Feature engineering and symbolic regression methods for detecting hidden physics from sparse sensor observation data

Cite as: Phys. Fluids 32, 015113 (2020); doi: 10.1063/1.5136351
Submitted: 12 November 2019 • Accepted: 16 December 2019 •
Published Online: 16 January 2020

Harsha Vaddireddy,[1] Adil Rasheed,[2] Anne E. Staples,[3] and Omer San[1,a)]

AFFILIATIONS

[1] School of Mechanical and Aerospace Engineering, Oklahoma State University, Stillwater, Oklahoma 74078, USA
[2] Department of Engineering Cybernetics, Norwegian University of Science and Technology, N-7465 Trondheim, Norway
[3] Department of Biomedical Engineering and Mechanics, Virginia Tech, Blacksburg, Virginia 24061, USA

[a)] Electronic mail: osan@okstate.edu

ABSTRACT

We put forth a modular approach for distilling hidden flow physics from discrete and sparse observations. To address functional expressiblity, a key limitation of the black-box machine learning methods, we have exploited the use of symbolic regression as a principle for identifying relations and operators that are related to the underlying processes. This approach combines evolutionary computation with feature engineering to provide a tool for discovering hidden parameterizations embedded in the trajectory of fluid flows in the Eulerian frame of reference. Our approach in this study mainly involves gene expression programming (GEP) and sequential threshold ridge regression (STRidge) algorithms. We demonstrate our results in three different applications: (i) equation discovery, (ii) truncation error analysis, and (iii) hidden physics discovery, for which we include both predicting unknown source terms from a set of sparse observations and discovering subgrid scale closure models. We illustrate that both GEP and STRidge algorithms are able to distill the Smagorinsky model from an array of tailored features in solving the Kraichnan turbulence problem. Our results demonstrate the huge potential of these techniques in complex physics problems, and reveal the importance of feature selection and feature engineering in model discovery approaches.

I. INTRODUCTION

Since the dawn of mathematical modeling of complex physical processes, scientists have been attempting to formulate predictive models to infer current and future states. These first principle models are generally conceptualized from conservation laws, sound physical arguments, and empirical heuristics drawn from either conducting experiments or hypotheses made by an insightful researcher. However, there are many complex systems (some being climate science, weather forecasting, and disease control modeling) with their governing equations known partially, and their hidden physics wait to be modeled. In the last decade, there have been rapid advances in machine learning[1,2] and easy access to rich data, thanks to the plummeting costs of sensors and high performance computers.

This paradigm shift in data driven techniques can be readily exploited to distill new or improved physical models for nonlinear dynamical systems. Extracting predictive models based on observing complex patterns from vast multimodal data can be loosely termed reverse engineering nature. This approach is not particularly new; for example, Kepler used planets' positional data to approximate their elliptic orbits. The reverse engineering approach is most appropriate in the modern age as we can leverage computers to directly infer physical laws from data collected from omnipresent sensors that otherwise might not be comprehensible to humans. Symbolic regression methods are a class of data driven algorithms that aim to find a mathematical model that can describe and predict hidden physics from observed input-response data. Some of the popular machine learning techniques that are adapted for the task of symbolic regression are neural networks,[3,4] compressive sensing/sparse optimization,[5,6] and evolutionary algorithms.[7,8]

Symbolic regression (SR) approaches based on evolutionary computation[7,9] are a class of frameworks that are capable of finding analytically tractable functions. Traditional deterministic regression algorithms assume a mathematical form and only find parameters that best fit the data. On the other hand, evolutionary SR approaches aim to simultaneously find parameters and also learn the best-fit functional form of the model from input-response data. Evolutionary algorithms search for functional abstractions with a preselected set of mathematical operators and operands while minimizing the error metrics. Furthermore, the optimal model is selected from Pareto front analysis with respect to minimizing accuracy vs model complexity. Genetic programming (GP)[7] is a popular choice leveraged by most of the SR frameworks. GP is an extended and improved version of a genetic algorithm (GA),[10,11] which is inspired by Darwin's theory of natural evolution. Seminal work was done in identifying hidden physical laws[12,13] from the input-output response using the GP approach. GP has been applied in the context of the SR approach in digital signal processing,[14] nonlinear system identification,[15] and aerodynamic parametric estimation.[16] Furthermore, GP as an SR tool was applied to identify complex closed-loop feedback control laws for turbulent separated flows.[17–20] Hidden physical laws of the evolution of a harmonic oscillator based on sensor measurements and the real world prediction of solar power production at a site were identified using GP as an SR approach.[21]

Improved versions of GP focus on better representation of the chromosome, which helps in the free evolution of the chromosome with constraints on the complexity of its growth, and faster searches for the best chromosome. Some of these improved versions of GP are gene expression programming (GEP),[8] parse matrix evolution (PME),[22] and linear genetic programming (LGP).[23] GEP takes advantage of the linear coded chromosome approach from GA and the parse tree evolution of GP to alleviate the disadvantages of both GA and GP. GEP was applied to diverse applications as an SR tool to recover nonlinear dynamical systems.[24–27] Recently, GEP was modified for tensor regression, termed as multi-GEP, and has been applied to recover functional models approximating the nonlinear behavior of the stress tensor in the Reynolds-averaged Navier-Stokes (RANS) equations.[28] Furthermore, this novel algorithm was extended to identify closure models in a combustion setting for large eddy simulations (LES).[29] Similarly, a new damping function has been discovered using the GEP algorithm for the hybrid RANS/LES methodology.[30] Generally, evolutionary based SR approaches can identify models with complex nonlinear compositions given enough computational time.

Compressive sensing (CS)[5,6] is predominately applied to signal processing in seeking the sparsest solution (i.e., a solution with the fewest number of features). Basis pursuit algorithms,[31] also identified as sparsity promoting optimization techniques,[32,33] play a fundamental role in CS. Ordinary least squares (OLS) optimization generally results in identifying models with large complexity, which are prone to overfitting. In sparse optimization, the OLS objective function is regularized by an additional constraint on the coefficient vector. This regularization helps in taming and shrinking large coefficients and thereby promoting sparsity in feature selection and avoiding overfitted solutions. The least absolute shrinkage and selection operator (LASSO)[32,34] is one of the most popular regularized least squares (LS) regression methods. In LASSO, an L_1 penalty is added to the LS objective function to recover sparse solutions.[35] In Bayesian terms, LASSO is a maximum *a posteriori* estimate (MAP) of LS with Laplacian priors. LASSO performs feature selection and simultaneously shrinks large coefficients, which may manifest to overfit the training data. Ridge regression[36] is another regularized variant where an L_2 penalty is added to the LS objective function. Ridge regression is also defined as a MAP estimate of LS with a Gaussian prior. The L_2 penalty helps in grouping multiple correlated basis functions and increases robustness and convergence stability for ill-conditioned systems. The elastic net approach[37,38] is a hybrid of the LASSO and ridge approaches combining the strengths of both algorithms.

Derived from these advances, a seminal work was done in employing sparse regression to identify the physical laws of nonlinear dynamical systems.[39] This work leverages the structure of sparse physical laws, i.e., only a few terms represent the dynamics. The authors have constructed a large feature library of potential basis functions that has the expressive power to define the dynamics and then seek to find a sparse feature set from this overdetermined system. To achieve this, a sequential threshold least squares (STLS) algorithm[39] has been introduced in such a way that a hard threshold on OLS coefficients is performed recursively to obtain sparse solutions. This algorithm was leveraged to form a framework called sparse identification of nonlinear dynamics (SINDy)[39] to extract the physical laws of nonlinear dynamical systems represented by ordinary differential equations (ODEs). This work re-envisioned model discovery from the perspective of sparse optimization and compressive sensing. The SINDy framework recovered various benchmark dynamical systems such as the chaotic Lorenz system and vortex shedding behind a cylinder. However, STLS regression finds it challenging to discover physical laws that are represented by spatiotemporal data or high-dimensional measurements and have highly correlated features in the basis library. This limitation was addressed using a regularized variant of STLS called the sequential threshold ridge regression (STRidge) algorithm.[40] This algorithm was intended to discover unknown governing equations that are represented by partial differential equations (PDEs), hence forming a framework termed as PDE-functional identification of nonlinear dynamics (PDE-FIND).[40] PDE-FIND was applied to recover canonical PDEs representing various nonlinear dynamics. This framework also performs reasonably well under the addition of noise to data and measurements. These sparse optimization frameworks generally have a free parameter associated with the regularization term that is tuned by the user to recover models ranging from highly complex to parsimonious.

In a similar direction of discovering governing equations using sparse regression techniques, L_1 regularized LS minimization was used to recover various nonlinear PDEs[41,42] using both high fidelity and distorted (noisy) data. Additionally, limited and distorted data samples were used to recover chaotic and high-dimensional nonlinear dynamical systems.[43,44] To automatically filter models with respect to model complexity (number of terms in the model) vs test accuracy, Bayes information criteria were used to rank the most informative models.[45] Furthermore, SINDy coupled with model information criteria is used to infer canonical biological models[46] and introduce a reduced order modeling (ROM) framework.[47] STRidge[40] was applied as a deterministic SR method to derive algebraic Reynolds-stress models for the RANS equations.[48] Recently, various sparse regression algorithms such as LASSO,[32] STRidge,[40] sparse relaxed regularized regression,[49] and the forward-backward greedy algorithm[50] were investigated to recover truncation error terms of various modified differential equations (MDEs) coming from canonical PDEs.[51] The frameworks discussed above assume that the structure of the model to be recovered is sparse in nature; that is, only a small number of terms govern the dynamics of the system. This assumption holds for many physical systems in science and engineering.

Fast function extraction (FFX)[52] is another deterministic SR approach based on pathwise regularized learning that is also called the elastic net algorithm.[37] The resulting models of FFX are selected through nondominated filtering concerning accuracy and model complexity, similar to evolutionary computations. FFX is influenced by both GP and CS to better distill physical models from data. FFX has been applied to recover hidden physical laws,[21] canonical governing equations,[53] and Reynolds stress models for the RANS equations.[54] Some other potential algorithms for deterministic SR are elite bases regression (EBR)[55] and prioritized grammar enumeration (PGE).[56] EBR uses only elite features in the search space selected by measuring correlation coefficients of features for the target model. PGE is another deterministic approach that aims for the substantial reduction of the search space where the genetic operators and random numbers from GP are replaced with grammar production rules and systematic choices.

An artificial neural network (ANN), also referred to as deep learning if multiple hidden layers are used, is a machine learning technique that transforms input features through nonlinear interactions and maps to output target features.[3,4] ANNs attracted attention in recent times due to their exemplary performance in modeling complex nonlinear interactions across a wide range of applications including image processing,[57] video classification,[58] and autonomous driving.[59] ANNs produce black-box models that are not quite open to physical inference or interpretability. Recently, physics-informed neural networks (PINNs)[60] were proposed in the flavor of SR that is capable of identifying scalar parameters for known physical models. PINNs use a loss function in symbolic form to help ANNs adhere to the physical structure of the system. Along similar directions, a Gaussian process regression (GPR) has been also investigated for the discovery of coefficients by recasting unknown coefficients as GPR kernel hyperparameters for various time dependent PDEs.[61,62] As a nonlinear system identification tool, the GPR approach provides a powerful framework to model dynamical systems.[63,64] State calibration with the four dimensional variational data assimilation (4D VAR)[65] and deep learning techniques such as long short-term memory (LSTM)[66] have been used for model identification in ROM settings. Convolutional neural networks (CNNs) are constructed to produce hidden physical laws from using the insight of establishing direct connections between filters and finite difference approximations of differential operators.[67,68] This approach has been demonstrated to discover underlying PDEs from learning the filters by minimizing the loss functions.[69,70]

In this paper, we have exploited the use of SR in three different applications: equation discovery, truncation error analysis, and hidden physics discovery. We demonstrate the use of the evolutionary computation algorithm, GEP, and the sparse regression algorithm, STRidge, in the context of the SR approach to discover various physical laws represented by linear and nonlinear PDEs from observing input-response data. We begin by demonstrating the identification of canonical linear and nonlinear PDEs that are up to the fifth order in space. For identifying one particular PDE, we demonstrate the natural feature extraction ability of GEP and the limits in the expressive and predictive power of using a feature library when dealing with STRidge in discovering physical laws. We then demonstrate the discovery of highly nonlinear truncation error terms of the Burgers MDE using both GEP and STRidge. We highlight that the analysis of truncation errors is very important in the implicit large eddy simulation as a way to determine inherent turbulence models. This analysis is usually very tedious and elaborate, and our study provides a clear example of how SR tools are suitable in such research. Following truncation error terms identification, we apply GEP using sparse data to recover hidden source terms represented by complex function compositions for a one-dimensional (1D) advection-diffusion process and a two-dimensional (2D) vortex-merger problem. Furthermore, both GEP and STRidge are used to demonstrate the identification of the eddy viscosity kernel along with its *ad hoc* modeling coefficient closing LES equations simulating the 2D decaying turbulence problem. An important result is the ability of the proposed methodology to distill the Smagorinsky model from an array of tailored features in solving the Kraichnan turbulence problem.

The rest of the paper is organized as follows: Section II gives a brief description of the GEP and STRidge algorithms. In Sec. III, GEP, and STRidge are tested on identifying different canonical PDEs. Section IV deals with the identification of nonlinear truncation terms of the Burgers MDE using both STRidge and GEP. In Sec. V we exploit GEP for identification of hidden source terms in a 1D advection-diffusion process and a 2D vortex-merger problem. We additionally demonstrate recovery of the eddy viscosity kernel and its modeling coefficient by both GEP and STRidge for closing the LES equations simulating the 2D decaying turbulence problem in the same section. Finally, Sec. VI draws our conclusions and highlights some ideas for future extensions of this work.

II. METHODOLOGY

We recover various physical models from data using two symbolic regression tools, namely, GEP, an evolutionary computing algorithm, and STRidge, which is a deterministic algorithm that draws its influences from compressive sensing and sparse optimization. We take the example of the equation discovery problem that is discussed in Sec. III to elaborate on the methodology of applying GEP and STRidge for recovering various physical models. We restrict the PDEs to be recovered to quadratic nonlinear and up to the fifth order in space. The general nonlinear PDE to be recovered is in the form of

$$u_t = \mathscr{F}(\sigma, u, u^2, u_x, u_x^2, uu_x, u_{2x}, \ldots, u_{5x}^2), \tag{1}$$

where subscripts denote the order of partial differentiation and σ is an arbitrary parameter. For example, consider the problem of identifying the viscous Burgers equation

$$u_t + uu_x = \nu u_{2x}, \tag{2}$$

where $u(x,t) \in \mathbb{R}^{m\times n}$ is the velocity field and ν is the kinematic viscosity. In our study, m is the number of time snapshots and n is the number of spatial locations. The solution field $u(x, t)$ is generally obtained by solving Eq. (2) analytically or numerically. The solution field might also be obtained from sensor measurements that can be arranged as shown follows:

$$\mathbf{u} = \overbrace{\begin{bmatrix} u_1(t_1) & u_2(t_1) & \dots & u_n(t_1) \\ u_1(t_2) & u_2(t_2) & \dots & u_n(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ u_1(t_m) & u_2(t_m) & \dots & u_n(t_m) \end{bmatrix}}^{\text{spatial locations}} \Bigg\} \text{time snapshots.} \tag{3}$$

For recovering PDEs, we need to construct a library of basis functions called feature library that contains higher order derivatives of the solution field $u(x, t)$. Higher order spatial and temporal partial derivative terms can be approximated using any numerical scheme once the recording of the discrete data set given by Eq. (3) is available. In our current setup, we use the leapfrog scheme for approximating the temporal derivatives and central difference schemes for spatial derivatives as follows:

$$\left.\begin{aligned} u_t &= \frac{u_j^{p+1} - u_j^{p-1}}{2dt}, \\ u_{2t} &= \frac{u_j^{p+1} - 2u_j^p + u_j^{p-1}}{dt^2}, \\ u_x &= \frac{u_{j+1}^p - u_{j-1}^p}{2dx}, \\ u_{2x} &= \frac{u_{j+1}^p - 2u_j^p + u_{j-1}^p}{dx^2}, \\ u_{3x} &= \frac{u_{j+2}^p - 2u_{j+1}^p + 2u_{j-1}^p - u_{j-2}^p}{2dx^3}, \\ u_{4x} &= \frac{u_{j+2}^p - 4u_{j+1}^p + 6u_j^p + -4u_{j-1}^p - u_{j-2}^p}{dx^4}, \\ u_{5x} &= \frac{u_{j+3}^p - 4u_{j+2}^p + 5u_{j+1}^p - 5u_{j-1}^p + 4u_{j-2}^p - u_{j-3}^p}{2dx^5}, \end{aligned}\right\} \tag{4}$$

where temporal and spatial steps are given by dt and dx, respectively. Within the expressions presented in Eq. (4), the spatial location is denoted using subscript index j, and the temporal instant using superscript index p.

We note that other approaches such as automatic differentiation or spectral differentiation for periodic domains can easily be adopted within our study. Both GEP and STRidge take the input library consisting of features (basis functions) that are built using Eqs. (2) and (3). This core library, used for the equation discovery problem in Sec. III, is shown as follows:

$$\left.\begin{aligned} \mathbf{V}(\mathbf{t}) &= \left[\mathbf{U}_t\right] \\ \widetilde{\mathbf{\Theta}}(\mathbf{U}) &= \left[\mathbf{U} \;\; \mathbf{U}_x \;\; \mathbf{U}_{2x} \;\; \mathbf{U}_{3x} \;\; \mathbf{U}_{4x} \;\; \mathbf{U}_{5x}\right] \end{aligned}\right\}. \tag{5}$$

The solution $u(x, t)$ and its spatial and temporal derivatives are arranged with size $m \cdot n \times 1$ in each column of Eq. (5). For example, the features (basis functions) $\mathbf{U}$ and $\mathbf{U}_{2x}$ are arranged as follows:

$$\mathbf{U} = \begin{bmatrix} u(x_0, t_0) \\ u(x_0, t_1) \\ | \\ u(x_j, t_p) \\ | \\ u(x_n, t_m) \end{bmatrix}, \quad \mathbf{U}_{2x} = \begin{bmatrix} u_{2x}(x_0, t_0) \\ u_{2x}(x_0, t_1) \\ | \\ u_{2x}(x_j, t_p) \\ | \\ u_{2x}(x_n, t_m) \end{bmatrix}, \tag{6}$$

where subscript j denotes the spatial location and subscript p denotes the time snapshot. The features (basis functions) in the core library $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ are expanded to include interacting features limited to quadratic nonlinearity and also a constant term. The final expanded library is given as

$$\mathbf{\Theta}(\mathbf{U}) = \left[\mathbf{1} \;\; \mathbf{U} \;\; \mathbf{U}^2 \;\; \mathbf{U}_x \;\; \mathbf{U}\mathbf{U}_x \;\; \mathbf{U}_x^2 \;\; \dots \;\; \mathbf{U}_{5x}^2\right], \tag{7}$$

where the size of the library is $\mathbf{\Theta}(\mathbf{U}) \in \mathbb{R}^{m \cdot n \times N_\beta}$ and N_β is the number of features (basis functions, i.e., $N_\beta = 28$ for our setup). For example, if we have 501 spatial points and 101 time snapshots with 28 bases, then $\mathbf{\Theta}(\mathbf{U})$ [Eq. (7)] contains 501 × 101 rows and 28 columns.

Note that the core feature library $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ in Eq. (5) is given as an input to GEP to recover PDEs and the algorithm extracts higher degree nonlinear interactions of core features in $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ automatically. However, for sparse optimization techniques such as STRidge, explicit input of all possible combinations of core features in Eq. (5) is required. Therefore, $\mathbf{\Theta}(\mathbf{U})$ in Eq. (7) forms the input to the STRidge algorithm for equation identification. This forms the fundamental difference in terms of feature building for both algorithms. Subsection II A gives a brief introduction to GEP and its specific hyperparameters that control the efficacy of the algorithm in identifying physical models from observing data. Furthermore, Subsection II B describes how to form linear system representations in terms of $\mathbf{V}(\mathbf{t})$ and $\mathbf{\Theta}(\mathbf{U})$ and briefly describes the STRidge optimization approach to identifying sparse features and thereby building parsimonious models using spatiotemporal data.

A. Gene expression programming

Gene expression programming (GEP)[8,71] is a genotype-phenotype evolutionary optimization algorithm which takes advantage of the simple chromosome representation of genetic algorithm (GA)[10] and the free expansion of complex chromosomes of genetic programming (GP).[7] As in most evolutionary algorithms, this technique also starts with generating initial random populations, iteratively selecting candidate solutions according to a fitness function, and improving candidate solutions by modifying through genetic variations using one or more genetic operators. The main difference between GP and GEP is how both techniques define the nature of their individuals. In GP, the individuals are nonlinear entities of different sizes and shapes represented as parse trees, and in GEP, the individuals are encoded as linear strings of fixed length called genomes and chromosomes, similar to GA representation of individuals, and later expressed as nonlinear entities of different size and shape called phenotypes or expression trees (ET). GEP is used for a very broad range of applications, but here it is introduced as a symbolic regression tool to extract constraint-free solutions from input-response data.

The arrangement of a typical gene/chromosome in GEP is shown in Fig. 1. The GEP gene is composed of head and tail regions as illustrated in Fig. 1. The head of a gene consists of both symbolic terms from functions (elements from a function set F) and terminals (elements from a terminal set T) whereas the tail consists of only terminals. The function set F may contain arithmetic mathematical operators (e.g., +, ×, −, /), nonlinear functions (e.g., sin, cos, tan, arctan, sqrt, and exp), or Boolean operators (e.g., NOT, NOR, OR, and AND) and the terminal set contains the symbolic variables. The

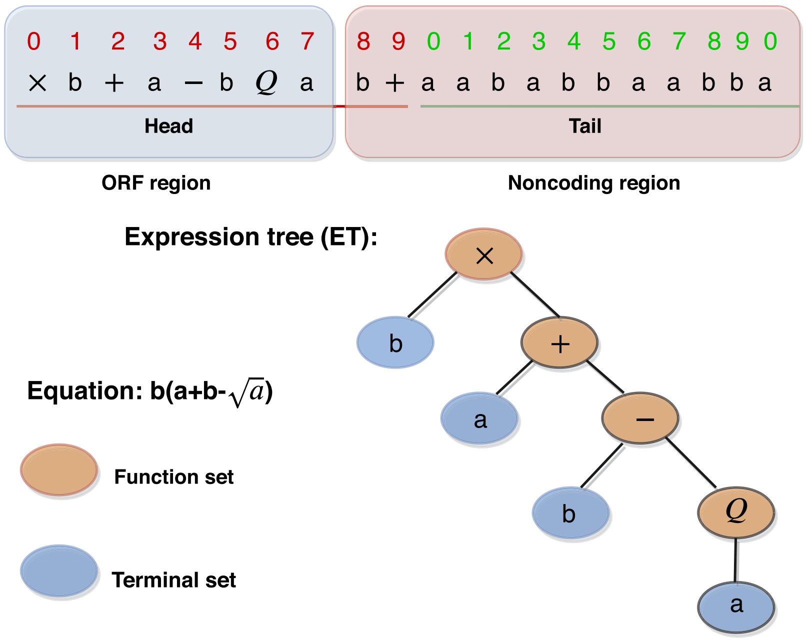

FIG. 1. ET of a gene/chromosome with its structure in GEP. Q represents the square root operator.

gene always starts with a randomly generated mathematical operator from the function set F. The head length is one of the important hyperparameters of GEP, and it is determined using trial and error as there is no definite method to assign it. Once the head length is determined, the size of the tail is computed as a function of the head length and the maximum arity of a mathematical operator in the function set F.[9] It can be calculated by the following equation:

$$\text{tail length} = \text{head length} \times (a_{max} - 1) + 1, \tag{8}$$

where a_{max} is the maximum argument of a function in F. The single gene can be extended to multigenic chromosomes where individual genes are linked using a linking function (e.g., +, ×, /, −). The general rule of thumb is to have a larger head and higher number of genes when dealing with complex problems.[9]

The structural organization of the GEP gene is arranged in terms of open reading frames (ORFs) inspired from biology where the coding sequence of a gene equivalent to an ORF begins with a start codon, continues with an amino acid codon, and ends with a termination codon. In contrast to a gene in biology, the start site is always the first position of a gene in GEP, but the termination point does not always coincide with the last position of a gene. These regions of the gene are termed noncoding regions downstream of the termination point. Only the ORF region is expressed in the ET and can be clearly seen in Fig. 1.

Even though the noncoding regions in GEP genes do not participate in the final solution, the power of GEP evolvability lies in this region. The syntactically correct genes in GEP evolve after modification through diverse genetic operators due to this region's chromosome. This is the paramount difference between GEP and GP implementations, where in the latter, many syntactically invalid individuals are produced and need to be discarded while evolving the solutions and additional special constraints are imposed on the depth/complexity of the candidate solution to be evolved to avoid bloating problem.[19]

Figure 2 displays the typical flowchart of the GEP algorithm. The process is described briefly below:

1. The optimization procedure starts with a random generation of chromosomes built upon combinations of functions and terminals. The size of the random population is a hyperparameter and the larger the population size, better the probability of finding the best candidate solution.
2. After the population is generated, the chromosomes are expressed as ETs, and then this population is converted to a numerical expression. This expression is then evaluated using a fitness function. In our setup, we employ the mean squared error between the best predicted model f^* and the true model f as the fitness function given by
$$MSE = \frac{1}{N}\sum_{l=1}^{N}\left(f^*_{(lk)} - f_{(l)}\right)^2, \tag{9}$$
where f^*_{lk} is the value predicted by the chromosome k for the fitness case l (out of N samples cases) and f_l is the true or measurement value for the l^{th} fitness case.
3. The termination criteria are checked after all fitness evaluations, to continue evolving or to save the best fitness chromosome as our final predicted model. In our current setup, we terminate after a specified number of generations.
4. The evolvability/reproduction of the chromosome through genetic operators, which is the core part of the GEP evolutionary algorithm, executes if termination criteria are not met. Before the genetic operations on the chromosome begin, the best chromosome according to the fitness function is cloned to

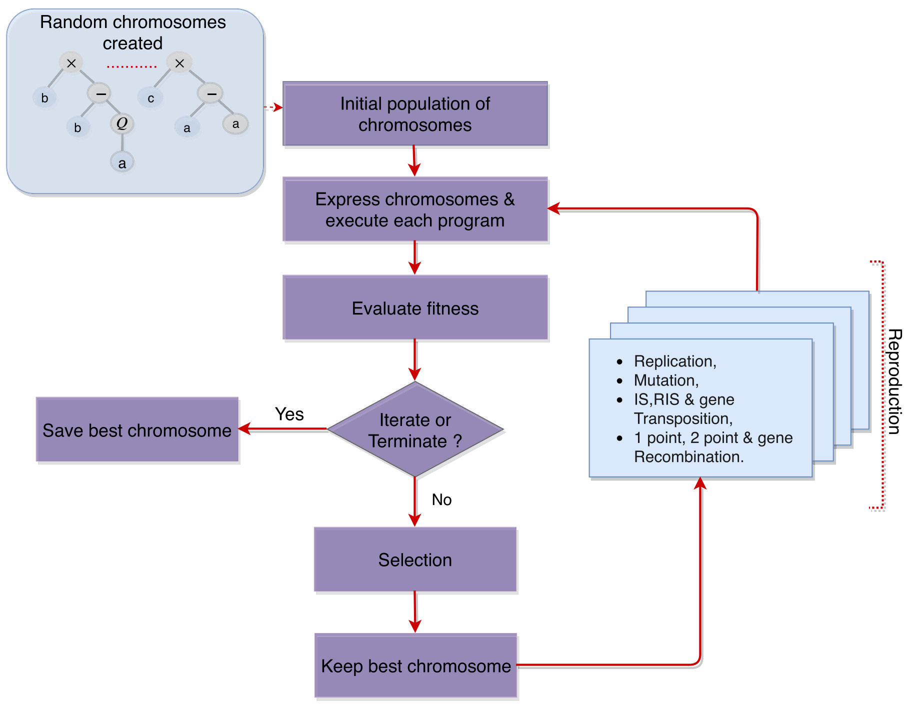
Random chromosomes created
Initial population of chromosomes
Express chromosomes & execute each program
Evaluate fitness
Iterate or Terminate ?
Yes
Save best chromosome
No
Selection
Keep best chromosome
Replication,
Mutation,
IS,RIS & gene Transposition,
1 point, 2 point & gene Recombination.
Reproduction

FIG. 2. Flowchart of the gene expression programming.

the next generations using a selection method. Popular selection methods include tournament selection with elitism and roulette-wheel selection with elitism. In our current setup, we use tournament selection with elitism.

5. The four genetic operators that introduce variation in populations are mutation, inversion, transposition, and recombination. The GEP transposition operator is applied to the elements of the chromosome in three ways: insertion sequence (IS), root insertion sequence (RIS), and gene insertion sequence, and similarly, three kinds of recombination are applied, namely, one point, two point, and gene recombination.
6. The process is continued until the termination criteria are met, which is the number of generations in our current setup.

Numerical constants occur in most mathematical models, and therefore, it is important to any symbolic regression tools to effectively integrate floating point constants in their optimization search. GP[7] handles numerical constants by introducing random numerical constants in a specified range to its parse trees. The random constants are moved around the parse trees using the crossover operator. GEP handles the creation of random numerical constants (RNCs) by using an extra terminal "?" and a separate domain Dc composed of symbols chosen to represent random numerical constants.[9] This Dc specific domain starts from the end of the tail of the gene.

For each gene, RNCs are generated during the creation of a random initial population and kept in an array. To maintain the genetic variations in the pool of RNCs, additional genetic operators are introduced to take effect on Dc specific regions. Hence, in addition to the usual genetic operators such as mutation, inversion, transposition, and recombination, the GEP-RNC algorithm has Dc specific inversion, transposition, and random constant mutation operators. Hence, with these modifications to the algorithm, an appropriate diversity of random constants can be generated and evolved through operations of genetic operators. The values for each genetic operator selected for this study are listed in Table I. These values are selected from various examples given by Ferreira[9] combined with the trial and error approach. Additionally, to simplify our study, we use the same parameters for all the test cases even though they may not be the best values for the test case under investigation.

Once decent values of genetic operators that can explore the search space are selected, the size of the head length, population, and the number of genes form the most important hyperparameters for GEP. Generally, larger head length and a greater number of genes are selected for identifying complex expressions. Larger population size helps in a diverse set of initial candidates, which may help GEP in finding the best chromosome in a lower number of generations. However, computational overhead increases with an increase in the size of the population. Furthermore, the best chromosome can be identified in fewer generations with the right selection of the linking function between genes. The GEP algorithm inherently performs poorly in predicting the numerical constants that are ubiquitous in physical laws. Hence, the GEP-RNC algorithm is used where a range of random constants are predefined to help GEP to find numerical constants. This also becomes important in GEP for identifying the underlying expression in fewer generations. Finally, we note that due to the heuristic nature of evolutionary algorithms, any other combinations of hyperparameters might work perfectly in identifying the symbolic expressions. In this study, we use geppy,[72] an open source library for symbolic regression using GEP, which is built as an extension to the distributed evolutionary algorithms in Python (DEAP)

TABLE I. GEP hyperparameters for various genetic operators selected for all the test cases in this study.

Hyperparameters	Value
Selection	Tournament selection
Mutation rate	0.05
Inversion	0.1
IS transposition rate	0.1
RIS transposition rate	0.1
Gene transposition rate	0.1
One point recombination	0.3
Two point recombination	0.2
Gene recombination	0.1
Dc specific mutation rate	0.05
Dc specific inversion rate	0.1
Dc specific transposition rate	0.1
Random constant mutation rate	0.02

package.[73] All codes used in this study are made available on Github (https://github.com/sayin/SR).

B. Sequential threshold ridge regression

Compressive sensing/sparse optimization[5,74] has been exploited for sparse feature selection from a large library of potential candidate features and recovering dynamical systems represented by ODEs and PDEs[39,40,45] in a highly efficient computational manner. In our setup, we use this STRidge[40] algorithm to recover various hidden physical models from observed data. In continuation with Sec. II where we define feature library $\mathbf{\Theta}(\mathbf{U})$ and target/output data $\mathbf{V}(\mathbf{t})$, this subsection briefly explains the formation of an overdetermined linear system for STRidge optimization to identify various physical models from data.

The Burgers PDE given in Eq. (2) or any other PDE under consideration can be written in the form of linear system representation in terms of $\mathbf{\Theta}(\mathbf{U})$ and $\mathbf{V}(\mathbf{t})$,

$$\mathbf{V}(t) = \mathbf{\Theta}(\mathbf{U}) \cdot \boldsymbol{\beta}, \tag{10}$$

where $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_{N_\beta}]$ is the coefficient vector of size $\mathbb{R}^{N_\beta}$, where N_β is the number of features (basis functions) in library $\mathbf{\Theta}(\mathbf{U})$. Note that $\mathbf{\Theta}(\mathbf{U})$ is an over-complete library (the number of measurements is greater than the number of features) and has rich feature (column) space to represent the dynamics under consideration. Thus, we form an overdetermined linear system in Eq. (10). The goal of STRidge is to find a sparse coefficient vector $\boldsymbol{\beta}$ that only consists of active features, which best represent the dynamics. The rest of the features are hard thresholded to zero. For example, in the Burgers equation given by Eq. (2), STRidge ideally has to find the coefficient vector $\boldsymbol{\beta}$ that corresponds to the features uu_x and u_{2x}, and simultaneously, it should set all other feature coefficients to zero.

The linear system defined in Eq. (10) can be solved for β using the ordinary least squares (OLS) problem. However, OLS minimization tries to form a functional relationship with all the features in $\mathbf{\Theta}(\mathbf{U})$ resulting in all nonzero values in the coefficient vector $\boldsymbol{\beta}$. Thus, solving Eq. (10) using OLS infers a radically complex functional form to represent the underlying PDE and generally results in overfitted models. Regularized least square minimization can be applied to constrain the coefficients and avoid overfitting. Hence, regularized LS optimization is preferred to identify the sparse features (basis functions) along with their coefficient estimation. Typical estimation of a sparse coefficient vector with P nonzero entries in $\boldsymbol{\beta}$ is shown in Fig. 3. A general sparse regression objective function to approximate the solution of the coefficient vector $\boldsymbol{\beta}$ is given by

$$\beta^* = \arg\min_\beta \|\mathbf{\Theta} \cdot \beta - \mathbf{V}(\mathbf{t})\|_2^2 + \lambda\|\beta\|_0, \tag{11}$$

where λ is the regularizing weight and $\|\beta\|_0$ corresponds to the L_0 penalty, which makes the problem *np*-hard. Hence, to arrive at the convex optimization problem of Eq. (12), L_1 and L_2 penalties are generally used to approximate the solution of the coefficient vector $\boldsymbol{\beta}$.

The addition of the L_1 penalty to the LS objective function, which corresponds to a maximum *a posteriori* estimate (MAP) of a Laplacian prior and is termed the least absolute shrinkage and selection operator (LASSO) in compressive sensing, is defined by

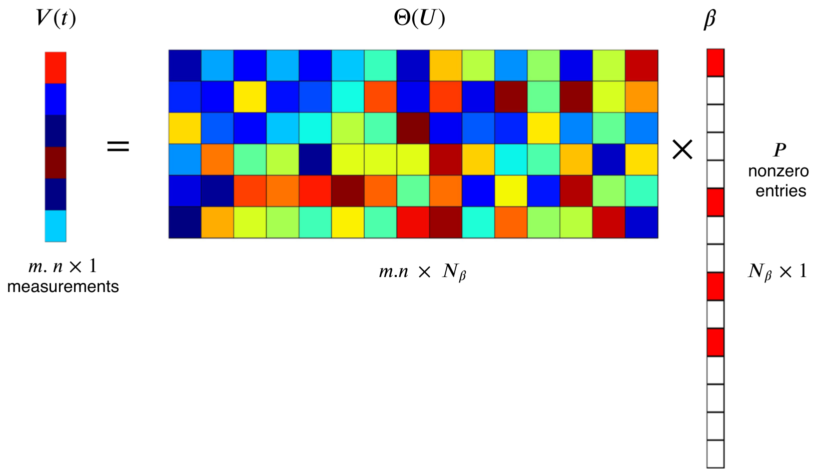

FIG. 3. Structure of compressive matrices with sparse nonzero entries in coefficient vector $\boldsymbol{\beta}$. Red boxes in $\boldsymbol{\beta}$ vector correspond to active feature coefficients and all other coefficients being set to zero.

ALGORITHM 1. STRidge(**Θ**, **V**(**t**), λ, *tol*, iters).[40]

```
Input: $\boldsymbol{\Theta}$, $\mathbf{V}(\mathbf{t})$, $\lambda$, $tol$, iters
Output: $\beta^*$
$\beta^* = \arg\min_\beta \|\boldsymbol{\Theta}\cdot\beta - \mathbf{V}(\mathbf{t})\|_2^2 + \lambda\|\beta\|_2^2$
large = $\{p : |\beta_p^*| \geq tol\}$
$\beta^*[\sim\text{large}] = 0$
$\beta^*[\text{large}]$ = STRidge($\boldsymbol{\Theta}$[:, large], $\mathbf{V}(\mathbf{t})$, $\lambda$, $tol$, iters − 1)
return $\beta^*$
```

$$\beta^* = \arg\min_\beta \|\boldsymbol{\Theta}\cdot\beta - \mathbf{V}(\mathbf{t})\|_2^2 + \lambda\|\beta\|_1. \tag{12}$$

However, the performance of LASSO deteriorates when the feature space is correlated.[40] The sequential threshold least squares (STLS) algorithm was proposed to identify dynamical systems represented by ODEs.[39] In STLS, a hard threshold is performed on least square estimates of regression coefficients and hard thresholding is recursively performed on remaining nonzero coefficients. However, the efficacy of STLS reduces when dealing with the identification of systems containing multiple correlated columns in **Θ**. Hence, L_2 regularized least squares termed ridge regression,[36] which corresponds to the maximum *a posteriori* estimate using a Gaussian prior, is proposed to handle the identification of PDEs. Ridge regression is defined by

$$\beta^* = \arg\min_\beta \|\boldsymbol{\Theta}\cdot\beta - \mathbf{V}(\mathbf{t})\|_2^2 + \lambda\|\beta\|_2 = (\boldsymbol{\Theta}^T\boldsymbol{\Theta} + \lambda^T I)\boldsymbol{\Theta}^T\mathbf{V}(\mathbf{t}). \tag{13}$$

Ridge regression is substituted for ordinary least squares in STLS and the resulting algorithm is termed sequential threshold ridge regression (STRidge).[40] The STRidge framework[40] is illustrated in Algorithm 1 for the sake of completeness. Note that, if λ = 0, STRidge becomes the STLS procedure. For more details on updating tolerance (*tol*) to perform hard thresholding in Algorithm 1, readers are encouraged to refer to the supplementary document of Rudy *et al.*[40]

We use the framework provided by Rudy *et al.*[40] in our current study. The hyperparameters in STRidge include the regularization weight λ and tolerance level *tol*, which are to be tuned to identify appropriate physical models. In the present study, the sensitivity of feature coefficients for various values of λ and the final value of λ where the best model is identified is shown. The following sections deal with various numerical experiments to test the GEP and STRidge frameworks.

III. EQUATION DISCOVERY

Partial differential equations (PDEs) play a prominent role in all branches of science and engineering. They are generally derived from conservation laws, sound physical arguments, and empirical heuristic from an insightful researcher. The recent explosion of machine learning algorithms provides new ways to identify hidden physical laws represented by PDEs using only data. In this section, we demonstrate the identification of various linear and nonlinear canonical PDEs using the GEP and STRidge algorithms from data alone. Analytical solutions of PDEs are used to form the data.

TABLE II. Summary of canonical PDEs selected for recovery.

PDE	Exact solution	Constant parameters	Discretization n (spatial) m (temporal)
Wave equation $u_t = -au_x$	$u(t,x) = \sin(2\pi(x - at))$	$a = 1.0$	$x \in [0, 1]$ $(n = 101)$, $t \in [0, 1]$ $(m = 101)$
Heat equation $u_t = -\alpha u_{2x}$	$u(t,x) = -\sin(x)\exp(-\alpha t)$	$\alpha = 1.0$	$x \in [-\pi, \pi]$ $(n = 201)$, $t \in [0, 1]$ $(m = 101)$
Burgers equation (i) $u_t = -uu_x + \nu u_{2x}$	$u(t,x) = \dfrac{x}{(t+1)\left(1 + (\sqrt{t+1})\exp\left(\frac{1}{16\nu}\frac{4x^2-t-1}{t+1}\right)\right)}$	$\nu = 0.01$	$x \in [0, 1]$ $(n = 101)$, $t \in [0, 1]$ $(m = 101)$
Burgers equation (ii) $u_t = -uu_x + \nu u_{2x}$	$u(t,x) = \dfrac{2\nu\pi\exp(-\pi^2\nu t)\sin(\pi x)}{a + \exp(-\pi^2\nu t)\cos(\pi x)}$	$\nu = 0.01$, $a = 5/4$	$x \in [0, 1]$ $(n = 101)$, $t \in [0, 100]$ $(m = 101)$
Korteweg-de Vries equation $u_t = -\alpha uu_x - \beta u_{3x}$	$u(t,x) = 12\left(\dfrac{4\cosh(2x-8t) + \cosh(4x-64t) + 3}{(3\cosh(x-28t) + \cosh(3x-36t))^2}\right)$	$\alpha = 6.0$, $\beta = 1.0$	$x \in [-10, 10]$ $(n = 501)$, $t \in [0, 1]$ $(m = 201)$
Kawahara equation $u_t = -uu_x - \alpha u_{3x} - \beta u_{5x}$	$u(t,x) = \dfrac{105}{169}\mathrm{sech}\left(\dfrac{1}{2\sqrt{13}}(x - at)\right)^4$	$\alpha = 1.0$, $\beta = 1.0$, $a = 36/169$	$x \in [-20, 20]$ $(n = 401)$, $t \in [0, 1]$ $(m = 101)$
Newell-Whitehead-Segel equation $u_t = \kappa u_{2x} + \alpha u - \beta u^q$	$u(t,x) = \dfrac{1}{\left(1 + \exp\left(\frac{x}{\sqrt{6}} - \frac{5t}{6}\right)\right)^2}$	$\kappa = 1.0$, $\alpha = 1.0$, $\beta = 1.0$, $q = 2$	$x \in [-40, 40]$ $(n = 401)$, $t \in [0, 2]$ $(m = 201)$
Sine-Gordon equation $u_{2t} = \kappa u_{2x} - \alpha\sin(u)$	$u(t,x) = 4\tan^{-1}(\mathrm{sech}(x)t)$	$\kappa = 1.0$, $\alpha = 1.0$	$x \in [-2, 2]$ $(n = 401)$, $t \in [0, 1]$ $(m = 101)$

TABLE III. GEP functional and terminal set used for equation discovery. "?" is a random constant.

Parameter	Value
Function set	+, −, ×, /, sin, cos
Terminal set	$\widetilde{\mathbf{\Theta}}(\mathbf{U})$, ?
Linking function	+

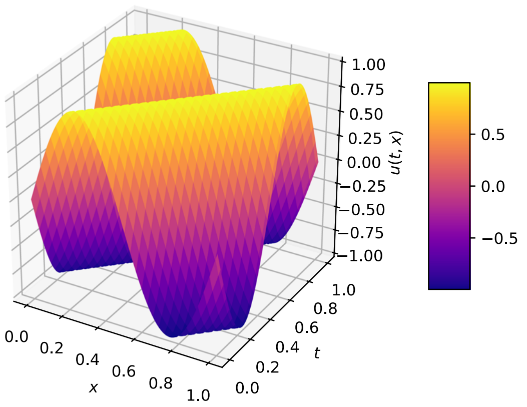

FIG. 4. Analytical solution of the wave equation.

Table II summarizes various PDEs along with their analytical solutions $u(t, x)$ and domain discretization. Building a feature library and corresponding response data to identify PDEs is discussed in detail in Sec. II.

We reiterate the methodology for PDE identification in Sec. II. The analytical solution $u(t, x)$ is solved at discrete spatial and temporal locations resulting from the discretization of the space and time domains as given in Table II. The discrete analytical solution is used as input data for calculating higher order spatial and temporal data using the finite difference approximations listed in Eq. (4). Furthermore, the feature library is built using discrete solution $u(t, x)$ and higher order derivative, which is discussed in Sec. II. As GEP is a natural feature extractor, core feature library $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ given in Eq. (5) is enough to form input data, i.e., GEP terminal set. Table III shows the function set and terminal set used for equation identification, and Table I lists the hyperparameter values for various genetic operators. However, the extended core feature library $\mathbf{\Theta}(\mathbf{U})$, which contains a higher degree of interactions of features, is used as input for STRidge as the expressive power of STRidge depends on exhaustive combinations of features in the input library. The temporal derivative of $u(t, x)$ is target or response data $\mathbf{V}(\mathbf{t})$ given in Eq. (5) for both GEP and STRidge.

A. Wave equation

Our first test case is the wave equation, which is a first order linear PDE. The PDE and its analytical solution are listed in Table II. We choose the constant wave speed $a = 1.0$ for propagation of the solution $u(t, x)$. Figure 4 shows the analytical solution $u(t, x)$ of the wave equation. The GEP hyperparameters used for identification of the wave equation are listed in Table IV. We use a smaller head length and a single gene for simple cases like a linear wave PDE. We note that any other combinations of hyperparameters may identify the underlying PDE. Figure 5 illustrates the identified PDE in the ET form. When the ET form is simplified, we can show that the resulting equation is the correct wave PDE, identified with its wave propagation speed parameter a.

The regularization weight (λ) in STRidge is swept across various values, as shown in Fig. 6. The yellow line in Fig. 6 represents the value of λ at which the best identified PDE is selected. Note that in this simple case, STRidge was able to find the wave equation for almost all the values of λ's that are selected. Table V shows the wave PDE recovered by both GEP and STRidge.

B. Heat equation

We use the heat equation, which is a second order linear PDE to test both SR approaches. The PDE and its analytical solution are listed in Table II. The physical parameter $\alpha = 1.0$ may represent thermal conductivity. Figure 7 displays the analytical solution $u(t, x)$ of the heat equation. Table IV lists the GEP hyperparameters used for the identification of the heat equation. Figure 8 shows the identified PDE in the form of an ET. When the ET form is simplified, we can show that the resulting model is the heat equation identified with its coefficient α.

The regularization weight (λ) in STRidge is swept across various values as shown in Fig. 9. The yellow line in Fig. 9 represents the value of λ selected at which STRidge finds the heat equation accurately. Note that STRidge was able to find the heat equation for low

TABLE IV. GEP hyperparameters selected for identification of various PDEs.

Hyperparameters	Wave equation	Heat equation	Burgers equation (i)	Burgers equation (ii)
Head length	2	2	4	2
Number of genes	1	2	1	2
Population size	25	25	20	50
Generations	100	100	500	500
Length of RNC array	10	10	30	5
Random constant minimum	−10	−1	−1	−1
Random constant maximum	10	1	1	1

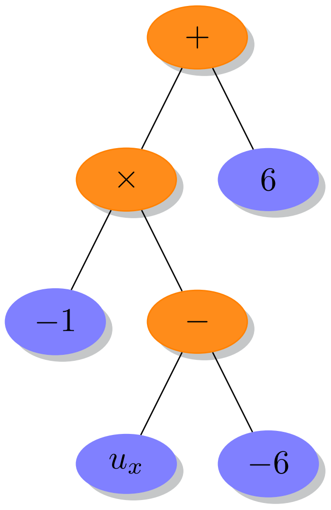

FIG. 5. Wave equation in terms of ET identified by GEP.

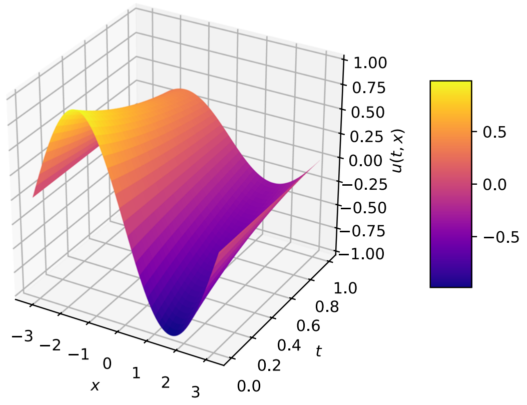

FIG. 7. Analytical solution of the heat equation.

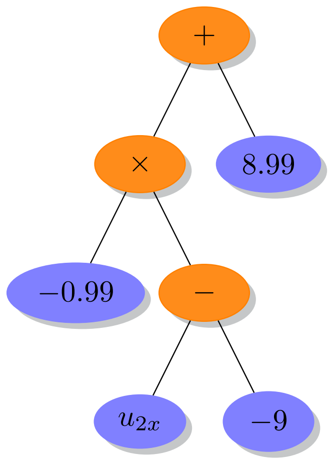

FIG. 8. Heat equation in terms of ET identified by GEP.

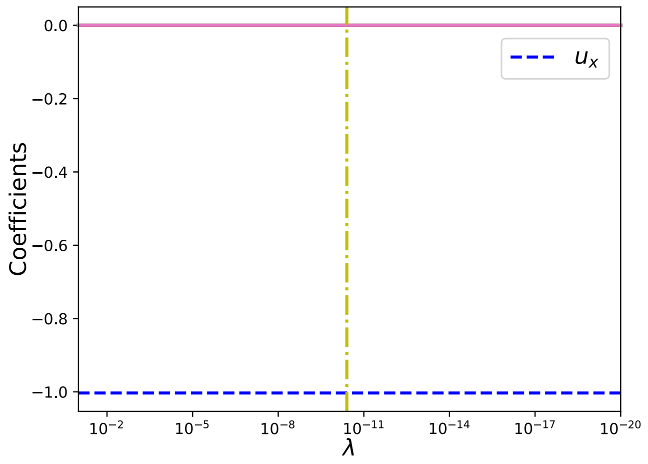

FIG. 6. STRidge coefficients as a function of regularization parameter λ for the wave equation.

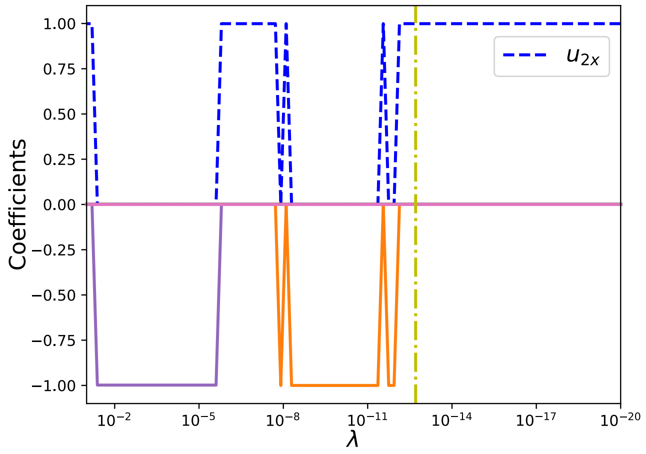

FIG. 9. STRidge coefficients as a function of regularization parameter λ for the heat equation.

values of the regularization weight λ as shown in Fig. 9. Table VI shows the heat equation recovered by both GEP and STRidge. STRidge was able to find a more accurate coefficient (α) value than GEP. Furthermore, a small constant value is also identified along with the heat equation by GEP.

TABLE V. Wave equation identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = -1.00\ u_x$	
GEP	$u_t = -1.00\ u_x$	1.72×10^{-28}
STRidge	$u_t = -1.00\ u_x$	9.01×10^{-29}

TABLE VI. Heat equation identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = -1.00\ u_{2x}$	
GEP	$u_t = -0.99\ u_{2x} - 5.33 \times 10^{-15}$	5.55×10^{-24}
STRidge	$u_t = -1.00\ u_{2x}$	4.09×10^{-30}

C. Burgers equation (i)

The Burgers equation is a fundamental nonlinear PDE occurring in various areas such as fluid mechanics, nonlinear acoustics, gas dynamics, and traffic flow.[75,76] The interest in the Burgers equation arises due to the nonlinear term uu_x and presents a challenge to both GEP and STRidge in the identification of its PDE using data. The form of the Burgers PDE and its analytical solution[77] is listed in Table II. The physical parameter $\nu = 0.01$ can be considered as the kinematic viscosity in fluid flows. Figure 10 shows the analytical solution $u(t, x)$ of the Burgers equation. Table IV shows the GEP hyperparameters used for the identification of the Burgers equation. Figure 11 shows the identified PDE in the form of the ET. When the ET form is simplified, we can show that the resulting model is the Burgers equation identified along with the coefficient of the nonlinear term and the kinematic viscosity. GEP uses more generations for identifying the Burgers PDE due to its nonlinear behavior along with the identification of feature interaction term uu_x.

The regularization weight (λ) in STRidge is swept across various values as shown in Fig. 12. The yellow line in Fig. 12 represents the value of λ at which the best identified PDE is selected. Note that the STRidge algorithm was able to find the Burgers equation at multiple values of regularization weights λ. Table VII shows the Burgers PDE recovered by both GEP and STRidge. There is an additional constant coefficient term recovered by GEP. Furthermore, the recovery of the nonlinear term using a limited set of input features shows the usefulness of GEP.

D. Burgers equation (ii)

The Burgers PDE with a different analytical solution is used to test the effectiveness of GEP and STRidge as the input data are changed but represented by the same physical law. The analytical solution of the Burgers equation (ii) is listed in Table II. The physical parameter $\nu = 0.01$ is used to generate the data. Figure 13 shows the alternate analytical solution $u(t, x)$ of the Burgers equation. Table IV shows the GEP hyperparameters used for the identification of the Burgers equation (ii). Figure 14 shows the identified PDE in the form of ET. When the ET form is simplified, we can show that the resulting model is the Burgers equation identified along with the coefficient of nonlinear term and kinematic viscosity. With an alternate solution, GEP uses a larger head length, more genes, and a larger population for identifying the same Burgers PDE.

The regularization weight (λ) in STRidge is swept across various values as shown in Fig. 15. The yellow line in Fig. 15 represents

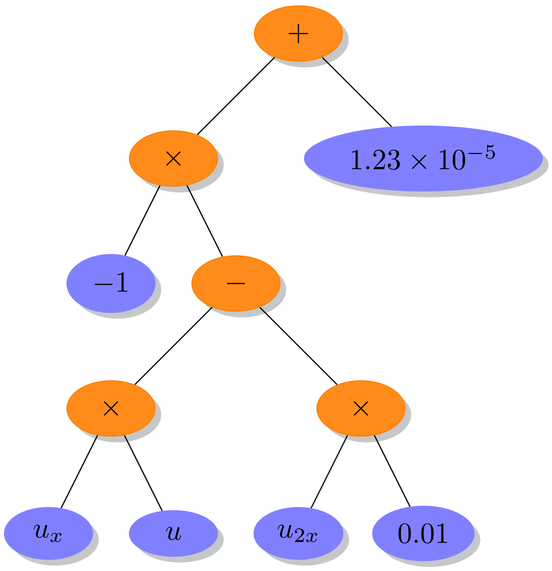

FIG. 11. Burgers equation (i) in terms of ET identified by GEP.

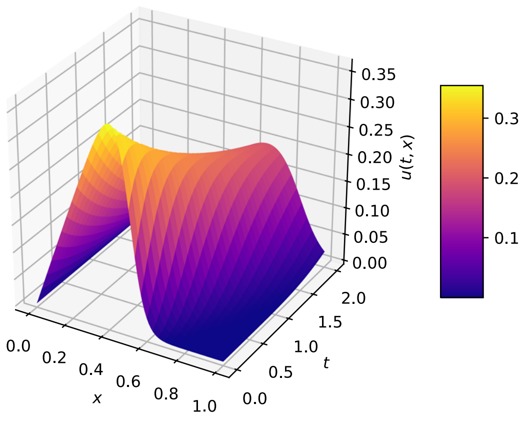

FIG. 10. Analytical solution of the Burgers equation (i).

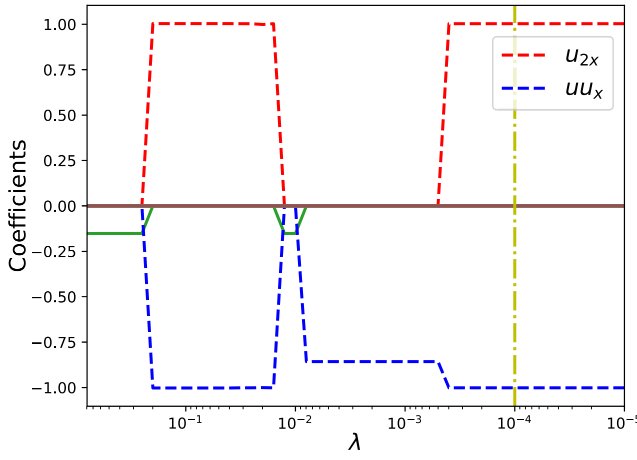

FIG. 12. STRidge coefficients as a function of regularization parameter λ for the Burgers equation (i).

TABLE VII. Burgers equation (i) identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = -uu_x + 0.01\ u_{2x}$	
GEP	$u_t = -uu_x + 0.01\ u_{2x} - 1.23 \times 10^{-5}$	6.10×10^{-8}
STRidge	$u_t = -uu_x + 0.01\ u_{2x}$	5.19×10^{-8}

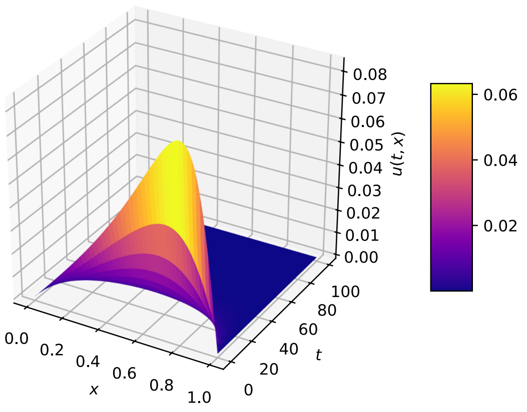

FIG. 13. Analytical solution of the Burgers equation (ii).

the value of λ at which the best identified PDE is selected. Note that STRidge was able to find the Burgers equation at various values of regularization weight λ. Table VIII shows the Burgers PDE recovered by both GEP and STRidge.

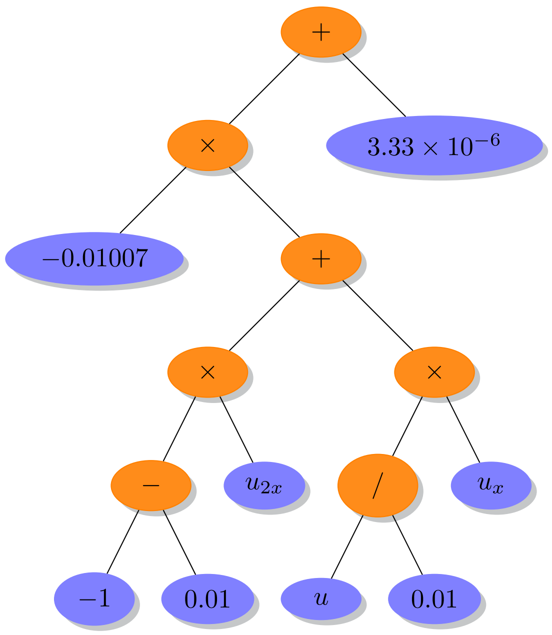

FIG. 14. Burgers equation (ii) in terms of ET identified by GEP.

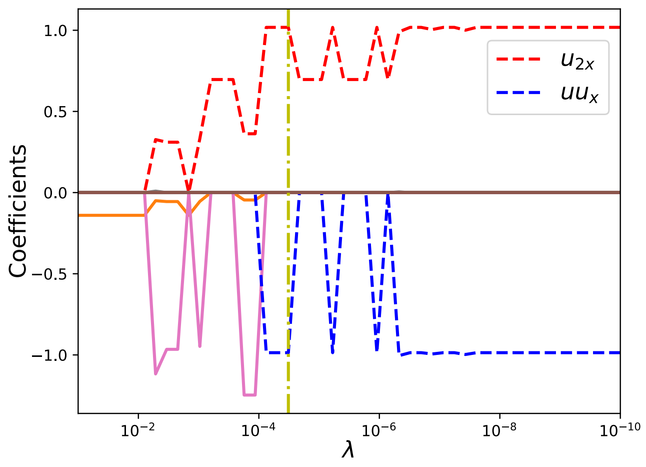

FIG. 15. STRidge coefficients as a function of regularization parameter λ for the Burgers equation (ii).

TABLE VIII. Burgers equation (ii) identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = -1.00\ uu_x + 0.01\ u_{2x}$	
GEP	$u_t = -1.01\ uu_x + 0.01\ u_{2x} - 3.33 \times 10^{-6}$	1.94×10^{-9}
STRidge	$u_t = -0.99\ uu_x + 0.01\ u_{2x}$	1.85×10^{-8}

E. Korteweg-de Vries (KdV) equation

Korteweg and de Vries derived the KdV equation to model Russell's phenomenon of solitons.[78,79] The KdV equation also appears when modeling the behavior of magnetohydrodynamic waves in warm plasmas, acoustic waves in an inharmonic crystal, and ion-acoustic waves.[80] Many different forms of the KdV equation are available in the literature, but we use the form given in Table II. Figure 16 shows the analytical solution $u(t, x)$ of the KdV equation.[81] It can be seen that this analytical solution refers to two solutions

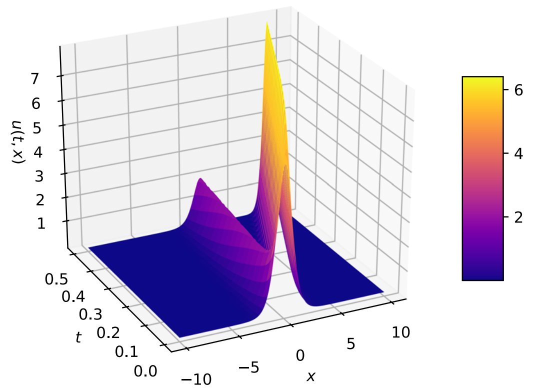

FIG. 16. Analytical solution of the KdV equation.

TABLE IX. GEP hyperparameters selected for identification of various PDEs.

Hyperparameters	KdV equation	Kawahara equation	NWS equation	Sine-Gordon equation
Head length	6	2	5	3
Number of genes	5	1	3	2
Population size	20	20	30	100
Generations	500	100	100	500
Length of RNC array	30	5	25	20
Random constant minimum	1	−1	−10	−10
Random constant maximum	10	1	10	10

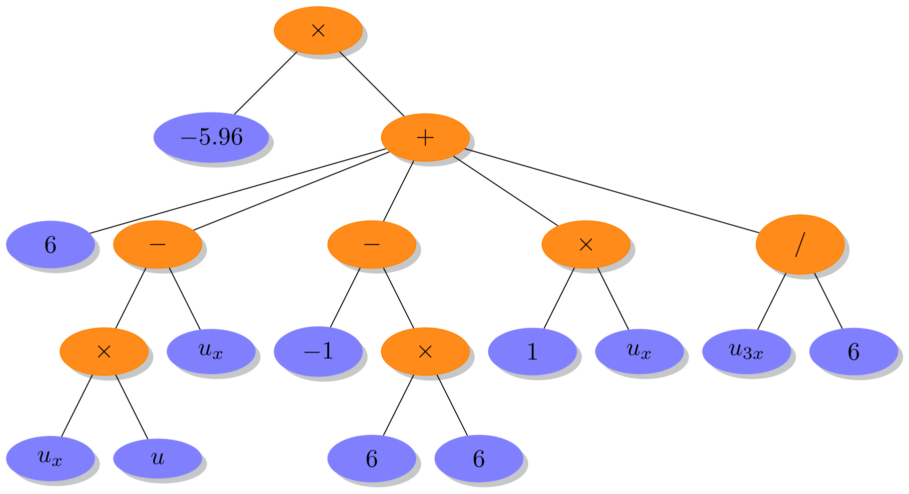

FIG. 17. KdV equation in terms of ET identified by GEP.

colliding together, which forms a good test case for SR techniques such as GEP and STRidge. Table IX shows the GEP hyperparameters used for the identification of the KdV equation. Due to the higher order nonlinear dynamics represented by a higher order PDE, GEP requires a large head length and genes compared to other test cases in equation discovery. Figure 17 shows the identified PDE in the form of the ET. When the ET form is simplified, we can observe that the resulting model is the KdV equation identified along with its coefficients.

The regularization weight (λ) in STRidge is swept across various values as shown in Fig. 18. The yellow line in Fig. 18 represents the value of λ at which the best identified PDE is selected. Note that STRidge was able to find the KdV equation at various values of the regularization weights (λ). Table X shows the KdV equation recovered by both GEP and STRidge. The physical model identified by STRidge is more accurate to the true PDE than the model identified by GEP.

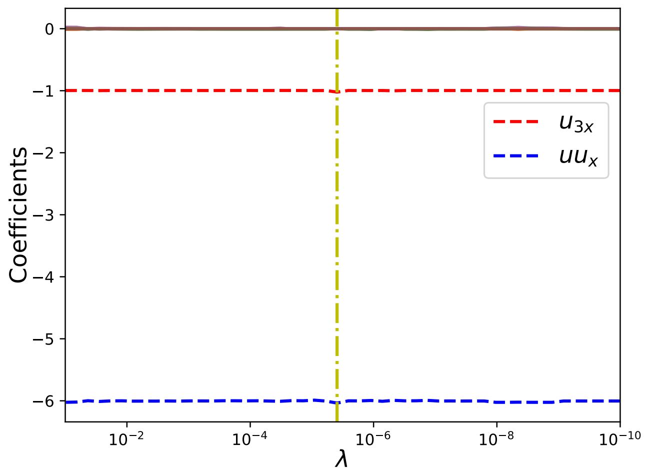

FIG. 18. STRidge coefficients as a function of regularization parameter λ for the KdV equation.

TABLE X. KdV equation identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = -6.00\ uu_x + 1.00\ u_{3x}$	
GEP	$u_t = -5.96\ uu_x + 0.99\ u_{3x} - 5.84 \times 10^{-4}$	0.29
STRidge	$u_t = -6.04\ uu_x + 1.02\ u_{3x}$	0.02

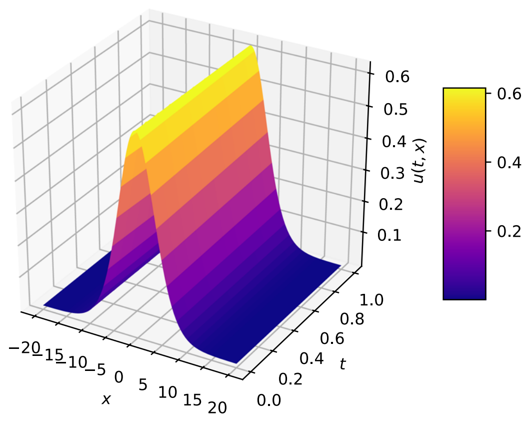

FIG. 19. Analytical solution of the Kawahara equation.

F. Kawahara equation

We consider the Kawahara equation, which is a fifth-order nonlinear PDE[82] shown in Table II. This equation is sometimes also referred to as a fifth-order KdV equation or singularly perturbed KdV equation. The fifth-order KdV equation is one of the most well known nonlinear evolution equation, which is used in the theory of magnetoacoustic waves in a plasma,[82] capillary-gravity waves,[83] and the theory of shallow water waves.[84] This test case is intended to test GEP and STRidge for identifying higher order derivatives from observing data. We use an analytical solution,[85] which is a traveling wave solution given in Table II. This analytical solution also satisfies the linear wave equation and hence both GEP and STRidge may recover a wave PDE (not shown here) as this is the sparsest model represented by observed data (Fig. 19). For simplifying the analysis, we remove the potential basis u_x from the feature library[42] $[\mathbf{\Theta}(\mathbf{U})]$ for STRidge and additionally include uu_x basis in core feature library $[\widetilde{\mathbf{\Theta}}(\mathbf{U})]$ for GEP.

Table IX shows the GEP hyperparameters used for the identification of the Kawahara equation. Due to simplifying the feature library, GEP requires smaller head length and single gene. Figure 20 shows the identified PDE in the form of ET. When the ET form is simplified, we can show that the resulting model is the Kawahara equation identified correctly along with its coefficients. For STRidge, the regularization weight (λ) is swept across various values as shown in Fig. 21. The yellow line in Fig. 21 represents the value of λ at which the best identified PDE is selected. Note that STRidge was able to find the Kawahara equation at various values of regularization weights (λ). Table XI shows the Kawahara equation identified by both GEP and STRidge.

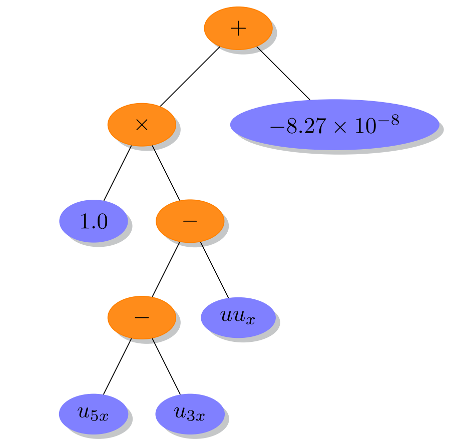

FIG. 20. Kawahara equation in terms of ET identified by GEP.

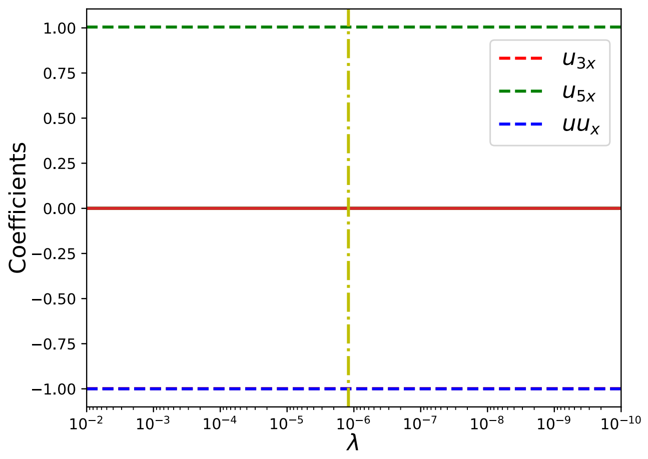

FIG. 21. STRidge coefficients as a function of regularization parameter λ for the Kawahara equation.

G. Newell-Whitehead-Segel equation

Newell-Whitehead-Segel (NWS) equation is a special case of the Nagumo equation.[86] Nagumo equation is a nonlinear reaction-diffusion equation that models pulse transmission line simulating a nerve axon,[87] population genetics,[88] and circuit theory.[89] The NWS equation and its analytical solution are shown in Table II. We use a traveling wave solution[90] that satisfies both wave and NWS equations (Fig. 22). We carry similar changes to the feature library that was applied to discovering the Kawahara equation.

Table IX shows the GEP hyperparameters used for the identification of the NWS equation. However, in contrast to identifying the Kawahara equation with smaller head length and single gene from simplifying the feature library, for NWS case, GEP requires larger head length and more genes for identifying PDE as shown in Table IX. This is due to the identification of nonlinear interaction

TABLE XI. Kawahara equation identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = -1.0\ uu_x - 1.00\ u_{3x} - 1.0\ u_{5x}$	
GEP	$u_t = -1.0\ uu_x - 1.00\ u_{3x} - 1.0\ u_{5x} - 8.27 \times 10^{-8}$	5.29×10^{-11}
STRidge	$u_t = -1.0\ uu_x - 0.99\ u_{3x} - 1.0\ u_{5x}$	1.35×10^{-12}

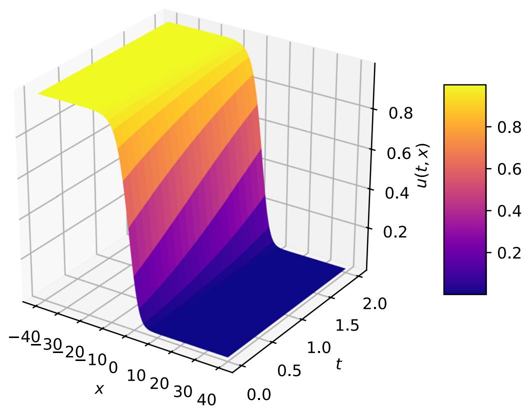

FIG. 22. Analytical solution of the NWS equation.

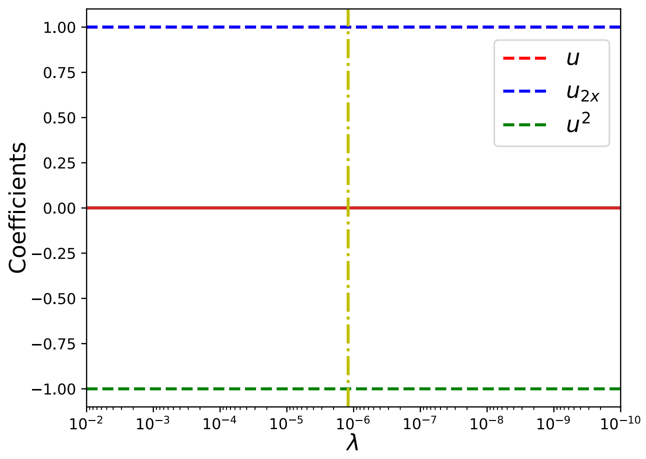

FIG. 24. STRidge coefficients as a function of regularization parameter λ for the NWS equation.

feature u^2 that appears in the NWS equation. Figure 23 shows the identified PDE in the form of ET. When the ET form is simplified, we can show that the resulting model is the NWS equation identified along with its coefficients. For STRidge, the regularization weight (λ) is swept across various values, as shown in Fig. 24. The yellow line in Fig. 24 represents the value of λ at which the best identified PDE is selected. Note that STRidge was able to find the NWS equation at various values of regularization weights (λ). Table XII shows the NWS equation identified by both GEP and STRidge.

H. Sine-Gordon equation

The Sine-Gordon equation is a nonlinear PDE that appears in propagating fluxions in Josephson junctions,[91] dislocation in crystals,[92] and nonlinear optics.[76] The Sine-Gordon equation has a sine term that needs to be identified by GEP and STRidge by observing data (Fig. 25). This test case is straightforward for GEP as the function set includes trigonometric operators that help to identify the equation. However, the application of STRidge is suitable if the feature library is limited to basic interactions and does not contain a basis with trigonometric dependencies. STRidge may recover infinite series approximations if higher degree basic feature interactions are included in the feature library.[39] Note that the output or target data for the Sine-Gordon equation consists of a second order temporal derivative of velocity field $u(t, x)$. Hence, $\mathbf{V(t)}$ consists of u_{2t} measurements instead of u_t.

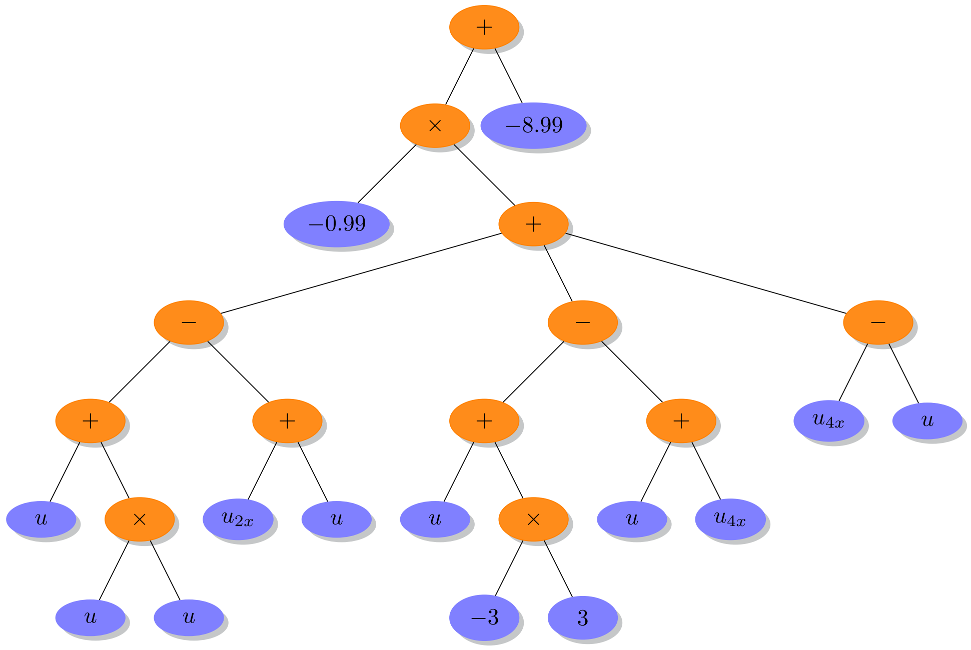

FIG. 23. NWS equation in terms of ET identified by GEP.

TABLE XII. NWS equation identified by GEP and STRidge.

	Recovered	Test error
True	$u_t = 1.00\ u_{2x} + 1.00\ u - 1.00\ u^2$	
GEP	$u_t = 0.99\ u_{2x} + 0.99\ u - 0.99\ u^2 - 8.27 \times 10^{-8}$	3.02×10^{-11}
STRidge	$u_t = 1.00\ u_{2x} + 0.99\ u - 0.99\ u^2$	1.36×10^{-11}

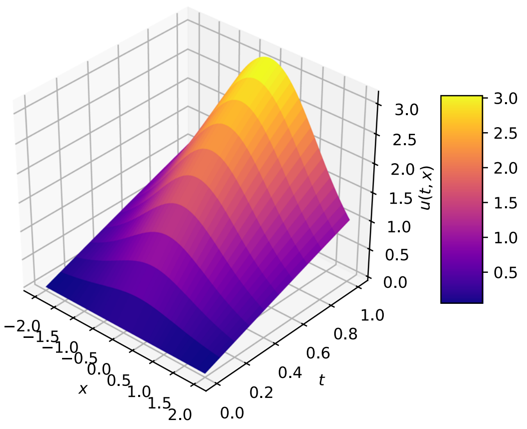

FIG. 25. Analytical solution of the Sine-Gordon equation.

Table IX shows the GEP hyperparameters used for identifying the Sine-Gordon equation. For our analysis, GEP was found to be the best model when the larger population size was used. Figure 26 shows the identified PDE in the form of ET. When the ET form is simplified, we can show that the resulting model is the Sine-Gordon equation identified along with its coefficients.

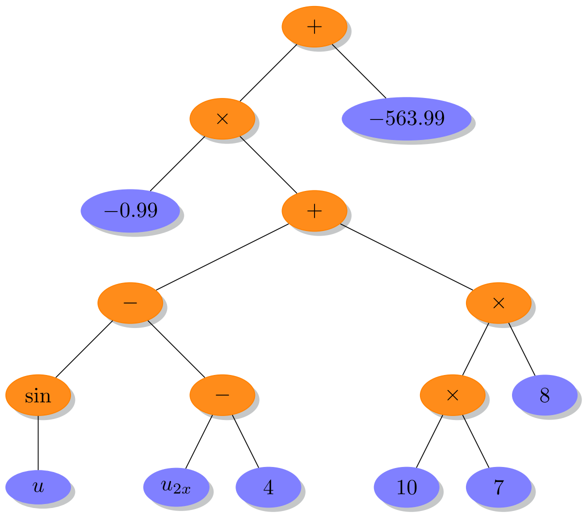

FIG. 26. Sine-Gordon equation in terms of ET identified by GEP.

TABLE XIII. Sine-Gordon equation identified by GEP.

	Recovered	Test error
True	$u_{2t} = 1.00\ u_{2x} - 1.00\ \sin(u)$	
GEP	$u_{2t} = 0.99\ u_{2x} - 0.99\ \sin(u) - 1.82 \times 10^{-5}$	1.57×10^{-4}

Table XIII shows the equation identified by GEP. This test case demonstrates the usefulness of GEP in identifying models with complex function composition and the limitations of the expressive and predictive power of the feature library in STRidge.

IV. TRUNCATION ERROR ANALYSIS

This section deals with constructing a modified differential equation (MDE) for the Burgers equation. We aim at demonstrating both GEP and STRidge techniques as SR tools in the identification of truncation errors resulting from an MDE of the Burgers nonlinear PDE. MDEs provide valuable insights into discretization schemes along with their temporal and spatial truncation errors. Initially, MDE analysis was developed to connect the stability nonlinear difference equations with the form of the truncation errors.[93] In continuation, the symbolic form of MDEs were developed and a key insight was proposed that only the first few terms of the MDE dominate the properties of the numerical discretization.[94] These developments of MDE analysis lead to increasing accuracy by eliminating leading order truncation error terms,[95] improving stability of schemes by adding artificial viscosity terms,[96] preserving symmetries,[97,98] and ultimately sparse identification of truncation errors.[51] Therefore, MDE analysis plays a prominent role in implicit large eddy simulations (ILES)[99] as truncation errors are shown to have inherent turbulence modeling capabilities.[100] Discretization schemes are tuned in the ILES approach to model the subgrid scale tensor using truncation errors. As the construction of MDEs becomes cumbersome and intractable for complex flow configurations, data driven SR tools such as GEP and STRidge can be exploited for the identification of MDEs by observing the data.

For demonstration purposes, we begin by constructing an MDE of the Burgers equation,

$$u_t + uu_x = \nu u_{2x}, \tag{14}$$

and discretizing Eq. (14) using first order schemes (i.e., forward in time and backward in space approximations for the spatial and temporal derivatives, respectively) and a second order accurate central difference approximation for the second order spatial derivatives. The resulting discretized Burgers PDE is shown below,

$$\frac{u_j^{p+1} - u_j^p}{dt} + u_j^p \frac{u_j^p - u_{j-1}^p}{dx} = \nu \frac{u_{j+1}^p - 2u_j^p + u_{j-1}^p}{dx^2}, \tag{15}$$

where the temporal and spatial steps are given by dt and dx, respectively. Within the expressions presented in Eq. (15), the spatial location is denoted using subscript index j and the temporal snapshot using superscript index p.

To derive the modified differential equation (MDE) of the Burgers PDE, we substitute the Taylor approximations for each term,

TABLE XIV. GEP functional and terminal sets used for truncation error term recovery. "?" is a random constant.

Parameter	Value
Function set	+, −, ×
Terminal set	$\widetilde{\boldsymbol{\Theta}}(\mathbf{U})$, ?
Linking function	+

TABLE XV. GEP hyperparameters selected for identification of truncation error terms of MDEs.

Hyperparameters	Burgers equation (i)	Burgers equation (ii)
Head length	8	8
Number of genes	5	4
Population size	70	70
Generations	1000	1000
Length of RNC array	20	20
Random constant minimum	1.0×10^{-6}	1.0×10^{-5}
Random constant maximum	0.01	0.01

$$\left.\begin{aligned} u_j^{p+1} &= u_j^p + dt(u_t)_j^p + \frac{dt^2}{2}(u_{2t})_j^p + \frac{dt^3}{6}(u_{3t})_j^p + \ldots \\ u_{j+1}^p &= u_j^p + dx((u_x))_j^p + \frac{dx^2}{2}(u_{2x})_j^p + \frac{dx^3}{6}(u_{3x})_j^p + \ldots \\ u_{j-1}^p &= u_j^p - dx(u_x)_j^p + \frac{dx^2}{2}(u_{2x})_j^p - \frac{dx^3}{6}(u_{3x})_j^p + \ldots \end{aligned}\right\}. \tag{16}$$

When we substitute these approximations into Eq. (15), we obtain the Burgers MDE as follows:

$$(u_t + uu_x - \nu u_{2x})_j^p = -R, \tag{17}$$

where R represents the truncation error terms of the Burgers MDE given as

$$R = \frac{dt}{2}(u_{2t})_j^p + \frac{dx}{2}(uu_x)_j^p - \frac{\nu dx^2}{12}(u_{4x})_j^p + O(dt^2, dx^4). \tag{18}$$

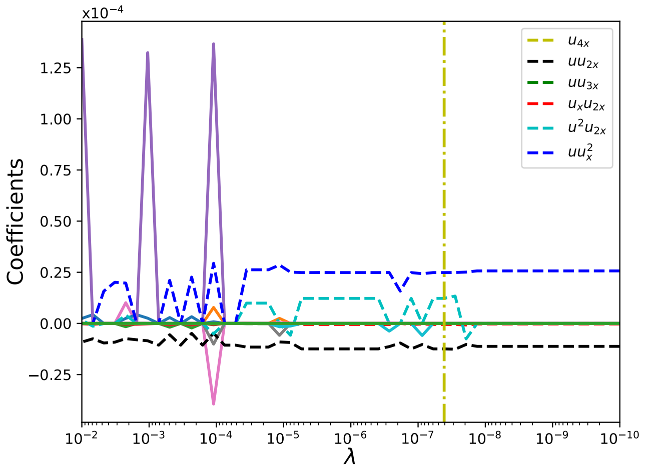

FIG. 28. STRidge coefficients as a function of regularization parameter λ for truncation error of the Burgers MDE (i).

Furthermore, the temporal derivative in Eq. (18) is substituted with the spatial derivatives resulting in

$$\begin{aligned} R = {}& dtuu_x^2 - dt\nu u_x u_{2x} - dt\nu u u_{3x} - \frac{dx}{2}uu_{2x} \\ &+ \frac{dt}{2}u^2 u_{2x} - \frac{\nu dx^2}{12}u_{4x} + O(dt^2, dx^4). \end{aligned} \tag{19}$$

The truncation error or residual of the discretized equation considering $u(t, x)$ as exact solution to the Burgers PDE is equal to the difference between the numerical scheme [Eq. (15)] and differential equation [Eq. (14)].[101] This results in discretized equation with the residual as shown as

$$u_j^{p+1} - u_j^p + u_j^p dt \frac{u_j^p - u_{j-1}^p}{dx} - \nu dt \frac{u_{j+1}^p - 2u_j^p + u_{j-1}^p}{dx^2} = Rdt. \tag{20}$$

We follow the same methodology for constructing the output data and feature library as discussed in Sec. II for the equation discovery. However, the output or target data $\mathbf{V}(\mathbf{t})$ are stored with the left hand side of Eq. (20) denoted from now as $\mathbf{U_{er}}$. The resulting output and

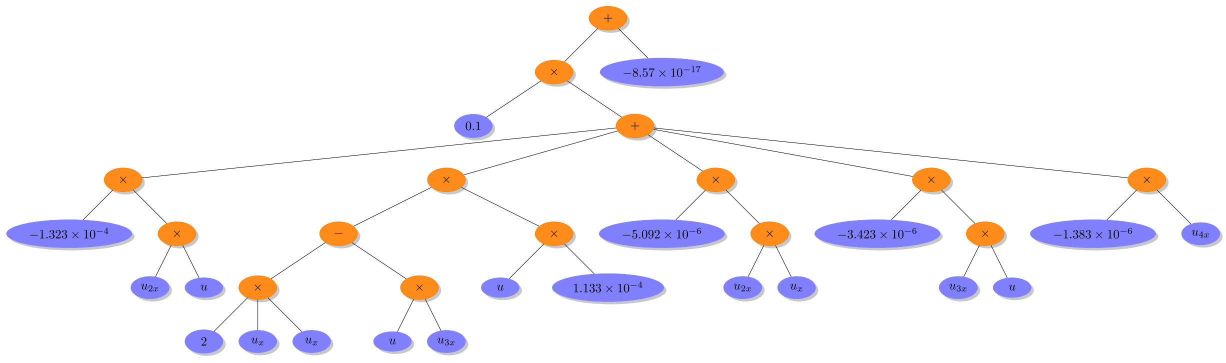

FIG. 27. Truncation error of the Burgers MDE using analytical solution of the Burgers equation (i) in terms of ET identified by GEP.

TABLE XVI. Identified truncation error terms along with coefficients for the Burgers MDE (i) by GEP and STRidge.

	True	GEP	Relative error (%)	STRidge	Relative error (%)
uu_x^2	2.5×10^{-5}	2.26×10^{-5}	9.6	2.48×10^{-5}	0.8
$u_x u_{2x}$	-5.0×10^{-7}	-5.09×10^{-7}	1.8	-5.02×10^{-7}	0.4
uu_{3x}	-2.5×10^{-7}	-3.42×10^{-7}	36.8	-2.29×10^{-7}	8.4
$u^2 u_{2x}$	1.25×10^{-5}	1.13×10^{-5}	9.6	1.22×10^{-5}	2.4
u_{4x}	1.25×10^{-9}	1.38×10^{-9}	10.4	1.16×10^{-9}	7.2
uu_{2x}	-1.25×10^{-5}	-1.33×10^{-5}	6.4	-1.24×10^{-5}	0.8

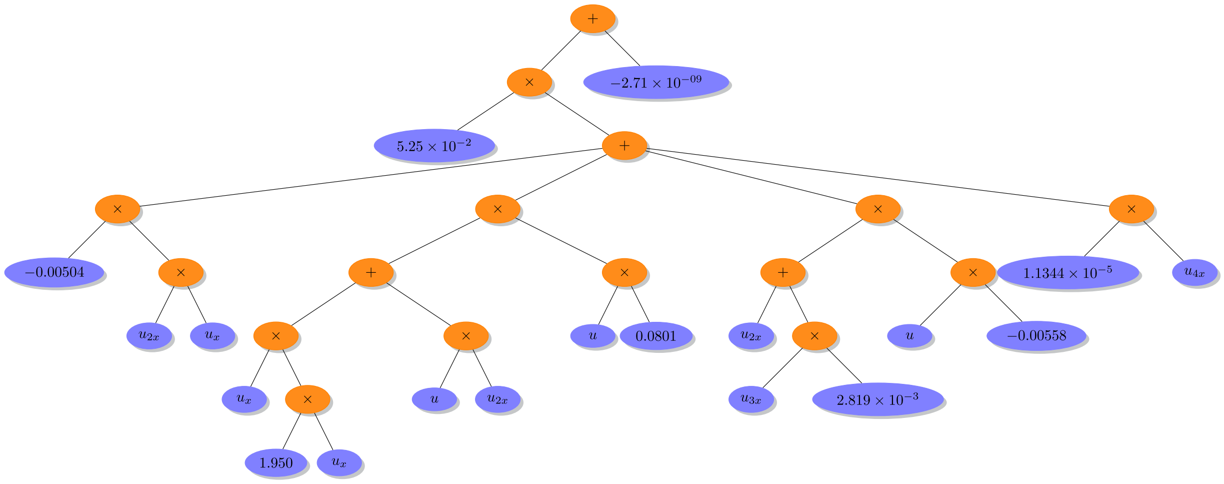

FIG. 29. Truncation error term of the Burgers MDE using analytical solution of the Burgers equation (ii) in terms of ET identified by GEP.

core feature library are shown as follows:

$$\left.\begin{aligned} \mathbf{V(t)} &= \left[\mathbf{U_{er}}\right] \\ \widetilde{\mathbf{\Theta}}(\mathbf{U}) &= \left[\mathbf{U}\ \ \mathbf{U}_x\ \ \mathbf{U}_{2x}\ \ \mathbf{U}_{3x}\ \ \mathbf{U}_{4x}\right] \end{aligned}\right\}. \tag{21}$$

The computation of the output data $\mathbf{V(t)}$ in Eq. (21) can be obtained using the analytical solution of the Burgers PDE. Furthermore, the derivatives in the core feature library $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ are calculated using the finite difference approximations given by Eq. (4). We use both analytical solutions listed in Table II for the Burgers equation (i) and the Burgers equation (ii) to test GEP and STRidge for recovering truncation error terms.

We use the same extended feature library $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ as input to STRidge given in Eq. (7), but without the fifth order derivative. However, we add an additional third degree interaction of features to $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ to recover the truncation error terms containing third degree nonlinearities. The extra nonlinear features that are added to $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ are given as follows:

$$\left[\mathbf{U^2U_x}\ \ \mathbf{U^2U_{2x}}\ \ \mathbf{U^2U_{3x}}\ \ \mathbf{U^2U_{4x}}\ \ \mathbf{UU_x^2}\ \ \mathbf{UU_xU_{2x}}\ \ \mathbf{UU_xU_{3x}}\ \ \mathbf{UU_xU_{4x}}\right].$$

In contrast, GEP uses the core feature $\widetilde{\mathbf{\Theta}}(\mathbf{U})$ as input as it identifies the higher order nonlinear feature interactions automatically. This test case shows the natural feature extraction capability of GEP and the need to modify the feature library to increase the expressive power of STRidge.

The functional and terminal sets used for the identification of the truncation error are listed in Table XIV. First, we test the recovery of truncation errors using the analytical solution of the Burgers equation (i) with the same spatial and temporal domain listed in

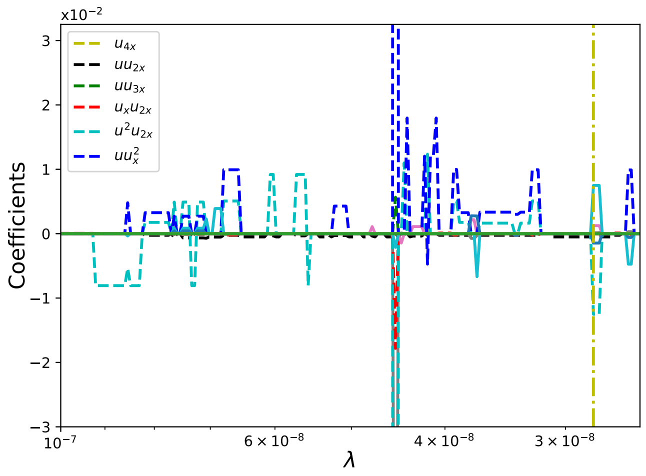

FIG. 30. STRidge coefficients as a function of regularization parameter λ for truncation error of the Burgers MDE (ii).

TABLE XVII. Identified truncation error terms along with coefficients for the Burgers MDE (ii) by GEP and STRidge.

	True	GEP	Relative error (%)	STRidge	Relative error (%)
uu_x^2	1.0×10^{-2}	8.19×10^{-3}	18.1	9.92×10^{-3}	0.8
$u_x u_{2x}$	-2.0×10^{-4}	-2.64×10^{-4}	32.0	-1.99×10^{-4}	0.5
uu_{3x}	-1.0×10^{-4}	-1.55×10^{-4}	55.0	-9.91×10^{-5}	0.9
$u^2 u_{2x}$	5.0×10^{-3}	4.21×10^{-3}	15.8	5.08×10^{-3}	1.6
u_{4x}	5.0×10^{-7}	5.65×10^{-7}	13.0	4.94×10^{-7}	1.2
uu_{2x}	-2.5×10^{-4}	-2.75×10^{-4}	10	-2.54×10^{-4}	1.6

Table II. However, we set spatial discretization to $dx = 0.005$ and temporal discretization to $dt = 0.005$ for storing the analytical solution $u(t, x)$. This test case needs a large population size, bigger head length, more genes, and more iterations as given in Table XV, as the truncation error terms consist of nonlinear combinations of features and the coefficients of error terms that are generally difficult for GEP to identify. Figure 27 shows the ET form of the identified truncation error terms. The regularization weight λ for STRidge is swept across a range of values, as shown in Fig. 28. The vertical yellow line in Fig. 28 is the value of λ where STRidge identifies the best truncation error model. Table XVI shows the recovered error terms by GEP and STRidge along with their coefficients. Both GEP and STRidge perform well in identifying the nonlinear spatial error terms with STRidge predicting the error coefficient better than GEP.

In the second case, we test the recovery of truncation errors using an analytical solution of the Burgers equation (ii) with the same the spatial and the temporal domain listed in Table II. We select spatial discretization $dx = 0.005$ and the temporal discretization $dt = 0.1$ for propagating the analytical solution $u(t, x)$. This test case also follows the previous case where a large population size, bigger head length, more genes, and more iterations are needed, as shown in Table XV. Figure 29 shows the ET form of the identified truncation error terms. The regularization weight λ for STRidge is swept across a range of values as shown in Fig. 30. In this test case, the coefficients change rapidly with respect to λ, and the best model is recovered only at the value of λ shown by the vertical yellow line in Fig. 30. Table XVII shows the recovered error terms by GEP and STRidge along with their coefficients. Similar to the previous test case, STRidge predicts the truncation error coefficients better than GEP.

V. HIDDEN PHYSICS DISCOVERY

In this section, we demonstrate the identification of hidden physical laws from sparse data mimicking sensor measurements using GEP and STRidge. Furthermore, we demonstrate the usefulness of GEP as a natural feature extractor that is capable of identifying complex functional compositions. However, STRidge in its current form is limited by its expressive power, which depends on its input feature library. Many governing equations of complex systems in the modern world are only partially known or in some cases still awaiting first principle equations. For example, atmospheric radiation models or chemical reaction models might not be fully known in governing equations of environmental systems.[102,103] These unknown models are generally manifested in the right hand side of the known governing equations (i.e., dynamical core) behaving as a source or forcing term. The recent explosion of rapid data gathering using smart sensors[104] has enabled researchers to collect data that represent the true physics of complex systems but their governing equations are only known partially. To this end, SR approaches might be able to recover these unknown physical models when exposed to data representing full physics.

To demonstrate the proof of concept for identification of unknown physics, we formulate a 1D advection-diffusion PDE and a 2D vortex-merger problem. These problems include a source term that represents the hidden physical law. We generate synthetic data that contains true physics and substitute this data set into the known governing equations. This results in an unknown physical model left as a residual that must be recovered by GEP when exposed to a target or output containing the known part of the underlying processes. Furthermore, both GEP and STRidge are tested to recover eddy viscosity kernels for the 2D Kraichnan turbulence problem. These eddy viscosity kernels are manifested as source terms in the LES equations that model unresolved small scales. Additionally, the value of the *ad hoc* free modeling parameter that controls the dissipation in eddy viscosity models is also recovered using GEP and STRidge.

A. 1D advection-diffusion PDE

In the first test case, we consider a 1D nonhomogeneous advection-diffusion PDE, which appears in many areas such as fluid dynamics,[105] heat transfer,[106] and mass transfer.[107] The nonhomogeneous PDE takes the form

$$u_t + cu_x = \alpha u_{2x} + S(t, x), \tag{22}$$

where $c = \frac{1}{3\pi}$, $\alpha = \frac{1}{4}$ and $S(t, x)$ is the source term.

We use an analytical solution $u(t, x)$ for solving Eq. (22). The exact solution for this nonhomogeneous PDE is as follows:

$$u(t, x) = \exp\left(\frac{\pi^2 t}{4}\right)\sin(\pi x), \tag{23}$$

where the spatial domain $x \in [0, 1]$ and the temporal domain $t \in [0, 1]$. We discretize the space and time domains with $n = 501$ and $m = 1001$, respectively. Figure 31 shows the corresponding analytical solution $u(t, x)$.

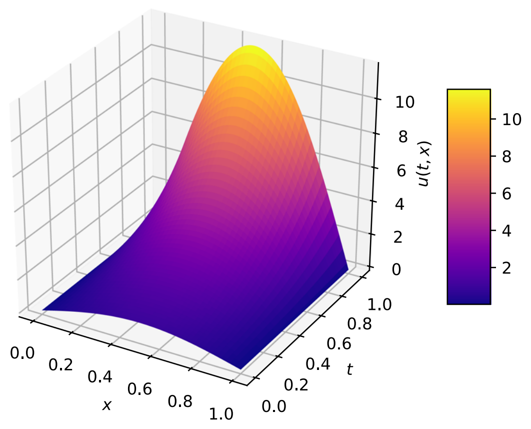

FIG. 31. Solution to the 1D advection-diffusion PDE with source term.

The source term $S(t,x)$, which satisfies Eq. (22) for the analytical solution provided by Eq. (23), is given as

$$S(t,x) = \frac{\pi^2}{2}\exp\left(\frac{\pi^2 t}{4}\right)\sin(\pi x) + \frac{1}{3}\exp\left(\frac{\pi^2 t}{4}\right)\cos(\pi x). \tag{24}$$

Our goal is to recover this hidden source term once the solution $u(t,x)$ is available either by solving the analytical equation given by Eq. (23) or by sensor measurements in real world applications. Furthermore, we select 64 random sparse spatial locations to mimic experimental data collection. After the solution $u(t, x)$ is stored at selected sparse spatial locations, we follow the same procedure for constructing output data and feature building as discussed in Sec. II. The corresponding output data **V** and feature library for recovering source term using GEP are given as

$$\left.\begin{aligned} \mathbf{V} &= \left[\mathbf{U_t} + c\mathbf{U_x} - \alpha\mathbf{U_{2x}}\right] \\ \widetilde{\mathbf{\Theta}} &= \left[\mathbf{x}\ \ \mathbf{t}\right] \end{aligned}\right\}. \tag{25}$$

The derivatives in the output data **V** are calculated using Eq. (4). Hence, to calculate spatial derivatives, we also store additional stencil data $u(t, x)$ around the randomly selected sparse locations $(u)_j^p$ i.e., $(u)_{j+1}^p$, $(u)_{j-1}^p$. Table XVIII gives the functional and terminal sets used by GEP to recover the source term $S(t, x)$ given in Eq. (24).

Table XIX lists the hyperparameters used by GEP for recovering source term of the 1D advection-diffusion equation. As the hidden physical law given in Eq. (24) consists of complex functional compositions, GEP requires a larger head length, and more generations are required by GEP for identification. The ET form of the source term $S(t, x)$ found by GEP is shown in Fig. 32. The identified source term after simplifying the ET form found by GEP is listed in Table XX. GEP was able to identify the source term $S(t, x)$ given in Eq. (24) from sparse data.

B. 2D vortex-merger problem

In this section, we demonstrate the recovery of a hidden physical law from the data generated by solving the vortex-merger problem with source terms. The initial two vortices merge to form a single vortex when they are located within a certain critical distance from each other. This two-dimensional process is one of the fundamental processes of fluid motion and it plays a key role in a variety of simulations, such as decaying two-dimensional turbulence[108,109] and mixing layers.[110] This phenomenon also occurs in other fields such as astrophysics, meteorology, and geophysics.[111] The vortex-merger problem is simulated using the 2D incompressible Navier-Stokes equations in the domain with periodic boundary conditions.

We specifically solve the system of PDEs in the vorticity-streamfunction formulation. This system of PDEs contains the vorticity transport equation derived by taking the curl of the 2D incompressible Navier-Stokes equations and the Poisson equation representing the kinematic relationship between the streamfunction (ψ) and vorticity (ω). The resulting vorticity-streamfunction formulation with a source term is given as

$$\left.\begin{aligned} \omega_t + J(\omega,\psi) &= \frac{1}{\text{Re}}\nabla^2\omega + S(t,x,y) \\ \nabla^2\psi &= -\omega \end{aligned}\right\} \tag{26}$$

where the Reynolds number is set to Re = 2000. In Eq. (26), $S(t, x, y)$ is the source term and $J(\omega, \psi)$ is the Jacobian term given as $\psi_y\omega_x - \psi_x\omega_y$. We use the Cartesian domain $(x, y) \in [0, 2\pi] \times [0, 2\pi]$ with a spatial resolution of 128 × 128. The initial vorticity field consisting of a corotating vortex pair is generated using the superposition of two Gaussian-distributed vortices given by

$$\begin{aligned}\omega(0,x,y) = {}& \Gamma_1\exp\left(-\rho\left[(x-x_1)^2 + (y-y_1)^2\right]\right) \\ &+ \Gamma_2\exp\left(-\rho\left[(x-x_2)^2 + (y-y_2)^2\right]\right),\end{aligned} \tag{27}$$

TABLE XVIII. GEP functional and terminal sets used for source term identification. "?" is a random constant.

Parameter	Value
Function set	+, −, ×, /, exp, sin, cos
Terminal set	$\widetilde{\mathbf{\Theta}}$, ?
Linking function	+

TABLE XIX. GEP hyperparameters selected for identifying source terms for the 1D advection-diffusion and the 2D vortex-merger problem.

Hyperparameters	1D advection-diffusion equation	2D vortex-merger problem
Head length	6	5
Number of genes	2	3
Population size	50	50
Generations	1000	500
Length of RNC array	5	8
Random constant minimum	$\frac{\pi}{4}$	$-\pi$
Random constant maximum	π	π

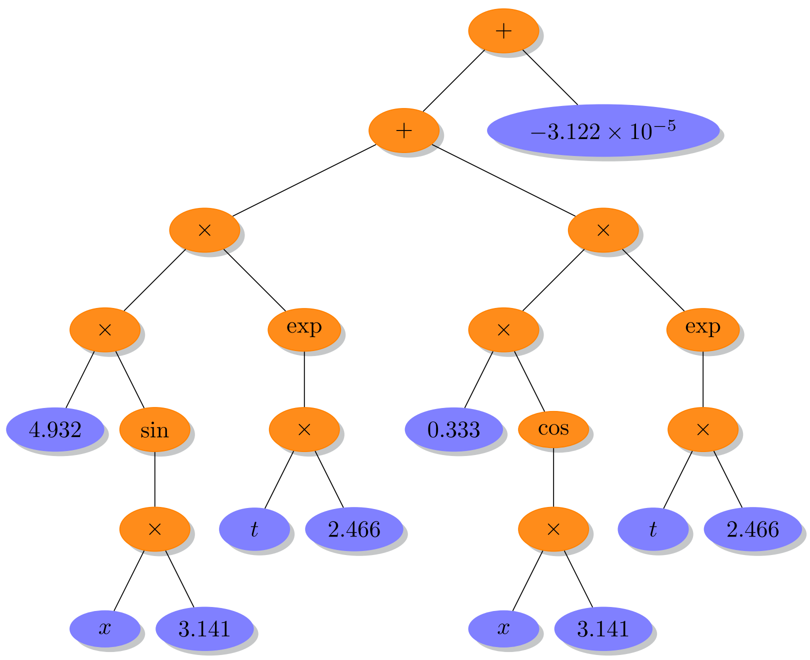

FIG. 32. Hidden source term of the 1D advection-diffusion PDE in terms of ET identified by GEP.

where the circulation $\Gamma_1 = \Gamma_2 = 1$, the interacting constant $\rho = \pi$, and the initial vortex centers are located near each other with coordinates $(x_1, y_1) = (\frac{3\pi}{4}, \pi)$ and $(x_2, y_2) = (\frac{5\pi}{4}, \pi)$. We choose the source term $S(t, x)$ as

$$S(t,x,y) = \Gamma_0 \sin(x)\cos(y)\exp\left(\frac{-4\pi^2}{\text{Re}}t\right), \tag{28}$$

where the magnitude of the source term is set to $\Gamma_0 = 0.01$.

The vorticity field ω and streamfunction field ψ are obtained by solving Eq. (26) numerically. We use a third-order Runge-Kutta scheme for the time integration, and a second order Arakawa scheme[112] for the discretization of the Jacobian term $J(\omega, \psi)$. As we have a periodic domain, we use a fast Fourier transform (FFT) for solving the Poisson equation in Eq. (26) to obtain the streamfunction at every time step. Numerical details for solving the vortex-merger problem can be found in San *et al.*[110,113] We integrate the solution from time $t = 0$ to $t = 20$ with a temporal step $dt = 0.01$.

Figure 33 shows the merging process of two vortices at the initial and final times. The red markers in Fig. 33 are 64 randomly selected sparse locations to collect both streamfunction ψ and vorticity ω data. Once the streamfunction and vorticity data at sparse locations are available, we can construct the target data $\mathbf{V}$ and feature library $\widetilde{\mathbf{\Theta}}$ as discussed in Sec. II. The resulting input-response data are given as

$$\left.\begin{aligned}\mathbf{V} &= \left[\boldsymbol{\omega}_t + \boldsymbol{J}(\boldsymbol{\omega}, \boldsymbol{\psi}) - \frac{1}{\text{Re}}\nabla^2\boldsymbol{\omega}\right] \\ \widetilde{\mathbf{\Theta}} &= \left[\mathbf{x}\ \ \mathbf{y}\ \ \mathbf{t}\right]\end{aligned}\right\}. \tag{29}$$

The derivatives in the output data $\mathbf{V(t)}$ are calculated using finite difference approximations similar to Eq. (4). As streamfunction $(\psi)^p_{i,j}$ and vorticity $(\omega)^p_{i,j}$ data are selected only at sparse spatial locations, we also store the surrounding stencil, i.e., $(\psi)^p_{i+1,j}$, $(\psi)^p_{i-1,j}$, $(\psi)^p_{i,j+1}$, $(\psi)^p_{i,j-1}$, and $(\omega)^p_{i+1,j}$, $(\omega)^p_{i-1,j}$, $(\omega)^p_{i,j+1}$, $(\omega)^p_{i,j-1}$ in order to calculate the derivatives. The index i represents spatial location in the x direction, and j represents spatial location in the y direction.

In this test case, we demonstrate the identification of hidden physics, which is the source term $S(t, x, y)$ given by Eq. (28) from the data obtained at sparse spatial locations using GEP. Table XIX lists the hyperparameters used by GEP to recover the hidden physical law. We use the same function and terminal sets as shown in Table XVIII, but × is used as a linking function. Figure 34 shows the ET form of hidden physical law (source term) obtained by GEP. Simplification of the ET form shows the identified source term, which is close to true source term, as shown in Table XXI.

The 1D advection-diffusion and 2D vortex-merger problem demonstrate the usefulness of GEP in recovering hidden physics, i.e., a source term composed of complex functions using

TABLE XX. Hidden source term (S) of the 1D advection-diffusion PDE identified by GEP.

	Recovered	Test error
True	$S = 4.93\exp(2.47\,t)\sin(3.14\,x) + 0.33\exp(2.47\,t)\cos(3.14\,x)$	
GEP	$S = 4.93\exp(2.46\,t)\sin(3.14\,x) + 0.33\exp(2.46\,t)\cos(3.14\,x) - 3.12\times10^{-5}$	3.34×10^{-7}

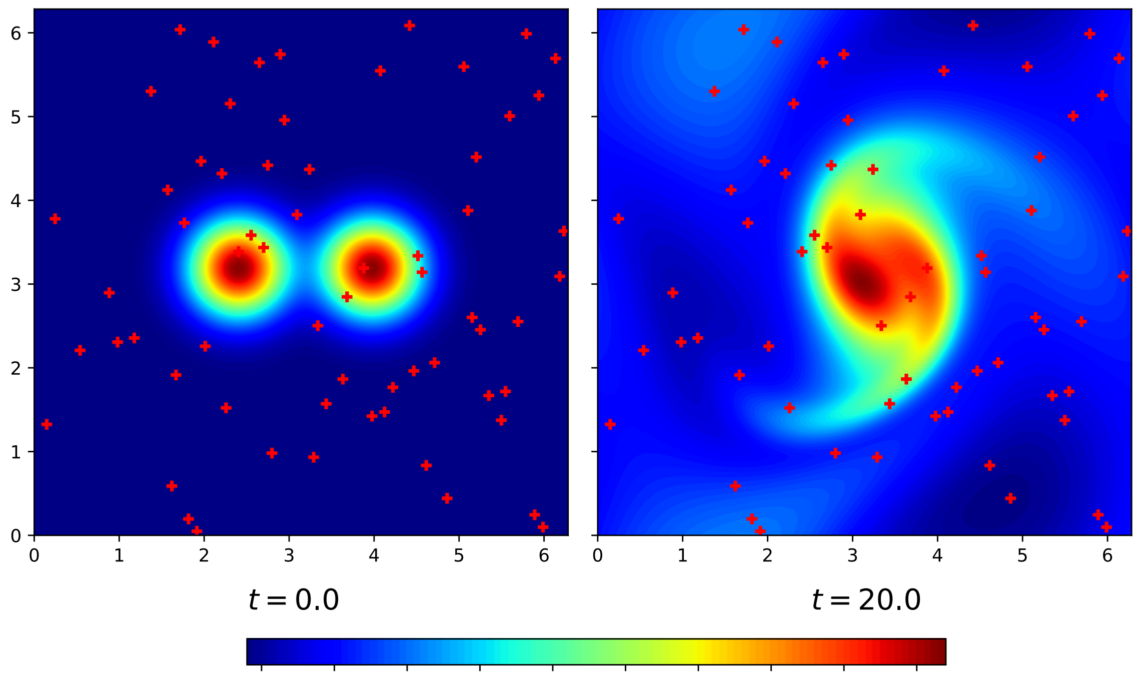

FIG. 33. The 2D vortex-merger problem with source term at time t = 0.0 and t = 20.0. The red markers shows 64 random sensor locations used to collect vorticity (ω) and streamfunction (ψ) data for recovering source term $S(t, x, y)$.

randomly selected sparse data. The expressive power of the feature library limits the applications of STRidge for identifying complex composition models. However, STRidge might be able to identify the infinite series approximations of these nonlinear functions.[39] In the next test case, we use both STRdige and GEP to identify eddy viscosity kernels along with their free modeling coefficient that controls the dissipation of these kernels.

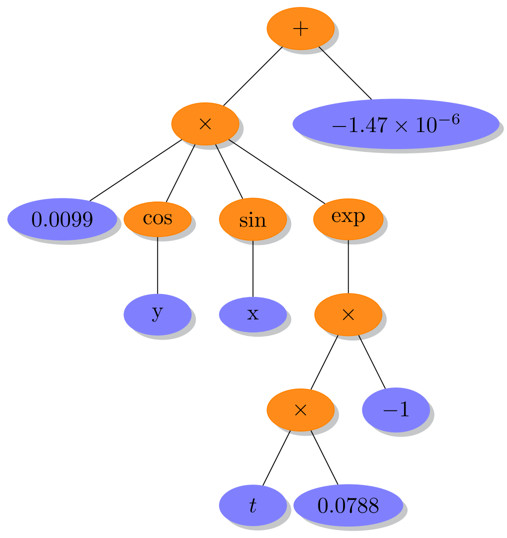

FIG. 34. Hidden source term of the 2D vortex-merger problem in terms of ET identified by GEP.

C. 2D Kraichnan turbulence

The concept of two-dimensional turbulence helps in understanding many complex physical phenomena such as geophysical and astrophysical flows.[114,115] The equations of two-dimensional turbulence can model idealized flow configurations restricted to two-dimensions such as flows in rapidly rotating systems and in thin films over rigid bodies. The physical mechanism associated with the two-dimensional turbulence is explained by the Kraichnan-Batchelor-Leith (KBL) theory.[116–118] Generally, large eddy simulation (LES) is performed for both two and three dimensional flows to avoid the fine resolution and thereby computational requirements of direct numerical simulation (DNS) computations.[119,120] In LES, the flow variables are decomposed into resolved low wavenumber (or large scale) and unresolved high wavenumber (or small scale). This is achieved by the application

TABLE XXI. Hidden source term (S) of the 2D vortex-merger problem identified by GEP.

	Recovered	Test error
True	$S = 0.0100\,\sin(x)\cos(y)\exp(-0.078\,t)$	
GEP	$S = 0.0099\,\sin(x)\cos(y)\exp(-0.078\,t) - 1.47\times10^{-6}$	1.35×10^{-8}

of a low pass spatial filter to the flow variables. By arresting high wavenumber content (small scales), we can reduce the high resolution requirement of DNS, and hence faster simulations and reduced storage requirements. However, the procedure of introducing a low pass filtering results in an unclosed term for the LES governing equations representing the finer scale effects in the form of a source term.

Thus, the quality of LES depends on the modeling approach used to close the spatially filtered governing equations to capture the effects of the unresolved finer scales.[121] This model, also called the subgrid scale model, is a critical part of LES computations. A functional or eddy viscosity approach is one of the popular approaches to model this closure term. These approaches propose an artificial viscosity to mimic the dissipative effect of the fine scales. Some of the popular functional models are the Smagorinsky,[122] Leith,[123] Balwin-Lomax,[124] and Cebeci-Smith models.[125] All these models require the specification of a model constant that controls the quantity of dissipation in the simulation, and its value is often set based on the nature of the particular flow being simulated. In this section, we demonstrate the identification of an eddy viscosity kernel (model) along with its *ad hoc* model constant from observing the source term of the LES equation using both GEP and STRidge as robust SR tools. To this end, we use the vorticity-streamfunction formulation for two-dimensional fluid flows given in Eq. (26). We derive the LES equations for the two dimensional Kraichnan turbulence by applying a low pass spatial filter to the vorticity-streamfunction PDE given in Eq. (26). The resulting filtered equation is given as

$$\overline{\omega}_t + \overline{J(\psi,\omega)} = \frac{1}{\mathrm{Re}}\nabla^2\overline{\omega}, \tag{30}$$

where Re is the Reynolds number of the flow and $J(\omega,\psi)$ is the Jacobian term given as $\psi_y\omega_x - \psi_x\omega_y$. Furthermore, Eq. (30) is rearranged as

TABLE XXII. GEP functional and terminal sets used for identifying eddy viscosity kernel. "?" is a random constant.

Parameter	Value
Function set	$+, -, \times, /$
Terminal set	$\widetilde{\boldsymbol{\Theta}}$, ?
Linking function	+

TABLE XXIII. GEP hyperparameters selected for identification of the eddy viscosity kernel for the Kraichnan turbulence.

Hyperparameters	Kraichnan turbulence
Head length	2
Number of genes	2
Population size	20
Generations	500
Length of RNC array	3
Random constant minimum	−1
Random constant maximum	1

TABLE XXIV. LES source term (Π) for two-dimensional Kraichnan turbulence problem identified by GEP and STRidge.

	Recovered
GEP	$\Pi = 0.000\,128\,\lvert\overline{S}\rvert\,\overline{w}_{2x} + 0.000\,128\,\lvert\overline{S}\rvert\,\overline{w}_{2y} - 0.362$
STRidge	$\Pi = 0.000\,132\,\lvert\overline{S}\rvert\,\overline{w}_{2x} + 0.000\,129\,\lvert\overline{S}\rvert\,\overline{w}_{2y}$

$$\overline{\omega}_t + J(\overline{\psi},\overline{\omega}) = \frac{1}{\mathrm{Re}}\nabla^2\overline{\omega} + \Pi, \tag{31}$$

where the LES source term Π is given as

$$\Pi = J(\overline{\psi},\overline{\omega}) - \overline{J(\psi,\omega)}. \tag{32}$$

The source term Π in Eq. (32) represents the influence of the subgrid scales on the larger resolved scales. The term $\overline{J(\psi,\omega)}$ is not

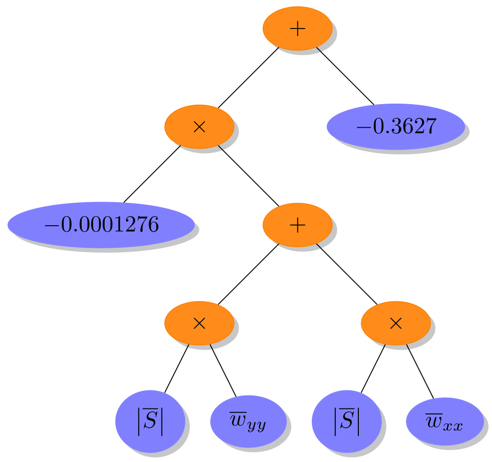

FIG. 35. Samgorisnsky kernel in terms of ET identified for the two-dimensional Kraichnan turbulence problem by GEP.

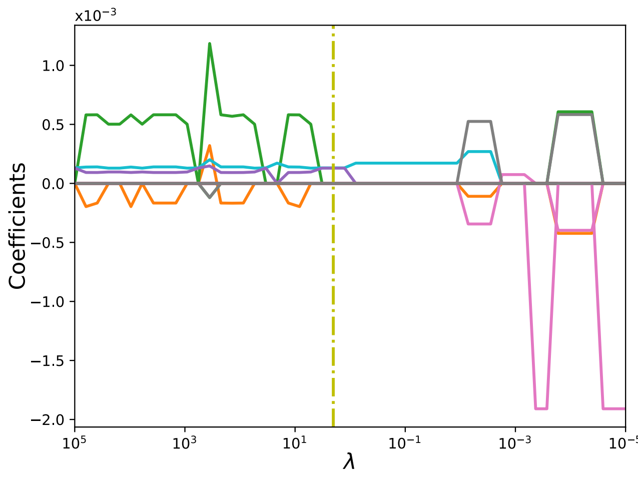

FIG. 36. STRidge coefficients as a function of regularization parameter λ for the two-dimensional Kraichnan turbulence problem.

available, which necessitates the use of a closure modeling approach. In functional or eddy viscosity models, the source term of LES equations is represented as

$$\Pi = \nu_e \nabla^2 \overline{\omega}, \tag{33}$$

where the eddy viscosity ν_e is given by, but not limited to, the Smagorinsky, Leith, Baldwin-Lomax, and Cebeci-Smith kernels. The choice of these eddy viscosity kernels essentially implies the choice of a certain function of local field variables such as the strain rate or gradient of vorticity as a control parameter for the magnitude of ν_e.

In Smagorisnky model, the eddy viscosity kernel is given by

$$\nu_e = (c_s \delta)^2 |\overline{S}|, \tag{34}$$

where c_s is a free modeling constant that controls the magnitude of the dissipation and δ is a characteristic grid length scale given by the square root of the product of the cell sizes in each direction. The $|\overline{S}|$ is based on the second invariant of the filtered field deformation, and given by

$$|\overline{S}| = \sqrt{4\overline{\psi}_{xy}^2 + (\overline{\psi}_{2x} - \overline{\psi}_{2y})^2}. \tag{35}$$

The Leith model proposes that the eddy viscosity kernel is a function of vorticity and is given as

$$\nu_e = (c_s \delta)^3 |\nabla \overline{\omega}|, \tag{36}$$

where $|\nabla \overline{\omega}|$ controls the dissipative characteristic of eddy viscosity as against resolved strain rate used in the Smagorinsky model. The magnitude of the gradient of vorticity is defined as

$$|\nabla \overline{\omega}| = \sqrt{\overline{\omega}_x^2 + \overline{\omega}_y^2}. \tag{37}$$

The Baldwin-Lomax approach is an alternative approach that models the eddy viscosity kernel as

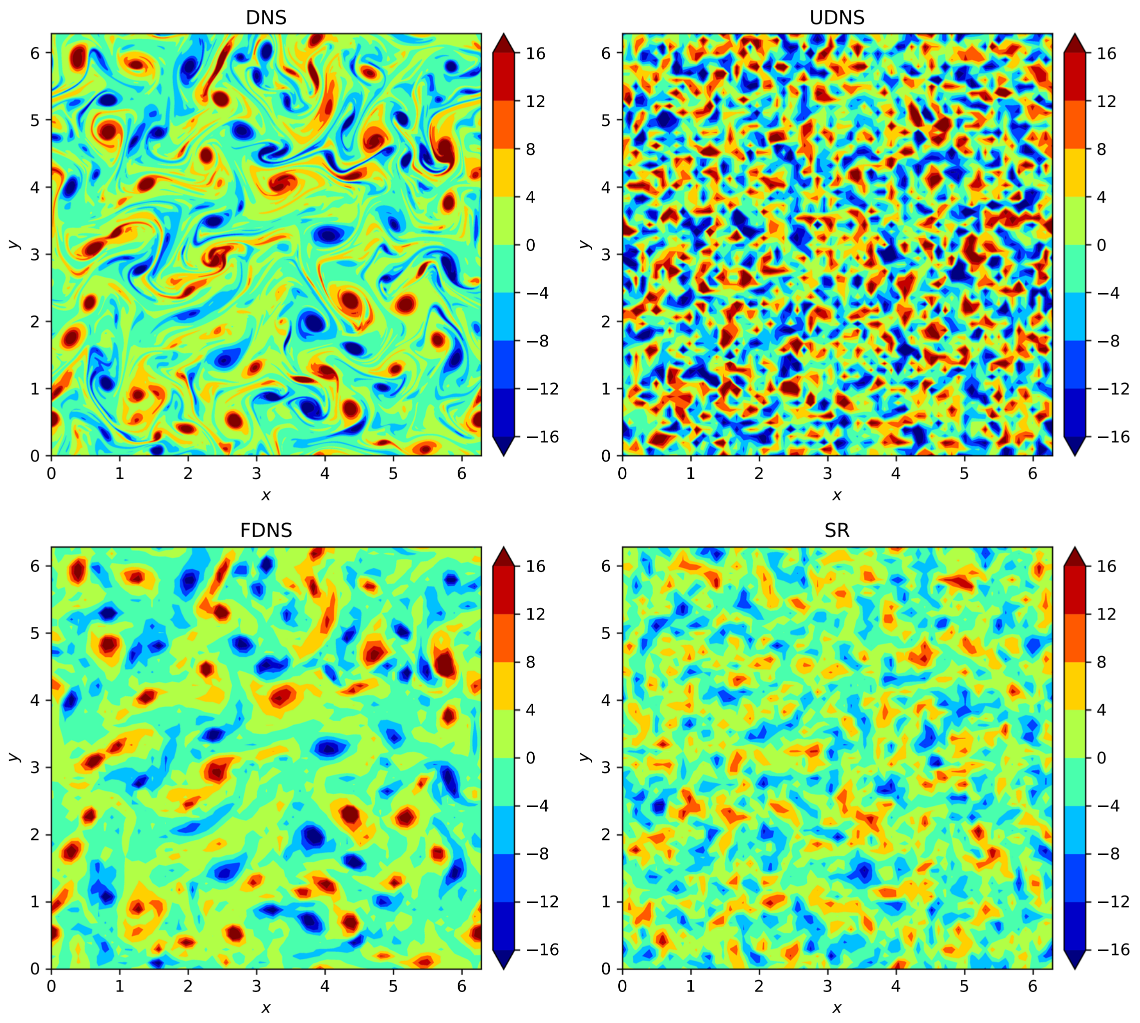

FIG. 37. Contour plots for the two-dimensional Kraichnan turbulence problem at $t = 4$. SR refers to the identified model of the Smagorinsky kernel with $c_s = 0.12$. UDNS and FDNS refer to the no-model and filtered DNS simulations, respectively.

$$\nu_e = (c_s\delta)^2|\overline{\omega}|, \tag{38}$$

where $|\overline{\omega}|$ is the absolute value of the vorticity considered as a measure of the local energy content of the flow at a grid point and also a measure of the dissipation required at that location.

The Cebeci-Smith model was devised for the Reynolds Averaged Navier-Stokes (RANS) applications. The model is modified for the LES setting, and is given as

$$\nu_e = (c_s\delta)^2|\overline{\Omega}|, \tag{39}$$

where $|\overline{\Omega}|$ is given as

$$|\overline{\Omega}| = \sqrt{\overline{\psi}_{2x}^2 + \overline{\psi}_{2y}^2}. \tag{40}$$

High fidelity DNS simulations are performed for Eq. (30). We use a square domain of length 2π with periodic boundary conditions in both directions. We simulate homogeneous isotropic decaying turbulence, which may be specified by an initial energy spectrum that decays through time. High fidelity DNS simulations are carried out for Re = 4000 with 1024 × 1024 resolution from time $t = 0$ to $t = 4.0$ with time step 0.001. The filtered flow quantities and LES source term Π in Eq. (32) are obtained from coarsening the DNS quantities to obtain quantities with a 64 × 64 resolution. Further details of the solver and coarsening can be found in San and Staples.[109] Once the LES source term Π in Eq. (32) and filtered flow quantities are obtained, we build the feature library and output data similar to the discussion in Sec. II. The resulting input-response data are given as

$$\left.\begin{aligned} \mathbf{V} &= [\boldsymbol{\Pi}] \\ \widetilde{\boldsymbol{\Theta}} &= \left[\overline{\boldsymbol{\omega}}_{2x}\ \overline{\boldsymbol{\omega}}_{2y}\ |\overline{S}|\ |\nabla\overline{\boldsymbol{\omega}}|\ |\overline{\boldsymbol{\omega}}|\ |\overline{\Omega}|\right] \end{aligned}\right\}. \tag{41}$$

GEP uses the output and feature library given in Eq. (41) to automatically extract the best eddy viscosity kernel for decaying turbulence problems along with the model's *ad hoc* coefficient.

The extended feature library is constructed to include nonlinear interactions up to the quadratic degree to expand the expressive power for the STRidge algorithm. The resulting extended feature library is given as

$$\boldsymbol{\Theta} = \left[\mathbf{1}\ \overline{\boldsymbol{\omega}}_{2x}\ \overline{\boldsymbol{\omega}}_{2x}^2\ \overline{\boldsymbol{\omega}}_{2y}\ \overline{\boldsymbol{\omega}}_{2x}\overline{\boldsymbol{\omega}}_{2y}\ \overline{\boldsymbol{\omega}}_{2y}^2\ \ldots\ |\overline{\boldsymbol{\Omega}}|^2\right]. \tag{42}$$

The function and terminal sets used for identification of the eddy viscosity kernel by GEP are listed in Table XXII. Furthermore, the hyperparameters of GEP are listed in Table XXIII. Both GEP and STRidge identify the Smagorinsky kernel with approximately the same coefficients as shown in Table XXIV. The ET form of the Smagorinsky kernel found by GEP is shown in Fig. 35. The regularization weight λ is varied to recover multiple models of different complexity as shown in Fig. 36. The yellow line in Fig. 36 corresponds to the value of λ where STRidge identifies the Smagorinsky kernel. We can take the average coefficient from both SR tools and derive the value of the free modeling constant identified by SR approaches. The average model of both approaches is given by

$$\Pi = 0.000\,129\,(|S|\,w_{2x} + |S|\,w_{2y}). \tag{43}$$

By comparing with Eqs. (33) and (34) and using the spatial cell size $\delta = \frac{2\pi}{64}$, the value of the free modeling constant is retrieved as $c_s = 0.12$.

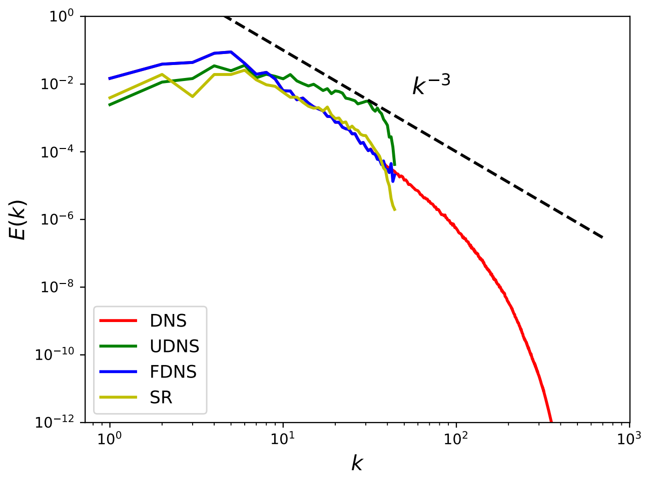

FIG. 38. Energy spectra for the two-dimensional Kraichnan turbulence problem at t = 4. SR refers to the identified model of the Smagorinsky kernel with c_s = 0.12. UDNS and FDNS refer to the no-model and filtered DNS simulations, respectively.

The SR identified Smagorinsky kernel with $c_s = 0.12$ is plugged into the LES source term Π in Eq. (31) and a forward LES simulation is run for the 2D decaying turbulence problem. Figure 37 shows the vorticity fields at time $t = 4.0$ for the DNS, under-resolved no-model simulation (UDNS), filtered DNS (FDNS), and LES with the SR-retrieved Smagorinsky kernel. Energy spectra at time $t = 4.0$ are shown in Fig. 38. We can observe that SR approaches satisfactorily identify the value of the modeling constant c_s, which controls reasonably well the amount of dissipation needed to account for the unresolved small scales. We also highlight that several deep learning frameworks such as ANNs have been exploited for subgrid scale modeling for 2D Kraichnan turbulence.[126–128] The importance of feature selection can be seen in these works where different invariant kernels, like those listed in the feature library given in Eq. (41), are used as inputs to improve the ANN's predictive performance. The authors compared *a posteriori* results with different free modeling coefficients of the Smagorinsky and Leith models. Furthermore, it is evident from the energy spectrum comparisons in their studies that the appropriate addition of dissipation with the right tuning of the free modeling coefficient can lead to better predictions of the energy spectrum. To this end, SR approaches automatically distill traditional models along with the right values for the *ad hoc* free modeling coefficients. Although the present study establishes a modular regression approach for discovering the relevant free parameters in LES models, we highlight that it can be extended easily to a dynamic closure modeling framework reconstructed automatically by sparse data on the fly based on the flow evolution, a topic we would like to address in future studies.

VI. CONCLUSION

Data driven symbolic regression tools can be extremely useful for researchers for inferring complex models from sensor data when the underlying physics is partially or completely unknown. Sparse optimization techniques are envisioned as an SR tool that is capable of recovering hidden physical laws in a highly efficient computational manner. Popular sparse optimization techniques such as LASSO, ridge, and elastic-net are also known as feature selection methods in machine learning. These techniques are regularized variants of least squares regression adapted to reduce overfitting and promote sparsity. The model prediction ability of sparse regression methods is primarily dependent on the expressive power of its feature library, which contains exhaustive combinations of nonlinear basis functions that might represent the unknown physical law. This limits the identification of physical models that are represented by complex functional compositions. GEP is an evolutionary optimization algorithm widely adapted for the SR approach. This genotype-phenotype algorithm takes advantage of the simple chromosome representations of GA and the free expansion of complex chromosomes of GP. GEP is a natural feature extractor that may not need *a priori* information of nonlinear bases other than the basic features as a terminal set. Generally, with enough computational time, GEP may recover unknown physical models that are represented by complex functional compositions by observing the input-response data.

In this paper, we demonstrate that the sparse regression technique STRidge and the evolutionary optimization algorithm GEP are effective SR tools for identifying hidden physical laws from observed data. We first identify various canonical PDEs using both STRidge and GEP. We demonstrate that STRidge is limited by its feature library for identifying the Sine-Gordon PDE. Following equation discovery, we demonstrate the power of both algorithms in identifying the leading truncation error terms for the Burgers MDE. While both algorithms find the truncation terms, coefficients found by STRidge were more accurate than coefficients found by GEP. We note that, when the feature library is capable of expressing the underlying physical model, the application of STRidge is suitable due to its fewer hyperparameters and lower computational overhead. Next, we illustrate the recovery of hidden physics that is supplied as the source or forcing term of a PDE. We use randomly selected sparse measurements that mimic real world data collection. STRdige is not applied in this setting as the feature library was limited to represent the unknown physical model that consists of complex functional compositions. GEP was able to identify the source term for both the 1D advection-diffusion PDE and the 2D vortex-merger problem using sparse measurements. Finally, both STRdige and GEP were applied to discover the eddy viscosity kernel along with its *ad hoc* modeling coefficient as a subgrid scale model for the LES equations simulating the 2D Kraichnan turbulence problem. This particular example demonstrates the capability of inverse modeling or parametric estimation for turbulence closure models using SR approaches. Future studies will focus on identifying LES closure models that augment the known closure models by accounting for the various nonlinear physical process. Furthermore, various SR tools are being investigated for the identification of nonlinear truncation error terms of MDEs for implicit LES approaches that can be exploited for modeling turbulent flows without the need for explicit subgrid scale models.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Award No. DE-SC0019290. O.S. gratefully acknowledges their support.

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

REFERENCES

[1] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," Science **349**, 255–260 (2015).
[2] V. Marx, "Biology: The big challenges of big data," Nature **498**, 255–260 (2013).
[3] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," Psychol. Rev. **65**, 386 (1958).
[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436 (2015).
[5] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted l_1 minimization," J. Fourier Anal. Appl. **14**, 877–905 (2008).
[6] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," IEEE Signal Process. Mag. **25**, 21–30 (2008).
[7] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (MIT Press, Cambridge, MA, USA, 1992), Vol. 1.
[8] C. Ferreira, "Gene expression programming: A new adaptive algorithm for solving problems," preprint arXiv:cs/0102027 (2001).
[9] C. Ferreira, *Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence* (Springer, 2006), Vol. 21.
[10] M. Mitchell, *An Introduction to Genetic Algorithms* (MIT Press, 1998).
[11] J. H. Holland, *Adaptation in Natural and Artificial Systems, 1975* (University of Michigan Press, Ann Arbor, MI, 1992).
[12] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," Science **324**, 81–85 (2009).
[13] J. Bongard and H. Lipson, "Automated reverse engineering of nonlinear dynamical systems," Proc. Natl. Acad. Sci. U. S. A. **104**, 9943–9948 (2007).
[14] Y. Yang, C. Wang, and C. Soh, "Force identification of dynamic systems using genetic programming," Int. J. Numer. Methods Eng. **63**, 1288–1312 (2005).
[15] L. Ferariu and A. Patelli, "Elite based multiobjective genetic programming for nonlinear system identification," in *International Conference on Adaptive and Natural Computing Algorithms* (Springer, 2009), pp. 233–242.
[16] C. Luo, Z. Hu, S.-L. Zhang, and Z. Jiang, "Adaptive space transformation: An invariant based method for predicting aerodynamic coefficients of hypersonic vehicles," Eng. Appl. Artif. Intell. **46**, 93–103 (2015).
[17] S. L. Brunton and B. R. Noack, "Closed-loop turbulence control: Progress and challenges," Appl. Mech. Rev. **67**, 050801 (2015).
[18] N. Gautier, J.-L. Aider, T. Duriez, B. Noack, M. Segond, and M. Abel, "Closed-loop separation control using machine learning," J. Fluid Mech. **770**, 442–457 (2015).
[19] T. Duriez, V. Parezanović, K. von Krbek, J.-P. Bonnet, L. Cordier, B. R. Noack, M. Segond, M. Abel, N. Gautier, J.-L. Aider *et al.*, "Feedback control of turbulent shear flows by genetic programming," preprint arXiv:1505.01022 (2015).
[20] A. Debien, K. A. Von Krbek, N. Mazellier, T. Duriez, L. Cordier, B. R. Noack, M. W. Abel, and A. Kourta, "Closed-loop separation control over a sharp edge ramp using genetic programming," Exp. Fluids **57**, 40 (2016).
[21] M. Quade, M. Abel, K. Shafi, R. K. Niven, and B. R. Noack, "Prediction of dynamical systems by symbolic regression," Phys. Rev. E **94**, 012214 (2016).
[22] C. Luo and S.-L. Zhang, "Parse-matrix evolution for symbolic regression," Eng. Appl. Artif. Intell. **25**, 1182–1193 (2012).

[23]M. F. Brameier and W. Banzhaf, *Linear Genetic Programming* (Springe-Verlag, New York, 2007).
[24]R. S. Faradonbeh and M. Monjezi, "Prediction and minimization of blast-induced ground vibration using two robust meta-heuristic algorithms," Eng. Comput. **33**, 835–851 (2017).
[25]R. S. Faradonbeh, A. Salimi, M. Monjezi, A. Ebrahimabadi, and C. Moormann, "Roadheader performance prediction using genetic programming (GP) and gene expression programming (GEP) techniques," Environ. Earth Sci. **76**, 584 (2017).
[26]F. S. Hoseinian, R. S. Faradonbeh, A. Abdollahzadeh, B. Rezai, and S. Soltani-Mohammadi, "Semi-autogenous mill power model development using gene expression programming," Powder Technol. **308**, 61–69 (2017).
[27]H. Çanakcı, A. Baykasoğlu, and H. Güllü, "Prediction of compressive and tensile strength of Gaziantep basalts via neural networks and gene expression programming," Neural Comput. Appl. **18**, 1031 (2009).
[28]J. Weatheritt and R. Sandberg, "A novel evolutionary algorithm applied to algebraic modifications of the RANS stress–strain relationship," J. Comput. Phys. **325**, 22–37 (2016).
[29]M. Schoepplein, J. Weatheritt, R. Sandberg, M. Talei, and M. Klein, "Application of an evolutionary algorithm to LES modelling of turbulent transport in premixed flames," J. Comput. Phys. **374**, 1166–1179 (2018).
[30]J. Weatheritt and R. D. Sandberg, "Hybrid Reynolds-averaged/large-eddy simulation methodology from symbolic regression: Formulation and application," AIAA J. **55**, 3734–3746 (2017).
[31]H. Rauhut, "Compressive sensing and structured random matrices," in *Theoretical Foundations and Numerical Methods for Sparse Recovery* (Walter de Gruyter GmbH & Co. KG, Berlin, 2010), Vol. 9, pp. 1–92.
[32]R. Tibshirani, "Regression shrinkage and selection via the LASSO," J. R. Stat. Soc.: Ser. B **58**, 267–288 (1996).
[33]G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning* (Springer Science+Business Media, New York, 2013), Vol. 112.
[34]R. Tibshirani, M. Wainwright, and T. Hastie, *Statistical Learning with Sparsity: The LASSO and Generalizations* (Chapman and Hall/CRC, Florida, USA, 2015).
[35]E. J. Candes, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," Commun. Pure Appl. Math. **59**, 1207–1223 (2006).
[36]K. P. Murphy, *Machine Learning: A Probabilistic Perspective* (MIT Press, Cambridge, MA, USA, 2012).
[37]H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," J. R. Stat. Soc.: Ser. B **67**, 301–320 (2005).
[38]J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," J. Stat. Software **33**, 106182 (2010).
[39]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. U. S. A. **113**, 3932–3937 (2016).
[40]S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Data-driven discovery of partial differential equations," Sci. Adv. **3**, e1602614 (2017).
[41]H. Schaeffer, R. Caflisch, C. D. Hauck, and S. Osher, "Sparse dynamics for partial differential equations," Proc. Natl. Acad. Sci. U. S. A. **110**, 6634–6639 (2013).
[42]H. Schaeffer, "Learning partial differential equations via data discovery and sparse optimization," Proc. R. Soc. A **473**, 20160446 (2017).
[43]G. Tran and R. Ward, "Exact recovery of chaotic systems from highly corrupted data," Multiscale Model. Simul. **15**, 1108–1129 (2017).
[44]H. Schaeffer, G. Tran, and R. Ward, "Extracting sparse high-dimensional dynamics from limited data," SIAM J. Appl. Math. **78**, 3279–3295 (2018).
[45]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. R. Soc. A **473**, 20170009 (2017).
[46]N. M. Mangan, S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Inferring biological networks by sparse identification of nonlinear dynamics," IEEE Trans. Mol. Biol. Commun. Multi-Scale **2**, 52–63 (2016).
[47]J.-C. Loiseau, B. R. Noack, and S. L. Brunton, "Sparse reduced-order modelling: Sensor-based dynamics to full-state estimation," J. Fluid Mech. **844**, 459–490 (2018).
[48]M. Schmelzer, R. Dwight, and P. Cinnella, "Data-driven deterministic symbolic regression of nonlinear stress-strain relation for rans turbulence modelling," in *2018 Fluid Dynamics Conference* (AIAA, 2018), p. 2900.
[49]P. Zheng, T. Askham, S. L. Brunton, J. N. Kutz, and A. Y. Aravkin, "A unified framework for sparse relaxed regularized regression: SR3," IEEE Access **7**, 1404–1423 (2018).
[50]T. Zhang, "Adaptive forward-backward greedy algorithm for sparse learning with linear models," in *Advances in Neural Information Processing Systems* (NIPS, 2009), pp. 1921–1928.
[51]S. Thaler, L. Paehler, and N. A. Adams, "Sparse identification of truncation errors," J. Comput. Phys. **397**, 108851 (2019).
[52]T. McConaghy, "FFX: Fast, scalable, deterministic symbolic regression technology," in *Genetic Programming Theory and Practice IX* (Springer, 2011), pp. 235–260.
[53]H. Vaddireddy and O. San, "Equation discovery using fast function extraction: A deterministic symbolic regression approach," Fluids **4**, 111 (2019).
[54]M. Schmelzer, R. P. Dwight, and P. Cinnella, "Machine learning of algebraic stress models using deterministic symbolic regression," preprint arXiv:1905.07510 (2019).
[55]C. Chen, C. Luo, and Z. Jiang, "Elite bases regression: A real-time algorithm for symbolic regression," in *2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)* (IEEE, 2017), pp. 529–535.
[56]T. Worm and K. Chiu, "Prioritized grammar enumeration: Symbolic regression by dynamic programming," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation* (ACM, 2013), pp. 1021–1028.
[57]D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2012), pp. 3642–3649.
[58]A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2015), pp. 3128–3137.
[59]A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," Electron. Imaging **2017**, 70–76.
[60]M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys. **378**, 686–707 (2019).
[61]M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Numerical Gaussian processes for time-dependent and nonlinear partial differential equations," SIAM J. Sci. Comput. **40**, A172–A198 (2018).
[62]M. Raissi and G. E. Karniadakis, "Hidden physics models: Machine learning of nonlinear partial differential equations," J. Comput. Phys. **357**, 125–141 (2018).
[63]J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith, "Dynamic systems identification with Gaussian processes," Math. Comput. Modell. Dyn. Syst. **11**, 411–424 (2005).
[64]G. Gregorčič and G. Lightbody, "Nonlinear system identification: From multiple-model networks to Gaussian processes," Eng. Appl. Artif. Intell. **21**, 1035–1055 (2008).
[65]L. Cordier, B. R. Noack, G. Tissot, G. Lehnasch, J. Delville, M. Balajewicz, G. Daviller, and R. K. Niven, "Identification strategies for model-based control," Exp. Fluids **54**, 1580 (2013).
[66]Z. Wang, D. Xiao, F. Fang, R. Govindan, C. C. Pain, and Y. Guo, "Model identification of reduced order fluid dynamics systems using deep learning," Int. J. Numer. Methods Fluids **86**, 255–268 (2018).
[67]J.-F. Cai, B. Dong, S. Osher, and Z. Shen, "Image restoration: Total variation, wavelet frames, and beyond," J. Am. Math. Soc. **25**, 1033–1089 (2012).
[68]B. Dong, Q. Jiang, and Z. Shen, "Image restoration: Wavelet frame shrinkage, nonlinear evolution PDEs, and beyond," Multiscale Model. Simul. **15**, 606–660 (2017).
[69]Z. Long, Y. Lu, X. Ma, and B. Dong, "PDE-net: Learning PDEs from data," in *Proceedings of the 35th International Conference on Machine Learning, Proceedings of Machine Learning Research*, edited by J. Dy and A. Krause (PMLR, Stockholmsmässan, Stockholm, Sweden, 2018), Vol. 80, pp. 3208–3216.

[70]Z. Long, Y. Lu, and B. Dong, "PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network," J. Comput. Phys. **399**, 108925 (2019).
[71] C. Ferreira, "Gene expression programming in problem solving," in *Soft Computing and Industry* (Springer, 2002), pp. 635–653.
[72]G. Shuhua, geppy: A gene expression programming framework in python, 2019, https://github.com/ShuhuaGao/geppy.
[73]F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," J. Mach. Learn. Res. **13**, 2171–2175 (2012).
[74]R. G. Baraniuk, "Compressive sensing," IEEE Signal Process. Mag. **24**, 118–124 (2007).
[75]H. Bateman, "Some recent researches on the motion of fluids," Mon. Weather Rev. **43**, 163–170 (1915).
[76]G. B. Whitham, *Linear and Nonlinear Waves* (John Wiley & Sons, 2011), Vol. 42.
[77]M. Maleewong and S. Sirisup, "On-line and off-line POD assisted projective integral for non-linear problems: A case study with Burgers' equation," Int. J. Math., Comput. Phys., Electr., Comput. Eng. **5**, 984–992 (2011).
[78]D. J. Korteweg and G. de Vries, "On the change of form of long waves advancing in a rectangular canal, and on a new type of long stationary waves," Philos. Mag. Series 5 **39**(240), 422–443 (1895) [reprinted in Philos. Mag. **91**, 1007–1028 (2011)].
[79]L. Wazzan, "A modified tanh–coth method for solving the KdV and the KdV–Burgers' equations," Commun. Nonlinear Sci. Numer. Simul. **14**, 443–450 (2009).
[80]T. Ozis and S. Ozer, "A simple similarity-transformation-iterative scheme applied to Korteweg–de Vries equation," Appl. Math. Comput. **173**, 19–32 (2006).
[81] G. L. Lamb, Jr., *Elements of Soliton Theory* (Wiley-Interscience, New York, 1980).
[82]T. Kawahara, "Oscillatory solitary waves in dispersive media," J. Phys. Soc. Jpn. **33**, 260–264 (1972).
[83]T. Kawahara, N. Sugimoto, and T. Kakutani, "Nonlinear interaction between short and long capillary-gravity waves," J. Phys. Soc. Jpn. **39**, 1379–1386 (1975).
[84]J. K. Hunter and J. Scheurle, "Existence of perturbed solitary wave solutions to a model equation for water waves," Physica D **32**, 253–268 (1988).
[85]Sirendaoreji, "New exact travelling wave solutions for the Kawahara and modified Kawahara equations," Chaos, Solitons Fractals **19**, 147–150 (2004).
[86]C. Zhi-Xiong and G. Ben-Yu, "Analytic solutions of the Nagumo equation," IMA J. Appl. Math **48**, 107–115 (1992).
[87]J. Nagumo, S. Arimoto, and S. Yoshizawa, "An active pulse transmission line simulating nerve axon," Proc. IRE **50**, 2061–2070 (1962).
[88]D. G. Aronson and H. F. Weinberger, "Multidimensional nonlinear diffusion arising in population genetics," Adv. Math. **30**, 33–76 (1978).
[89]A. Scott, "Neuristor propagation on a tunnel diode loaded transmission line," Proc. IEEE **51**, 240 (1963).
[90]M. Dehghan and F. Fakhar-Izadi, "Pseudospectral methods for Nagumo equation," Int. J. Numer. Methods Biomed. Eng. **27**, 553–561 (2011).
[91] A. Barone, F. Esposito, C. Magee, and A. Scott, "Theory and applications of the sine-gordon equation," La Riv. Nuovo Cimento **1**, 227–267 (1971).
[92]J. Perring and T. Skyrme, "A model unified field equation," Nucl. Phys. **31**, 550–555 (1962).
[93]C. W. Hirt, "Heuristic stability theory for finite-difference equations," J. Comput. Phys. **2**, 339–355 (1968).
[94]R. D. Ritchmyer and K. Norton, *Difference Methods for Initial Value Problems* (Jonh Wiley & Sons, New York, 1967).
[95]G. Klopfer and D. S. McRae, "Nonlinear truncation error analysis of finite difference schemes forthe Euler equations," AIAA J. **21**, 487–494 (1983).
[96]A. Majda and S. Osher, "A systematic approach for correcting nonlinear instabilities," Numer. Math. **30**, 429–452 (1978).
[97]E. Ozbenli and P. Vedula, "Numerical solution of modified differential equations based on symmetry preservation," Phys. Rev. E **96**, 063304 (2017).
[98]E. Ozbenli and P. Vedula, "High order accurate finite difference schemes based on symmetry preservation," J. Comput. Phys. **349**, 376–398 (2017).
[99]N. Adams, S. Hickel, and S. Franz, "Implicit subgrid-scale modeling by adaptive deconvolution," J. Comput. Phys. **200**, 412–431 (2004).
[100]L. G. Margolin and W. J. Rider, "A rationale for implicit turbulence modelling," Int. J. Numer. Methods Fluids **39**, 821–841 (2002).
[101] C. Hirsch, *Numerical Computation of Internal and External Flows: The Fundamentals of Computational Fluid Dynamics* (Elsevier, Burlington, MA, 2007).
[102]V. M. Krasnopolsky and M. S. Fox-Rabinovitz, "Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction," Neural Networks **19**, 122–134 (2006).
[103]V. M. Krasnopolsky and M. S. Fox-Rabinovitz, "A new synergetic paradigm in environmental numerical modeling: Hybrid models combining deterministic and machine learning components," Ecol. Modell. **191**, 5–18 (2006).
[104]S. Dhingra, R. B. Madda, A. H. Gandomi, R. Patan, and M. Daneshmand, "Internet of things mobile-air pollution monitoring system (IoT-Mobair)," IEEE Internet Things J. **6**, 5577–5584 (2019).
[105]N. Kumar, "Unsteady flow against dispersion in finite porous media," J. Hydrol. **63**, 345–358 (1983).
[106]J. Isenberg and C. Gutfinger, "Heat transfer to a draining film," Int. J. Heat Mass Transfer **16**, 505–512 (1973).
[107]V. Guvanasen and R. Volker, "Numerical solutions for solute transport in unconfined aquifers," Int. J. Numer. Methods Fluids **3**, 103–123 (1983).
[108]P. Meunier, S. Le Dizès, and T. Leweke, "Physics of vortex merging," C. R. Phys. **6**, 431–450 (2005).
[109]O. San and A. E. Staples, "High-order methods for decaying two-dimensional homogeneous isotropic turbulence," Comput. Fluids **63**, 105–127 (2012).
[110]O. San and A. E. Staples, "A coarse-grid projection method for accelerating incompressible flow computations," J. Comput. Phys. **233**, 480–508 (2013).
[111]J. N. Reinaud and D. G. Dritschel, "The critical merger distance between two co-rotating quasi-geostrophic vortices," J. Fluid Mech. **522**, 357–381 (2005).
[112]A. Arakawa, "Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow. Part I," J. Comput. Phys. **1**, 119–143 (1966).
[113]S. Pawar and O. San, "CFD Julia: A learning module structuring an introductory course on computational fluid dynamics," Fluids **4**, 159 (2019).
[114]G. Boffetta and S. Musacchio, "Evidence for the double cascade scenario in two-dimensional turbulence," Phys. Rev. E **82**, 016307 (2010).
[115]G. Boffetta and R. E. Ecke, "Two-dimensional turbulence," Annu. Rev. Fluid Mech. **44**, 427–451 (2012).
[116]R. H. Kraichnan, "Inertial ranges in two-dimensional turbulence," Phys. Fluids **10**, 1417–1423 (1967).
[117]G. K. Batchelor, "Computation of the energy spectrum in homogeneous two-dimensional turbulence," Phys. Fluids **12**, II–233 (1969).
[118]C. Leith, "Atmospheric predictability and two-dimensional turbulence," J. Atmos. Sci. **28**, 145–161 (1971).
[119]U. Piomelli, "Large-eddy simulation: Achievements and challenges," Prog. Aerosp. Sci. **35**, 335–362 (1999).
[120]C. Meneveau and J. Katz, "Scale-invariance and turbulence models for large-eddy simulation," Annu. Rev. Fluid Mech. **32**, 1–32 (2000).
[121]P. Sagaut, *Large Eddy Simulation for Incompressible Flows: An Introduction* (Springer Science & Business Media, 2006).
[122]J. Smagorinsky, "General circulation experiments with the primitive equations: I. The basic experiment," Mon. Weather Rev. **91**, 99–164 (1963).
[123]C. E. Leith, "Diffusion approximation for two-dimensional turbulence," Phys. Fluids **11**, 671–672 (1968).
[124]B. Baldwin and H. Lomax, "Thin-layer approximation and algebraic model for separated turbulentflows," in *16th Aerospace Sciences Meeting* (AIAA, 1978), p. 257.
[125]A. Smith and T. Cebeci, "Numerical solution of the turbulent-boundary-layer equations," Technical Report DAC 33735, DTIC, 1967.
[126]R. Maulik, O. San, A. Rasheed, and P. Vedula, "Subgrid modelling for two-dimensional turbulence using neural networks," J. Fluid Mech. **858**, 122–144 (2019).
[127]R. Maulik, O. San, A. Rasheed, and P. Vedula, "Data-driven deconvolution for large eddy simulations of Kraichnan turbulence," Phys. Fluids **30**, 125109 (2018).
[128]R. Maulik, O. San, J. D. Jacob, and C. Crick, "Sub-grid scale model classification and blending through deep learning," J. Fluid Mech. **870**, 784–812 (2019).