function in a given sample set (the test set). The discretized ordinary least squares problem from (2) for operator regression using this data is then to minimize the loss

$$J = \frac{1}{N} \sum_{i=1}^N \|\mathcal{L}\mathbf{u}^i - \mathbf{v}^i\|_{\ell^2(\mathbb{R}^{d_2})}^2. \quad (6)$$

We stress that the objective is to infer the mapping from noiseless input to noiseless output, having access only to noisy observations of input and output data for training.

2.1 Bias in operators inferred from Ordinary Least-Squares (OLS) operator regression

The presence of noise given by (5) while minimizing a loss function as in (2) leads to systematic bias in the prediction of \mathcal{L} , in a way that is analogous to the attenuation bias described previously. We generate pairs of noisy functions,

$$(u^i, v^i) = (\hat{u}^i + \epsilon_u^i, \partial_x \hat{u}^i + \epsilon_v^i),$$

where the u^i and v^i are generated as described in Section 6, and attempt to infer the Burgers operator $\mathcal{L}u = \partial_x u^2$ using the MOR-Physics operator regression method, described in Section 4.1. Figure 1 demonstrates the bias in the learned operator by plotting the action on a test function; the predicted output function exhibits ℓ_2 -norm smaller than the true output function. This is typical of the output on test functions, suggesting that the operator norm of \mathcal{L} itself is attenuated, i.e., that the OLS learned operator “underpredicts” the action of the true Burgers operator. In Table 1, we compute the maximum and average norms for the action of the OLS learned and true Burgers operator over a set of 1000 test functions in the unit ball of ℓ^2 ; note that the former approximates the operator norm of the learned \mathcal{L} over the sample space. These statistics strongly suggest that the operator norm is biased to zero. The hyperparameters used in this study are available in Appendix B.

	OLS operator	True operator
$\max \ \mathcal{L}u\ _{\ell_2(\mathbb{R}^{d_2})}$	4.42	12.45
$\text{average} \ \mathcal{L}u\ _{\ell_2(\mathbb{R}^{d_2})}$	2.40	7.75

Table 1: Operator norm statistics for true Burgers operator and OLS inferred operator. Computed over 1000 samples with $\|u\|_{\ell_2(\mathbb{R}^{d_1})} = 1$.

The following theorem shows that this can be expected for linear operators. It extends the classical result on attenuation bias to the case of a linear operator learned by OLS. We obtain the theorem for an idealized case of infinitely many finite-dimensional samples. That is, we assume a discretization of the form (3), but for infinitely many

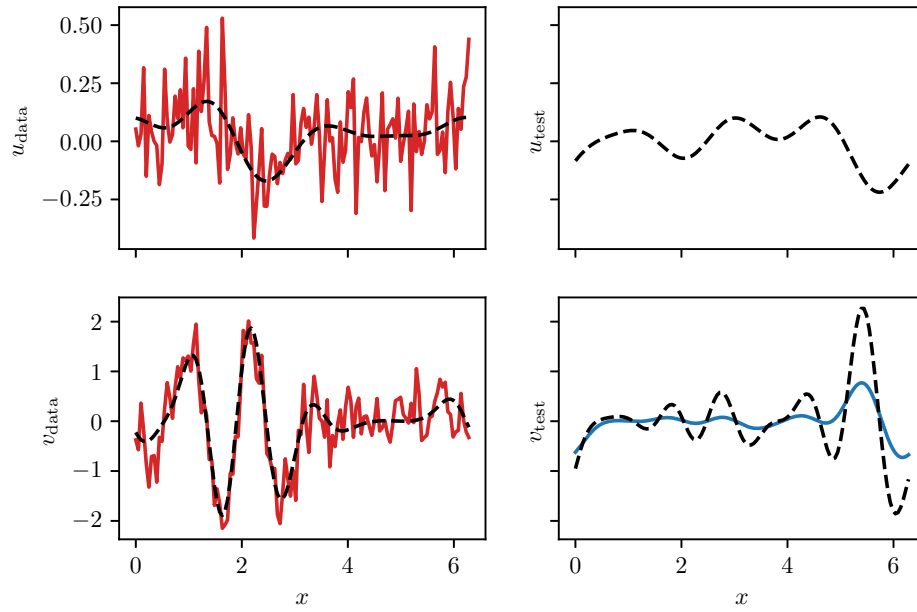


Figure 1: Operator learned from noisy independent variables with OLS underpredicts the action of the true operator. (Top left) sample of training u (—) and underlying smooth function \hat{u} (----). (Bottom left) sample of training v (—) and underlying smooth function $\hat{v} = \partial_x^2 \hat{u}$ (----). (Top right) smooth test function. (Bottom right) action of true Burgers operator on smooth test function (----) and action of OLS learned operator (—).

pairs, consistent with the implicit distribution of samples. Then, rather than the fully discrete loss (6), the theorem is stated for a loss of the form

$$\bar{J} = \mathbb{E} \left[\left\| \hat{\mathcal{L}} \mathbf{u} - \mathbf{v} \right\|_{\ell^2(\mathbb{R}^{d_2})}^2 \right], \quad \mathbf{u} \in \mathbb{R}^{d_1}, \mathbf{v} \in \mathbb{R}^{d_2}. \quad (7)$$

The fully discrete loss given in (6) converges to \bar{J} as the sample size increases. The following theorem is proven in Appendix A.

Theorem 1 *The minimizer to (7) among linear $\hat{\mathcal{L}}$ is*

$$\mathcal{L} = \mathbb{E} [\hat{\mathbf{v}} \hat{\mathbf{u}}^\top] (\mathbb{E} [\hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + \sigma_u^2 I)^{-1}.$$

The norm upperbound for this operator is

$$\|\mathcal{L}\| \leq \frac{\|\mathbb{E} [\hat{\mathbf{v}} \hat{\mathbf{u}}^\top]\|}{\|\mathbb{E} [\hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + \sigma_u^2 I\|}.$$

This norm upperbound decreases with increasing levels of noise, σ_u , suggesting a zero bias for the OLS inferred operator with large noise in u . Theorem 1 should be compared to the attenuation bias result for classical regression (1). The significance of this result is that while zero-bias noise in the dependent variable v can be compensated for by a large sample size, the presence of seemingly innocuous zero-bias noise in the independent variable u persists in biasing the optimal solution of the ordinary least squares problem (7) itself. The numerical results shown in Figure 1 and Table 1 above suggests similar behavior for nonlinear operator regression as well. This necessitates the error-in-variables model we propose below. In particular, ad-hoc denoising that does not address the limitations of OLS is not guaranteed to remove this bias, in contrast to our error-in-variables approach we discuss below.

3 Error-in-variables (EiV) model

Instead of applying OLS to infer the operator, we model the error for both u and v with the joint probability density function (PDF),

$$\begin{bmatrix} \tilde{u}^i - u^i \\ v^i - \mathcal{L} \tilde{u}^i \end{bmatrix} \sim \mathcal{N} \left(\mathbf{0}, \begin{bmatrix} \sigma_u^2 I & \\ & \sigma_v^2 I \end{bmatrix} \right)$$

where \tilde{u} is a “denoised” version of u . \mathcal{L} along with $\tilde{u}, \sigma_u, \sigma_v$ can be recovered via maximum likelihood estimation (MLE),

$$\mathcal{L}, \tilde{u}, \sigma_u, \sigma_v = \operatorname{argmax}_{\mathcal{L}, \tilde{u}, \sigma_u, \sigma_v} \prod_i p \left(\begin{bmatrix} \tilde{u}^i - u^i \\ v^i - \mathcal{L} \tilde{u}^i \end{bmatrix} \right), \quad (8)$$

where $p(\cdot)$ refers to the probability density of the indicated random variable. Real-world functional data is often smooth. Therefore, we simplify recovery of \tilde{u} via a

low-pass filter, $\tilde{u} = \mathcal{G}u$, and optimize over \mathcal{G} instead of \tilde{u} . We select for \mathcal{G} , the smooth spectral filter,

$$\mathcal{G}(u) = \mathcal{F}^{-1} \text{erfc}(a(\kappa - \kappa_c)) \mathcal{F}u$$

where a and κ_c are the filter bandwidth and cutoff wavenumber, respectively. The optimization over \mathcal{G} in (8) is replaced with optimization over the two filter parameters.

In our numerical studies, we have observed that priors on the filter parameters can improve robustness. For certain problems we place a Beta distribution prior on κ_c ,

$$\kappa_c / \beta_{\kappa_c} \sim \text{Beta}(1 + \varepsilon, 1 + \varepsilon).$$

where β_{κ_c} is a hyperparameter that specify the maximum allowable κ_c . With this prior, we perform maximum a posteriori (MAP) estimation to obtain \mathcal{L} ,

$$\mathcal{L}, \kappa_c, a, \sigma_u, \sigma_v = \underset{\mathcal{L}, \kappa_c, a, \sigma_u, \sigma_v}{\text{argmax}} \prod_i p\left(\begin{bmatrix} \mathcal{G}u^i - u^i \\ v^i - \mathcal{L}\mathcal{G}u^i \end{bmatrix}\right) p(\kappa_c / \beta_{\kappa_c}). \quad (9)$$

For small ε , Beta is a smooth approximation to the uniform distribution, and therefore MAP is amenable to gradient descent based algorithms. This prior forces our EiV model to select a smaller κ_c so that $\mathcal{G}u^i$ is smooth. For our numerical studies, we use $\varepsilon = 0.01$.

4 Operator regression methods

In this section, we summarize the two operator regression methods examined in our experiments.

4.1 MOR-Physics

The MOR-Physics operator regression method uses the following operator parameterization,

$$\mathcal{L}(u) = \sum_i^{N_o} \mathcal{F}^{-1} g_i(\kappa) \mathcal{F} h_i(u),$$

where \mathcal{F} is the Fourier transform, g_i and h_i are neural networks, κ is the wavenumber, and N_o is a hyperparameter. Here, $h_i(u)$ is applied pointwise, such that, $h_i(u(x)) = (h_i \circ u)(x)$. The optimization over \mathcal{L} in (8) is replaced here with optimization over the neural network parameters. For all numerical studies, we select the network width=5, depth=5, and activation function as elu for both h_i and g_i .

4.2 DeepONet

The DeepONet operator regression method includes a branch network, b , that takes as input u and a trunk network, t , that takes as input x , a single grid point on the function

v . In this work, we use the unstacked variant of the DeepONet with the following operator parameterization to approximate v at a single grid point x ,

$$\mathcal{L}(u, x) = \sum_{k=1}^p b_k(u) t_k(x) + b_0,$$

where p is the network width and b_0 is an additional bias term.

For all computational studies involving DeepONet, we use fully connected networks with tanh activation functions for both the trunk and branch. We chose depths of 3 and 2 for the trunk and branch, respectively and 70 for the network width, p .

5 Operator regression with error-in-variables model for time-evolving systems

In addition to the inference problem discussed in the previous sections, we are also interested in inferring time-evolving PDE's from noise corrupted solutions. Given a solution, $u(x, t) = \hat{u}(x, t) + \epsilon(x, t)$, we seek to recover a PDE in the following form,

$$\begin{aligned} \partial_t u &= \mathcal{L}(u), \quad x \in \Omega \\ u(x, 0) &= u_0(x) \\ \mathcal{B}u &= 0, \quad x \in \partial\Omega \end{aligned}$$

We discretize this PDE in time with forward Euler, obtaining,

$$u_{n+1} = u_n + \Delta t \mathcal{L}(u_n) = \mathcal{P}(u_n) \quad n = 1, 2, \dots, \hat{N}_t,$$

This temporal discretization results in a ResNet-type architecture which has been effectively used to parametrize and discover governing evolution equations [38, 21, 12, 14]. We still parameterize \mathcal{L} with either MOR-Physics or DeepONet. However, we can now ideally model the error in u over all time steps with the following joint PDF,

$$\begin{bmatrix} \mathcal{G}u_0 - u_0 \\ \mathcal{P}\mathcal{G}u_0 - u_1 \\ \vdots \\ \mathcal{P}^{\hat{N}_t}\mathcal{G}u_0 - u_n \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \sigma_u^2 I).$$

Due to the computational complexity of evaluating probability densities from the above distribution, we instead consider the following marginalized distribution as a simplification,

$$\begin{bmatrix} \mathcal{G}u_0 - u_0 \\ \mathcal{P}^{N_t}\mathcal{G}u_0 - u_{N_t} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \sigma_u^2 I),$$

where N_t is a hyperparameter indicating a maximum timestep.

6 Results

In the following sections, we compare OLS and EiV operator regression using both MOR-Physics and DeepONet. We perform all test cases on the periodic domain, $\Omega = [0, L]^d$, where d is the spatial dimension of the problem. For each test, we generate samples of smooth functions, $\hat{\mathbf{u}}^i$ by applying a spectral low pass filter to samples of white noise. For time independent problems, we then apply the *a priori* known operator to $\hat{\mathbf{u}}^i$ and obtain $\hat{\mathbf{v}}^i$.

We next corrupt the data by adding white noise and obtain \mathbf{u}^i and \mathbf{v}^i . To standardize the amount of noise added to $\hat{\mathbf{u}}$ and $\hat{\mathbf{v}}$, we tune the standard deviation of the white noise to target specific signal-to-noise ratios (SNR's),

$$\text{SNR} = 10 \log_{10} \left(\frac{\text{RMS}(\mathbf{u})}{\sigma_u^2} \right)^2,$$

and likewise for \mathbf{v} . For all test cases, we target the same SNR for both \mathbf{u} and \mathbf{v} . With this noisy data, we attempt to infer the operator or PDE. Finally we evaluate our inference by generating noiseless test data, u_{test} using the same method discussed above, and comparing the action of the inferred operator to the action of the true operator on u_{test} .

For time evolving problems, we numerically integrate the *a priori* known PDE, using $\hat{\mathbf{u}}_0^i = \hat{\mathbf{u}}^i$ as the initial condition, obtaining $\hat{\mathbf{u}}_n^i$ for all n timesteps. We next corrupt this data with white noise as discussed above, obtaining \mathbf{u}_n , from which we attempt to infer an evolution PDE. We evaluate our inference by generating a noiseless initial condition $\mathbf{u}_0 = \mathbf{u}_{\text{test}}$, evolving the inferred PDE, and comparing to the evolution of the true PDE using the same \mathbf{u}_{test} as an initial condition.

To perform the optimization in (8) and (9), we use the stochastic gradient based optimizer, ADAM [39]. Batch sizes, learning rates, sample sizes, and other hyperparameters for all numerical studies are listed in Appendix B.

6.1 Learning operators with the MOR-Physics EiV model

In these sections, we compare the OLS and EiV models using the MOR-Physics operator parameterization discussed in Section 4.1.

6.1.1 Recovering the Burgers operator

We consider the learning problem discussed in Figure 1 where we attempt to recover the Burgers operator, $\mathcal{L}u = \partial_x u^2$. In Figure 2, we apply OLS and EiV inferred error models to this dataset, recover operators, compute the action of these operators on a noiseless test function, u_{test} , and compare to the action of the true operator, v_{test} . We observe that the OLS model recovered operators only reproduce the action of the true Burgers operator for the highest SNR examined, while underpredicting the action for lower SNR's. In contrast, the EiV model is able to recover operators down to $\text{SNR} = 4$ with high accuracy. However, for lower SNR's the EiV model also fails to recover a suitable operator.

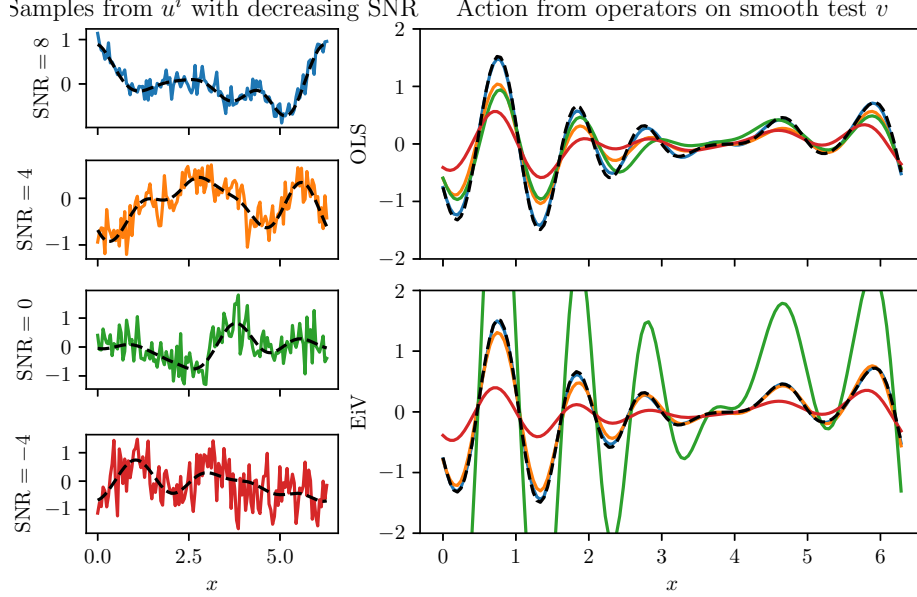


Figure 2: EiV model improves recovery of true Burgers operator in the presence of noisy independent variables. (Left) Underlying smooth function \hat{u} (----) and training u for increasing SNR = 8 (—), SNR = 4 (—), SNR = 0 (—), and SNR = -4 (—). (Right) Action of true Burgers operator (----) on noiseless test \mathbf{u}_{test} and action of learned operators from data with decreasing SNR for OLS (Top right) and EiV (Bottom right).

On the left four subplots of Figure 3, we examine the action of the filter \mathcal{G} on samples of noisy input functions, u^i , with decreasing SNR. For the high SNR data, the EiV model successfully identifies filter parameters that smooth the input functions. However, for low SNR data, the EiV model selects filters that fail to smooth the input functions, suggesting the method finds high κ_c .

6.1.2 Improved operator learning with smoothness prior

The results in Figure 2 can be improved by adding priors on the filter parameters as discussed in Section 3 to constrain κ_c to smaller values and enforce smoothness on $\mathcal{G}u^i$. In Figure 4, we plot the action of the learned operators with priors on κ_c and four different hyperparameters, β_{κ_c} . We find, for a wide range of β_{κ_c} , that the prior enables successful recovery of the Burgers operator even for the lowest SNR. However for the largest $\beta_{\kappa_c} = 80$, our EiV model can no longer consistently recover the Burgers operator. Notably, these results are relatively insensitive to β_{κ_c} , so careful hyperparameter selection is not necessary.

On the right four subplots of Figure 3, we examine the action of the filter \mathcal{G} on sample noisy input functions, u^i , with decreasing SNR and the four β_{κ_c} hyperparameters.

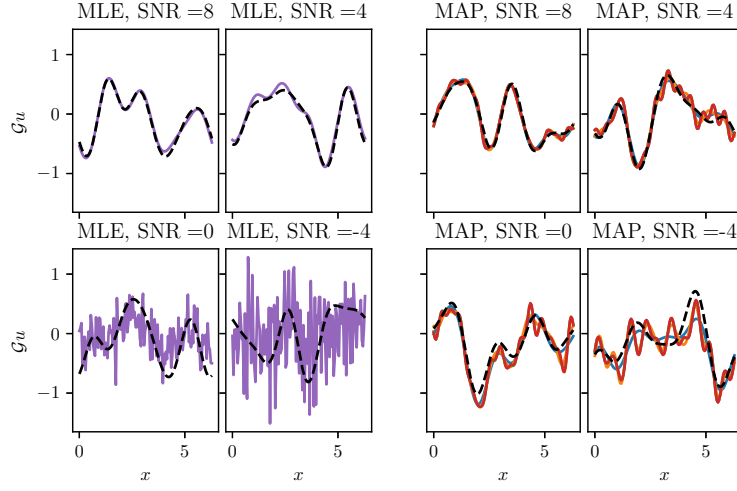


Figure 3: Effect of cutoff wavenumber prior on filter for EiV model. (*Left*) Action of MLE estimate of filters on noisy u^i (—) for decreasing SNR and corresponding noiseless \hat{u}^i (---). (*Right*) Action of MAP estimate of filters (κ_c prior) on u^i with hyperparameters, $\beta_{\kappa_c} = 10$ (—), $\beta_{\kappa_c} = 20$ (—), $\beta_{\kappa_c} = 40$ (—), and $\beta_{\kappa_c} = 80$ (—).

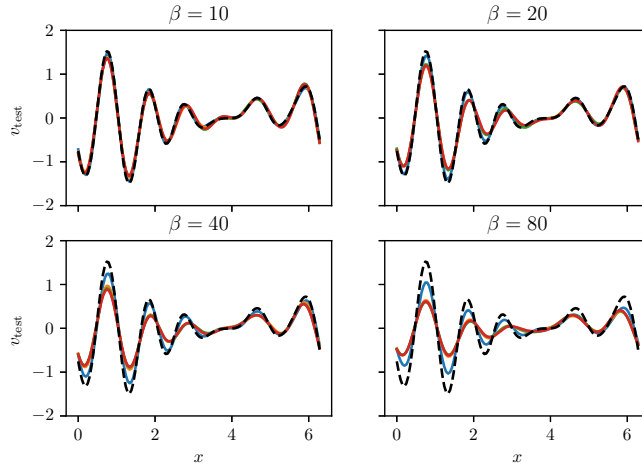


Figure 4: Cutoff wavenumber prior improves EiV model. Action of EiV operator on u_{test} learned from SNR = 8 (—), SNR = 4 (—), SNR = 0 (—), and SNR = -4 (—) for various β_{κ_c} . Action of true operator (---).

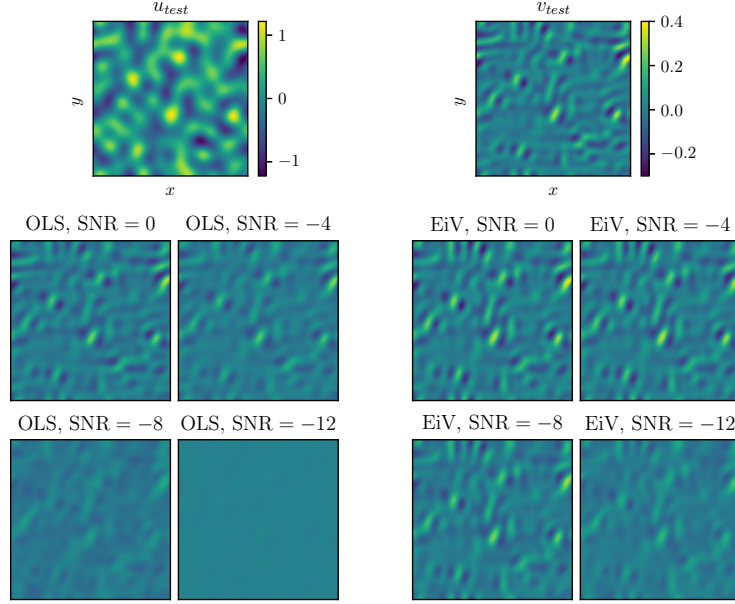


Figure 5: EiV model improves recovery of 2D Burgers operator from noisy data. (*Top*) Noiseless test data, u_{test} , and action of true operator, v_{test} . (*Bottom left*) Action of OLS (*Bottom left*) and EiV (*Bottom right*) learned operators on u_{test} for decreasing SNR.

We find that the MAP estimate always produces a operative filter that smooths u^i , even for the lowest SNR.

6.1.3 Learning the 2D Burgers operator

In this section, we attempt to recover a 2D generalization of the Burgers operator [40], $\mathcal{L} = \partial_x u^2 + \partial_y u^2$. Figure 5 compares the OLS and EiV learned operators for decreasing SNR. For the EiV model, we included the κ_c prior with $\beta_{\kappa_c} = 10$. As in the 1D Burgers operator test, we find that action of the OLS learned operator severely underpredicts the action of the true operator, especially at very low SNR. The EiV learned operator, however, captured the action of the true operator with SNR down to -4 . Even with lower SNR's, the EiV operator did not underpredict the true operator as severely as the OLS operator.

6.1.4 Learning the Kuramoto–Sivashinsky equation

In this section we attempt to recover the Kuramoto–Sivashinsky (KS) equation,

$$\partial_t u + 0.5 \partial_x u^2 + \partial_x^2 u + \partial_x^4 u = 0,$$

using the method outlined in Section 5. To generate the data, we solve KS equation using Pseudospectral methods. We use Orszag's 3/2 zero-padding technique to eliminate

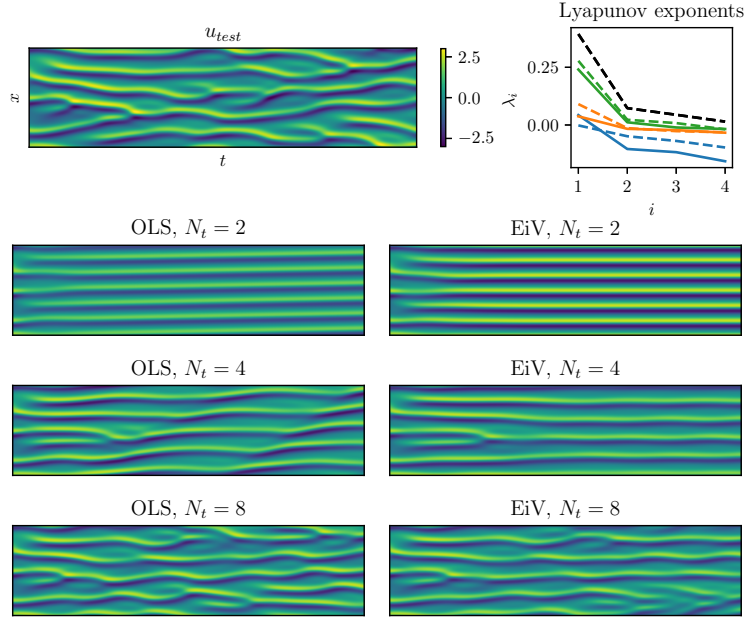


Figure 6: OLS and EiV models perform similarly for KS equation inference. (*Top left*) Noiseless test data, u_{test} . OLS (*Bottom left*) and EiV (*Bottom right*) inferred operators for increasing hyperparameter, N_t . *Top right* Lyapunov exponents for true equation (----); OLS equation with $N_t = 2$ (- - -), $N_t = 4$ (---), $N_t = 8$ (- - -); and EiV equation with $N_t = 2$ (—), $N_t = 4$ (—), $N_t = 8$ (—).

aliases from the Fourier transform of u^2 , which is computed in the (x, t) domain [41]. We use the low-storage Runge-Kutta method for temporal discretization [42] which has been widely used in the direct numerical simulation of turbulent flows [43, 44]. It treats the nonlinear term explicitly with third-order accuracy. Linear terms are implicitly treated, similarly to the Crank-Nicolson method, with second-order accuracy.

In Figure 6, we infer the equation using the OLS and EiV models for increasing final timestep, N_t , and found that both performed similarly. For this study, we used $\text{SNR} = 4$. Qualitatively, neither model was able to capture the dynamics of the KS equation with $N_t = 2$ and $N_t = 4$, but both models were able to capture the dynamics with $N_t = 8$.

As a chaotic system, the KS equation can be characterized by computing Lyapunov exponents as outlined in [45]. In Figure 6, we also compute the first four Lyapunov exponents for the true and learned equations. As observed in the qualitative comparison, OLS and EiV perform similarly, only successfully recovering the true KS equation for $N_t = 8$. This is in contrast to regressing the Burgers operator, in which EiV outperformed OLS. We hypothesize that is due to the difference in regularity of the target operators; the Burgers operator, a differential operator, reduces the regularity of the input function, while the KS evolution operator possesses smoothing properties [46]. As a result, one can expect the operator norm over most subspaces of sample functions to

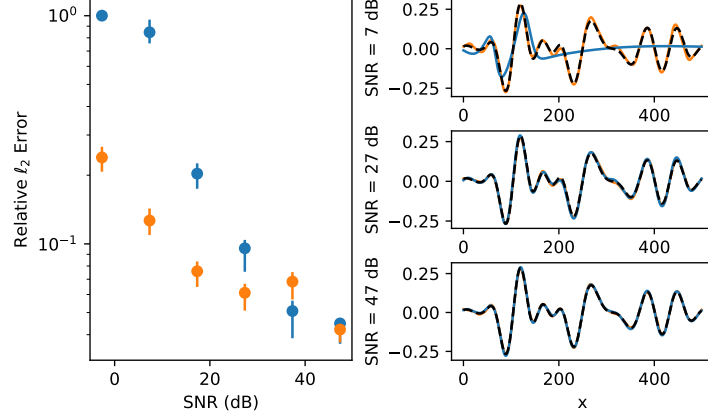


Figure 7: EiV inference performs better than OLS inference for low SNR in DeepONet. (Left) Relative ℓ_2 error vs. SNR for OLS (●) and EiV (●). (Right) Action of true (---), OLS (—), and EiV (—) operators on u_{test} for increasing SNR. Error bars indicate 25th and 75th percentile over 100 test samples.

be much higher for the Burgers operator than for the KS evolution operator, rendering the OLS attenuation bias more significant in regressing the former. We plan to explore in more detail the advantages of EiV over OLS in time dependent PDE learning and time independent operator learning in future work.

6.2 Learning the Burgers operator with DeepONet EiV model

In this section, we examine EiV operator learning for the 1D Burgers operator using the DeepONet method outlined in Section 4.2. In Figure 7, we compute the ℓ_2 error between the actions of OLS and EiV learned operators on test functions and the true actions. As in MOR-Physics, the EiV learned operator robustly learns from noisy data, even at very low SNR. Finally, we examine the effect of training set size on EiV learning in Figure 8. We find that for very low samples, EiV fails to recover the Burgers operator to high accuracy. With larger numbers of samples, EiV can successfully recover the operator. Future work may focus on examining the posterior distribution of operators in the small data regime.

7 Conclusion

We have demonstrated that operator regression performed by minimizing a least-squares loss is prone to attenuation bias when the input, or independent variable, is corrupted by white noise. This is supported both by an upper bound on the operator norm for linear operator regression using ordinary least squares (OLS) as well as numerical results for nonlinear operator regression for problems such as recovering the Burgers operator from noisy observations. We proposed an Error-in-Variables (EiV) model for

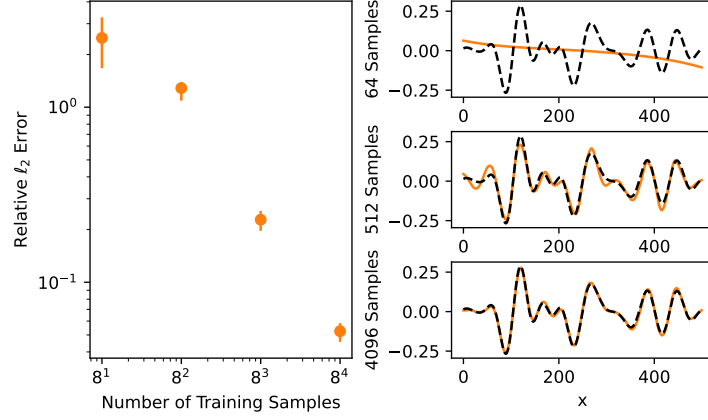


Figure 8: EiV inference improves with training set size in DeepONet. (Left) Relative ℓ_2 error vs. training set size for EiV (●). (Right) Action of true (---) and EiV (—) operators on u_{test} for increasing training set size. Results computed for $\text{SNR} = 17$. Error bars indicate 25th and 75th percentile over 100 test samples.

operator regression that replaces the least-squares loss with maximum likelihood estimation or maximum a posteriori estimation with an appropriate smoothness prior. The EiV method is applicable to a wide variety of existing operator regression methods; we demonstrate this by combining it with the MOR-physics method of [13] and the DeepONet method of [19]. For these methods, the EiV model significantly improves the operator prediction fidelity for a variety of problems given noise in the input variable, as demonstrated for the Burgers equation in one and two dimensions. Finally, we introduce an EiV model for time evolving PDEs and show OLS and EiV perform similarly in recovering the Kuramoto-Sivashinsky equation. This suggests that the regularity properties of the target operator effect the severity of OLS attenuation bias in practice.

Acknowledgements

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government. SAND Number: SANDx.

The work of R.G. Patel, I. Manickam, and M. Gulian are supported by the U.S. Department of Energy, Office of Advanced Scientific Computing Research under the Collaboratory on Mathematics and Physics-Informed Learning Machines for Multi-scale and Multiphysics Problems (PhILMs) project.

References

- [1] Nathaniel Trask, Ravi G. Patel, Ben J. Gross, and Paul J. Atzberger. GMLS-Nets: A framework for learning from unstructured data. *arXiv preprint arXiv:1909.05371*, 2019.
- [2] Huaqian You, Yue Yu, Marta D’Elia, Tian Gao, and Stewart Silling. Nonlocal Kernel Network (NKN): a stable and resolution-independent deep neural network. *arXiv preprint arXiv:2201.02217*, 2022.
- [3] Shengze Cai, Zhicheng Wang, Lu Lu, Tamer A Zaki, and George Em Karniadakis. DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks. *Journal of Computational Physics*, 436:110296, 2021.
- [4] Karthik Duraisamy, Gianluca Iaccarino, and Heng Xiao. Turbulence modeling in the age of data. *Annual Review of Fluid Mechanics*, 51:357–377, 2019.
- [5] Guofei Pang, Paris Perdikaris, Wei Cai, and George Em Karniadakis. Discovering variable fractional orders of advection–dispersion equations from field data using multi-fidelity Bayesian optimization. *Journal of Computational Physics*, 348:694–714, 2017.
- [6] Andrew M Stuart. Inverse problems: a Bayesian perspective. *Acta Numerica*, 19:451–559, 2010.
- [7] Nicolas Garcia Trillos and Daniel Sanz-Alonso. The Bayesian formulation and well-posedness of fractional elliptic inverse problems. *Inverse Problems*, 33(6):065006, 2017.
- [8] Marta D’Elia and Max Gunzburger. Identification of the diffusion parameter in nonlocal steady diffusion problems. *Applied Mathematics & Optimization*, 73(2):227–249, 2016.
- [9] Olena Burkovska, Christian Glusa, and Marta D’Elia. An optimization-based approach to parameter learning for fractional type nonlocal models. *Computers & Mathematics with Applications*, 2021.
- [10] Mamikon Gulian, Maziar Raissi, Paris Perdikaris, and George Karniadakis. Machine learning of space-fractional differential equations. *SIAM Journal on Scientific Computing*, 41(4):A2485–A2509, 2019.
- [11] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using Gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017.
- [12] Ravi G. Patel and Olivier Desjardins. Nonlinear integro-differential operator regression with neural networks. *arXiv preprint arXiv:1810.08552*, 2018.

- [13] Ravi G Patel, Nathaniel A Trask, Mitchell A Wood, and Eric C Cyr. A physics-informed operator regression framework for extracting data-driven continuum models. *Computer Methods in Applied Mechanics and Engineering*, 373:113500, 2021.
- [14] Tong Qin, Kailiang Wu, and Dongbin Xiu. Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics*, 395:620–635, 2019.
- [15] Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021.
- [16] Anima Anandkumar, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Nikola Kovachki, Zongyi Li, Burigede Liu, and Andrew Stuart. Neural operator: Graph kernel network for partial differential equations. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.
- [17] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020.
- [18] Kaushik Bhattacharya, Bamdad Hosseini, Nikola B Kovachki, and Andrew M Stuart. Model reduction and neural networks for parametric PDEs. *arXiv preprint arXiv:2005.03180*, 2020.
- [19] Lu Lu, Pengzhan Jin, and George Em Karniadakis. DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv preprint arXiv:1910.03193*, 2019.
- [20] Huaqian You, Yue Yu, Nathaniel Trask, Mamikon Gulian, and Marta D’Elia. Data-driven learning of nonlocal physics from high-fidelity synthetic data. *Computer Methods in Applied Mechanics and Engineering*, 374:113553, Feb 2021.
- [21] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-net: Learning PDEs from data. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3208–3216. PMLR, 10–15 Jul 2018.
- [22] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *Science advances*, 7(40):eabi8605, 2021.
- [23] Bernard Walter Silverman and James O Ramsay. Functional data analysis. In *International Encyclopedia of the Social and Behavioral Sciences*. Amsterdam: Elsevier, 2001.

- [24] Ming Yuan and T. Tony Cai. A reproducing kernel Hilbert space approach to functional linear regression. *The Annals of Statistics*, 38(6):3412 – 3444, 2010.
- [25] Hachem Kadri, Emmanuel Duflos, Philippe Preux, Stéphane Canu, Alain Rakotomamonjy, and Julien Audiffren. Operator-valued kernels for learning from functional response data. *Journal of Machine Learning Research*, 17(20):1–54, 2016.
- [26] Janet S. Kim, Ana-Maria Staicu, Arnab Maity, Raymond J. Carroll, and David Ruppert. Additive function-on-function regression. *Journal of Computational and Graphical Statistics*, 27(1):234–244, 2018.
- [27] Mark J Meyer, Brent A Coull, Francesco Versace, Paul Cinciripini, and Jeffrey S Morris. Bayesian function-on-function regression for multilevel functional data. *Biometrics*, 71(3):563–574, 2015.
- [28] Aniruddha Rajendra Rao and Matthew Reimherr. Modern non-linear function-on-function regression. *arXiv preprint arXiv:2107.14151*, 2021.
- [29] Connie Khor Li Kou, Hwee Kuan Lee, and Teck Khim Ng. A compact network learning model for distribution regression. *Neural Networks*, 110:199–212, 2019.
- [30] Jeffrey S. Morris. Functional regression. *Annual Review of Statistics and Its Application*, 2(1):321–359, 2015.
- [31] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- [32] Yuichi Yoshida and Takeru Miyato. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*, 2017.
- [33] Jennifer A. Hutcheon, Arnaud Chiolerio, and James A. Hanley. Random measurement error and regression dilution bias. *BMJ (Online)*, 340, 2010.
- [34] Ivan Markovsky and Sabine Van Huffel. Overview of total least-squares methods. *Signal Processing*, 87(10):2283–2302, 2007. Special Section: Total Least Squares and Errors-in-Variables Modeling.
- [35] Kristian Linnet. Evaluation of regression procedures for methods comparison studies. *Clinical chemistry*, 39(3):424–432, 1993.
- [36] Silvelyn Zwanzig. *Estimation in nonlinear functional error-in-variables models*. PhD thesis, Universität Hamburg, 2000.
- [37] Xiaohong Chen, Han Hong, and Denis Nekipelov. Nonlinear models of measurement errors. *Journal of Economic Literature*, 49(4):901–937, 2011.
- [38] Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, dec 2017.

- [39] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [40] N. A. Mohamed. Solving one- and two-dimensional unsteady Burgers' equation using fully implicit finite difference schemes. *Arab Journal of Basic and Applied Sciences*, 26(1):254–268, 2019.
- [41] Steven A. Orszag. On the elimination of aliasing in finite-difference schemes by filtering high-wavenumber components. *Journal of the Atmospheric sciences*, 28:1074, 1971.
- [42] Philippe R. Spalart, Robert D. Moser, and Michael M. Rogers. Spectral methods for the navier-stokes equations with one infinite and two periodic directions. *Journal of Computational Physics*, 96:297–324, 10 1991.
- [43] Sergio Hoyas and Javier Jiménez. Scaling of the velocity fluctuations in turbulent channels up to $Re = 2003$. *Physics of Fluids*, 18:011702, 2006.
- [44] Myoungkyu Lee and Robert D. Moser. Direct numerical simulation of turbulent channel flow up to $Re = 5200$. *Journal of Fluid Mechanics*, 774:395–415, 7 2015.
- [45] Russell A. Edson, J. E. Bunder, Trent W. Mattner, and A. J. Roberts. Lyapunov exponents of the Kuramoto–Sivashinsky PDE. *The ANZIAM Journal*, 61(3):270–285, 2019.
- [46] Pierre Collet, J-P Eckmann, Henri Epstein, and Joachim Stubbe. Analyticity for the Kuramoto–Sivashinsky equation. *Physica D: Nonlinear Phenomena*, 67(4):321–326, 1993.

A Proof of Theorem 1

We write the objective function in (7) as

$$\begin{aligned}
 \bar{J} &= \mathbb{E} [(\mathcal{L}\mathbf{u} - \mathbf{v})^\top (\mathcal{L}\mathbf{u} - \mathbf{v})] \\
 &= \mathbb{E} [(\mathbf{u}^\top \mathcal{L}^\top - \mathbf{v}^\top)(\mathcal{L}\mathbf{u} - \mathbf{v})] \\
 &= \mathbb{E} [\mathbf{u}^\top \mathcal{L}^\top \mathcal{L}\mathbf{u} - \mathbf{v}^\top \mathcal{L}\mathbf{u} - \mathbf{u}^\top \mathcal{L}^\top \mathbf{v} + \mathbf{v}^\top \mathbf{v}] \\
 &= \mathbb{E} [\mathbf{u}^\top \mathcal{L}^\top \mathcal{L}\mathbf{u} - 2\mathbf{v}^\top \mathcal{L}\mathbf{u} + \mathbf{v}^\top \mathbf{v}].
 \end{aligned}$$

Using linearity of expectation, this can be written as a sum of three terms, namely

$$\begin{aligned}
 I &= \mathbb{E} [\mathbf{u}^\top \mathcal{L}^\top \mathcal{L}\mathbf{u}] \\
 &= \mathbb{E} [(\hat{\mathbf{u}} + \boldsymbol{\epsilon}_u)^\top \mathcal{L}^\top \mathcal{L}(\hat{\mathbf{u}} + \boldsymbol{\epsilon}_u)] \\
 &= \mathbb{E} [(\hat{\mathbf{u}}^\top + \boldsymbol{\epsilon}_u^\top) \mathcal{L}^\top \mathcal{L}(\hat{\mathbf{u}} + \boldsymbol{\epsilon}_u)] \\
 &= \mathbb{E} [\hat{\mathbf{u}}^\top \mathcal{L}^\top \mathcal{L}\hat{\mathbf{u}} + \hat{\mathbf{u}}^\top \mathcal{L}^\top \mathcal{L}\boldsymbol{\epsilon}_u + \boldsymbol{\epsilon}_u^\top \mathcal{L}^\top \mathcal{L}\hat{\mathbf{u}} + \boldsymbol{\epsilon}_u^\top \mathcal{L}^\top \mathcal{L}\boldsymbol{\epsilon}_u] \\
 &= \mathbb{E} [\hat{\mathbf{u}}^\top \mathcal{L}^\top \mathcal{L}\hat{\mathbf{u}} + 2\boldsymbol{\epsilon}_u^\top \mathcal{L}^\top \mathcal{L}\hat{\mathbf{u}} + \boldsymbol{\epsilon}_u^\top \mathcal{L}^\top \mathcal{L}\boldsymbol{\epsilon}_u] \\
 &= \mathbb{E} [\hat{\mathbf{u}}^\top \mathcal{L}^\top \mathcal{L}\hat{\mathbf{u}}] + \mathbb{E} [\boldsymbol{\epsilon}_u^\top \mathcal{L}^\top \mathcal{L}\boldsymbol{\epsilon}_u];
 \end{aligned}$$

$$\begin{aligned}
II &= -\mathbb{E} [\mathbf{v}^\top \mathcal{L} \mathbf{u}] \\
&= -2\mathbb{E} [(\hat{\mathbf{v}} + \boldsymbol{\epsilon}_v)^\top \mathcal{L}(\hat{\mathbf{u}} + \boldsymbol{\epsilon}_u)] \\
&= -2\mathbb{E} [\hat{\mathbf{v}}^\top \mathcal{L} \hat{\mathbf{u}} + \hat{\mathbf{v}}^\top \mathcal{L} \boldsymbol{\epsilon}_u + \boldsymbol{\epsilon}_v^\top \mathcal{L} \hat{\mathbf{u}} + \boldsymbol{\epsilon}_v^\top \mathcal{L} \boldsymbol{\epsilon}_u] \\
&= -2\mathbb{E} [\hat{\mathbf{v}}^\top \mathcal{L} \hat{\mathbf{u}}] + \hat{\mathbf{v}}^\top \mathcal{L} [\mathbb{E} \boldsymbol{\epsilon}_u] + [\mathbb{E} \boldsymbol{\epsilon}_v^\top] \mathcal{L} \hat{\mathbf{u}} + [\mathbb{E} \boldsymbol{\epsilon}_v^\top] \mathcal{L} [\mathbb{E} \boldsymbol{\epsilon}_u] \\
&= -2\mathbb{E} [\hat{\mathbf{v}}^\top \mathcal{L} \hat{\mathbf{u}}] ;
\end{aligned}$$

and $III = \mathbb{E} [\mathbf{v}^\top \mathbf{v}]$. While $\partial III / \partial \mathcal{L} = 0$, we have

$$\begin{aligned}
\frac{\partial I}{\partial \mathcal{L}} &= \mathbb{E} \left[\frac{\partial}{\partial \mathcal{L}} \hat{\mathbf{u}}^\top \mathcal{L}^\top \mathcal{L} \hat{\mathbf{u}} \right] + \mathbb{E} \left[\frac{\partial}{\partial \mathcal{L}} \boldsymbol{\epsilon}_u^\top \mathcal{L}^\top \mathcal{L} \boldsymbol{\epsilon}_u \right] \\
&= \mathbb{E} [2\mathcal{L} \hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + \mathbb{E} [2\mathcal{L} \boldsymbol{\epsilon}_u \boldsymbol{\epsilon}_u^\top] \\
&= 2\mathcal{L} \mathbb{E} [\hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + 2\mathcal{L} \mathbb{E} [\boldsymbol{\epsilon}_u \boldsymbol{\epsilon}_u^\top] \\
&= 2\mathcal{L} (\mathbb{E} [\hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + \sigma_u^2 I) ,
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial II}{\partial \mathcal{L}} &= -\mathbb{E} \left[\frac{\partial}{\partial \mathcal{L}} \hat{\mathbf{v}}^\top \mathcal{L} \hat{\mathbf{u}} \right] \\
&= -2\mathbb{E} [\hat{\mathbf{v}} \hat{\mathbf{u}}^\top] .
\end{aligned}$$

Setting the derivative of the objective function equal to zero, we obtain

$$\mathcal{L} (\mathbb{E} [\hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + \sigma_u^2 I) = \mathbb{E} [\hat{\mathbf{v}} \hat{\mathbf{u}}^\top]$$

which has solution

$$\mathcal{L} = \mathbb{E} [\hat{\mathbf{v}} \hat{\mathbf{u}}^\top] (\mathbb{E} [\hat{\mathbf{u}} \hat{\mathbf{u}}^\top] + \sigma_u^2 I)^{-1} .$$

B Hyperparameters

The following tables lists the hyperparameters used in our numerical examples. The multiple learning rates indicate that we reinitialized the Adam optimizer during training and lowered the learning rate.

	Figures 1, 2, 3, and 4	Figure 5	Figure 6	Figures 7 and 8
Domain size, L	2π	$2\pi \times 2\pi$	30π	512
Grid size	128	128×128	256	512
Training set size	64	64	128	1500
Batch size	4	4	32	1500
Learning rate 1	1E-3	1E-3	1E-3	1e-4
Learning rate 2	1E-4	1E-4	1E-3	5e-5
Learning rate 3	1E-5	N/A	1E-3	1e-5
Epochs	400	200	1	1.5E6
Number of operators, N_o	1	1	2	N/A

Table 2: Hyperparameters used for all computational studies.