Coarse-scale PDEs from fine-scale observations via machine learning F

Cite as: Chaos **30**, 013141 (2020); https://doi.org/10.1063/1.5126869
Submitted: 05 September 2019 . Accepted: 12 December 2019 . Published Online: 24 January 2020

Seungjoon Lee, Mahdi Kooshkbaghi, Konstantinos Spiliotis, Constantinos I. Siettos, and Ioannis G. Kevrekidis

COLLECTIONS

Paper published as part of the special topic on When Machine Learning Meets Complex Systems: Networks, Chaos and Nonlinear Dynamics
Note: This paper is part of the Focus Issue, "When Machine Learning Meets Complex Systems: Networks, Chaos and Nonlinear Dynamics."

F This paper was selected as Featured

Coarse-scale PDEs from fine-scale observations via machine learning

Cite as: Chaos **30**, 013141 (2019); doi: 10.1063/1.5126869
Submitted: 5 September 2019 · Accepted: 12 December 2019 ·
Published Online: 24 January 2020

Seungjoon Lee,[1] Mahdi Kooshkbaghi,[2] Konstantinos Spiliotis,[3] Constantinos I. Siettos,[4] and Ioannis G. Kevrekidis[1,a)]

AFFILIATIONS

[1]Department of Chemical and Biomolecular Engineering, Johns Hopkins University, Baltimore, Maryland 21218, USA
[2]Program in Applied and Computational Mathematics, Princeton University, Princeton, New Jersey 08544, USA
[3]Institute of Mathematics, University of Rostock, 18051 Rostock, Germany
[4]Dipartimento di Matematica e Applicazioni “Renato Caccioppoli,” Universitá degli Studi di Napoli Federico II, Corso Umberto I, 40, 80138 Napoli NA, Italy

Note: This paper is part of the Focus Issue, “When Machine Learning Meets Complex Systems: Networks, Chaos and Nonlinear Dynamics.”
[a)]**Also at:** Department of Applied Mathematics and Statistics and Department of Medicine, Johns Hopkins University, Baltimore, Maryland 21218, USA. **Electronic mail:** yannisk@jhu.edu.

ABSTRACT

Complex spatiotemporal dynamics of physicochemical processes are often modeled at a microscopic level (through, e.g., atomistic, agent-based, or lattice models) based on first principles. Some of these processes can also be successfully modeled at the macroscopic level using, e.g., partial differential equations (PDEs) describing the evolution of the right few macroscopic observables (e.g., concentration and momentum fields). Deriving good macroscopic descriptions (the so-called “closure problem”) is often a time-consuming process requiring deep understanding/intuition about the system of interest. Recent developments in data science provide alternative ways to effectively extract/learn accurate macroscopic descriptions approximating the underlying microscopic observations. In this paper, we introduce a data-driven framework for the identification of unavailable coarse-scale PDEs from microscopic observations via machine-learning algorithms. Specifically, using Gaussian processes, artificial neural networks, and/or diffusion maps, the proposed framework uncovers the relation between the relevant macroscopic space fields and their time evolution (the right-hand side of the explicitly unavailable macroscopic PDE). Interestingly, several choices equally representative of the data can be discovered. The framework will be illustrated through the data-driven discovery of macroscopic, concentration-level PDEs resulting from a fine-scale, lattice Boltzmann level model of a reaction/transport process. Once the coarse evolution law is identified, it can be simulated to produce long-term macroscopic predictions. Different features (pros as well as cons) of alternative machine-learning algorithms for performing this task (Gaussian processes and artificial neural networks) are presented and discussed.

The behavior of microscopic complex systems is often described in terms of effective, macroscopic governing equations, leading to simple and efficient prediction. Yet, the discovery/derivation of such macroscopic governing equations generally relies on deep understanding and prior knowledge about the system, as well as extensive and time-consuming mathematical justification. Recent developments in data-driven computational approaches suggest alternative ways toward uncovering useful coarse-scale governing equations directly from fine-scale data. Interestingly, even deciding what the “right” coarse-scale variables are may present a significant challenge. In this paper, we introduce and implement a framework for systematically extracting coarse-scale observables from microscopic/fine-scale data and for discovering the underlying governing equations using machine-learning techniques (e.g., Gaussian processes and artificial neural networks) enhanced by feature selection methods. Intrinsic representations of the coarse-scale behavior via manifold learning techniques (in particular, diffusion maps) generating alternative possible forms of the governing equations are also explored and discussed.

I. INTRODUCTION

The successful description of the spatiotemporal evolution of complex systems typically relies on detailed mathematical models operating at a fine scale (e.g., molecular dynamics, agent-based, stochastic, or lattice-based methods). Such microscopic, first-principles models, keeping track of the interactions between huge numbers of microscopic level degrees of freedom, typically lead to prohibitive computational cost for large-scale spatiotemporal simulations.

To address this issue (and since we are typically interested in macroscale features—pressure drops, reaction rates—rather than the position and velocity of each individual molecule), reduced, coarse-scale models are developed and used, leading to significant computational savings in large-scale spatiotemporal simulations.[1]

Macroscopically, the fine-scale processes may often be successfully modeled using partial differential equations (PDEs) in terms of the right macroscopic observables ("coarse variables": not molecules and their velocities, say, but rather pressure drops and momentum fields). Deriving the macroscopic PDE that effectively models the microscopic physics (the so-called "closure problem") requires, however, deep understanding/intuition about the complex system of interest and often extensive mathematical operations; the discovery of macroscopic governing equations is typically a difficult and time-consuming process.

To bypass the first-principles discovery of a macroscopic PDE directly, several data-driven approaches provide ways to effectively determine good coarse observables and the approximate coarse-scale relations between them from simulation data. In the early 1990s, researchers (including our group) employed artificial neural networks for system identification (both lumped and distributed).[2–6] Projective time integration in dynamical systems[7] and fluid dynamics[8,9] also provides a good data-driven approximation of long-time prediction based not on closed-form equations, but rather on a "black box" simulator. Furthermore, the equation-free framework for designing fine-scale computational experiments and systematically processing their results through "coarse time-steppers" has proven its usefulness/computational efficiency in analyzing macroscopic bifurcation diagrams.[10,11] The easy availability of huge simulation data sets, and recent developments in the efficient implementation of machine-learning algorithms, has made the revisiting of the identification of nonlinear dynamical systems from simulation time series an attractive—and successful—endeavor. Working with observations at the macroscopic level, hidden macroscopic PDEs can be recovered directly by artificial neural networks[6] (see also Ref. 12). Sparse identification of nonlinear dynamics (SINDy)[13] as well as Gaussian processes[14,15] have also been successfully used, resulting in *explicit* data-driven PDEs. All these approaches rely on macroscopic observations.

In this paper, we discuss the identification of unavailable coarse-scale PDEs *from fine-scale observations* through a combination of machine learning and manifold learning algorithms. Specifically, using Gaussian processes, artificial neural networks, and/or diffusion maps, and starting with candidate coarse fields (e.g., densities), our procedure extracts relevant macroscopic features (e.g., coarse derivatives) from the data and then uncovers the relations between these macroscopic features and their time evolution (the right-hand side of the explicitly unavailable macroscopic PDE).

To effectively reduce the input data domain, we employ two feature selection methods: (1) a sensitivity analysis via Automatic Relevance Determination (ARD)[16–18] in Gaussian processes and (2) a manifold learning technique, diffusion maps.[19] Having selected the relevant macrofeatures in terms of which the evolution can be modeled, we employ two machine-learning algorithms to approximate a "good" right-hand side of the underlying PDEs: (1) Gaussian process regression and (2) artificial neural networks.

Our framework is illustrated through the data-driven discovery of the macroscopic, concentration-level PDE resulting from a fine-scale, lattice Boltzmann (LB) model of a reaction/transport process [the FitzHugh-Nagumo (FHN) process in one spatial dimension]. Long-term macroscopic prediction is enabled by numerical simulation of the coarse-scale PDE *identified from the lattice Boltzmann data*. Different possible feature combinations (leading to different realizations of the same evolution) will also be discussed.

The remainder of the paper is organized as follows: In Sec. II, we present an overview of our proposed framework and briefly review theoretical concepts of Gaussian process regression, artificial neural networks, and diffusion maps. Two methods for feature selection are also presented. In Sec. III, we describe two simulators at different scales: (1) the FitzHugh-Nagumo model at the macroscale and (2) its lattice Boltzmann realization at the microscale. In Sec. IV, we demonstrate the effectiveness of our proposed framework and discuss the advantages and challenges of different feature selection methods and regression models for performing this task. In Sec. V, we summarize our results and discuss open issues for further development of the data-driven discovery of the underlying coarse PDE from microscopic observations.

II. FRAMEWORK FOR RECOVERING A COARSE-SCALE PDE VIA MACHINE LEARNING

A. Overview

The workflow of our framework for recovering hidden coarse-scale PDEs from microscopic observations is schematically shown in Figs. 1 and 2. Specifically, this framework consists of two subsections: (1) computing coarse-scale observables and (2) identifying coarse-scale PDEs and then numerically simulating them.

To clarify the algorithm, consider a single field (the activator u; later in this paper, we will use two densities, u and v, for the activator and the inhibitor, respectively). As shown in Fig. 1, we compute the coarse-scale observable (here, the u concentration field) through the zeroth moment of the microscopic LB simulation [averaging the particle distribution functions (f_i) on a given lattice point, see Sec. III B for more details].

Given the coarse-scale observable, we estimate its time derivative and several of its spatial derivatives (e.g., u_t, u_x, and u_{xx}), typically using finite difference schemes in time and space as necessary. A PDE of the form $u_t = L(u) = F(u, u_x, u_{xx}, \ldots)$ is a relation between the time derivative and a number of spatial derivatives; this relation holds at every moment in time and every point in space. For a simple reaction-diffusion equation, say, $u_t = u_{xx} - u$, the data triplets (u, u_t, u_{xx}) will in general lie on a two-dimensional

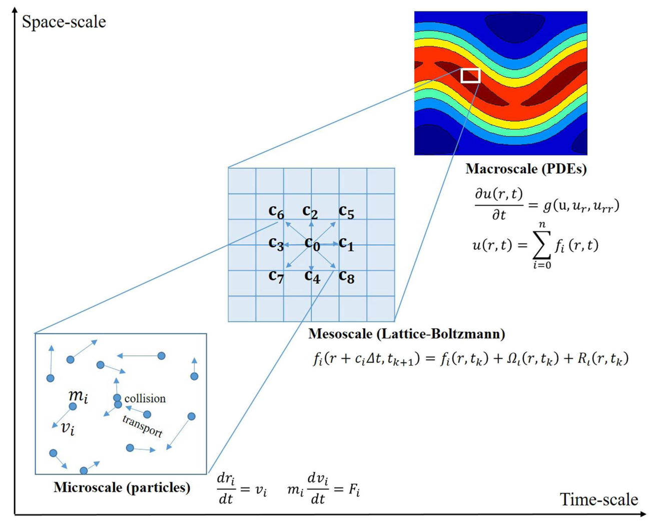

Space-scale
Macroscale (PDEs)
$\frac{\partial u(r,t)}{\partial t} = g(u, u_r, u_{rr})$
$u(r,t) = \sum_{i=0}^{n} f_i(r,t)$
c_6 c_2 c_5 c_3 c_0 c_1 c_7 c_4 c_8
Mesoscale (Lattice-Boltzmann)
$f_i(r + c_i\Delta t, t_{k+1}) = f_i(r, t_k) + \Omega_i(r, t_k) + R_i(r, t_k)$
m_i
v_i
collision
transport
Microscale (particles)
$\frac{dr_i}{dt} = v_i \quad m_i\frac{dv_i}{dt} = F_i$
Time-scale

FIG. 1. Schematic illustration of the extraction of coarse-scale observables u from microscopic observations. Through a lattice Boltzmann model (here, D2Q9), we obtain particle distribution functions (f_i) on a given lattice. Using the zeroth moment field of the particle distribution function at the grid point x_n, we extract the coarse observable u (in this paper, we have two coarse observables, u and v, which represent the density of the activator and the inhibitor, respectively).

manifold in a three-dimensional space, since u_t is a function of u and u_{xx}. Knowing that this manifold is two-dimensional suggests (in the spirit of the Whitney and Takens embedding theorems[20,21]) that any five generic observables suffice to create an embedding—and thus learn u_t, a function on the manifold, as a function of these five observables. One might choose, for example, as observables the values of u at any five spatial points at a given time moment, possibly the five points used in a finite difference stencil for estimating spatial derivatives. In the study of time series through delay embeddings, one uses observations on a temporal stencil; it is interesting that here, one might use observations on a spatial stencil—encoding information analogous to spatial derivatives (see Ref. 12). Motivated by this perspective, in order to learn the time derivative u_t, we use an original input data domain including several (say, all up to some order) spatial derivatives. We also consider the selection of a reduced input data domain via two feature selection methods: (1) a sensitivity analysis by automatic relevance determination (ARD) in Gaussian processes[16,18,22] and (2) a manifold learning approach, diffusion maps,[23,24] with a regression loss (see Sec. IV B in more detail). Then, we consider two different machine-learning methods (Gaussian process regression and artificial neural networks) to learn u_t based on the selected feature input data domain.

After training, simulation of the learned coarse-scale PDE given a coarse initial condition $u_0(x,t), v_0(x,t)$ can proceed with any acceptable discretization scheme in time and space (from simple finite differences to, say, high order spectral or finite element methods).

B. Gaussian process regression

One of the two approaches we employ to extract dominant features and uncover the RHS of coarse-scale PDEs is Gaussian process regression. In Gaussian processes, to represent a probability distribution over target functions (here, the time derivative), we assume that our observations are a set of random variables whose finite collections have a multivariate Gaussian distribution with an *unknown*

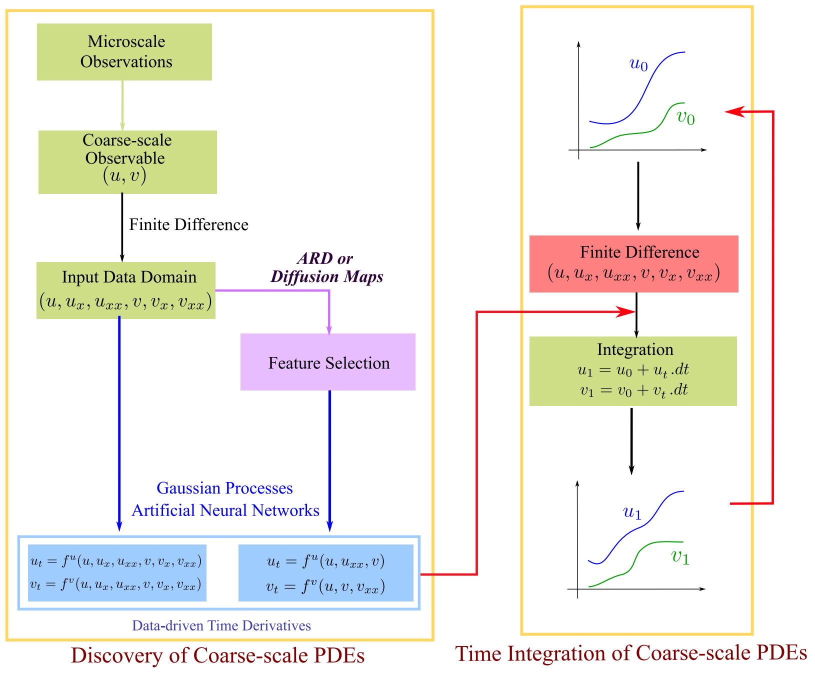

Microscale Observations
Coarse-scale Observable (u, v)
Finite Difference
Input Data Domain $(u, u_x, u_{xx}, v, v_x, v_{xx})$
ARD or Diffusion Maps
Feature Selection
Gaussian Processes
Artificial Neural Networks
$u_t = f^u(u, u_x, u_{xx}, v, v_x, v_{xx})$
$v_t = f^v(u, u_x, u_{xx}, v, v_x, v_{xx})$
$u_t = f^u(u, u_{xx}, v)$
$v_t = f^v(u, v, v_{xx})$
Data-driven Time Derivatives
Discovery of Coarse-scale PDEs
u_0
v_0
Finite Difference $(u, u_x, u_{xx}, v, v_x, v_{xx})$
Integration
$u_1 = u_0 + u_t.dt$
$v_1 = v_0 + v_t.dt$
u_1
v_1
Time Integration of Coarse-scale PDEs

FIG. 2. Workflow for uncovering coarse-scale PDEs. First, we compute macroscopic variables u and v from the lattice Boltzmann simulation data [see Eq. (18) and Fig. 1] and estimate their spatial derivatives (e.g., by finite difference schemes on the lattice). After that, we employ machine-learning algorithms (here, Gaussian process regression or artificial neural networks) to identify "proper" time derivatives u_t and v_t from an original input data domain directly (no feature selection among several spatial derivatives) or from a reduced input data domain (feature selection among several spatial derivatives) using ARD in Gaussian processes or diffusion maps. We then simulate the identified coarse-scale PDE for given coarse initial conditions ($\mathbf{u_0}, \mathbf{v_0}$).

mean (usually set to zero) and an *unknown* covariance matrix K. This covariance matrix is commonly formulated by a Euclidean distance-based kernel function κ in the input space, whose hyperparameters are optimized by training data. Here, we employ a radial basis kernel function (RBF), which is the *de facto* default kernel function in Gaussian process regression, with ARD,[17]

$$K_{ij} = \kappa(\mathbf{x_i}, \mathbf{x_j}; \theta) = \theta_0 \exp\left(-\frac{1}{2}\sum_{l=1}^{k}\frac{(x_{i,l} - x_{j,l})^2}{\theta_l}\right), \tag{1}$$

where $\theta = [\theta_0, \ldots, \theta_k]^T$ is a $k+1$ dimensional vector of hyperparameters and k is the number of dimensions of the input data domain. The optimal hyperparameter set θ^* can be obtained by minimizing a negative log marginal likelihood with the training data set $\{\mathbf{x}, \mathbf{y}\}$,

$$\begin{aligned}\theta^* &= \text{argmin}_\theta\{-\log p(\mathbf{y}|\mathbf{x}, \theta)\}\\ &= \frac{1}{2}\mathbf{y}^T(K + \sigma^2 I)^{-1}\mathbf{y} + \frac{1}{2}\log|(K + \sigma^2 I)| + \frac{N}{2}\log 2\pi,\end{aligned} \tag{2}$$

where N is the number of training data points and σ^2 and I represent the variance of the (Gaussian) observation noise and an $N \times N$ identity matrix, respectively.

To find the Gaussian distribution of the function values (here, the time derivative) at test data points, we represent the multivariate Gaussian distribution with the covariance matrix constructed by Eq. (1) as

$$\begin{bmatrix}\mathbf{y}\\ \mathbf{y}^*\end{bmatrix} = N\left(\mathbf{0}, \begin{bmatrix}K + \sigma^2 I & K_*\\ K_*^T & K_{**}\end{bmatrix}\right), \tag{3}$$

where $\mathbf{y}^*$ is a predictive distribution for test data $\mathbf{x}^*$ and K_* represents a covariance matrix between training and test data while K_{**} represents a covariance matrix between test data.

Finally, we represent a Gaussian distribution for time derivatives at the test point in terms of a predictive mean and its variance, through conditioning a multivariate Gaussian distribution as

$$\bar{\mathbf{y}}^* = K_*(K + \sigma^2 I)^{-1}\mathbf{y}, \rightarrow \bar{\mathbf{y}}^* = K_*^T(K + \sigma^2 I)^{-1}\mathbf{y}, \tag{4}$$

$$K(\mathbf{y}^*) = K_{**} - K_*^T(K + \sigma^2 I)^{-1}K_*, \tag{5}$$

and we assign the predictive mean ($\bar{\mathbf{y}}^*$) as the estimated time derivative for the corresponding data point.

C. Artificial neural networks

Next, we consider (artificial, possibly deep) neural networks (ANNs or NNs or DNNs) for identifying the RHS of coarse-scale PDEs. Generally, neural networks consist of an input layer, one or more hidden layers, and an output layer, all comprising several computational neurons, typically fully connected by weights (ω), biases (b), and an activation function [$\psi(\cdot)$]. Macroscopic observables and their spatial derivatives are assigned at the input layer, while the corresponding time derivative is obtained at the output layer (here, we are considering only first order PDEs in time; higher order equations, like the wave equation, involving second derivatives in time can also be accounted for within the framework). In (feedforward) neural networks, a universal approximation theorem[25] guarantees that for a single hidden layer with (sufficient) a finite number of neurons, an approximate realization $\tilde{y}$ of the target function, y, can be found. Here, approximation implies that the target and learned functions are sufficiently close in an appropriately chosen norm ($\forall\delta > 0 : \|y - \tilde{y}\| < \delta$). The approximate form of the target function obtained through the feedforward neural net can be written as

$$\tilde{y}(\mathbf{x}) = \sum_{i=1}^{N}\psi\left(\boldsymbol{\omega}_i^T\mathbf{x} + b_i\right). \tag{6}$$

The root-mean-square error cost function,

$$E_D = \frac{1}{N}\sum_{j=1}^{N}(y_j - \tilde{y}(x_j))^2, \tag{7}$$

typically measures the goodness of the approximation.

In order to obtain a *generalizable* network, with good performance on the test data set as well as on the training data set (e.g., preventing overfitting), several regularization approaches have been proposed, mostly relying on modifications of the cost function. Foresee and Hagan[26] showed that modifying the cost function by adding the regularization term $E_\omega = \Sigma_{j=1}^{N_\omega}\omega_j^2$ results in a network that will maximize the posterior probability based on Bayes' rule. We thus trained our network based on a total cost function of the form

$$E_{total} = \beta_1 E_D + \beta_2 E_\omega, \tag{8}$$

in which β_1 and β_2 are network tuning parameters. Here, we employ Bayesian regularized backpropagation for training, which updates weight and bias values through Levenberg-Marquardt optimization;[27] we expect that, for our data, comparable results would be obtained through other modern regularization/optimization algorithms.

D. Diffusion maps

Diffusion maps (DMAPs) have successfully been employed for dimensionality reduction and nonlinear manifold learning.[23,24,28,29] The diffusion maps algorithm can guarantee, for data lying on a smooth manifold—and at the limit of infinite data—that the eigenvectors of the large normalized kernel matrices constructed from the data converge to the eigenfunctions of the Laplace-Beltrami operator on the manifold on which the data lie. These eigenfunctions can also provide nonlinear parametrizations (i.e., sets of coordinates) for such (Riemannian) manifolds. To approximate the Laplace-Beltrami operator from scattered data points on the manifold, a normalized diffusion kernel matrix between observation (data) points is commonly used,

$$\mathbf{W_{ij}} = \exp\left(-\frac{\|\mathbf{y_i} - \mathbf{y_j}\|_2^2}{\varepsilon}\right), \tag{9}$$

where $\mathbf{y_i}$ are real-valued observations and ε is the kernel width. After that, one obtains a normalized matrix $\mathbf{W}^{(\alpha)}$ by

$$\mathbf{W}^{(\alpha)} = \mathbf{D}^{-\alpha}\mathbf{W}\mathbf{D}^{-\alpha}, \tag{10}$$

where $\mathbf{D}$ is a diagonal matrix whose ith entry is the sum of corresponding row of W. Here, $\alpha \in \{0, 1\}$ is a tuning parameter: $\alpha = 0$

corresponds to the classical normalized graph Laplacian,[30,31] while $\alpha = 1$, which takes into account the local data density, yields the Laplace-Beltrami operator;[24] in this paper, we set $\alpha = 1$. Then, $\tilde{\mathbf{W}}$ is calculated simply as

$$\tilde{\mathbf{W}} = \tilde{\mathbf{D}}^{-1}\mathbf{W}^{(\alpha)}, \tag{11}$$

where $\tilde{\mathbf{D}}$ is a diagonal matrix whose ith entry is the sum of the corresponding row of $\mathbf{W}^{(\alpha)}$.

Finally, an embedding of the manifold is constructed by the first few (say, m) nontrivial eigenvectors of $\tilde{\mathbf{W}}$,

$$\mathbf{y_i} \mapsto \left(\lambda_1^t \phi_{1,i}, \ldots, \lambda_m^t \phi_{m,i}\right), \quad i = 1, \ldots, n, \tag{12}$$

where t corresponds to the number of diffusion steps (here, $t = 0$) with descending ordered eigenvalues λ_i.

Instead of using the Euclidean distance between the data points in the diffusion map algorithm, it has recently been proposed to use a different, kernel-based similarity metric;[32] the associated kernel-based embeddings allow for control of the distortion of the resulting embedding manifold; we will return to this and its possible implications in our Conclusions (Sec. V).

E. Feature selection

Describing the coarse-scale spatiotemporal dynamics in the form of a (translationally invariant) PDE involves learning the local field time derivatives as a function of a few, relevant local field spatial derivatives. Starting with a "full" input data domain consisting of all local field values as well as all their coarse-scale spatial derivatives (up to some order), we must extract the few "relevant" spatial derivatives as dominant features of this input data domain. Such feature selection will typically reduce the dimensionality of the input data domain. Among various feature selection methods, we employ two algorithms based on (1) sensitivity analysis via ARD in Gaussian processes[16,18,22] and (2) manifold parametrization through input-output-informed diffusion maps.[19]

First, we employ sensitivity analysis via automatic relevance determination (ARD) in Gaussian processes, which effectively reduces the number of input data dimensions. In Gaussian processes, we obtain the optimal hyperparameter set θ^* by minimizing a negative log marginal likelihood [see Eq. (2)]. ARD assigns a different hyperparameter θ_i for each input dimension d_i. As can be seen in Eq. (1), a large value of θ_i nullifies the difference between target function values along the d_i dimension, allowing us to designate this dimension as "insignificant." Practically, we select the input dimensions with relatively small θ_j to build a reduced input data domain, which can still successfully represent the approximation of the right-hand side on the underlying PDEs.

Alternatively, we employ a manifold learning technique to find the intrinsic representation of the coarse-scale PDE and then examine the relation between these intrinsic coordinates and given input features (spatial field derivatives). Specifically, diffusion maps will provide an intrinsic parametrization of the combined input-output data domain (here, $\{u_t, u, u_x, u_{xx}, v, v_x, v_{xx}\}$ for u and $\{v_t, u, u_x, u_{xx}, v, v_x, v_{xx}\}$ for v). Selecting leading intrinsic coordinates (eliminating higher harmonic eigenfunctions), we can then find the lowest-dimensional embedding space for the PDE manifold (the manifold embodying u_t and v_t as a function of the embedding intrinsic coordinates). We then test several combinations of subsets of the input domain coordinates (spatial derivatives) as to their ability to parametrize the discovered intrinsic embedding coordinates. Each set of such inputs, which successfully parametrize the intrinsic embedding coordinates (see Fig. 3), provides us a new possibility of learning a PDE formulation that describes the spatiotemporal dynamics of our observation data set.

In principle, any subset of intrinsic coordinates that successfully parametrize the manifold can be used to learn functions on the manifold and, in particular, u_t and v_t. The success of any particular subset of leading intrinsic coordinates in describing u_t and v_t is confirmed through regression, via a mean-squared-error loss (L).

Next, we investigate which set of features of the input domain (which sets of spatial derivatives) can be best used to parametrize the intrinsic embedding (and thus learn the PDE right-hand side). One can find the subset of features from a user-defined dictionary (here, spatial derivatives) to parametrize the intrinsic embedding coordinates through a linear group LASSO.[33] In this paper, we examine several combinations of input domain variables and find subsets that can minimally parametrize the intrinsic embedding; this is quantified through a total regression loss (L_T) based on a mean-squared error as

$$L_T = \left(\sum_{j=1}^{d} L_{\phi_j}^2\right)^{\frac{1}{2}}, \tag{13}$$

where L_{ϕ_j} represents a GP regression loss for representing the intrinsic coordinate ϕ_j using selected input features and d represents the number of intrinsic coordinates we chose.

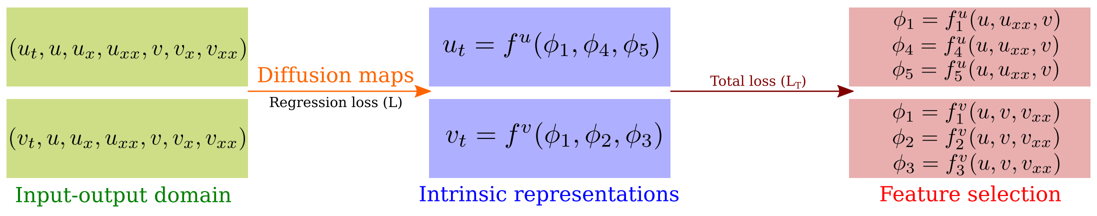

$(u_t, u, u_x, u_{xx}, v, v_x, v_{xx})$
$(v_t, u, u_x, u_{xx}, v, v_x, v_{xx})$
Diffusion maps
Regression loss (L)
$u_t = f^u(\phi_1, \phi_4, \phi_5)$
$v_t = f^v(\phi_1, \phi_2, \phi_3)$
Total loss (L_T)
$\phi_1 = f_1^u(u, u_{xx}, v)$
$\phi_4 = f_4^u(u, u_{xx}, v)$
$\phi_5 = f_5^u(u, u_{xx}, v)$
$\phi_1 = f_1^v(u, v, v_{xx})$
$\phi_2 = f_2^v(u, v, v_{xx})$
$\phi_3 = f_3^v(u, v, v_{xx})$
Input-output domain
Intrinsic representations
Feature selection

FIG. 3. Feature selection via input-output-informed diffusion maps. Diffusion maps provide intrinsic coordinatization of the output (the time derivatives) from combined input-output data. Guided by a regression loss (L), we find a low-dimensional intrinsic embedding space in which u_t (and v_t) can be represented as a function of just a few intrinsic diffusion map coordinates. After that, we search and find minimal subsets of the input data domain that can parametrize the selected intrinsic coordinates (e.g., ϕ_1, ϕ_4, ϕ_5) as quantified by a small total regression loss [see Eq. (13)].

ARD for Gaussian processes suggests the "best" input domain subset in terms of which we will try and predict u_t and v_t. In the manifold learning context, we may find several different input subsets capable of parametrizing the manifold on which the observed behavior lies. Different minimal parametrizing subsets will lead to different (but, in principle, on the data, equivalent) right-hand sides for the PDE evolution. One expects that some of them will be "better conditioned" (have better Lipschitz constants) than others.

III. DIFFERENT SCALE SIMULATORS FOR ONE-DIMENSIONAL REACTION-DIFFUSION SYSTEMS

A. Macroscale simulator: FitzHugh-Nagumo model

To describe a one-dimensional reaction-diffusion system that involves an activator u and an inhibitor v, the FitzHugh-Nagumo model consists of two coupled reaction-diffusion partial differential equations,

$$\begin{aligned}\frac{\partial u}{\partial t} &= D^u \frac{\partial^2 u}{\partial x^2} + u - u^3 - v,\\ \frac{\partial v}{\partial t} &= D^v \frac{\partial^2 v}{\partial x^2} + \epsilon(u - a_1 v - a_0),\end{aligned} \tag{14}$$

where a_1 and a_0 are model parameters, ϵ represents a kinetic bifurcation parameter, and D^u and D^v represent diffusion coefficients for u and v, respectively. Here, we set these parameters to $a_1 = 2$, $a_0 = -0.03$, $\epsilon = 0.01$, $D^u = 1$, and $D^v = 4$.[11] We discretize a spatial domain on $[0, 20]$ with $\Delta x = 0.2$ and a time domain on $[0, 450]$ with $\Delta t = 0.001$, respectively. We impose homogeneous Neumann boundary conditions at both boundaries and solve these equations (for various initial conditions) numerically via the finite element method using the COMSOL Multiphysics® software.[34]

B. Microscale simulator: The lattice Boltzmann model

We also introduce a lattice Boltzmann model (LBM),[35,36] which can be thought of as a mesoscopic numerical scheme for describing spatiotemporal dynamics using finite-difference-type discretizations of Boltzmann-Bhatnagar-Gross-Krook (BGK) equations,[37] retaining certain advantages of microscopic particle models. In this paper, the lattice Boltzmann model is our fine-scale "microscopic simulator," and its results are considered to be "the truth" from which the coarse-scale PDE will be learned.

The time evolution of the particle distribution function on a given lattice can be described by

$$f_i^l(x_{j+i}, t_{k+1}) = f_i^l(x_j, t_k) + \Omega_i^l(x_j, t_k) + R_i^l(x_j, t_k) \quad l \in \{u, v\}, \tag{15}$$

where a superscript l indicates the activator u and the inhibitor v and Ω_i^l represents a collision term defined by Bhatnagar-Gross-Krook (BGK),[37]

$$\Omega_i^l(x_j, t_k) = -\omega^l(f_i^l(x_j, t_k) - f_i^{l,eq}(x_j, t_k)), \tag{16}$$

where ω^l represents a BGK relaxation coefficient defined as[38]

$$\omega^l = \frac{2}{1 + 3D^l \frac{\Delta t}{\Delta x^2}}. \tag{17}$$

To compute our coarse-scale observables u and v, we employ the D1Q3 model, which uses three distribution functions on the one-dimensional lattice as (f_{-1}^l, f_0^l, f_1^l) for each density (totalling 6 distribution functions). Through the zeroth moment (in the velocity directions) of the overall distribution function, finally, we compute the coarse-scale observable u and v as

$$\begin{aligned}u(x_j, t_k) &= \sum_{i=-1}^{1} f_i^u(x_j, t_k),\\ v(x_j, t_k) &= \sum_{i=-1}^{1} f_i^v(x_j, t_k).\end{aligned} \tag{18}$$

Based on spatially uniform local diffusion equilibrium, for which f_i^{eq} is homogeneous in all velocity directions, the weights are chosen all equal to 1/3,

$$\begin{aligned}f_i^{u,eq}(x_j, t_k) &= \frac{1}{3}u(x_i, t_k),\\ f_i^{v,eq}(x_j, t_k) &= \frac{1}{3}v(x_i, t_k).\end{aligned} \tag{19}$$

Thus, the reaction terms R_i^l in Eq. (15) are modeled by

$$\begin{aligned}R_i^u(x_j, t_k) &= \frac{1}{3}\Delta t(u(x_j, t_k) - u(x_j, t_k)^3 - v(x_j, t_k)),\\ R_i^v(x_j, t_k) &= \frac{1}{3}\Delta t\epsilon(u(x_j, t_k) - a_1 v(x_j, t_k)^3 - a_0).\end{aligned} \tag{20}$$

All model parameters $(a_0, a_1, \epsilon, D^u, D^v)$ are the same as the FHN PDE. The corresponding spatiotemporal behavior of these coarse observables u and v is shown in Figs. 4(a) and 4(c), while the FHN PDE simulation for the same coarse initial conditions is shown in Figs. 4(b) and 4(d).

IV. RESULTS

A. Learning without feature selection

We begin by considering our proposed framework without feature selection, so as to later contrast with the results including feature selection. The data come from the fine-scale lattice Boltzmann simulation. For the parameter values selected, the long-term dynamics of the LB simulation lie, for all practical purposes, on (or very close to) a stable time-periodic solution. To predict the coarse time derivatives u_t and v_t, we collect training data from five different initial conditions near this stable periodic solution (see Fig. 5) with the following LB spatiotemporal discretization—in space, 99 discretized points on $[0.2, 19.8]$ with $dx = 0.2$ and in time, 451 discretized points on $[0, 450]$ with $dt = 1$ for each initial condition. Since our data come from the fine-scale LB code, we need to initialize at the fine, LB scale of particle distribution functions (and not just of the concentrations u and v). To initialize the particle distribution functions in the lattice Boltzmann model, we apply the equal weights rule, 1/3 for f_{-1}, f_0, and f_1, motivated by near-equilibrium considerations. We do expect that such initialization features will soon be "forgotten" as higher distribution moments become quickly slaved to the lower (here, the zeroth) moments (see, for example, Ref. 39). To ensure that our results are not affected by the initialization details, we only start collecting training data after relaxation by short time simulation (here, 2000 time steps with $\Delta t = 0.001$ or

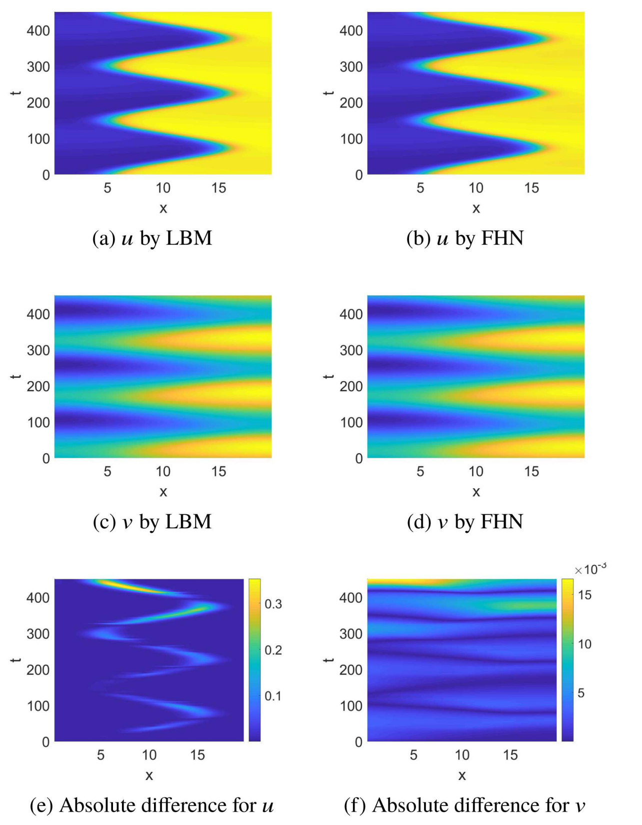


FIG. 4. Spatiotemporal behavior of u and v simulated by the lattice Boltzmann model and by the FitzHugh-Nagumo PDE. (a) and (c) u and v from the lattice Boltzmann model (LBM). (b) and (d) u and v from the FitzHugh-Nagumo PDE. (e) and (f) Normalized absolute difference between the simulations of the two models.

$t = 2$) (see the Appendix). We estimate the local coarse fields and their (several) spatial and temporal derivatives through finite differences and then apply machine-learning algorithms (here, Gaussian processes as well as neural networks) to learn the time derivatives of the activator u_t and the inhibitor v_t using as input variables the local u, v and all their spatial derivatives up to and including order two $(u, u_x, u_{xx}, v, v_x, v_{xx})$,

$$u_t(x,t) = f^u(u, u_x, u_{xx}, v, v_x, v_{xx}),$$
$$v_t(x,t) = f^v(u, u_x, u_{xx}, v, v_x, v_{xx}). \quad (21)$$

Specifically, for the neural network approach, we build two different networks, one for the prediction of the activator and one for the inhibitor. For both the activator and the inhibitor, we set 2 hidden layers each consist of 6 neurons using a hyperbolic tangent sigmoid activation function; as mentioned above, we use Levenberg-Marquardt optimization with a Bayesian regularization (see Sec. II C). Both networks use the mean-squared error as their loss function. For Gaussian processes, we employ a radial basis kernel function with ARD [see Eq. (1)]. Regression results obtained by each of the two methods for the time derivatives in the training data

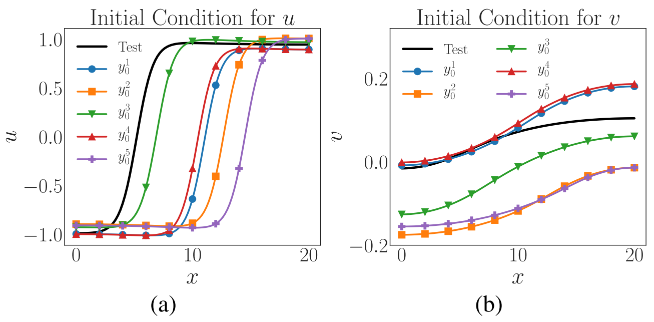


FIG. 5. Five different coarse initial conditions (y_i^0) for training and a test coarse initial condition (colored in black). (a) Coarse initial conditions for u. (b) Coarse initial conditions for v. Five initial conditions are randomly chosen near the stable periodic solution.

set are shown in Fig. 6. Both methods provide good approximations of the target time derivatives u_t and v_t. Given the test coarse initial condition (black curves in Fig. 5), simulation results *with the learned PDE* from $t = 0$ to $t = 450$ with $\Delta t = 0.001$ and their normalized absolute differences from the "ground truth" LB simulations are shown in Fig. 7. The order of magnitude of these absolute differences for both models is the same as those between the LB FHN and the explicitly known FHN PDE [see Figs. 4(e) and 4(f)].

B. Learning with feature selection

Now, we consider the possibility of feature selection in an attempt to learn the RHS of coarse-scale PDEs with a minimal number of input domain variables (spatial derivatives). First, we apply

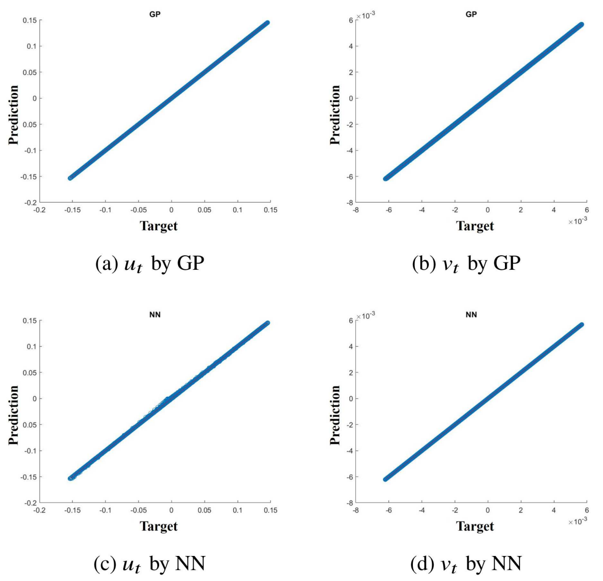
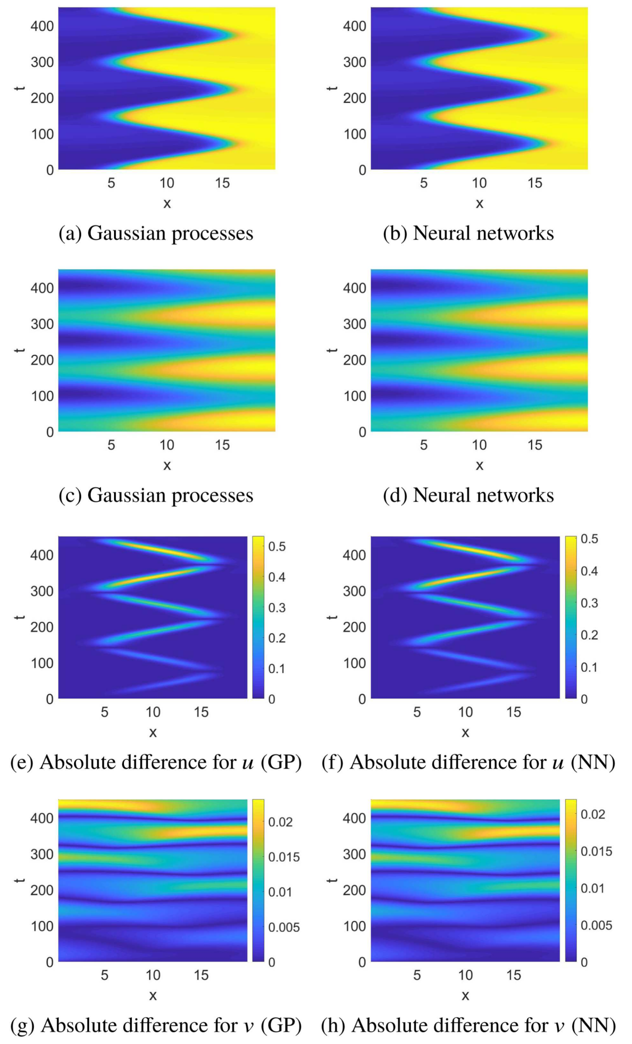


FIG. 6. No feature selection: $u_t = f^u(u, u_x, u_{xx}, v, v_x, v_{xx})$ and $v_t = f^v(u, u_x, u_{xx}, v, v_x, v_{xx})$. Regression results of the two methods for time derivatives: Gaussian processes (GPs) and neural networks (NNs).



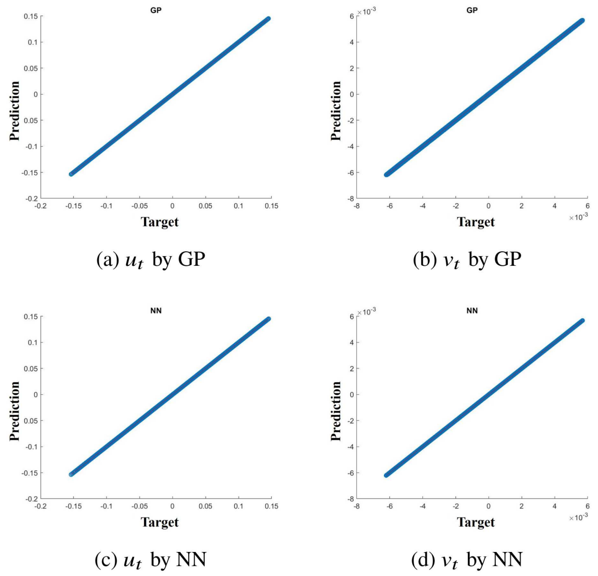
(a) Gaussian processes (b) Neural networks

(c) Gaussian processes (d) Neural networks

(e) Absolute difference for u (GP) (f) Absolute difference for u (NN)

(g) Absolute difference for v (GP) (h) Absolute difference for v (NN)

FIG. 7. No feature selection: $u_t = f^u(u, u_x, u_{xx}, v, v_x, v_{xx})$ and $v_t = f^v(u, u_x, u_{xx}, v, v_x, v_{xx})$. (a)–(d) Simulation results of the two methods for u and v. (e)–(h) The normalized absolute differences from the "ground truth" LB simulations for u and v.



(a) u_t by GP (b) v_t by GP

(c) u_t by NN (d) v_t by NN

FIG. 8. Feature selection 1: $u_t = f_1^u(u, u_{xx}, v)$ and $v_t = f_1^v(u, v, v_{xx})$. These selected variables are the same as those that appear in the right-hand side of the explicitly known FHN PDE. Regression results of the two methods for time derivatives: Gaussian processes (GPs) and neural networks (NNs).

the sensitivity analysis via ARD in the case of Gaussian process approximation. The optimal ARD weights (θ^*) for u_t and v_t are tabulated in Table I. u_t has three relatively small weights for (u, u_{xx}, v), and v_t has also three relatively small weights for (u, v, v_{xx}). It is interesting to observe that the selected features via ARD are the same as those in the explicitly known FHN PDE [see Eq. (14)]. This shows that ARD can effectively guide in selecting the appropriate dimensionality of the input data domain, resulting here in the same spatial derivative choices as in the explicitly known FHN PDE. Now, we use the reduced input data domain (u, u_{xx}, v) for u_t and (u, v, v_{xx}) for v_t to recover the RHS of the coarse-scale PDEs as

$$\begin{aligned} u_t(x,t) &= f_1^u(u, u_{xx}, v), \\ v_t(x,t) &= f_1^v(u, v, v_{xx}). \end{aligned} \tag{22}$$

Regression results of our two methods for the time derivatives are shown in Fig. 8. Results of long time simulation of the learned PDEs by each method, from $t = 0$ to $t = 450$, as well as normalized absolute differences from the simulation of the "ground truth" LB are shown in Fig. 9.

The two machine-learning methods operating with a reduced input data domain can still provide good approximations of the time derivatives and of the resulting dynamics. The order of magnitude of these absolute differences is effectively the same as the difference of the FHN LB from the explicitly known FHN PDE. It is, therefore,

TABLE I. Optimal ARD weights [θ^* for u_t and v_t in Eq. (2)]. As mentioned in Sec. II E, features that have relatively small ARD weights can be regarded as dominant features for the target functions u_t and v_t.

	u	u_x	u_{xx}	v	v_x	v_{xx}
u_t	5.28×10^{00}	4.23×10^{06}	9.13×10^{02}	2.13×10^{03}	5.32×10^{08}	4.78×10^{07}
v_t	1.33×10^{02}	6.69×10^{06}	1.94×10^{06}	5.09×10^{02}	4.20×10^{06}	1.75×10^{02}

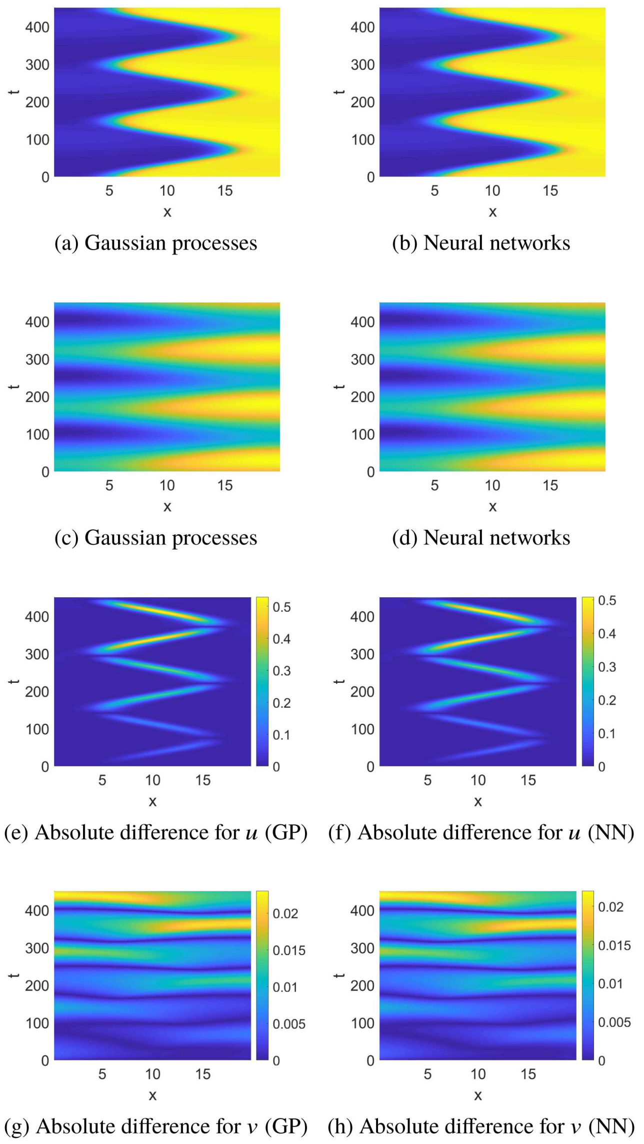


FIG. 9. Feature selection 1: $u_t = f^u(u, u_{xx}, v)$ and $v_t = f^v(u, v, v_{xx})$. (a)–(d) Simulation results of the two methods for u and v. (e)–(h) The normalized absolute differences from the "ground truth" LB simulations for u and v.

clear that our framework effectively recovers the coarse-scale PDE from fine-scale observation data; the difference is that the right-hand side of the PDE is now given in terms of the ANN right-hand side, or in terms of the observed data and the GP kernel/hyperparameters, rather than the simple algebraic formula of Eq. (14).

Next, we consider an alternative approach for feature selection, via our manifold learning technique, diffusion maps. The best candidate set among different combinations of intrinsic coordinates (varying the number of leading independent intrinsic dimensions and recording the corresponding Gaussian process regression loss) is shown in Table II. Since the three-dimensional intrinsic embedding space exhibits a (tiny) regression loss of order 10^{-8}, we choose an input domain for u_t consisting of $(\phi_1^u, \phi_4^u, \phi_5^u)$ as shown in Fig. 10(a). For v_t, by the same token, we choose the three-dimensional embedding space consisting of $(\phi_1^v, \phi_2^v, \phi_3^v)$ as shown in Fig. 10(b). Based on these identified intrinsic embedding spaces, we examined several subsets of input domain features (spatial derivatives) using the total loss of Eq. (13). "Good" subsets of input features (those that result in small regression losses with minimal input dimension) are presented in Table III. Clearly, different choices of such input feature subsets can give comparable total losses; this suggests that we may construct different right-hand sides of the unknown coarse-scale

TABLE II. The best candidates and the corresponding regression loss (L) for u_t and v_t with respect to the number of diffusion map coordinates.

	Optimal intrinsic coordinates		Regression loss (L)	
	u_t	v_t	u_t	v_t
1d	(ϕ_5^u)	(ϕ_2^v)	4.60×10^{-04}	7.69×10^{-06}
2d	(ϕ_1^u, ϕ_5^u)	(ϕ_1^v, ϕ_2^v)	1.40×10^{-06}	1.50×10^{-06}
3d	$(\phi_1^u, \phi_4^u, \phi_5^u)$	$(\phi_1^v, \phi_2^v, \phi_3^v)$	2.18×10^{-08}	4.74×10^{-08}
4d	$(\phi_1^u, \phi_3^u, \phi_4^u, \phi_5^u)$	$(\phi_1^v, \phi_2^v, \phi_3^v, \phi_4^v)$	1.64×10^{-08}	5.71×10^{-09}

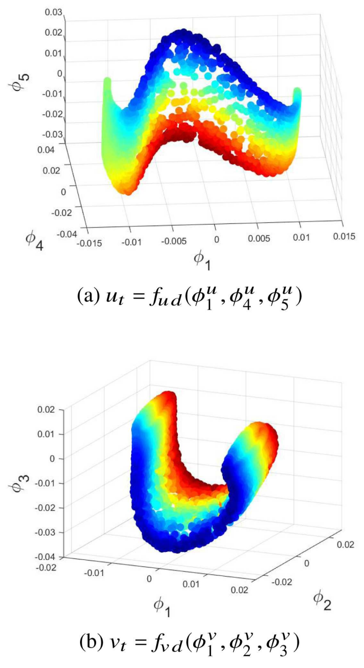


FIG. 10. Three leading diffusion map coordinates: Colors represent u_t in (a) and v_t in (b).

TABLE III. The best candidates and corresponding total loss for $u_t = (\phi_1^u, \phi_4^u, \phi_5^u)$ and $v_t = (\phi_1^v, \phi_2^v, \phi_3^v)$ with respect to the number of features.

	$u_t = (\phi_1^u, \phi_4^u, \phi_5^u)$		$v_t = (\phi_1^v, \phi_2^v, \phi_3^v)$	
	Features	Total loss (L_T)	Features	Total loss (L_T)
1d	(u)	6.51×10^{-05}	(u)	7.93×10^{-05}
2d	(u, v)	1.65×10^{-08}	(u, v)	1.49×10^{-05}
3d	(u, u_{xx}, v)	6.52×10^{-09}	(u, v, v_{xx})	3.32×10^{-07}
	(u, u_x, v)	7.39×10^{-09}	(u, u_x, v_{xx})	6.21×10^{-07}
4d	(u, u_x, u_{xx}, v)	2.68×10^{-09}	(u, v, v_x, v_{xx})	4.47×10^{-09}

PDE, which are comparably successful in representing the observed dynamics.

The good candidates for u_t and v_t identified this way, consisting of three input features, are (u, u_{xx}, v) and (u, v, v_{xx}); they are the same as those found from GPs via ARD and also the same as the ones in the explicitly known FHN PDE. Interestingly, another possible alternative candidate set is also identified: (u, u_x, v) for u_t and (u, u_x, v_{xx}) for v_t.

Using these alternative candidate feature sets, we model different "versions" of what, on the data, is effectively the same coarse-scale PDE. The "alternative" version of the PDE can be symbolically written as

$$\begin{aligned} u_t(x,t) &= f_2^u(u, u_x, v), \\ v_t(x,t) &= f_2^v(u, v, v_{xx}), \end{aligned} \tag{23}$$

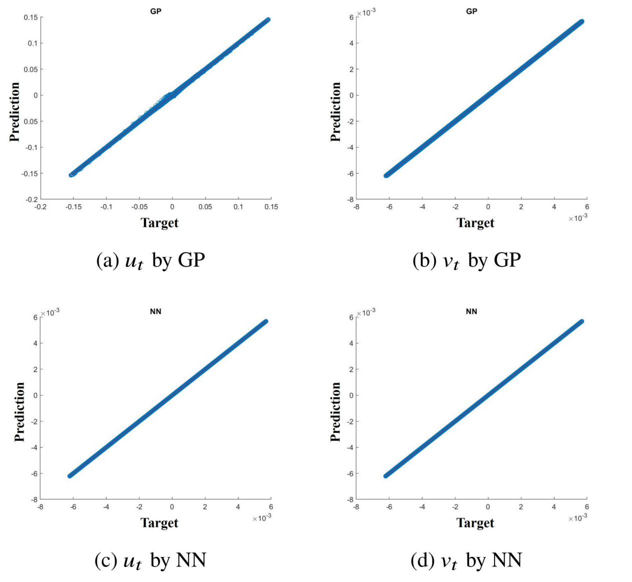


FIG. 11. Feature selection 2: $u_t = f_2^u(u, u_x, v)$ and $v_t = f_2^v(u, v, v_{xx})$. Regression results of the two methods for time derivatives: Gaussian processes (GPs) and neural networks (NNs).

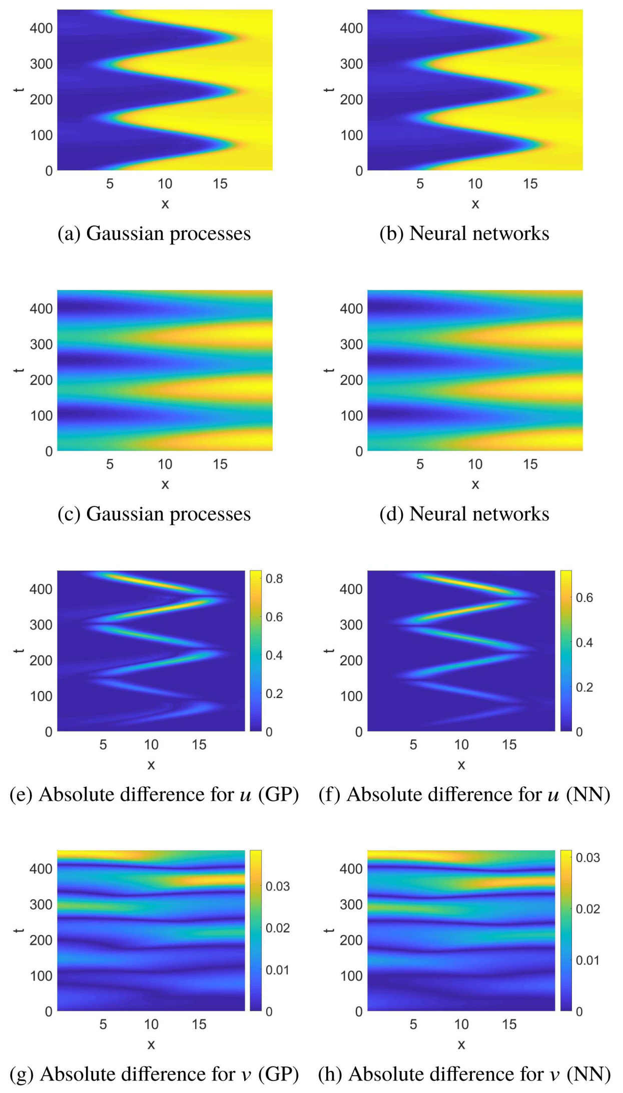


FIG. 12. Feature selection 2: $u_t = f^u(u, u_x, v)$ and $v_t = f^v(u, v, v_{xx})$. (a)–(d) Simulation results of the two methods for u and v. (e)–(h) The normalized absolute differences from the "ground truth" LB simulations for u and v.

and the corresponding regression results of the time derivatives are shown in Fig. 11. Specifically, we use the first spatial derivative u_x instead of the second spatial derivative u_{xx} for learning u_t.

As shown in Fig. 12, both models provide good predictions of the "ground truth" LB simulations; we observe, however, that the accuracy of the neural network based predictions is enhanced. These results confirm that, on the data, alternative coarse-scale PDE forms can provide successful macroscopic description.

To further explore this possibility of alternative PDE forms that represent the observed data with *qualitatively comparable accuracy*, we also explored the efficacy of a third coarse-scale PDE

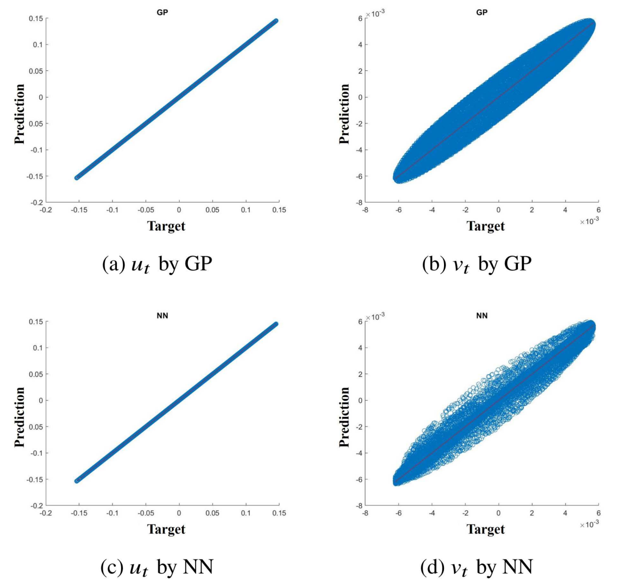


FIG. 13. Feature selection 3: $u_t = f_3^u(u, u_{xx}, v)$ and $v_t = f_3^v(u, u_x, v_{xx})$. Regression results of the two methods for time derivatives: Gaussian processes (GPs) and neural networks (NNs).

description, in terms of a yet different input feature set: (u, u_{xx}, v) for u_t and(u, u_x, v_{xx}) for v_t so that the PDE can symbolically be written as

$$u_t(x,t) = f_3^u(u, u_{xx}, v),\\ v_t(x,t) = f_3^v(u, u_x, v_{xx}). \tag{24}$$

The corresponding prediction results of the time derivatives are shown in Fig. 13.

As shown in Fig. 13, both regression methods provide an inaccurate approximation of v_t near $v_t = 0$; the order of magnitude of this error is 10^{-3}. The long-term prediction results are not as accurate representations of the ground truth LB simulation as the previous two coarse-scale PDE realizations; yet they may still be qualitatively informative. Normalized absolute differences of long-time simulation for both machine-learning methods are shown in Fig. 14. As was the case in the previous alternative PDE realizations, the NN model appears more accurate than the GP one.

To compare our identified coarse-scale PDEs with the explicitly known FHN PDE [see Eq. (14)], we also compare the predictions of our coarse-scale PDEs to those of the FHN PDE via mean normalized absolute differences for the test coarse initial condition followed from $t = 0$ to $t = 450$ as

$$\mathrm{MNAD_u} = \frac{1}{N_T}\sum_{i=1}^{99}\sum_{j=0}^{450}\frac{|u(i,j) - u_f(i,j)|}{\max(u_f) - \min(u_f)},\\ \mathrm{MNAD_v} = \frac{1}{N_T}\sum_{i=1}^{99}\sum_{j=0}^{450}\frac{|v(i,j) - v_f(i,j)|}{\max(v_f) - \min(v_f)}, \tag{25}$$

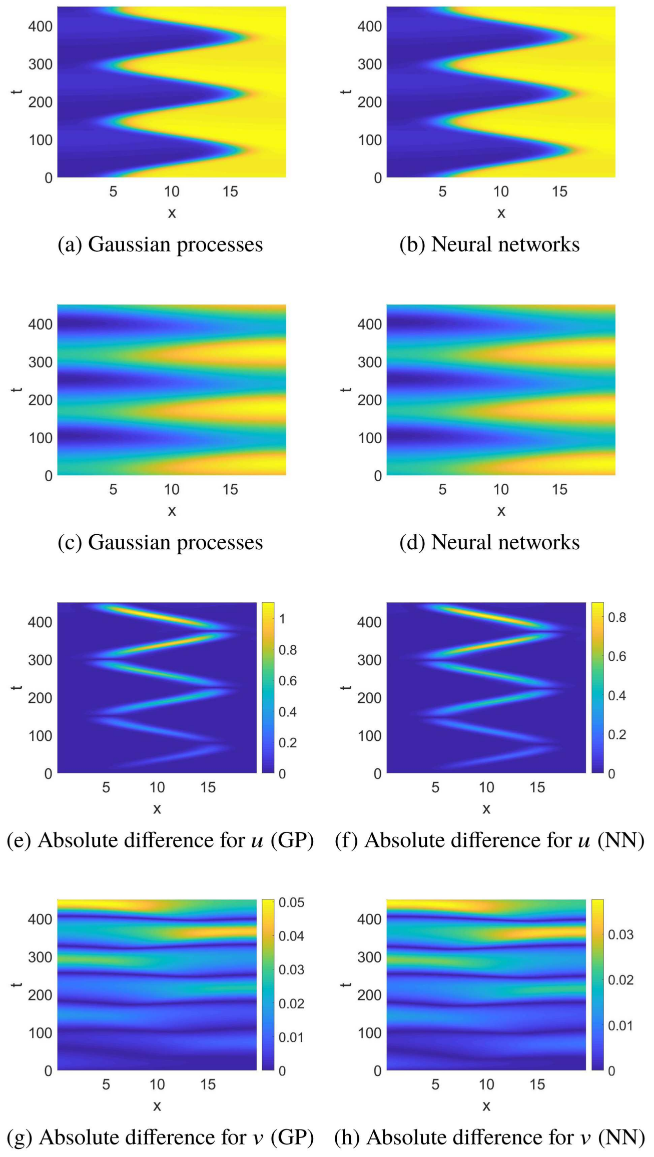


FIG. 14. Feature selection 3: $u_t = f^u(u, u_{xx}, v)$ and $v_t = f^v(u, u_x, v_{xx})$. (a)–(d) Simulation results of the two methods for u and v. (e)–(h) The normalized absolute differences from the "ground truth" LB simulations for u and v.

where N_T is the total number of data points and u_f and v_f represent simulation results of the FHN PDE, respectively. The comparison of these representative simulations of our various coarse-scale PDEs is summarized in Table IV. The differences across our various coarse-scale identified PDEs are of order 10^{-2} and below, comparable to the difference between each of them and the FHN PDE. Specifically, "feature selection 1" (Fig. 9), whose variables are the same as those of the explicit FHN PDE, provides the best PDE realization via *both* the GP and the NN models.

TABLE IV. Mean normalized absolute difference (MNAD) for different coarse-scale PDEs. "GP" and "NN" represent "Gaussian processes" and "Neural networks," respectively.

	MNAD_u	MNAD_v
No feature selection with GPs	1.59×10^{-02}	1.62×10^{-02}
No feature selection with NNs	1.53×10^{-02}	1.56×10^{-02}
Feature selection 1 with GPs	1.58×10^{-02}	1.62×10^{-02}
Feature selection 1 with NNs	1.54×10^{-02}	1.57×10^{-02}
Feature selection 2 with GPs	2.39×10^{-02}	2.20×10^{-02}
Feature selection 2 with NNs	2.00×10^{-02}	2.11×10^{-02}
Feature selection 3 with GPs	3.20×10^{-02}	3.31×10^{-02}
Feature selection 3 with NNs	2.08×10^{-02}	2.16×10^{-02}

V. CONCLUSION

In this paper, we demonstrated the data-driven discovery of macroscopic, concentration-level PDEs for reaction/transport processes resulting from fine-scale observations (here, from simulations of a lattice Boltzmann mesoscopic model). Long-term macroscopic prediction is then obtained by simulation of the identified (via machine-learning methods) coarse-scale PDE. We explored the effect of input feature selection capability on the effectiveness of our framework to identify the underlying macroscopic PDEs.

Our framework suggests four different PDEs (one without and three with feature selection), all comparable with the explicit FitzHugh-Nagumo PDE *on the data*: all of them provide good approximations of sample coarse-scale dynamic trajectories. The FHN PDE terms have a well-established mechanistic physical meaning (reaction and diffusion); it would be interesting to explore if any physical meaning can be ascribed to our alternative parametrizations of the right-hand side of the coarse-scale evolution PDE. Clearly, the identified PDE depends on our choice of observables—for example, our diffusion map embedding coordinates. We plan to explore the use of kernel-based embeddings (as discussed in Ref. 32 mentioned above) as an approach that can control the well-posedness of the embedding and the distortion of the resulting manifold; this will clearly affect the identified PDE, and it will be interesting to study the interplay between differently distorted embedding manifolds and different identified approximate PDEs.

In our framework, we employed finite differences to estimate spatial derivatives in the formulation of the PDE. Instead of numerical spatial derivatives, we may use the values of coarse observables at neighboring points directly to uncover the coarse evolution law. The effect of this alternative embedding for the PDE right-hand side, explored in Ref. 12, on the accuracy of the identified model predictions, is the subject of ongoing research.

We believe that the framework we presented is easily generalizable to multiscale and/or multifidelity data. Here, we worked across a single scale gap and a single fine-scale simulator providing the ground truth. We envision that data fusion tools can be combined with the approach to exploit data at several cascaded scales and taking advantage of simulation data from several heterogeneous simulators.[9,40]

ACKNOWLEDGMENTS

S. Lee, M. Kooshkbaghi, and I. G. Kevrekidis gratefully acknowledge partial support by the National Institutes of Health (NIH) and Defense Advanced Research Projects Agency (DARPA). Discussions of the authors with Dr. Felix Dietrich are gratefully acknowledged.

APPENDIX: "HEALING" FOR THE LATTICE BOLTZMANN MODEL

An important assumption underlying our work is that the fine-scale model can "close" at a coarse-scale level. In our particular case, this means that, even though our fine-scale Lattice Boltzmann model (LBM) evolves particle distribution functions, one can be predictive at the coarse level of the zeroth moments of these functions and the concentrations u and v of the activator and the inhibitor. The hypothesis that allows this reduction is that the problem is *singularly perturbed* in time: higher order moments of these distribution functions become quickly slaved to the "slow," governing, zeroth order moment fields. Yet, while initializing the FHN PDE only requires spatial profiles of u and v at the initial time, initializing the full FHN LBM requires initial conditions for *all* the evolving particle distributions. Constructing such detailed fine-scale initializations consistent with the coarse initial conditions is an important conceptual component of equation-free computation; the term used is "lifting."[7,41]

Here, in lifting the coarse-scale observable (a concentration field ρ) to the microscopic description (particle distribution function f) on each node, we employ an equal weight rule, i.e., the three particle distribution function values at the node x_n are chosen to be

$$f_{-1}(x_n) = f_0(x_n) = f_1(x_n) = \frac{\rho(x_n)}{3}. \tag{A1}$$

This equal weight choice (the local, spatially uniform diffusive equilibrium distribution) is not, in general, consistent with the (spatially

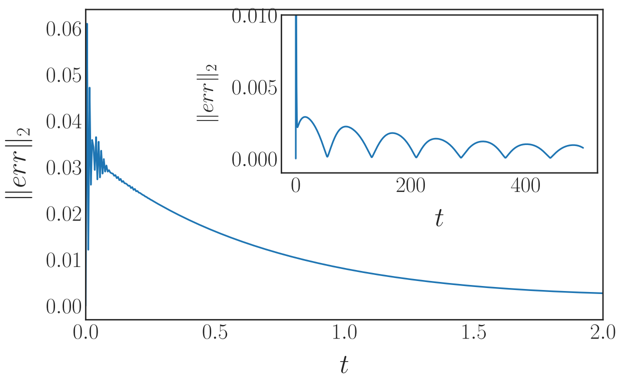


FIG. 15. Evolution of the L^2 norm [see Eq. (A2)] of the coarse difference between trajectories with the same coarse but different fine initial conditions. After the initial small (but violent) oscillation in error abates (for $t \lesssim 0.1$), the perturbed system relaxes to the vicinity of the base solution over $t \approx 2$.

nonuniform and not simply diffusive) macroscopic PDE model (here, the FitzHugh-Nagumo PDEs); yet, we expect that the fine-scale simulation features will become rapidly slaved to the local concentration field.[39] To estimate the appropriate slaving/relaxation time, we compare the L^2 norm of the density predicted by two differently initialized LBM simulations: the one that lies on the long-term stable limit cycle and the one that results from it by retaining the coarse state, but perturbing the associated fine-scale states according to the local diffusive equilibrium $\frac{1}{3}$ rule [for $\epsilon = 0.01$ in Eq. (20)].

To explore the slaving time scale, we trace the L^2 norm of the difference between the simulations resulting from these two initializations. This L^2 norm is defined as

$$\|err\|_2 = \|\rho_{eq}(x,t) - \rho(x,t)\|_2, \tag{A2}$$

where ρ_{eq} and ρ represent the density with equal weights and the density without equal weights (reference solution), respectively. As shown in Fig. 15, after a fast transient oscillation of L^2 (for $t < 0.1$), the norm decays smoothly until $t \approx 2$. There is still a small inherent bias (the trajectory will come back to a *nearby* point along the limit cycle); this does not affect our estimate of the slaving time. We, therefore, chose a relaxation time $t = 2$ (or 2000 LB time steps), and we started collecting coarse observations as training data from our various initializations only after $t = 2$.

REFERENCES

[1]W. Noid, "Perspective: Coarse-grained models for biomolecular systems," J. Chem. Phys. **139**, 090901 (2013).
[2]J. Hudson, M. Kube, R. Adomaitis, I. Kevrekidis, A. Lapedes, and R. Farber, "Nonlinear signal processing and system identification: Applications to time series from electrochemical reactions," Chem. Eng. Sci. **45**, 2075–2081 (1990).
[3]K. Krischer, R. Rico-Martinez, I. Kevrekidis, H. Rotermund, G. Ertl, and J. Hudson, "Model identification of a spatiotemporally varying catalytic reaction," AIChE J. **39**, 89–98 (1993).
[4]R. Rico-Martinez, J. Anderson, and I. Kevrekidis, "Continuous-time nonlinear signal processing: A neural network based approach for gray box identification," in *Proceedings of IEEE Workshop on Neural Networks for Signal Processing* (IEEE, 1994), pp. 596–605.
[5]J. Anderson, I. Kevrekidis, and R. Rico-Martinez, "A comparison of recurrent training algorithms for time series analysis and system identification," Comput. Chem. Eng. **20**, S751–S756 (1996).
[6]R. Gonzalez-Garcia, R. Rico-Martinez, and I. Kevrekidis, "Identification of distributed parameter systems: A neural net based approach," Comput. Chem. Eng. **22**, S965–S968 (1998).
[7]I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidid, O. Runborg, and C. Theodoropoulos, "Equation-free, coarse-grained multiscale computation: Enabling microscopic simulators to perform system-level analysis," Commun. Math. Sci. **1**, 715–762 (2003).
[8]S. Sirisup, G. E. Karniadakis, D. Xiu, and I. G. Kevrekidis, "Equation-free/Galerkin-free pod-assisted computation of incompressible flows," J. Comput. Phys. **207**, 568–587 (2005).
[9]S. Lee, I. G. Kevrekidis, and G. E. Karniadakis, "A resilient and efficient CFD framework: Statistical learning tools for multi-fidelity and heterogeneous information fusion," J. Comput. Phys. **344**, 516–533 (2017).
[10]C. Siettos, C. Gear, and I. Kevrekidis, "An equation-free approach to agent-based computation: Bifurcation analysis and control of stationary states," Europhys. Lett. **99**, 48007 (2012).
[11]C. Theodoropoulos, Y.-H. Qian, and I. G. Kevrekidis, ""Coarse" stability and bifurcation analysis using time-steppers: A reaction-diffusion example," Proc. Natl. Acad. Sci. U.S.A. **97**, 9840–9843 (2000).
[12]Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner, "Learning data-driven discretizations for partial differential equations," Proc. Natl. Acad. Sci. U.S.A. **116**, 15344–15349 (2019).
[13]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).
[14]M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Machine learning of linear differential equations using Gaussian processes," J. Comput. Phys. **348**, 683–693 (2017).
[15]M. Raissi and G. E. Karniadakis, "Hidden physics models: Machine learning of nonlinear partial differential equations," J. Comput. Phys. **357**, 125–141 (2018).
[16]Y. A. Qi, T. P. Minka, R. W. Picard, and Z. Ghahramani, "Predictive automatic relevance determination by expectation propagation," in *Proceedings of the Twenty-First International Conference on Machine Learning* (ACM, 2004), p. 85.
[17]C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (MIT Press, 2006).
[18]D. P. Wipf and S. S. Nagarajan, "A new view of automatic relevance determination," in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., 2008), pp. 1625–1632.
[19]A. Holiday, M. Kooshkbaghi, J. M. Bello-Rivas, C. W. Gear, A. Zagaris, and I. G. Kevrekidis, "Manifold learning for parameter reduction," J. Comput. Phys. **392**, 419–431 (2019).
[20]H. Whitney, "Differentiable manifolds," Ann. Math. **37**, 645–680 (1936).
[21]F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980*, edited by D. Rand and L.-S. Young (Springer, Berlin, 1981), pp. 366–381.
[22]C. K. Williams and C. E. Rasmussen, "Gaussian processes for regression," in *Advances in Neural Information Processing Systems* (MIT Press, 1996), pp. 514–520.
[23]R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker, "Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps," Proc. Natl. Acad. Sci. U.S.A. **102**, 7426–7431 (2005).
[24]R. R. Coifman and S. Lafon, "Diffusion maps," Appl. Comput. Harmon. Anal. **21**, 5–30 (2006).
[25]G. Cybenko, "Approximation by superpositions of a sigmoidal function," Math. Control Signals Syst. **2**, 303–314 (1989).
[26]F. D. Foresee and M. T. Hagan, "Gauss-Newton approximation to Bayesian learning," in *Proceedings of International Conference on Neural Networks (ICNN'97)* (IEEE, 1997), Vol. 3, pp. 1930–1935.
[27]M. T. Hagan and M. B. Menhaj, "Training feedforward networks with the Marquardt algorithm," IEEE Trans. Neural Netw. **5**, 989–993 (1994).
[28]B. Nadler, S. Lafon, R. R. Coifman, and I. G. Kevrekidis, "Diffusion maps, spectral clustering and reaction coordinates of dynamical systems," Appl. Comput. Harmon. Anal. **21**, 113–127 (2006).
[29]B. Nadler, S. Lafon, R. Coifman, and I. G. Kevrekidis, "Diffusion maps—A probabilistic interpretation for spectral embedding and clustering algorithms," in *Principal Manifolds for Data Visualization and Dimension Reduction* (Springer, 2008), pp. 238–260.
[30]M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems* (MIT Press, 2002), pp. 585–591.
[31]M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," Neural Comput. **15**, 1373–1396 (2003).
[32]A. Bittracher, S. Klus, B. Hamzi, and C. Schütte, "A kernel-based method for coarse graining complex dynamical systems," e-print arXiv:1904.08622 (2019).
[33]M. Meila, S. Koelle, and H. Zhang, "A regression approach for explaining manifold embedding coordinates," e-print arXiv:1811.11891.
[34]COMSOL, COMSOL Multiphysics (COMSOL AB, Stockholm), see www.comsol.com.
[35]S. Chen and G. D. Doolen, "Lattice Boltzmann method for fluid flows," Annu. Rev. Fluid Mech. **30**, 329–364 (1998).
[36]S. Succi, *The Lattice Boltzmann Equation: For Fluid Dynamics and Beyond* (Oxford University Press, 2001).
[37]P. L. Bhatnagar, E. P. Gross, and M. Krook, "A model for collision processes in gases. I. Small amplitude processes in charged and neutral one-component systems," Phys. Rev. **94**, 511 (1954).

[38] Y. Qian and S. Orszag, "Scalings in diffusion-driven reaction $A + B \rightarrow C$: Numerical simulations by lattice BGK models," J. Stat. Phys. **81**, 237–253 (1995).
[39] P. Van Leemput, K. Lust, and I. G. Kevrekidis, "Coarse-grained numerical bifurcation analysis of lattice Boltzmann models," Physica D **210**, 58–76 (2005).
[40] S. Lee, F. Dietrich, G. E. Karniadakis, and I. G. Kevrekidis, "Linking Gaussian process regression with data-driven manifold embeddings for nonlinear data fusion," Interface Focus **9**, 20180083 (2019).
[41] I. G. Kevrekidis and G. Samaey, "Equation-free multiscale computation: Algorithms and applications," Annu. Rev. Phys. Chem. **60**, 321–344 (2009).