SIAM J. Applied Dynamical Systems
Vol. 18, No. 2, pp. 643–660

Data-Driven Identification of Parametric Partial Differential Equations[*]

Samuel Rudy[†], Alessandro Alla[‡], Steven L. Brunton[§], and J. Nathan Kutz[†]

Abstract. In this work we present a data-driven method for the discovery of parametric partial differential equations (PDEs), thus allowing one to disambiguate between the underlying evolution equations and their parametric dependencies. Group sparsity is used to ensure parsimonious representations of observed dynamics in the form of a parametric PDE, while also allowing the coefficients to have arbitrary time series, or spatial dependence. This work builds on previous methods for the identification of constant coefficient PDEs, expanding the field to include a new class of equations which until now have eluded machine learning based identification methods. We show that group sequentially thresholded ridge regression outperforms group LASSO in identifying the fewest terms in the PDE along with their parametric dependency. The method is demonstrated on four canonical models with and without the introduction of noise.

Key words. data-driven method, sparse regression, parametric models

AMS subject classifications. 37M02, 65P02, 49M02

DOI. 10.1137/18M1191944

1. Introduction.

The extraction of physical laws from experimental data, often in the form of differential and partial differential equations, may be critical to science and engineering applications where governing equations are unknown. Time-series data collected from experiments, or as the macroscopic aggregate of small scale behavior, often obeys unknown governing equations that are parametrized by time-evolving parameters $\mu(t)$. In some cases, it may be possible to derive physical laws from first principles using data as well as some knowledge of the system, but there are many cases where this is elusive such as the large scale networked dynamical systems of the power grid and the brain, or chemical kinetics of, for example, the Belousov–Zhabotinsky reaction. Recently, there has been a substantial research effort towards automating the process of data-driven model discovery in order to identify interpretable expressions for the dynamics in the form of ordinary and partial differential equations (ODEs and PDEs):

$$u_t = N(u, u_x, u_{xx}, \dots, \mu(t)), \quad t \in [0, T], \tag{1}$$

[*] Received by the editors June 4, 2018; accepted for publication (in revised form) by T. Sauer February 5, 2019; published electronically April 9, 2019.

http://www.siam.org/journals/siads/18-2/M119194.html

Funding: The work of the first, third, and fourth authors was supported by the Defense Advanced Research Projects Agency (DARPA) grant HR0011-16-C-0016. The work of the third author was also supported by the Air Force Office of Scientific Research (AFOSR) grant FA9550-18-1-0200. The work of the fourth author was also supported by the Air Force Office of Scientific Research (AFOSR) grant FA9550-15-1-0385.

[†] Department of Applied Mathematics, University of Washington, Seattle, WA 98185 (shrudy@uw.edu, kutz@uw.edu).

[‡] Department of Mathematics, PUC-Rio, Rua Marques de Sao Vicente, 225, Rio de Janeiro, 22451-900, Brazil (alla@mat.puc-rio.br).

[§] Department of Mechanical Engineering, University of Washington, Seattle, WA 98195 (sbrunton@uw.edu).

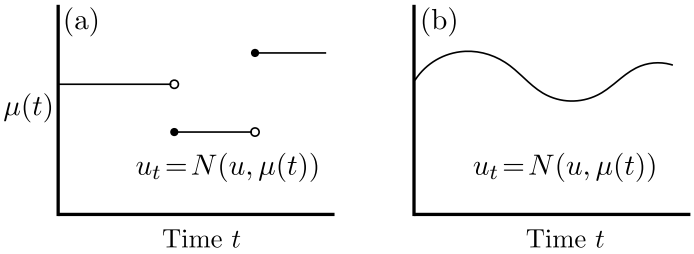

Figure 1. *Two prototypical scenarios for parametric model discovery:* (a) *Parameters $\mu(t)$ are piecewise constant and change at fixed points in time.* (b) *The underlying PDE model $u_t = N(u, \mu(t))$ depends on continuously varying parameter $\mu(t)$.*

where $N(\cdot)$ characterizes the evolution of the system and its parametric dependencies through the parameter $\mu(t) : [0, T] \to \mathbb{R}$. Although a number of automated discovery techniques have been developed for discovering the right-hand side of (1) with constant parameters $\mu(t) = \mu_0$, none have demonstrated the capability to infer the governing equations when the parametrization $\mu(t)$ has explicit time dependence. Through the usage of techniques imposing group sparsity, we develop a mathematical architecture that allows us to explicitly disambiguate the dynamical evolution (1) from its parametric dependencies $\mu(t)$. This is a critical innovation in model discovery as most realistic systems do indeed have time-dependent parametric dependencies that must be concurrently extracted during the model discovery process.

Figure 1 demonstrates two prototypical parametric dependencies: (a) a PDE model whose constant parameters change at fixed points in time and (b) a PDE model that depends continuously on the parameter $\mu(t)$. Our proposed model discovery method provides a principled approach to efficiently handle these two parametric cases, thus advancing the field of model discovery by disambiguating the dynamics from parametric time dependence. Even if the governing equations are known, parametric dependencies in PDEs complicate numerical simulation schemes and challenge one's ability to produce accurate future state predictions. Characterizing parametric dependence is also a critical task for model reduction in both time-independent [42, 24] and time-dependent PDEs [4, 5]. Thus the ability to explicitly extract the parametric dependence of a spatio-temporal system is necessary for accurate, quantitative characterization of PDEs.

More broadly, system identification using machine learning methods has emerged as a viable alternative to expert knowledge and first principles derivations. It is important to separate the field of system identification into two distinct categories: (i) methods that accurately reflect observed dynamics using black box functions (e.g., neural networks), and (ii) methods that recover closed form and interpretable expressions for the dynamics in the form of ordinary and partial differential equations (ODEs and PDEs). This duality reflects the two cultures narrative of machine learning and classical statistics made popular by Breiman [8]. On one hand, the research may assume a specific model for the data with known mechanism, while on the other the research is interested in algorithmic models that, while not necessarily reflecting the true mechanism, are accurate in prediction. While several recent works have made progress from both viewpoints, we focus on the former. The terms in a differential equation often have physical interpretations and motivation, e.g., diffusion and advection are ubiquitous in many physical systems and are characterized by prototypical expressions in a PDE. For systems where first principal derivations prove intractable, we may gain insights into the underlying physics of the system based on the terms in the identified PDE. Further, we view the process of extracting closed form equations as being more generalizable than fitting black box models to a specific dataset. Specifically, altering initial conditions will not change governing equations, but may break a machine learned black box solver.

Research towards the automated inference of dynamical systems from data is not new [18]. Methods for extracting linear systems from time series data include the eigensystem realization algorithm (ERA) [26] and dynamic mode decomposition (DMD) [50, 57, 62, 12, 31, 3]. Identification of nonlinear systems has, until very recently, relied on black box methods. These include NARMAX [15], neural networks [23], equation free methods [28, 29, 30], and Laplacian spectral analysis [22]. There has also been considerable recent work towards data-driven approximation of the Koopman operator [39, 13, 40] via extensions of DMD [64], diffusion maps [21], delay-coordinates [10, 2, 19], and neural networks [65, 60, 63, 38, 41, 34].

The use of genetic algorithms for nonlinear system identification [6, 58] allowed for the derivation of physical laws in the form of ordinary differential equations. Genetic algorithms are highly effective in learning complex functional forms but are slower computationally than simple regression. Sparsity promoting methods have been used previously in dynamical systems [53, 35, 14], and sparse regression has been leveraged to identify parsimonious ordinary differential equations from a large library of allowed functional forms [11, 16]. Much work building on the sparse regression framework has followed and includes inferring rational functions [36], the use of information criteria for model validation [37], constrained regression for conservation laws [32], model discovery with highly corrupt data [61], the learning of bifurcation parameters [56], stochastic dynamics [7], weak forms of the observed dynamics [54], and regression with small amounts of data [55, 27]. In contrast to sparse regression, a neural network based approach was proposed to identify ordinary differential equations, sacrificing some interpretability for a richer class of allowed functional forms [48].

Sparse regression based methods for PDEs were first used in [51, 52]. These methods were demonstrated on a large class of PDEs and have the benefit of being highly interpretable, but struggle with numerical differentiation of noisy data. In Rudy et al. [51] the noise was addressed by testing with only a small degree of noise (large SNR), while in Schaeffer [52] noise was added to the time derivative after it was computed from clean data. Alternatively, Gaussian processes were used to determine linear PDEs [45] and nonlinear PDEs known up to a set of coefficients [44]. Using Gaussian process regression requires less data than sparse regression and naturally manages noise, but the method is only applicable to PDEs with a known structure. Reference [33] makes a substantial contribution by using neural networks to accurately learn partial differential equations with nonconstant coefficients. A neural network is constructed that mimics a forward Euler timestepping scheme and the accuracy of potential models is evaluated based on their future state prediction accuracy. While seemingly more robust than sparse regression, the method in [33] does not penalize extraneous terms in the learned PDE and thus falls short of producing optimally parsimonious models. Furthermore, [33] only tests the method on a nonlinear problem using a relatively strong Ansatz. Neural networks were also used in [46] and [47] to solve and to estimate parameters in partial differential equations with known terms to a high degree of accuracy.

However, similar to [44], it is assumed that the PDE is known up to a set of coefficients. A more sophisticated neural network approach was used in [43] to learn dynamics of systems with unknown terms. However, the approach in [43] does not give closed form representations of the dynamics and the resulting neural network model therefore does not give insights into the underlying physics.

In this work, we present a sparse regression framework for identifying PDEs with nonconstant coefficients, something that none of the previous PDE discovery methods are equipped to do. Specifically, we allow for variation in the value of a coefficient across time or space, but maintain that the active terms in the PDE must be consistent. This is an important innovation in practice as the parameters of physical systems often vary during the measurement process, so that the parametric dependencies may be disambiguated from the PDE itself. Our method extends the sparse regression frameworks first proposed for PDE discovery [51, 52] by using group sparsity and results in a more parsimonious and interpretable model than neural networks. We are still limited by the accuracy of numerical differentiation and by the library terms in the sparse regression. Numerical differentiation using neural networks as shown in [43] appears promising as a method for obtaining more accurate time derivatives from noisy data. The limitation based on terms included in the library seems more permanent. Any closed-form model expression must be representable with a finite set of building blocks. Here we only use monomials in our data and its derivatives, since these are the common terms seen in physics, but there is no limitation to the terms included in the library.

2. Methods.

The parametric discovery method relies on several foundational mathematical tools. In the following subsections, we will discuss the identification of constant coefficient equations as well as regression methods for group sparsity. Finally, we combine these ideas to show how one may identify parametric PDEs and suggest a methodology for model selection that balances accuracy and the number of active terms in the PDE.

2.1. Identification of constant coefficient partial differential equations.

Several recent methods have been proposed for the identification of constant coefficient partial differential equations from data. In this work we expand on the sparse regression framework, PDE-FIND, used in [51]. We will briefly elaborate on the method and refer the reader to the original paper for details. The PDE-FIND algorithm provides a principled technique for discovering the underlying PDE from spatial time series measurements alone using a library of candidate functions for the PDE and sparse regression. For the identification of constant coefficient PDEs, we have a dataset $\mathbf{U}$, which is a discretization of a function $u(x,t)$ that we assume satisfies the PDE of the form given in (1):

$$u_t = N(u, u_x, u_{xx}, \dots) = \sum_{j=1}^{d} N_j(u, u_x, u_{xx}, \dots)\xi_j \,. \tag{2}$$

We assume that the nonlinear expression $N(\cdot)$ may be expanded as a sum of simple monomial basis functions N_j of u and its derivatives. Note that this sum is not unique and that we can include extra basis functions by simply setting the corresponding ξ_j to be zero. In PDE-FIND, which constructs an overcomplete library of many possible monomial basis functions and regresses to find ξ, sparsity is used to ensure that basis functions that do not appear in the PDE are set to zero in the sum.

To be more precise, given a dataset $\mathbf{U} \in \mathbb{R}^{n\times m}$ representing m timesteps of a PDE discretized with n gridpoints, we numerically differentiate in both x and t to form the linear regression problem given by (3),

$$\underbrace{\begin{pmatrix} u_t(x_1,t_1) \\ u_t(x_2,t_1) \\ \vdots \\ u_t(x_n,t_m) \end{pmatrix}}_{\mathbf{U}_t} = \underbrace{\begin{pmatrix} 1 & u(x_1,t_1) & u_x(x_1,t_1) & \dots & u^3u_{xxx}(x_1,t_1) \\ 1 & u(x_2,t_1) & u_x(x_2,t_1) & \dots & u^3u_{xxx}(x_2,t_1) \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & u(x_n,t_m) & u_x(x_n,t_m) & \dots & u^3u_{xxx}(x_n,t_m) \end{pmatrix}}_{\mathbf{\Theta}(\mathbf{U})} \xi, \tag{3}$$

which is a large, overdetermined linear system of equations $\mathbf{Ax} = \mathbf{b}$. Note that here we have shown a problem where derivatives up to third order are multiplied by powers of u up to cubic order, but one could include arbitrarily many library functions. Solving for ξ and ensuring sparsity gives the PDE. PDE-FIND has been shown to accurately identify several partial differential equations from data alone. The sparsity constraint is a regularizer for the linear regression [51].

2.2. Group sparsity.

In a typical sparse regression, we seek a sparse solution to the linear system of equations $\mathbf{Ax} = \mathbf{b}$. Accuracy of the predictor, $\|\mathbf{Ax} - \mathbf{b}\|$, is balanced against the number of nonzero coefficients in $\mathbf{x}$. Thus the sparse regularization enforces a solution $\mathbf{x}$ with many zeros (which is the variable ξ in (3)). In this paper, we use the notion of group sparsity to find time series representing each parameter in the PDE, rather than single values. We group collections of terms in $\mathbf{x}$ together and seek solutions to $\mathbf{Ax} = \mathbf{b}$ that minimize the number of groups with nonzero coefficients.

One well studied method for solving regression problems with group sparsity is the group LASSO [20]

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{w}} \frac{1}{2n} \left\| \mathbf{b} - \sum_{g\in\mathcal{G}} \mathbf{A}_{(g)}\mathbf{w}_{(g)} \right\|_2^2 + \lambda \sum_{g\in\mathcal{G}} \|\mathbf{w}_{(g)}\|_2. \tag{4}$$

Here $\mathcal{G}$ is a collection of groups, each of which contains a subset of the indices enumerating the columns of $\mathbf{A}$ and coefficients in $\mathbf{x}$. Note that the second term in the group Lasso corresponds to a convex relaxation of the number of groups containing a nonzero value, given by $\sum_{g\in\mathcal{G}} \delta_{\|\mathbf{w}_{(g)}\|,0}$. Optimal solutions to (4) satisfy the Karush–Kuhn–Tucker (KKT) conditions. For each $g' \in \mathcal{G}$, solutions must satisfy

$$\underbrace{\frac{1}{n}\mathbf{A}_{(g')}{}^T \left(\mathbf{b} - \sum_{g\neq g'} \mathbf{A}_{(g)}\mathbf{x}_{(g)} \right)}_{\mathbf{r}_{(g')}} - \frac{1}{n}\mathbf{A}_{(g')}{}^T\mathbf{A}_{(g')}\mathbf{x}_{(g')} \in \lambda\partial\|\mathbf{x}_{(g')}\|_2, \tag{5}$$

where the subgradient is a set valued function $\partial\|\mathbf{x}_{(g')}\|_2 = \{\mathbf{x}_{(g')}\|\mathbf{x}_{(g')}\|_2^{-1}\}$ if $\mathbf{x}_{(g')} \neq 0$ or $\partial\|\mathbf{x}_{(g')}\|_2 = \{\mathbf{y} : \|\mathbf{y}\|_2 \leq 1\}$ for $\mathbf{x}_{(g')} = 0$. The KKT condition reveals that there is a solution with $\mathbf{x}_{(g')} = 0$ when $\|\mathbf{r}_{(g')}\|_2 \leq \lambda$. Conditions for uniqueness of the solution to (4) are explored in [49].

The concept of group sparsity has been used in several previous methods for identifying dynamics given by ordinary differential equations [16, 56]. As it will be shown in the following, we find that the group LASSO performs poorly in the case of identifying PDEs. We instead use a sequential thresholding method based on ridge regression, similar to the method used in [51], but adapted for group sparsity. A sequential thresholding method was also used in [56] for group sparsity but for ordinary and not partial differential equations. Reference [56] also provides some theoretical results concerning the convergence of a hard thresholding algorithm for group sparsity while the case of hard thresholding for nongroup sparsity is extensively studied in [66]. Our method, which we call sequential grouped threshold ridge regression (SGTR), is summarized in algorithm 1.

Algorithm 1 SGTR($\mathbf{A}, \mathbf{b}, \mathcal{G}, \lambda, \epsilon, \text{maxit}, f(\mathbf{x}) = \|\mathbf{x}\|_2.$)

```
# Solves $\mathbf{x} \approx \mathbf{A}^{-1}\mathbf{b}$ with sparsity imposed on groups in $\mathcal{G}$

# Initialize coefficients with ridge regression
$\mathbf{x} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \mathbf{A}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$

# Threshold groups with small $f$ and repeat
for $iter = 1, \dots, \text{maxit}$:

    # Remove groups with sufficiently small $f(\mathbf{x}_{(g)})$
    $\mathcal{G} = \{g \in \mathcal{G} : f(\mathbf{x}_{(g)}) > \epsilon\}$

    # Refit these groups (note this sets $\mathbf{x}_{(g)} = 0$ for $g \notin \mathcal{G}$)
    $\mathbf{x} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \sum_{g\in\mathcal{G}} \mathbf{A}_{(g)}\mathbf{w}_{(g)}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$

# Get unbiased estimates of coefficients after finding sparsity
$\mathbf{x}_{(\mathcal{G})} = \arg\min_{\mathbf{w}} \|\mathbf{b} - \sum_{g\in\mathcal{G}} \mathbf{A}_{(g)}\mathbf{w}_{(g)}\|_2^2$

return $\mathbf{x}$
```

Throughout the training, $\mathcal{G}$ tracks the groups that have nonzero coefficients, and it is paired down as we threshold coefficients with sufficiently small relevance, as measured by f. If a group $g \in \mathcal{G}$ meets the thresholding criterion $f(\mathbf{x}_{(g)}) < \epsilon$, then $\mathbf{x}_{(g)}$ is set to zero and not considered in subsequent iterations of algorithm 1. We use the 2-norm of the coefficients in each group for f but one could also consider arbitrary functions. In particular, for problems where the coefficients within each group have a natural ordering, as they do in our case as time series or spatial functions, one could consider smoothness or other properties of the functions. In practice, we normalize each column of $\mathbf{A}$ and $\mathbf{b}$ so that differences in scale between the groups do not affect the result of the algorithm. For the group LASSO we always perform an unregularized least squares regression on the nonzero coefficients after the sparsity pattern has been discovered to debias the coefficients. We found SGTR to outperform the group LASSO for the problem of correctly identifying the active terms in parametric PDEs.

2.3. Data-driven identification of parametric partial differential equations.

In the identification of parametric PDEs, we consider equations of the form

$$u_t = N(u, u_x, \dots, \mu(t)) = \sum_{j=1}^{d} N_j(u, u_x, \dots)\xi_j(t). \tag{6}$$

Note that this equation is similar to (2) but has time-varying parametric dependence. To capture spatial variation in the coefficients, we simply replace $\xi(t)$ with $\xi(x)$. The PDE is assumed to contain a small number of active terms, each with a time varying coefficient $\xi(t)$. We seek to solve two problems: (i) determine which coefficients are nonzero and (ii) find the values of the coefficients for each ξ_j at each timestep or spatial location for which we have data.

Prior knowledge of an appropriate set of basis functions may be vital to the accuracy of the algorithm proposed in this work. In what follows we assume the true dynamics N to lie in the span of our candidate functions $\{N_j\}_{j=1}^{d}$. It is shown in [11] that ODEs with trigonometric functions may be approximated by a sparse regression to their Taylor series, though no such analysis has been done for PDEs. The supplementary material in [51] demonstrates that regression using an incomplete library yields unpredictable results. Indeed, [11] indicates that if the correct model term(s) is not present, sparse identification often fails in a stereotypical way, producing a nonparsimonious representation of the dynamical system. Thus it is typically easy to determine if the algorithm fails.

For time dependent problems, we construct a separate regression for each timestep, allowing for variation in the PDE between timesteps. Similar to the PDE-FIND method, we construct a library of candidate functions for the PDE using monomials in our data and its derivatives so that

$$\mathbf{\Theta}\left(u^{(j)}\right) = \begin{pmatrix} | & | & | & | \\ 1 & u^{(j)} & \dots & u^3 u_{xxx}^{(j)} \\ | & | & | & | \end{pmatrix}, \tag{7}$$

where the set of m equations is given by

$$u_t^{(j)} = \mathbf{\Theta}\left(u^{(j)}\right)\xi^{(j)}, \quad j = 1, \dots, m\,. \tag{8}$$

Our goal is to solve the set of equations given by (8) with the constraint that each $\xi^{(j)}$ is sparse and that they all share the same sparsity pattern. That is, we want a fixed set of active terms in the PDE. To do this, we consider the set of equations as a single linear system and use group sparsity. Expressing the system of equations for the parametric equation as a single linear system we get the block diagonal structure given by

$$\underbrace{\begin{pmatrix} u_t^{(1)} \\ u_t^{(2)} \\ \vdots \\ u_t^{(m)} \end{pmatrix}}_{u_t} = \underbrace{\begin{pmatrix} \mathbf{\Theta}\left(u^{(1)}\right) & & & \\ & \mathbf{\Theta}\left(u^{(2)}\right) & & \\ & & \ddots & \\ & & & \mathbf{\Theta}\left(u^{(m)}\right) \end{pmatrix}}_{\mathbf{\Theta}} \underbrace{\begin{pmatrix} \xi^{(1)} \\ \xi^{(2)} \\ \vdots \\ \xi^{(m)} \end{pmatrix}}_{\xi}. \tag{9}$$

We solve (9) using SGTR with columns of the block diagonal library matrix grouped by their corresponding term in the PDE. Thus for m timesteps and d candidate functions in the library, groups are defined as $\mathcal{G} = \{\{j + d \cdot i : i = 0, \dots, m-1\} : j = 1, \dots, d\}$. This ensures a sparse solution to the PDE while also allowing arbitrary time series for each variable. For problems with spatial, rather than temporal, variation in the coefficients, we simply group by spatial rather than time coordinate. A similar block diagonal structure is obtained but with n blocks of size $m \times d$ rather than m blocks of size $n \times d$. The groups are defined by $\mathcal{G} = \{\{j + d \cdot i : i = 0, \dots, n-1\} : j = 1, \dots, d\}$.

Since we are evaluating the relevance of groups based on their norm, it is important to consider differences in the scale of the candidate functions. For example, if $u \sim \mathcal{O}(10^{-2})$, then a cubic function will be $\mathcal{O}(10^{-6})$ and relatively large coefficients multiplying this data may not have a large effect on the dynamics, but will not be removed by the hard threshold due to its size. To remedy this, we normalize each candidate function represented in $\mathbf{\Theta}$ as well as each $u_t^{(j)}$ to have unit length prior to the group thresholding algorithm and then correct for the normalization after we have discovered the correct sparsity pattern.

2.4. Model selection.

For each model we test both the group LASSO as well as SGTR using an exhaustive range of parameter values. Let $\tilde{\mathbf{\Theta}}$ denote the block diagonal matrix $\mathbf{\Theta}$ shown in (9) but with all columns normalized to have unit length and $\tilde{u}_t$ be the vector of all time derivatives having been normalized to unit length so that $\|\tilde{u}_t\| = \sqrt{m}$. For the group LASSO, we find the minimal value of λ that will set all coefficients to zero which is given by

$$\lambda_{\text{max}} = \max_{g \in \mathcal{G}} \frac{1}{n} \|\tilde{\mathbf{\Theta}}_{(g)^T} \tilde{u}_t\|_2. \tag{10}$$

We check 50 evenly spaced values of λ between $10^{-5}\lambda_{\text{max}}$ and λ_{max} on a logarithmic scale.

For SGTR, we search over the range of tolerances between ϵ_{min} and ϵ_{max} defined as

$$\epsilon_{max/min} = \underset{g \in \mathcal{G}}{max/min} \|\xi_{\text{ridge},(g)}\|_2, \tag{11}$$

where $\xi_{\text{ridge}} = (\tilde{\mathbf{\Theta}}^T \tilde{\mathbf{\Theta}} + \lambda I)^{-1} \tilde{\mathbf{\Theta}}^T \tilde{u}_t$. Note that by definition, ϵ_{min} is the minimum tolerance that has any effect on the sparsity of the predictor and ϵ_{max} is the minimum tolerance that guarantees all coefficients to be zero. A set of 50 intermediate tolerances, equally spaced on a logarithmic scale, is tested between ϵ_{min} and ϵ_{max}.

To select the optimal model generated via each method, we evaluate the models using the Akaike information criterion (AIC)-inspired loss function

$$\mathcal{L}(\xi) = N \ln \left(\frac{\|\tilde{\mathbf{\Theta}}\xi - \tilde{u}_t\|_2^2}{N} + \epsilon \right) + 2k, \tag{12}$$

where k is the number of nonzero coefficients in the identified PDE given by the number of nonzero-elements of ξ divided by the number of time-steps, $\|\xi\|_0/m$, and N is the number of rows in $\mathbf{\Theta}$, which is equal to the size of our original dataset u.

Equation (12) is closely related to the AIC [1]. Typically, the mean square error of a linear model is used to evaluate goodness of fit, but in our case there is error in computing

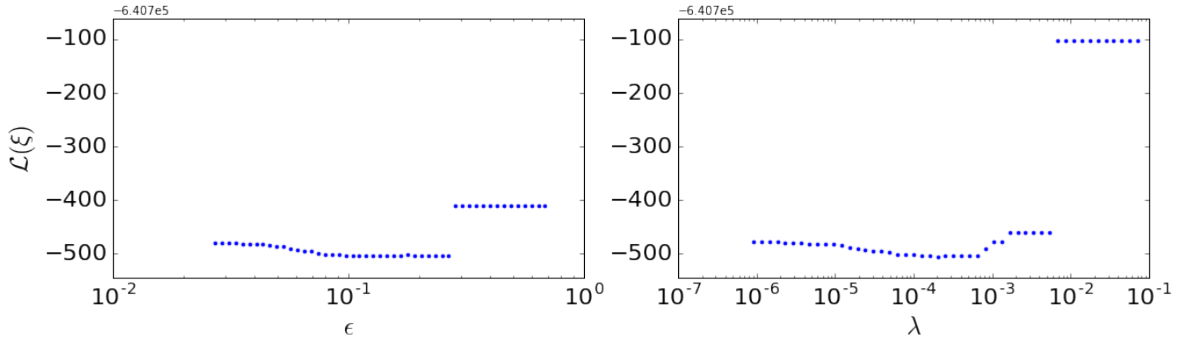

Figure 2. *Example of loss function evaluated for a number of candidate models for the parametric Burgers' equation. Library included derivatives up to 4th order multiplying powers of u up to cubic. Left: 50 models obtained via SGTR algorithm using values of ϵ between ϵ_{min} and ϵ_{max}. Right: 50 models obtained via group LASSO for λ between $10^{-5}\lambda_{max}$ and λ_{max}.*

the time derivative u_t, so we assume that any linear model which perfectly fits the data is overfit. We have added $\epsilon = 10^{-5}$ to the mean square error of each model as a floor in order to avoid overfitting. Without this addition, our algorithm selects insufficiently parsimonious representations of the dynamics.

Figure 2 illustrates the loss function from (12) evaluated on models derived from 50 values of ϵ and λ using SGTR and group LASSO, respectively. Initially, a low penalty in each algorithm yields a model that is overfit to the data given our sparsity criteria. For an intermediate value of ϵ or λ, a more parsimonious but still predictive model is obtained. For sufficiently high values, the model is too sparse and is no longer predictive.

3. Computational results of parametric PDE discovery.

We test our method for the discovery of parametric partial differential equations on four canonical models; Burgers' equation with a time varying nonlinear term, the Navier–Stokes equation for vorticity with a jump in Reynolds number, a spatially dependent advection equation, and a spatially dependent Kuramoto–Sivashinsky equation. In each case the method is also tested after introducing white noise with mean magnitude equal to 1% of the L^2-norm of the dataset. Noise is added directly to the data, $\mathbf{U}$ prior to numerical differentiation in order to replicate the effects of sensor noise. The proposed method is able to accurately identify the dynamics in each case except the Kuramoto–Sivashinsky equation, where the inclusion of a fourth order derivative makes numerical evaluation with noise highly challenging. Since final coefficient values for both group LASSO and SGTR are found using an unregularized regression on the nonzero terms found using sparse regression, results in any case where both algorithms determine the correct active terms are identical. We have therefore omitted side-by-side comparison and instead only show the group LASSO coefficients in cases where that method failed to identify the correct active terms. Python code used for each example is available online.[1]

[1] https://github.com/snagcliffs/parametric-discovery

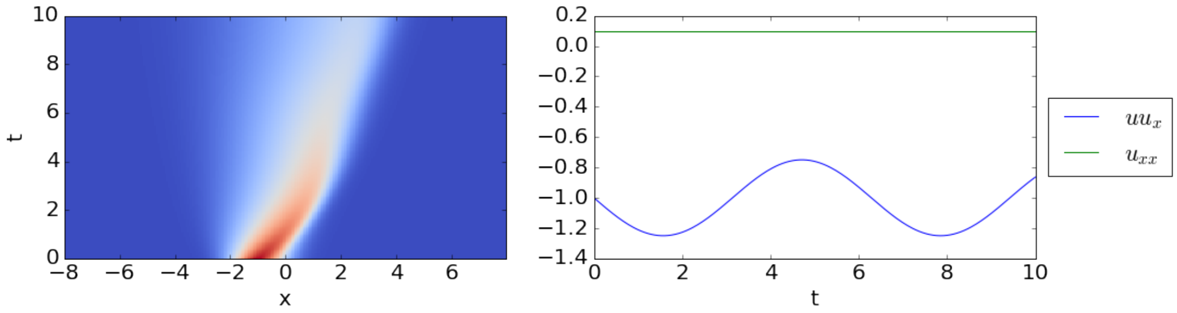

Figure 3. *Left: dataset for identification of the parametric diffusive Burgers' equation. Here the PDE was evolved on the interval* $[-8, 8]$ *with periodic boundary conditions and* $t \in [0, 10]$. *Right: coefficients for the terms in the parametric Burgers' equation. The diffusion was held constant at* 0.1 *while the nonlinear advection as coefficient is given by* $a(t) = -(1 + sin(t)/4)$.

3.1. Burgers' equation with diffusive regularization.

To test the parametric discovery of PDEs, we consider a solution of Burgers' equation with a sinusoidally oscillating coefficient $a(t)$ for the nonlinear advection term

$$\begin{aligned} u_t &= a(t)uu_x + 0.1u_{xx}, \\ a(t) &= -\left(1 + \frac{\sin(t)}{4}\right), \end{aligned} \tag{13}$$

where a small amount of diffusion is added to regularize the evolution dynamics.

Burgers' equation is solved numerically using the discrete Fourier transform (DFT) to evaluate spatial derivatives and the SciPy function *odeint* [25] for temporal integration on the interval $[-8, 8]$ with periodic boundary conditions and $t \in [0, 10]$ with $n = 256$ grid points and $m = 256$ time steps. See Figure 3 for a numerical solution and coefficients. We search for parsimonious representations of the dynamics by including powers of u up to cubic order, which can be multiplied by derivatives of u up to fourth order. For the noise-free dataset we use the DFT for computing derivatives. For the noisy dataset, we use polynomial interpolation to smooth the derivatives [51].

The resulting time series for the identified nonzero coefficients are shown in Figure 4. SGTR correctly identified the active terms in the PDE for both the noise-free and noisy datasets, whereas group LASSO fails in both cases to produce the correct PDE and its parametric dependencies. For group LASSO, the λ minimizing (12) is not the minimum value of $\lambda_{min} = 10^{-5}\lambda_{max}$, indicating that an appropriate range has been tested. Furthermore, no value of λ yields the correct sparsity pattern. This strongly suggests that there does not exist a λ such that the group LASSO realizes the correct sparsity pattern for the Burgers' equation dataset considered in this work. This may be due to the convex relaxation from $\delta_{\|\mathbf{w}_{(g)}\|,0}$ to $\|\mathbf{w}_{(g)}\|_2$ used in (4). We see from (5) that $\mathbf{x}_{(g')} = 0$ if $\mathbf{A}_{(g')}$ is sufficiently misaligned with $\mathbf{b} - \sum_{g \neq g'} \mathbf{A}_{(g)}\mathbf{x}_{(g)}$. For $\mathbf{A}_{(g')}{}^T\mathbf{b}$ large this may be less likely given small errors in the other coefficients. Indeed, since the Burgers' dataset is a traveling wave, $\langle u_x, u_t \rangle$ is large, which may

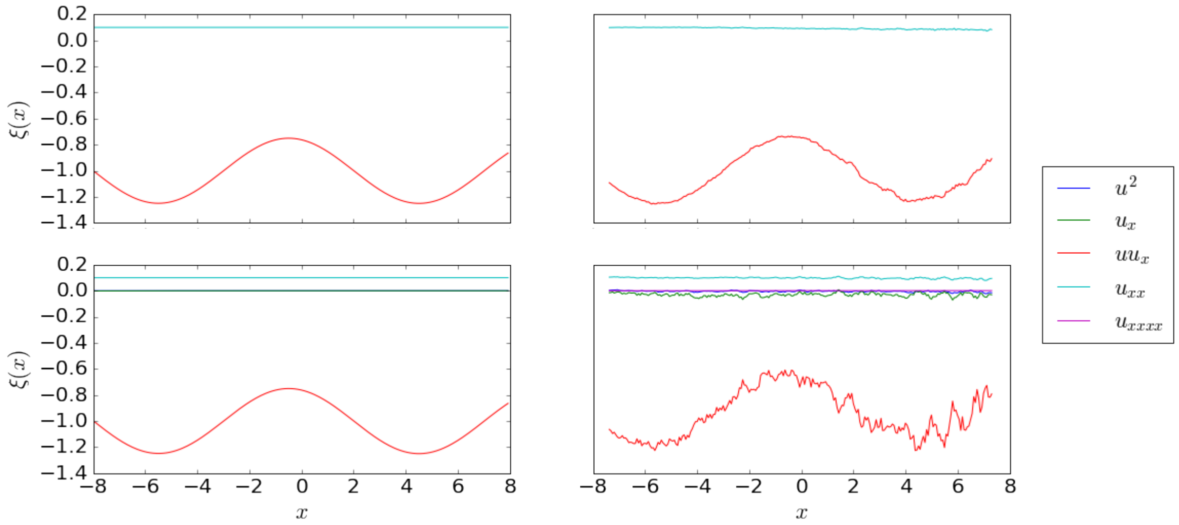

Figure 4. *Time series discovered for the coefficients of the parametric Burgers' equation. Top row: SGTR method, which correctly identifies the two terms. Bottom row: group LASSO method which adds several additional (incorrect) terms to the model. The left panels are noise-free, while the right panels contain 1% noise. This parametric dependency is illustrated in Figure 1b.*

explain why group LASSO fails to set the coefficient for u_x to zero.

3.2. Navier–Stokes: Flow around a cylinder. We consider the fluid flow around a circular cylinder by simulating the Navier–Stokes vorticity equation

$$\omega_t + \mathbf{u} \cdot \nabla\omega = \frac{1}{\nu(t)}\Delta\omega. \tag{14}$$

Data is generated using the immersed boundary projection method (IBPM) [59, 17] with $n_x = 449$ and $n_y = 199$ spatial points in x and y, respectively, and 1000 timesteps with $dt = 0.02$. The Reynolds number is adjusted half way through the simulation from $\nu = 100$ initially to $\nu = 75$. See Figure 5 for a snapshot of numerical solution and coefficient time series. This is representative of the fluid velocity exhibiting a sudden decrease midway through the data collection. Our library of candidate functions is constructed using up to second order derivatives of the vorticity and multiplied by up to quadratic functions of the data. To keep the size of the machine learning problem tractable, we subsample 1000 random spatial location from the wake of the cylinder to construct our library at every tenth timestep [51]. For the noise-free dataset, far fewer points are needed to accurately identify the dynamics. We suspect that with a more careful treatment of the numerical differentiation in the case of noisy data, such as that used in [43], the same would be true for the dataset with artificial noise; however, such work is not the focus of this paper. The identified time series for the Navier–Stokes equation are shown in Figure 6. SGTR and group LASSO both correctly identify the active terms in the PDE.

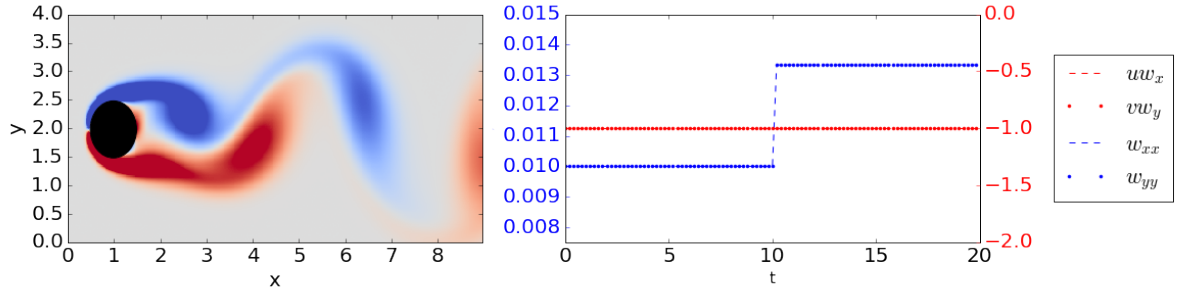

Figure 5. *Left: dataset for identification of the parametric Navier–Stokes equation* (14). *Right: coefficients for Navier–Stokes equations exhibiting jump in Reynolds number from* 100 *to* 75 *at* $t = 10$. *This parametric dependency is illustrated in Figure* 1a.

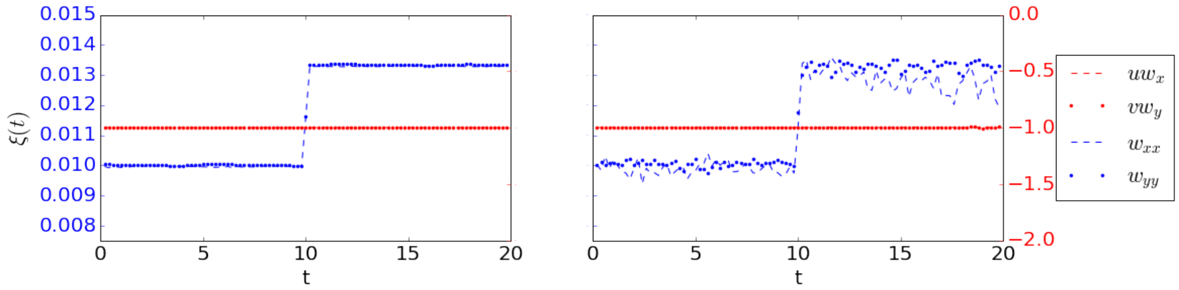

Figure 6. *Identified time series for coefficients for the Navier–Stokes equation. Distinct axes are used to highlight jump in Reynolds number. Left: no noise. Right:* 1% *noise.*

3.3. Spatially dependent advection-diffusion equation. The advection-diffusion equation is a simple model for the transport of a physical quantity in a velocity field with diffusion. Here, we adapt the equation to have a spatially dependent velocity

$$u_t = (c(x)u)_x + \epsilon u_{xx} = c(x)u_x + c'(x)u + \epsilon u_{xx} \tag{15}$$

which models transport through a spatially varying vector field due to $c = c(x)$. We solve (15) on a periodic domain $[-L, L]$ with $L = 5$ from $t = 0$ to 5, $\epsilon = 0.1$, and $c(x) = -1.5+\cos(2\pi x/L)$ using the DFT to evaluate spatial derivatives and the SciPy function *odeint* for temporal integration with $n = 256$ and $m = 256$. See Figure 7 for a numerical solution and coefficients. The library consists of powers of u up to cubic, multiplied by derivatives of u up to fourth order.

Results for the advection-diffusion equation are shown in Figure 8. In the noise-free and noisy datasets, both SGTR and group LASSO correctly identify the active terms in the PDE.

3.4. Spatially dependent Kuramoto–Sivashinsky equation. We now test the method on a Kuramoto–Sivashinsky equation with spatially varying coefficients

$$u_t = a(x)uu_x + b(x)u_{xx} + c(x)u_{xxxx}. \tag{16}$$

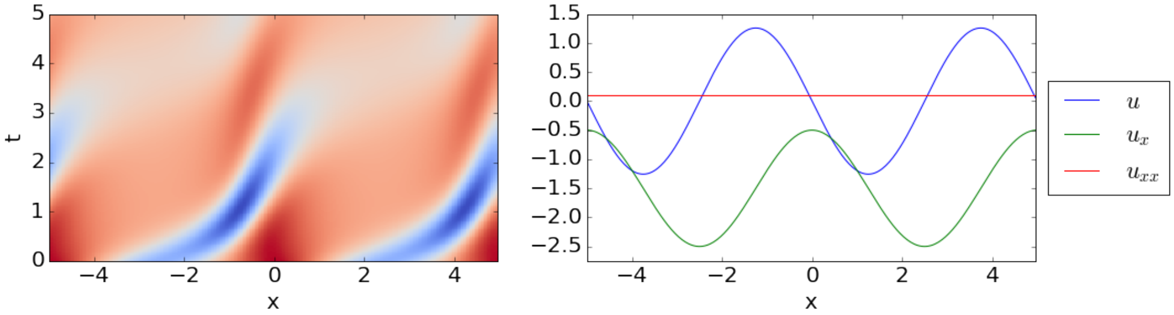

Figure 7. *Left: dataset for identification of the spatially dependent advection diffusion equation. Right: spatial dependence of PDE. In this case, the loadings $\xi_j(t)$ in (6) are replaced by $\xi_j(x)$.*

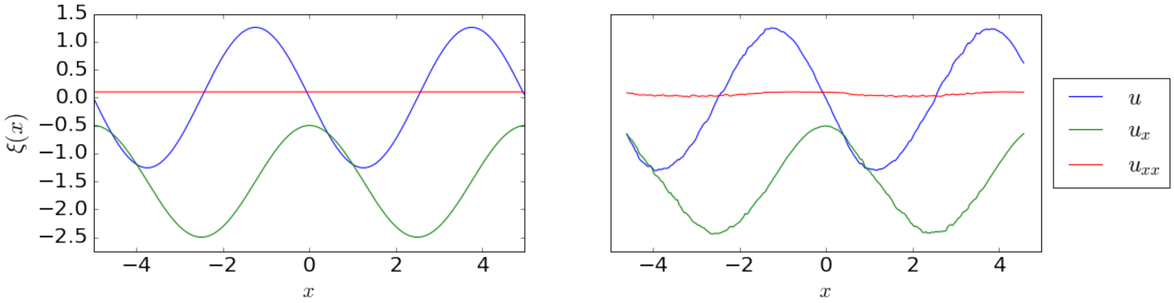

Figure 8. *Spatial dependence of advection diffusion equation. Left: no noise. Right: 1% noise. Both SGTR and group LASSO correctly identified the active terms.*

We use a periodic domain $[-L, L]$ with $L = 20$ and coefficients $a(x) = 1 + \sin(2\pi x/L)/4$, $b(x) = -1 + e^{-(x-2)^2/5}/4$ and $c(x) = -1 - e^{-(x+2)^2/5}/4$.

We solve (16) numerically using the DFT to evaluate spatial derivatives and the SciPy function *odeint* for temporal integration to $t = 200$ using $n = 512$ grid points and $m = 1024$ timesteps. See Figure 9 for a numerical solution and coefficients. The second half of the data set is used, so as to only consider the region where the dynamics exhibited spatio-temporal chaos, resulting in a dataset containing 512 snapshots of 512 gridpoints. The inclusion of a fourth order derivative in the Kuramoto–Sivashinsky equation makes accurate identification difficult due to error caused by noise in numerical differentiation. Indeed, our method fails to correctly identify the active terms when 1% noise is added. With 0.01% noise the correct terms were identified but with substantial error in coefficient value. We suspect that this shortcoming could at least be partially remedied by a more careful treatment of the numerical differentiation such as in [9]. The results of our parametric identification are shown in Figure 10.

4. Discussion.

We have presented a method for identifying governing laws for physical systems which exhibit either spatially or temporally dependent behavior. The method builds

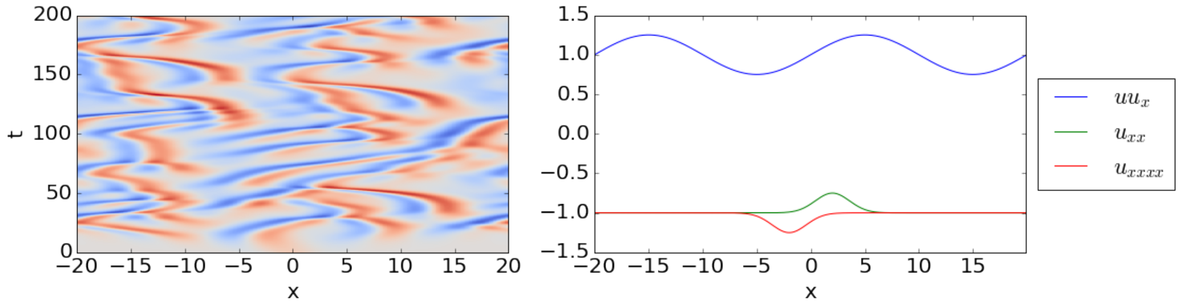

Figure 9. *Left: dataset for identification of the spatially dependent Kuramoto–Sivashinsky equation. Right: parametric dependency of the governing equations.*

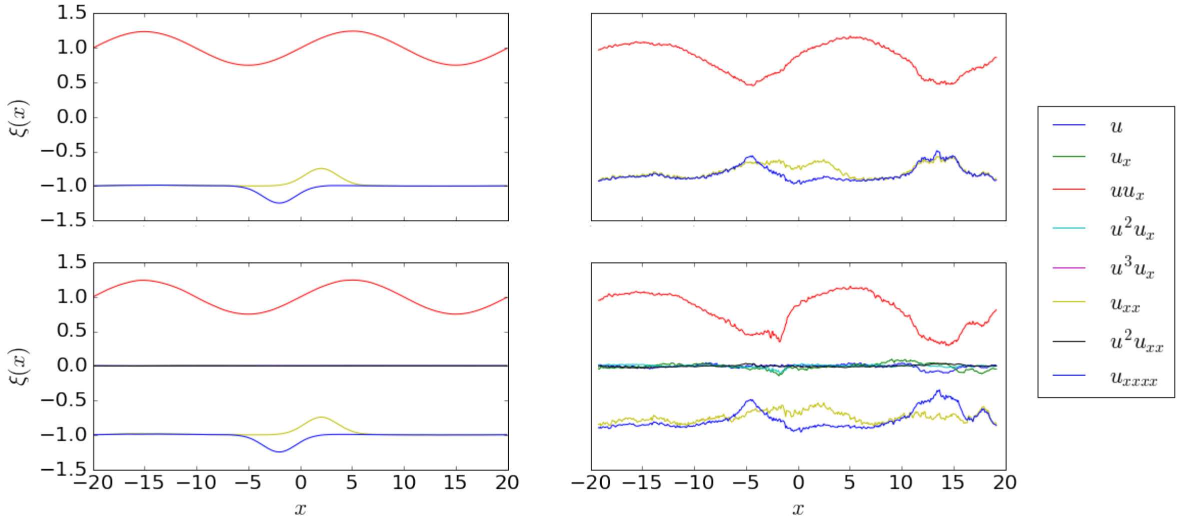

Figure 10. *Spatial dependence of Kuramoto–Sivashinsky. Top row: SGTR. Bottom row: group LASSO. Left: no noise. Right:* 0.01% *noise using SGTR. SGTR detects correct sparsity with significant parameter error. Group LASSO does not correctly identify the parsimonious model, nor does it do a good job at predicting the correct parametric values.*

on a growing body of work in the applied mathematics and machine learning community that seeks to automate the process of discovering physical laws. To the best of our knowledge, our method is the first approach for deriving parsimonious PDE expressions of spatio-temporal systems in the case of nonconstant coefficients. Specifically, we can disambiguate between the governing PDE evolution and its parametric dependencies. The SGTR algorithm exhibits equal or superior performance to the group Lasso in determining correct active terms on each of the example PDEs. Errors from the latter were generally in the form of extra terms added with small coefficient values throughout the time series. It may seem reasonable to threshold these time series after the discovery algorithm, but doing so assumes that the terms importance in the PDE is directly related to its magnitude, an assumption which we do not make given the normalization prior to sparse regression.

In this work we have split the data into distinct timesteps or spatial locations in order to find PDE models for each subset of the data, resulting in coefficients that can vary in space or time. However, with a sufficiently fine grid, it seems feasible that one could bin the data by areas localized in space and time to determine a coefficient varying in both space and time with some loss of resolution. Future work could consider spatial bins $\{S_i = [x_{min}^{(i)}, x_{max}^{(i)}]\}_{i=1}^{n_x}$ and temporal bins $\{T_i = [t_{min}^{(i)}, t_{max}^{(i)}]\}_{i=1}^{n_t}$. The block diagonal system of equations (9) would then have rows given by $u_t^{(i,j)} = \mathbf{\Theta}(u^{(i,j)})\xi^{(i,j)}$, where $u^{(i,j)} = u|_{S_i \times T_j}$ and have solution $\xi^{(i,j)}$ representing the course grained coefficients for the PDE. This same result may be achievable in a more stable manner by introducing a sparsity term to the work in [33].

We highlight that the results in this work are dataset specific. For different geometries and initial conditions the algorithm may yield variable results and determining sufficient conditions for accurate recovery of governing equations is an open problem. For a discussion of convergence results of hard-thresholding for the constant coefficient ordinary differential equation case, we refer the reader to [66].

As is the case with other sparse regression methods for identifying dynamical systems, this method is constrained by the ability of the user to accurately differentiate data. For ordinary differential equations, this may be circumvented by looking at the weak form of the dynamics [55], but doing so for PDEs seems difficult since there are derivatives that need to be evaluated with respect to multiple variables. We find the automatic differentiation approach used in [43] promising and suspect that the inclusion of neural network based differentiation could radically improve the ability of our method to identify dynamics from noisy data. With sufficient knowledge of data it may also be possible to obtain better estimates through tuning the polynomial based differentiation [9].

Automating the identification of closed form physical laws from data will hopefully boost scientific progress in areas where deriving the same laws from first principals proves intractable. There are several limitations to many methods proposed in the field thus far. In particular, current methods have generally studied equations of the form $u_t = N(u, x, t)$ but many equations in physics are not in this class. Indeed, if measuring a system with parametric dependencies, then past methods are unable to disambiguate between the evolution dynamics and its parametric dependencies $\mu(t)$, thus greatly limiting model discovery. There is also a trade off between methods that are able to derive parsimonious representations, though which are limited to a finite set of library elements, and those that use black box models to represent larger classes of possible functions. The researcher may also find difficulties in attempting to infer dynamics from the wrong set of measurements. For example, one could not derive the Schrödinger equation by only looking at measurements of intensity. While not addressing these issues, this work makes a step towards generalizing the class of equations which may be accurately identified via machine learning methods.

REFERENCES

[1] H. Akaike, *A new look at the statistical model identification*, IEEE Trans. Automatic Control, 19 (1974), pp. 716–723.

[2] H. Arbabi and I. Mezić, *Ergodic theory, dynamic mode decomposition and computation of spectral properties of the Koopman operator*, SIAM J. Appl. Dyn. Syst., 16 (2017), pp. 2096–2126, https://doi.org/10.1137/17M1125236.

[3] T. Askham and J. N. Kutz, *Variable projection methods for an optimized dynamic mode decomposition*, SIAM J. Appl. Dyn. Syst, 17 (2018), pp. 380–416, https://doi.org/10.1137/M1124176.

[4] P. Benner, S. Gugercin, and K. Willcox, *A survey of projection-based model reduction methods for parametric dynamical systems*, SIAM Rev., 57 (2015), pp. 483–531, https://doi.org/10.1137/130932715.

[5] P. Benner, M. Ohlberger, A. Patera, G. Rozza, and K. Urban, *Model Reduction of Parametrized Systems*, Springer, Cham, 2017.

[6] J. Bongard and H. Lipson, *Automated reverse engineering of nonlinear dynamical systems*, Proc. Natl. Acad. Sci. USA, 104 (2007), pp. 9943–9948.

[7] L. Boninsegna, F. Nüske, and C. Clementi, *Sparse Learning of Stochastic Dynamic Equations*, preprint, https://arxiv.org/abs/1712.02432, 2017.

[8] L. Breiman, *Statistical modeling: The two cultures (with comments and a rejoinder by the author)*, Statist. Sci., 16 (2001), pp. 199–231.

[9] O. Bruno and D. Hoch, *Numerical differentiation of approximated functions with limited order-of-accuracy deterioration*, SIAM J. Numer. Anal., 50 (2012), pp. 1581–1603, https://doi.org/10.1137/100805807.

[10] S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz, *Chaos as an intermittently forced linear system*, Nature Communications, 8 (2017), 19.

[11] S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, Proc. Natl. Acad. Sci. USA, 113 (2016), pp. 3932–3937.

[12] S. L. Brunton, J. L. Proctor, J. H. Tu, and J. N. Kutz, *Compressed sensing and dynamic mode decomposition*, J. Comput. Dyn., 2 (2015), pp. 165–191.

[13] M. Budišić, R. Mohr, and I. Mezić, *Applied Koopmanism*, Chaos, 22 (2012), 047510.

[14] R. E. Caflisch, S. J. Osher, H. Schaeffer, and G. Tran, *PDEs with compressed solutions*, Commun. Math. Sci., 13 (2015), pp. 2155–2176.

[15] S. Chen and S. A. Billings, *Representations of non-linear systems: the narmax model*, International Journal of Control, 49 (1989), pp. 1013–1032.

[16] S. Chen, A. Shojaie, and D. M. Witten, *Network reconstruction from high-dimensional ordinary differential equations*, J. Amer. Statist. Assoc., 112 (2017), pp. 1–11.

[17] T. Colonius and K. Taira, *A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions*, Comput. Methods Appl. Mech. Engrg., 197 (2008), pp. 2131–2146.

[18] J. Crutchfield and B. McNamara, *Equations of motion from a data series*, Complex Systems, 1 (1987), pp. 417–452.

[19] S. Das and D. Giannakis, *Delay-Coordinate Maps and the Spectra of Koopman Operators*, preprint, https://arxiv.org/abs/1706.08544, 2017.

[20] J. Friedman, T. Hastie, and R. Tibshirani, *A Note on the Group Lasso and a Sparse Group Lasso*, preprint, https://arxiv.org/abs/1001.0736, 2010.

[21] D. Giannakis, *Data-driven spectral decomposition and forecasting of ergodic dynamical systems*, Appl. Comput. Harmon. Anal., in press.

[22] D. Giannakis and A. J. Majda, *Nonlinear Laplacian spectral analysis for time series with intermittency and low-frequency variability*, Proc. Natl. Acad. Sci. USA, 109 (2012), pp. 2222–2227.

[23] R. Gonzalez-Garcia, R. Rico-Martinez, and I. Kevrekidis, *Identification of distributed parameter systems: A neural net based approach*, Computers & Chemical Engineering, 22 (1998), pp. S965–S968.

[24] J. S. Hesthaven, G. Rozza, and B. Stamm, *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*, Springer, Cham, 2016.

[25] E. Jones, T. Oliphant, P. Peterson, et al., *SciPy: Open sOurce Scientific Tools for Python*, 2001–, available online at http://www.scipy.org/.

[26] J.-N. Juang and R. S. Pappa, *An eigensystem realization algorithm for modal parameter identification and model reduction*, J. Guid. Control Dyn., 8 (1985), pp. 620–627.

[27] E. Kaiser, J. N. Kutz, and S. L. Brunton, *Sparse identification of nonlinear dynamics for model predictive control in the low-data limit*, Proc. A. 474 (2018), 20180335.

[28] I. Kevrekidis, C. Gear, J. Hyman, P. Kevrekidis, O. Runborg, and C. Theodoropoulos, *Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis*, Commun. Math. Sci., 1 (2003), pp. 715–762.

[29] I. G. Kevrekidis, C. W. Gear, and G. Hummer, *Equation-free: The computer-aided analysis of complex multiscale systems*, AIChE J., 50 (2004), pp. 1346–1355.

[30] I. G. Kevrekidis and G. Samaey, *Equation-free multiscale computation: Algorithms and applications*, Annu. Rev. Phys. Chem., 60 (2009), pp. 321–344.

[31] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*, Other Titles in Applied Mathematics 149, SIAM, Philadelphia, 2016, https://doi.org/10.1137/1.9781611974508.

[32] J.-C. Loiseau and S. L. Brunton, *Constrained sparse Galerkin regression*, J. Fluid Mech., 838 (2018), pp. 42–67.

[33] Z. Long, Y. Lu, X. Ma, and B. Dong, *PDE-net: Learning PDEs from Data*, preprint, https://arxiv.org/abs/1710.09668, 2017.

[34] B. Lusch, J. N. Kutz, and S. L. Brunton, *Deep Learning for Universal Linear Embeddings of Nonlinear Dynamics*, preprint, https://arxiv.org/abs/1712.09707, 2017.

[35] A. Mackey, H. Schaeffer, and S. Osher, *On the compressive spectral method*, Multiscale Model. Simul., 12 (2014), pp. 1800–1827, https://doi.org/10.1137/140965909.

[36] N. M. Mangan, S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Inferring Biological Networks by Sparse Identification Of Nonlinear Dynamics*, preprints, https://arxiv.org/abs/1605.08368, 2016.

[37] N. M. Mangan, J. N. Kutz, S. L. Brunton, and J. L. Proctor, *Model selection for dynamical systems via sparse regression and information criteria*, Proc. A. 473 (2017), 20170009.

[38] A. Mardt, L. Pasquali, H. Wu, and F. Noé, *VAMPnets: Deep learning of molecular kinetics*, Nature Communications, 9 (2018), 5.

[39] I. Mezić, *Spectral properties of dynamical systems, model reduction and decompositions*, Nonlinear Dynam., 41 (2005), pp. 309–325.

[40] I. Mezic, *Analysis of fluid flows via spectral properties of the Koopman operator*, Annu. Rev. Fluid Mech., 45 (2013), pp. 357–378.

[41] S. E. Otto and C. W. Rowley, *Linearly-Recurrent Autoencoder Networks for Learning Dynamics*, preprint, https://arxiv.org/abs/1712.01378, 2017.

[42] A. Quarteroni, A. Manzoni, and F. Negri, *Reduced Basis Methods for Partial Differential Equations: An Introduction*, Unitext 92, Springer, Cham, 2015.

[43] M. Raissi, *Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations*, preprint, https://arxiv.org/abs/1801.06637, 2018.

[44] M. Raissi and G. E. Karniadakis, *Hidden Physics Models: Machine Learning of Nonlinear Partial Differential Equations*, preprint, https://arxiv.org/abs/1708.00588, 2017.

[45] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Machine learning of linear differential equations using Gaussian processes*, J. Comput. Phys., 348 (2017), pp. 683–693.

[46] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics Informed Deep Learning (Part* I*): Data-Driven Solutions of Nonlinear Partial Differential Equations*, preprint, https://arxiv.org/abs/1711.10561, 2017.

[47] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics Informed Deep Learning (Part* II*): Data-Driven Discovery of Nonlinear Partial Differential Equations*, preprint, https://arxiv.org/abs/1711.10566, 2017.

[48] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Multistep Neural Networks for Data-Driven Discovery of Nonlinear Dynamical Systems*, preprint, https://arxiv.org/abs/1801.01236, 2018.

[49] V. Roth and B. Fischer, *The Group-Lasso for Generalized Linear Models: Uniqueness of Solutions and Efficient Algorithms*, in Proceedings of the 25th International Conference on Machine Learning, ACM, New York, 2008, pp. 848–855.

[50] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. Henningson, *Spectral analysis of nonlinear flows*, J. Fluid Mech., 645 (2009), pp. 115–127.

[51] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Data-driven discovery of partial differential equations*, Science Advances, 3 (2017), e1602614.

[52] H. Schaeffer, *Learning partial differential equations via data discovery and sparse optimization*, Proc. A. 473 (2017), 20160446.

[53] H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher, *Sparse dynamics for partial differential equations*, Proc. Natl. Acad. Sci. USA, 110 (2013), pp. 6634–6639.

[54] H. Schaeffer and S. G. McCalla, *Sparse model selection via integral terms*, Phys. Rev. E, 96 (2017), 023302.

[55] H. Schaeffer, G. Tran, and R. Ward, *Extracting Sparse High-Dimensional Dynamics from Limited Data*, preprint, https://arxiv.org/abs/1707.08528, 2017.

[56] H. Schaeffer, G. Tran, and R. Ward, *Learning Dynamical Systems and Bifurcation Via Group Sparsity*, preprint, https://arxiv.org/abs/1709.01558, 2017.

[57] P. J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, J. Fluid Mech., 656 (2010), pp. 5–28.

[58] M. Schmidt and H. Lipson, *Distilling free-form natural laws from experimental data*, Science, 324 (2009), pp. 81–85.

[59] K. Taira and T. Colonius, *The immersed boundary method: A projection approach*, J. Comput. Phys., 225 (2007), pp. 2118–2137.

[60] N. Takeishi, Y. Kawahara, and T. Yairi, *Learning Koopman invariant subspaces for dynamic mode decomposition*, in Adv. Neural Inf. Process. Syst. 30, 2017, pp. 1130–1140.

[61] G. Tran and R. Ward, *Exact recovery of chaotic systems from highly corrupted data*, Multiscale Model. Simul., 15 (2017), pp. 1108–1129, https://doi.org/10.1137/16M1086637.

[62] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, *On Dynamic Mode Decomposition: Theory and Applications*, preprint, https://arxiv.org/abs/1312.0041, 2013.

[63] C. Wehmeyer and F. Noé, *Time-Lagged Autoencoders: Deep Learning of Slow Collective Variables for Molecular Kinetics*, preprint, https://arxiv.org/abs/1710.11239, 2017.

[64] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, *A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition*, J. Nonlinear Sci., 25 (2015), pp. 1307–1346.

[65] E. Yeung, S. Kundu, and N. Hodas, *Learning Deep Neural Network Representations for Koopman Operators of Nonlinear Dynamical Systems*, preprint, https://arxiv.org/abs/1708.06850, 2017.

[66] L. Zhang and H. Schaeffer, *On the Convergence of the SINDy Algorithm*, preprint, https://arxiv.org/abs/1805.06445, 2018.