Chaos
An Interdisciplinary Journal of Nonlinear Science

RESEARCH ARTICLE | JUNE 27 2023

Learning dynamics on invariant measures using PDE-constrained optimization FREE

Jonah Botvinick-Greenhouse ; Robert Martin ; Yunan Yang

Chaos 33, 063152 (2023)
https://doi.org/10.1063/5.0149673

Learning dynamics on invariant measures using PDE-constrained optimization

Cite as: Chaos 33, 063152 (2023); doi: 10.1063/5.0149673
Submitted: 7 March 2023 · Accepted: 2 June 2023 ·
Published Online: 27 June 2023

Jonah Botvinick-Greenhouse,[1,a)] Robert Martin,[2] and Yunan Yang[3]

AFFILIATIONS

[1]Center for Applied Mathematics, Cornell University, Ithaca, New York 14850, USA
[2]DEVCOM Army Research Laboratory, Research Triangle Park, Durham, North Carolina 27709, USA
[3]Institute for Theoretical Studies, ETH Zürich, Zürich 8092, Switzerland

[a)]**Author to whom correspondence should be addressed:** jrb482@cornell.edu

ABSTRACT

We extend the methodology in Yang *et al.* [SIAM J. Appl. Dyn. Syst. **22**, 269–310 (2023)] to learn autonomous continuous-time dynamical systems from invariant measures. The highlight of our approach is to reformulate the inverse problem of learning ODEs or SDEs from data as a PDE-constrained optimization problem. This shift in perspective allows us to learn from slowly sampled inference trajectories and perform uncertainty quantification for the forecasted dynamics. Our approach also yields a forward model with better stability than direct trajectory simulation in certain situations. We present numerical results for the Van der Pol oscillator and the Lorenz-63 system, together with real-world applications to Hall-effect thruster dynamics and temperature prediction, to demonstrate the effectiveness of the proposed approach.

Published by AIP Publishing. https://doi.org/10.1063/5.0149673

Data-driven models have proven to be instrumental across numerous scientific disciplines for their ability to predict and control the behavior of complex physical systems.[1] Popular approaches for modeling dynamical trajectories typically adopt a Lagrangian perspective and seek pointwise matching with either the observed data or its approximate state derivatives. When the observed data have a poor temporal resolution and the state derivatives are difficult to approximate, these approaches may struggle. Such difficulties are further exaggerated when measurements are contaminated with noise, and the system in question exhibits sensitive dependence on initial conditions. In this paper, we propose an alternative approach that can circumvent some of these challenges by treating global statistics of the observed dynamics as the inference data.

I. INTRODUCTION

Differential equations are typically used to model trajectory data originating from physical systems. Common techniques for fitting differential equations to trajectory data include the shooting methods,[2,3] neural differential equations,[4–6] and SINDy.[7,8] Methods based on the Kalman filter are effective in data assimilation and in estimating unknown states and parameters of a system as it evolves in time.[9–14] When used to identify model parameters, these approaches fall under the broad category of system identification.

These approaches all adopt a Lagrangian perspective and directly fit the modeled trajectories or their state derivatives to the observed measurements. While these techniques have seen great success in modeling complex dynamics, their application is generally limited to inference trajectories sampled at a relatively high frequency. When the inference trajectory is sampled slowly or in the worst-case scenario when measurement times are unknown, these approaches may not be applicable. For example, see Fig. 1 in which we investigate the use of SINDy[7] and a neural ODE[4] for modeling the dynamics of a slowly sampled limit cycle.

There are at least three sources of instabilities when directly using the trajectory data to perform velocity reconstruction. First, for certain chaotic dynamical systems, a small perturbation in the initial condition can lead to a large deviation in the trajectory at a later time, which cannot be differentiated from inaccurate dynamics by looking at the data alone. Second, the estimation of the particle velocity suffers from slowly sampled trajectory data, which directly affects the reconstruction of the dynamics, as shown in Fig. 1. Third, the measurement (extrinsic) noise and the model intrinsic noise both change the state location. The small noise pollution can be

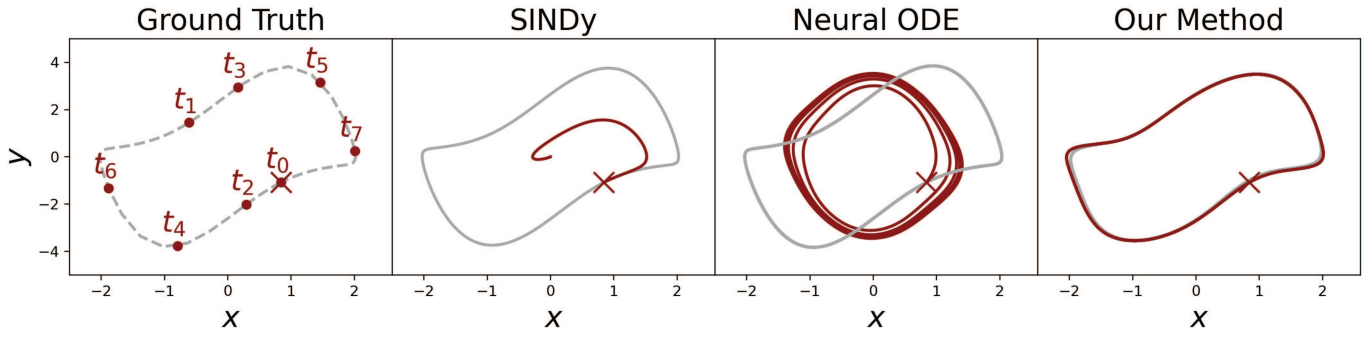

FIG. 1. Comparison with the SINDy[7,15,16] and the neural ODE[4] frameworks for learning slowly sampled dynamics. The left panel shows the original dynamics (in gray) and the first eight points of a slowly sampled trajectory (in red). While the SINDy and neural ODE frameworks can learn the quickly sampled dynamics (model output in gray), both struggle to learn from the slowly sampled trajectory (model output in red). On the other hand, our framework can learn from both quickly and slowly sampled dynamics (the red and gray model outputs coincide). Additional experiment details are provided in Sec. V A.

amplified more in the velocity estimation using the divided difference with a small time step. All three factors share the nature that a small perturbation in the trajectory data leads to a large deviation in the estimated velocity/learned dynamics.

In contrast with the Lagrangian approach to modeling dynamics, our method builds on an Eulerian perspective[17,18] in which velocity models are constructed to yield the same asymptotic statistics as the observed measurements. This approach converts what is traditionally regarded as an ordinary differential equation (ODE) or a stochastic differential equation (SDE) modeling problem into a partial differential equation (PDE)-constrained optimization problem. The motivation of our method is that, in certain situations, the PDE forward model yields better stability in solving the inverse problem than direct trajectory forward simulation based upon an ODE or SDE. Importantly, our method does not rely on prior knowledge of sampling times and can, thus, be used to learn the dynamics from slowly sampled trajectories.

There are two important differences between the line of work using Kalman filters and our proposed method. First, a Kalman filter is a particular case of the Bayes filter using the Bayes theorem, while our reconstruction follows a deterministic inverse problem (PDE-constrained optimization). Second, time is a crucial element in designing a Kalman filter, while in our approach, we use the invariant measure and a time-independent PDE surrogate model. Once the flow has been inferred, we can also perform uncertainty quantification for the forecasted dynamics, building toward extending grid-based Bayesian estimation of nonlinear low-dimensional systems[19] to slowly sampled unknown systems with nontrivial invariant measures.

More specifically, instead of directly treating the noisy observations $\{\tilde{\mathbf{x}}(t_i)\}_{i=1}^n$ from one single trajectory of an autonomous flow $\dot{\mathbf{x}} = v^*(\mathbf{x})$ as inference data, we consider the occupation measure ρ^* generated by a single trajectory, where for each measurable set B,

$$\rho^*(B) := \frac{1}{n}\sum_{i=1}^{n} \chi_B\left(\tilde{\mathbf{x}}(t_i)\right), \qquad \chi_B(x) = \begin{cases} 1, & x \in B, \\ 0, & x \notin B. \end{cases} \tag{1}$$

When the occupation measures generated by a nontrivial (see Sec. II A) set of initial conditions all weakly converge to the same invariant measure, the limiting measure is said to be physical.[20] In this work, we consider the class of autonomous systems for which the occupation measure of Lebesgue-almost all initial conditions converges to a unique physical measure. Notably, this encompasses chaotic attractors, such as the Lorenz-63 system.[21,22] This assumption guarantees the uniqueness of the invariant measure for the dynamical system under study. If we relax it and allow the existence of multiple invariant measures, further treatment of the PDE forward model is needed; for instance, the fact that different invariant measures are mutually singular as well as information on the initial condition, among other considerations, is necessary to guarantee that the steady-state solution picked up by the PDE model matches the observed invariant measure. We remark that the definition of a physical measure demonstrates its robustness to perturbations with respect to initial conditions.

Going forward, we write $v = v(\theta) = v(\mathbf{x};\theta)$ to denote the dependence of the reconstructed velocity fields on a set of parameters $\theta \in \Theta$ where $\Theta \subset \mathbb{R}^m$ is the admissible set of all parameter values. The concrete form of θ depends on the hypothesis space of v, which will be discussed in Sec. IV B. The task is now to find the best-parameterized model $v(\mathbf{x};\theta)$ approximating the true velocity v^* through the optimization

$$\inf_{\theta\in\Theta} \mathcal{J}(\theta), \qquad \mathcal{J}(\theta) := \mathcal{D}(\rho_\varepsilon(v(\theta)), \rho^*). \tag{2}$$

The formulation (2) represents an inverse data-matching problem, in which $\mathcal{D}$ denotes a metric or divergence on the space of probability measures and $\rho_\varepsilon(v(\theta))$ is a regularized approximation to the physical measure of the dynamical system, given some regularization parameter $\varepsilon > 0$ and the current velocity $v(\theta)$; that is, $v(\theta) \mapsto \rho_\varepsilon(v(\theta))$ is our new forward model.

Although one could approximate $\rho(v(\theta))$ by numerically integrating a trajectory and binning the observed states to a histogram,[17] this approach does not permit simple differentiation of the resulting measure with respect to the parameters θ. When the size of θ, i.e., m, is large, it is practical to use gradient-based optimization methods for solving the optimization problem (2), and one has to compute

the essential gradient $\partial_\theta \mathcal{J}$. In Ref. 18, this was handled by viewing $\rho_\varepsilon(\nu(\theta))$ as the dominant eigenvector of a regularized Markov matrix originating from an upwind finite-volume discretization of the continuity equation. The derivative $\partial_\theta \mathcal{J}$ was then seamlessly computed via the adjoint-state method.[18] The computation time of the adjoint-state method is independent of the size of θ, making the framework presented in Ref. 18 well-suited for large-scale computational inverse problems.

In this work, we build upon the framework proposed in Ref. 18 and study invariant measure-based velocity learning with a large-scale parameter space applied to real data. There are three essential new contributions:

1. We consider the Fokker–Planck equation as the partial differential equation (PDE) forward model for $\rho_\varepsilon(\nu(\theta))$, rather than the continuity equation. This is motivated by the Fokker–Planck equation's greater modeling capacity. Indeed, the Fokker–Planck equation reduces to the continuity equation when its diffusion term is zero, and it can fit intrinsic noise present in trajectories, which reduces over-fitting the parameterized velocity $\nu(\theta)$. Moreover, the Fokker–Planck equation can be seen as an alternative to the teleportation regularization used for the continuity equation in Ref. 18 in order to guarantee the uniqueness of the computed stationary solution $\rho_\varepsilon(\nu(\theta))$.
2. In contrast to only learning three coefficients as done in Ref. 18, we parameterize the velocity $\nu(\theta)$ using piecewise polynomial, global polynomial, and neural network discretizations, which can all yield large parameter spaces with thousands of dimensions. We compare the reconstructed velocity in each case and further discuss how the choice of parameterization affects the inverse problem's well-posedness and the reconstructed velocity's regularity. We also consider various metrics/divergences as the choice of the objective function.
3. We investigate velocity learning in time-delay coordinates, which can characterize the full dynamics from partial state measurements alone.[23] After performing the optimization (2), we evolve the learned Fokker–Planck equation forward in time to quantify the uncertainty in predictions of future dynamics. Based on this framework, we demonstrate that forecasts incur larger uncertainties when the embedding dimension is not sufficiently high. It is worth noting that there is no analytic form for the velocity in time-delay coordinates, even for well-studied dynamical systems. We also stress that our proposed approach permits larger-scale modeling of time-delayed dynamics than the approach considered in Ref. 17 due to the use of the adjoint-state method when solving the PDE-constrained optimization.

The rest of the paper is organized as follows. In Sec. II, we review essential background on dynamical systems, invariant measures, the Fokker–Planck equation, and time-delay coordinates. In Sec. III, we introduce the forward surrogate model $\rho_\varepsilon(\nu(\theta))$ and analyze its modeling errors. In Sec. IV, we present an efficient gradient calculation for the objective function $\mathcal{J}(\theta)$ by treating (2) as a PDE-constrained optimization problem and utilizing the adjoint-state method. We then adapt the gradient calculation to various velocity parameterizations, including neural network discretizations in which the gradient is computed along with the backpropagation technique.[24]

Finally, in Sec. V, we present velocity reconstructions for the Van der Pol oscillator and the Lorenz-63 system. We also model dynamics in time-delay coordinates based on real-world data from a Hall-effect thruster and actual temperature recordings. We perform uncertainty quantification on the last two real-data examples. Conclusions follow in Sec. VI.

II. BACKGROUND

This section reviews the essential background on invariant measures, stochastic dynamics, the Fokker–Planck equation, and time-delay coordinates. We also review the Eulerian approach for parameter identification proposed in Refs. 17 and 18, as well as past work on the discrete inverse Frobenius–Perron problem.[25]

A. Physical measures

Physical measures characterize the long-term statistical behavior of a significant collection of dynamical trajectories. When a dynamical system is chaotic and exhibits sensitive dependence on initial conditions, the existence of a physical measure unifies the statistical properties of trajectories that are pointwise dissimilar. While ergodic measures also describe the long-term statistical behavior of dynamical trajectories, they may have very small support or even be singular. On the other hand, when a dynamical system admits a physical measure, it holds that the trajectories corresponding to a positive Lebesgue measure subset of initial conditions will all share the same statistical behavior. We will now formalize these ideas in the language of ergodic theory. For a more thorough treatment of the topic, we refer to Refs. 20, 26, 27, and 28.

While we will review the theory of physical measures in the context of discrete-time dynamical systems, our applications will consider dynamics given by a time-Δt flow map for some $\Delta t > 0$. Following Ref. 20, we assume that $\mathcal{M}$ is a compact Riemannian manifold and that $T : \mathcal{M} \to \mathcal{M}$ is a diffeomorphism. A probability measure μ is said to be invariant with respect to the map T if $\mu(T^{-1}(B)) = \mu(B)$ for all $B \in \mathscr{B}$, where $\mathscr{B}$ denotes the Borel σ-algebra (see Ref. 29, Definition 2.1). Hereafter, we will assume that μ is an invariant measure. A point $x \in \mathcal{M}$ is said to be generic (see Ref. 20, Sec. 2.2) if for all $g \in C(\mathcal{M})$, it holds that

$$\lim_{N\to\infty} \frac{1}{N} \sum_{k=0}^{N-1} g(T^k(x)) = \int_{\mathcal{M}} g \, d\mu. \tag{3}$$

The left-hand side of (3) is known as the time-average of a function $g \in C(\mathcal{M})$, whereas the right-hand side of (3) is known as the space average. It follows from Birkhoff's pointwise ergodic theorem (see Theorem 2.30 in Ref. 29) that the time-average of any $g \in C(\mathcal{M})$ necessarily exists on a set of full μ-measure. To formally discuss the statistical properties of dynamical trajectories, we now define the N-step occupation measure given the initial condition $x \in \mathcal{M}$ as

$$\mu_{x,N}(B) := \frac{1}{N} \sum_{k=0}^{N-1} \chi_B(T^k(x)), \qquad \forall B \in \mathscr{B}. \tag{4}$$

The condition that a point $x \in \mathcal{M}$ is generic is equivalent to the condition

$$\lim_{N\to\infty} \mu_{x,N} = \mu\,, \tag{5}$$

where convergence takes place in the weak-* topology (see Ref. 29, Definition 4.19). Since the quantity $\mu_{x,N}(B)$ approximates the average amount of time for which the orbit $\{T^k(x)\}_{k=0}^{\infty}$ initiated at $x \in \mathcal{M}$ resides in a measurable set $B \in \mathscr{B}$, this convergence indicates that the collection of generic points all share the same asymptotic statistical behavior. When the measure μ is ergodic (see Ref. 29, Definition 4.19), it holds that μ-almost every $x \in \mathcal{M}$ is a generic point (see Ref. 29, Corollary 4.20). However, if μ is an ergodic measure that is singular with respect to the Lebesgue measure, the resulting collection of generic points may be physically insignificant and difficult to observe computationally. Motivated by this perspective, an invariant measure μ is said to be physical if there exists a collection of generic points with a positive Lebesgue measure (see Ref. 20, Definition 2.3).

We will next discuss the ways in which a physical invariant measure μ can be computationally approximated. If one collects the measurements $\{T^k(x)\}_{k=1}^{N}$, the weak-* convergence in (5) suggests that the physical measure μ will describe the statistics of our measurements provided that N is sufficiently large. Motivated by this perspective, we can discretize the domain $\mathcal{M}$ and directly compute the occupation measure (4) for each cell in the discretization to approximate the physical measure. This procedure has been previously used to approximate physical measures.[17,18,30] Other approaches have been proposed to compute the invariant measure as the stationary vector of the finite-dimensional approximation of the continuous Frobenius–Perron operator,[31] including Ulam's[27] and Galerkin-type methods.[32,33] More precisely, these discretizations are used to construct a Markov matrix that represents a random dynamical system approximating the deterministic map $T : \mathcal{M} \to \mathcal{M}$. An invariant measure for the discrete approximation is then recovered as a stationary vector of the resulting Markov matrix. As the discretization is refined, certain assumptions guarantee that the desired physical measure will be recovered in the weak-* limit (see Ref. 32, Theorem 4.14).

B. Stochastic dynamics and the Fokker–Planck equation

Consider an Itô stochastic differential equation (SDE) of the form

$$dX_t = v(X_t)dt + \sigma(X_t)dW_t, \qquad X_0 = x. \tag{6}$$

Above, W_t is a Brownian motion, v is the velocity, and σ determines the diffusion matrix $\Sigma(\mathbf{x}) = \frac{1}{2}\sigma(\mathbf{x})\sigma(\mathbf{x})^\top$. For simplicity, we will consider the case of constant diffusion. Similar to the deterministic setting, there are analogous notions of invariant measures, ergodicity, and physical measures in the stochastic setting.[34,35] One may use the Euler–Maruyama method to obtain the numerical solution to (6) on the time interval $[0, T]$, which assigns

$$X_{j+1} = X_j + v(X_j)\Delta t + \sigma(X_j)\xi_j\sqrt{\Delta t},$$

where $\{\xi_j\}$ are independently and identically distributed (i.i.d.) from $\mathcal{N}(0, I)$, the standard normal distribution on $\mathbb{R}^d$, $\Delta t := T/N$, and $j \in \{0, \ldots, N-1\}$.

The Fokker–Planck equation provides a PDE description of the probability density $\rho(\mathbf{x}, t)$ of the random variable X_t. The density evolves as (see Ref. 36, p. 88)

$$\frac{\partial \rho(\mathbf{x},t)}{\partial t} = -\nabla\cdot(\rho(\mathbf{x},t)v(\mathbf{x})) + \nabla\cdot\Big(\nabla\cdot(\Sigma(\mathbf{x})\rho(\mathbf{x},t))\Big). \tag{7}$$

By assuming a constant diffusion, we may write $\Sigma(\mathbf{x}) = DI$, where I denotes the identity and $D > 0$ is a constant representing the scale of the diffusion. Equation (7) can then be simplified to read

$$\frac{\partial \rho(\mathbf{x},t)}{\partial t} = -\nabla\cdot(\rho(\mathbf{x},t)v(\mathbf{x})) + D\nabla^2\rho(\mathbf{x},t). \tag{8}$$

We leave the study of a non-constant or anisotropic diffusion for later work. We remark that if $D = 0$, (8) reduces to the so-called continuity equation, which instead models the probability flow of the ODE given by $\dot{\mathbf{x}} = v(\mathbf{x})$. Under certain conditions,[37] the steady-state solution $\rho(\mathbf{x})$ of (8) exists and satisfies

$$\nabla\cdot(\rho(\mathbf{x})v(\mathbf{x})) = D\nabla^2\rho(\mathbf{x}). \tag{9}$$

Since (9) describes a limiting distribution $\lim_{t\to\infty}\rho(\mathbf{x}, t)$, it has been previously used to provide approximations of invariant measures for stochastically forced dynamical systems.[30] In Ref. 38, an SDE learning problem was studied using (7) as the modeling equation with different data assumptions.

C. Delay coordinates and Takens' theorem

The technique of time-delay embedding is a popular approach for reconstructing chaotic dynamical systems from limited observations.[17,39–41] The procedure involves embedding time-series measurements $\psi(t) = \psi(\mathbf{x}(t))$ of the state $\mathbf{x}(t)$ into d-dimensional Euclidean space by considering the vector of time-lagged observations,

$$\Psi_{d,\tau}(t) = (\psi(t), \psi(t-\tau), \ldots, \psi(t-(d-1)\tau)),$$

for some $\tau > 0$. Takens' theorem[23] provides suitable assumptions under which $\Psi_{d,\tau}(t)$ and $\mathbf{x}(t)$ are related via diffeomorphism, implying that the time-lagged vector of partial observations $\Psi_{d,\tau}(t)$ is sufficient for reconstructing the full state $\mathbf{x}(t)$. Notably, the embedding dimension provided in Ref. 23 is $d = 2m + 1$, where m is the dimension of a compact manifold $\mathcal{M}$ on which the flow map f_t for the original dynamics is defined. In cases when trajectories are attracted to a compact subset A with a box-counting dimension (see Ref. 42, p. 586) d_A strictly less than m, it turns out that lower-dimensional embeddings can be obtained.

When a time-series projection $\psi(t)$ of an unknown system $\dot{\mathbf{x}} = v(\mathbf{x})$ is observed, one can try to numerically determine a suitable embedding dimension d and time delay τ; see, for example, Refs. 43–46. Choosing a proper embedding dimension and time delay is important for obtaining a reliable surrogate model of the original dynamics in time-delayed coordinates. Notably, in Sec. V B, we demonstrate that models for the velocity in time-delayed coordinates can incur excess uncertainties when the embedding dimension is not sufficiently large.

D. Prior work on learning dynamics from invariant measures

For chaotic systems, trajectories are sensitive to initial conditions and estimation parameters. Sometimes, the approximate reference velocity field $\{\hat{v}(\mathbf{x}(t_i))\}$ cannot be accurately estimated from a trajectory $\{\mathbf{x}(t_i)\}$ due to the lack of observational data, slow sampling, discontinuous or inconsistent time trajectories, and noisy measurements. To tackle such difficulties, instead of working with the Lagrangian trajectories, Refs. 17 and 18 propose an Eulerian approach by treating the occupation measure (4) as the data. When enough samples are available, the occupation measure can be treated as an approximation to the invariant measure; see Sec. II A. Finding the optimal parameter θ is then translated into the optimization problem (2). The reference measure ρ^* is the occupation measure converted from the observed trajectories $\{\hat{\mathbf{x}}(t_i)\}$; see (4). In Ref. 17, the approximated synthetic $\rho_\varepsilon(v(\theta))$ is generated by first simulating the synthetic trajectories $\{\mathbf{x}(t_i;\theta)\}$ based on the dynamical system and then computing its histogram following (4). Since this approach requires lengthy trajectory simulation, each evaluation of $\rho_\varepsilon(v(\theta))$ for a given θ is relatively costly. Moreover, it is difficult to compute the derivative of $\rho_\varepsilon(v(\theta))$ with respect to θ due to the histogram approximation of nonlinear trajectories. As an improvement to the original idea in Ref. 17, Ref. 18 proposes a surrogate model to approximate $\rho_\varepsilon(v(\theta))$ that is differentiable in θ and sometimes faster to compute. The key idea is to solve for $\rho_\varepsilon(v(\theta))$ as the distributional steady-state solution to the continuity equation [i.e., (9) with $D = 0$] using a finite-volume upwind scheme together with the teleportation regularization. The gradient of the objective function $\mathcal{J}$ in (2) with respect to the parameter θ can be efficiently computed based on the adjoint-state method (see Sec. 5 in Ref. 18). The problem of learning an SDE from an invariant measure is also studied in Ref. 47, which uses a deep learning framework to invert the drift and diffusion terms.

The task of learning a dynamical system from an invariant measure has also been studied in the discrete-time setting under the inverse Frobenius–Perron problem.[25,48–50] The Frobenius–Perron operator, also known as the transfer operator, characterizes the time evolution of an initial measure μ_0 according to some prespecified dynamical system. Given a probability measure μ, the inverse Frobenius–Perron problem seeks to construct a dynamical system for which μ is a fixed point of the associated transfer operator. The most widely studied case involves recovering an ergodic map T on $[0, 1]$ for which a prescribed absolutely continuous measure is the unique fixed point of the discrete transfer operator. In this particular setting, various approaches, such as topological conjugation[51] and matrix methods,[52] have been introduced to solve the inverse problem. The multivariate inverse Frobenius–Perron problem was also studied in Ref. 53, where ergodic maps were constructed to adhere to the statistics of two-dimensional densities. Moreover, due to inherent non-uniqueness in the inverse problem, recent approaches further restrict the solution space of the discrete ergodic maps to those with a prescribed power spectrum.[54] To the best of our knowledge, Refs. 17, 18, and 47, and our contributions here are the first works that numerically solve the inverse Frobenius–Perron problem in the continuous-time setting. Notably, we do not assume that μ is absolutely continuous, as we use a finite-volume discretization to approximate the Frobenius–Perron operator.

III. THE FORWARD MODEL AND MODELING ERRORS

A central contribution of this work is to consider a different regularized forward model than the one in Ref. 18, especially for trajectory measurements containing intrinsic noise, which can be interpreted as sample paths of stochastic dynamical systems (6). In those cases, the Fokker–Planck equation (7) is a better candidate as the PDE surrogate model, as it contains a diffusion term that can fit noise present in the data. Based on the relationship between (6) and (7), one can learn both the velocity field $v(\mathbf{x})$ and the diffusion tensor $\Sigma(\mathbf{x})$ in the optimization framework (2). For simplicity, we only consider a fixed diffusion constant and leave the investigation of multi-parameter inversion to future work.

We will use (9) as the forward model to fit invariant measures generated by trajectories with intrinsic noise. While the diffusion term allows the model to fit the intrinsic noise and prevent over-fitting the noise into the target velocity component, it also controls the scaling of the reconstructed velocity $v(\mathbf{x};\theta)$. Indeed, when $D = 0$ and $\tilde{v}(\mathbf{x}) = a\,v(\mathbf{x})$, we have $\nabla\cdot(\rho(\mathbf{x})\tilde{v}(\mathbf{x})) = 0$ as long as $\nabla\cdot(\rho(\mathbf{x})v(\mathbf{x})) = 0$ for any $a > 0$. However, for most cases, $\tilde{v}$ and v will not solve the stationary Fokker–Planck equation (9) for $D > 0$.

A. Finite-volume discretization

We assume that our system evolves on the d-dimensional rectangular state space,

$$\Omega = [a_1, b_1] \times \cdots \times [a_d, b_d] \subset \mathbb{R}^d,$$

with a spatially dependent velocity $v : \Omega \to \mathbb{R}^d$. We define $n_i \in \mathbb{Z}^+$, $1 \le i \le d$, to be the number of equally spaced points along the ith spatial dimension at which we wish to approximate the solution of (8), as well as the mesh spacing

$$\Delta x_i := \frac{b_i - a_i}{n_i - 1}.$$

We are, thus, interested in obtaining a solution to the forward problem at points of the form

$$x_{k_1,\ldots,k_d} := (a_1 + k_1\Delta x_1, \ldots, a_d + k_d\Delta x_d),$$

where $k_i \in \{1, \ldots, n_i\}$. We will index our coordinates using column-major order and write $x_{k_1,\ldots,k_d} = x_j$ where

$$j = k_1 + \sum_{i=2}^{d}(k_i - 1)S_i, \qquad S_i := \prod_{j=1}^{i-1} n_j. \tag{10}$$

We will regard x_j as the center of the cell C_j where

$$C_j = \prod_{i=1}^{d}\left[a_i + \left(k_i - \frac{1}{2}\right)\Delta x_i,\ a_i + \left(k_i + \frac{1}{2}\right)\Delta x_i\right).$$

Following the approach in Ref. 19, we implement a first-order upwind finite-volume discretization of the continuity equation, adding a diffusion term using the central difference scheme and enforcing a zero-flux boundary condition.[55] This allows us to obtain an explicit time evolution of the probability vector $\rho = \begin{bmatrix}\rho_1 & \rho_2 & \cdots & \rho_N\end{bmatrix}^\top \in \mathbb{R}^N$, where $N = \prod_{i=1}^{d} n_i$. While ρ is

a discrete probability measure over the cells C_j, it also corresponds to a piecewise-constant probability density function on Ω.

With an abuse of notation, we will refer to both the piecewise-constant density and the discrete probability measure as ρ. We discretize the time domain with a step size Δt. Based on (8), the probability vector at the lth time step evolves as

$$\rho^{(l+1)} = \rho^{(l)} + K\rho^{(l)}, \qquad K = \sum_{i=1}^{d} \frac{\Delta t}{\Delta x} K_i,$$

where each K_i is a tridiagonal matrix of the form

$$K_i := S_i \begin{bmatrix} \ddots & & & & & & \\ & -v_{j-1}^{i,-} + \frac{D}{\Delta x_i} & & & & & \\ \ddots & \vdots & & & & & \\ & v_{j-1}^{i,-} - w_{j-1}^{i,+} - \frac{2D}{\Delta x_i} & -v_j^{i,-} + \frac{D}{\Delta x_i} & & & & \\ \ddots & \vdots & \vdots & & & & \\ & w_{j-1}^{i,+} + \frac{D}{\Delta x_i} & v_j^{i,-} - w_j^{i,+} - \frac{2D}{\Delta x_i} & -v_{j+1}^{i,-} + \frac{D}{\Delta x_i} & & & \\ & & \vdots & \vdots & \ddots & & \\ & & w_j^{i,+} + \frac{D}{\Delta x_i} & v_{j+1}^{i,-} - w_{j+1}^{i,+} - 2\frac{D}{\Delta x_i} & & & \\ & & & \vdots & \ddots & & \\ & & & w_{j+1}^{i,+} + \frac{D}{\Delta x_i} & & & \\ & & & & \ddots & & \end{bmatrix} \in \mathbb{R}^{N\times N}.$$

Above, we have defined for each $j \in \{1, \ldots, N\}$ the upwind velocities

$$v_j^{i,-} := \min\left\{0, v_j^i\right\}, \qquad v_j^{i,+} := \max\left\{0, v_j^i\right\},$$
$$w_j^{i,-} := \min\left\{0, w_j^i\right\}, \qquad w_j^{i,+} := \max\left\{0, w_j^i\right\},$$

where $v_j^i := v\left(x_j - \mathbf{e}_i \Delta x_i/2\right) \cdot \mathbf{e}_i$ and $w_j^i := v\left(x_j + \mathbf{e}_i \Delta x_i/2\right) \cdot \mathbf{e}_i$ denote the ith components of the velocity field at the center of cell faces, and $\{\mathbf{e}_i\}$ is the standard basis in $\mathbb{R}^d$. We remark that if x_j is away from $\partial\Omega$, then $w_j^{i,\pm} = v_{j+1}^{i,\pm}$. To enforce the zero-flux boundary condition, we set both the velocity v and diffusion D to be zero on $\partial\Omega$. As a result, the columns of K each sum to zero, and the total probability

$$\rho^{(l)} \cdot \mathbf{1} = 1, \qquad \mathbf{1} := \begin{bmatrix} 1 & \ldots & 1 \end{bmatrix}^\top \in \mathbb{R}^N$$

is conserved under time evolution. Since numerical artifacts cause the flux accumulation along the boundary, we also enforce $\rho = 0$ on $\partial\Omega$. When the boundary $\partial\Omega$ is sufficiently far from the trajectory data, this artifact is insignificant. Hereafter, we assume the uniform spatial discretization $\Delta x_i = \Delta x$ for all $i = 1, \ldots, d$. Here, we used an explicit time stepping scheme. The Courant–Friedrichs–Lewy (CFL) stability condition enforces $\Delta t = \mathcal{O}(\Delta x^2)$ to ensure the stability of the scheme. To be more specific, we choose

$$\Delta t < \frac{1}{2d} \frac{\Delta x^2}{D + \Delta x \|v\|_\infty},$$

where $\|v\|_\infty = \max_i \|v(\mathbf{x}) \cdot \mathbf{e}_i\|_\infty$. In this way, we can enforce that all entries of $I + K$ are non-negative with columns summed to one, which implies that $I + K$ is a Markov matrix.

For a complete description of the finite-volume scheme, we refer to Ref. 55. We remark that there are many higher-order structure-preserving schemes to solve (8), which also yield a Markov matrix; see Ref. 56 for example. A more accurate numerical scheme can further reduce the forward modeling error, which is left for future work.

B. Teleportation and diffusion regularization

We use the finite-volume discretization of the Fokker–Planck equation in Sec. III A to approximate its steady-state solution. After discretization, finding such stationary distributions to (9) is equivalent to solving the linear system,

$$(I + K)\rho = \rho.$$

Since the columns of K sum to zero, we have that $M := I + K$ is a column-stochastic Markov matrix. When $D \neq 0$, M is a transition matrix for an ergodic Markov chain, which has a unique equilibrium. When $D = 0$, to guarantee the uniqueness of the equilibrium, Ref. 18 applies the so-called teleportation regularization[57] and considers

$$M_\varepsilon := (1-\varepsilon)M + \varepsilon\, U, \quad U = N^{-1}\mathbf{1}\mathbf{1}^\top \in \mathbb{R}^{N\times N}.$$

There is now a unique solution to the linear system

$$M_\varepsilon \rho = \rho, \quad \rho \cdot \mathbf{1} = 1, \quad \rho > 0. \tag{11}$$

From a computational aspect, it is useful to take advantage of the fact that $M - I$ is sparse where $I \in \mathbb{R}^{N\times N}$ is the identity matrix and to instead solve

$$(1-\varepsilon)(M-I)\rho = -N^{-1}\varepsilon\mathbf{1},$$

where we have simply rearranged terms in (11) and used the fact that $\rho \cdot \mathbf{1} = 1$.

Since U is also a column stochastic Markov matrix with the uniform probability of visiting any point of the mesh, using M_ε amounts to stopping the dynamics based on M at a random time and restarting it from a uniformly randomly chosen initial point. The size of ε represents the restarting frequency—the smaller ε, the rarer we restart.[18]

On the other hand, adding the diffusion component D to the tridiagonal matrix K can be seen as another way of regularizing the noise-free Markov matrix by adding scaled Brownian motion after each discrete evolution of the deterministic dynamics. For deterministic dynamics with $D = 0$, the solution to (9) might not be unique if there is more than one attractor. The use of teleportation connects all attractors through the "random restart," and the solution ρ_ε to the linear system (11) has support that connects all the disjoint attractors. Similarly, when $D \neq 0$, the Brownian motion connects all disjoint attractors of the deterministic dynamics, giving a unique steady-state solution. In this scenario, the use of teleportation for the diffusive case is simply a numerical treatment to improve the conditioning of matrix M rather than to guarantee the uniqueness of ρ.

It is worth noting that both the teleportation regularization and an incorrect diffusion coefficient could be sources of modeling error when we perform parameter identification. Although these regularizations enable faster evaluation of $\rho_\varepsilon(\nu(\theta))$ and better posedness of the forward problem, they may reduce the accuracy of the inverse problem solution.

C. Numerical diffusion

In Fig. 2, we illustrate ρ_ε computed as the steady-state solution to the Fokker–Planck equation in the top row and the approximation to physical invariant measures of the corresponding SDE in the bottom row. From Fig. 2, we see that on a coarse mesh, the first-order finite-volume scheme incurs significant numerical error, which gives a computed solution with an artificial diffusion effect and, thus, is often referred to as the numerical diffusion.[19] The amount of numerical diffusion is reduced as the mesh is refined since it is incurred by the first-order scheme. In particular, it is expected to decay as $\mathcal{O}(\max_i \Delta x_i)$ in the L^∞ norm as we refine the mesh.[55] Besides the teleportation and the modeling diffusion D, the presence of numerical diffusion is another modeling error incurred from solving the forward problem.

IV. GRADIENT CALCULATION AND VELOCITY PARAMETERIZATION

Another main contribution of this paper is to reconstruct the velocity field $\nu(\mathbf{x})$ using large-scale parameterizations $\nu(\mathbf{x};\theta)$, which turns an infinite-dimensional problem of searching for $\nu(\mathbf{x})$ in a function space to a finite-dimensional optimization problem of finding $\theta \in \Theta \subset \mathbb{R}^m$. Here, we introduce parameterizations based on piecewise-constant, neural network, and global polynomial functions. We also investigate various data-fitting objective functions $\mathcal{J}$ that compare the mismatch between the observed and simulated invariant measures, ρ^* and $\rho_\varepsilon(\nu(\theta))$. We compute the gradient of such functions with respect to the coefficients θ in the parameterized velocity model $\nu(\mathbf{x};\theta)$ based on the adjoint-state method for the PDE-constrained part and the backpropagation technique[24] for the neural network part. Thanks to these techniques, we can then efficiently evaluate the gradients of $\mathcal{J}$ with respect to θ and, thus, conveniently use gradient-based optimization algorithms to iteratively update θ, e.g., steepest descent, L-BFGS, conjugate gradient descent methods as well as stochastic methods such as Adam.[58] For notational simplicity, we will write $\rho(\nu(\theta)) = \rho_\varepsilon(\nu(\theta))$ throughout this section.

A. Gradient calculation through the adjoint-state method

Recall the finite-volume scheme in Sec. III A for solving (9). The forward model yields a discrete measure $\rho(\nu(\theta)) = \rho(\theta) = [\rho_1(\theta)\ldots\rho_j(\theta)\ldots\rho_N(\theta)]^\top$ over the cells $\{C_j\}$, which converges to the solution to (9) in the weak sense as we refine the discretization parameters. For the explicit form of $\rho(\nu(\theta))$, we refer to Eq. (5.1) in Ref. 18. Note that we have highlighted the dependence of our approximate steady-state distributional solution to the Fokker–Planck equation (9) on the velocity $\nu(\mathbf{x};\theta)$. Our goal is to solve the optimization problem (2),

$$\inf_{\theta\in\Theta} \mathcal{J}(\rho(\nu(\theta)), \rho^*),$$

by using gradient-based methods, where $\mathcal{J}$ is the cost function and ρ^* represents our inference data. The adjoint-state method is an efficient technique by which we can evaluate the derivative $\partial_\theta \mathcal{J}$, as the computation time is largely independent of the size of θ. One can derive the adjoint-state method for gradient computations by differentiating the discrete constraint,[59] which in our case is the eigenvector problem,

$$g(\rho(\theta),\theta) = M_\varepsilon(\theta)\rho(\theta) - \rho(\theta) = \mathbf{0},$$

where $\rho(\theta)\cdot\mathbf{1} = 1$. Specifically, we compute $\partial_\theta\mathcal{J} = \lambda^\top\partial_\theta g$ where λ solves $(\partial_\rho g)^\top\lambda = -(\partial_\rho\mathcal{J})^\top$. In our case, this linear system is the adjoint equation [see Ref. 18, Eq. (5.8)]

$$(M_\varepsilon^\top - I)\lambda = -\left(\partial_\rho\mathcal{J}\right)^\top + \left(\partial_\rho\mathcal{J}\right)^\top \rho\,\mathbf{1}, \tag{12}$$

and the derivative

$$\partial_\theta\mathcal{J} = \lambda^\top\left(\partial_\theta M_\varepsilon\right)\rho. \tag{13}$$

As a result, we only need to compute the derivatives $\partial_\rho \mathcal{J}$ and $\partial_\theta M_\varepsilon$ to determine the gradient $\nabla_\theta \mathcal{J} = (\partial_\theta \mathcal{J})^\top$. The former depends on the choice of the objective function, while the latter is based on a specific parameterization of the velocity field $v(\mathbf{x}; \theta)$ determined by its hypothesis space.

1. The computation of $\partial_\rho \mathcal{J}$

For the objective function $\mathcal{J}$, we consider the quadratic Wasserstein distance, the squared L^2 norm, the Kullback–Leibler (KL) divergence, and the Jensen–Shannon (JS) divergence.

a. Quadratic Wasserstein distance. For probability measures ρ and ρ^* on Ω, with finite second-order moments, the squared quadratic Wasserstein distance is defined by

$$W_2^2(\rho, \rho^*) := \inf_{T_{\rho,\rho^*} \in \mathcal{T}} \int_\Omega |x - T_{\rho,\rho^*}(x)|^2 d\rho(x),$$

where

$$\mathcal{T} := \{T : \Omega \to \Omega : \rho(T^{-1}(B)) = \rho^*(B),\ B \in \mathscr{B}\}$$

is the set of maps that push ρ forward into ρ^*.[60] With an abuse of notation, we also use $\rho(x)$ and $\rho^*(x)$ to denote the densities of ρ and ρ^*, respectively. For efficient computation of the W_2 distance, we utilize the back-and-forth method,[61] which instead uses the dual Kantorovich formulation,[60]

$$W_2^2(\rho, \rho^*) = \sup_{\phi_1, \phi_2} \left(\int_\Omega \phi_1(x)\rho^*(x)dx + \int_\Omega \phi_2(x)\rho(x)dx \right),$$

where $\phi_1 \in L^1_{\rho^*}(\Omega)$ and $\phi_2 \in L^1_\rho(\Omega)$ are required to satisfy $\phi_1(x) + \phi_2(y) \le |x - y|^2$. In this case, the Fréchet derivative of $\mathcal{J} = W_2^2(\rho, \rho^*)$ with respect to ρ is given by

$$\frac{\partial \mathcal{J}}{\partial \rho} = \phi_2.$$

b. Squared L^2 norm. The squared L^2 distance as the objective function and its Fréchet derivative are given by

$$\mathcal{J} = \frac{1}{2} \int_\Omega |\rho(x) - \rho^*(x)|^2 dx,$$

$$\frac{\partial \mathcal{J}}{\partial \rho} = \rho - \rho^*.$$

c. KL divergence. The KL divergence and its Fréchet derivative are given by

$$\mathcal{J} = D_{\mathrm{KL}}(\rho, \rho^*) := \int_\Omega \rho^*(x) \log\left(\frac{\rho^*(x)}{\rho(x)}\right) dx,$$

$$\frac{\partial D_{\mathrm{KL}}}{\partial \rho} = -\frac{\rho^*(x)}{\rho(x)}.$$

We remark that our definition of the KL divergence differs from many applications in which it is commonly computed as $\mathcal{J} = D_{\mathrm{KL}}(\rho^*, \rho)$.

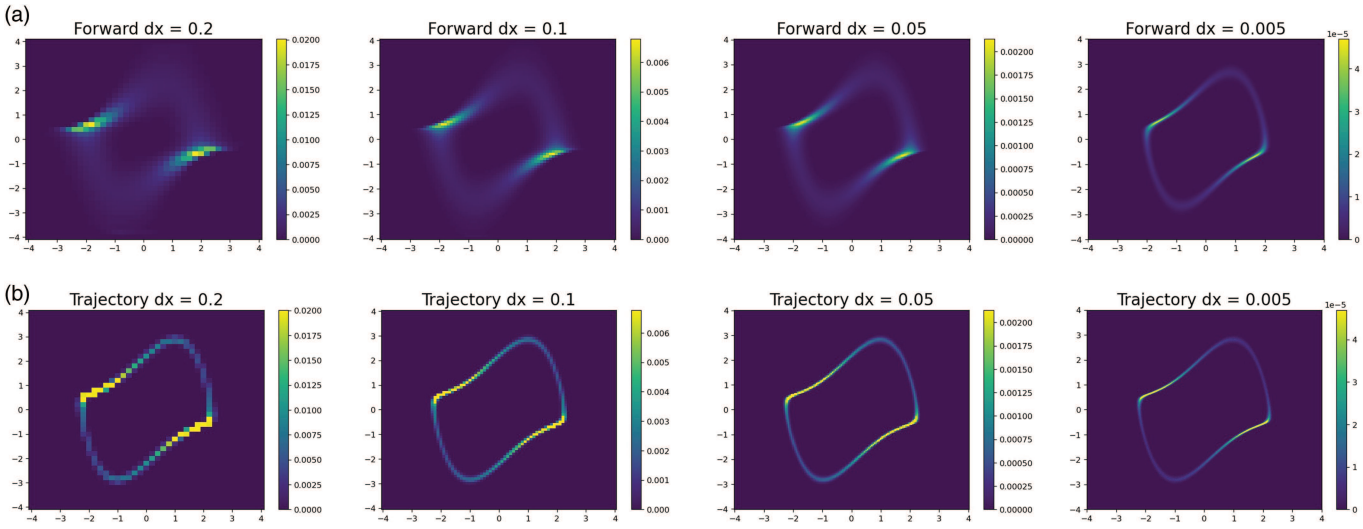

FIG. 2. As the mesh size of the forward model discretization is refined, we visually observe the convergence of the computed steady-state solution (a) to the approximate physical measure (b). The Van der Pol oscillator (19) with $c = 1$ and $D = 0.001$ is used in this example, and the histograms indicate mass per cell. (a) The computed steady-state solution to (9) for decreasing values of Δx. (b) The approximate physical measure obtained by binning a time trajectory based on the SDE (6) for decreasing values of Δx.

d. JS divergence. Defining $\rho' := (\rho + \rho^*)/2$, the JS divergence and its Fréchet derivative are given by

$$\mathcal{J} = D_{\mathrm{JS}}(\rho, \rho^*) = \frac{1}{2} D_{\mathrm{KL}}(\rho, \rho') + \frac{1}{2} D_{\mathrm{KL}}(\rho^*, \rho'),$$
$$\frac{\partial D_{\mathrm{JS}}}{\partial \rho} = \frac{1}{2} \log\left(\frac{2\rho}{\rho + \rho^*}\right).$$

Based on definitions of the KL and JS divergence, it is clear that we may encounter numerical instability issues if either ρ or ρ^* is not supported on the entire domain Ω. Thus, we remark that for the computation of both the KL and JS divergences, we restrict the domain Ω to regions where both ρ and ρ^* are strictly positive. This is equivalent to the definition of the KL and JS divergence based upon the so-called Csiszar divergence [see Ref. 62, Eq. (1)].

2. The computation of $\partial_\theta \mathcal{J}$

We have presented a few cases of $\partial_\rho \mathcal{J}$ for different choices of $\mathcal{J}$. Next, we show how to obtain $\partial_\theta M_\varepsilon$, which is the other necessary component in the adjoint-state method for gradient calculation; see (12) and (13). To begin with, we consider $\theta = \{v_j^i\}$ for all $i = 1, \ldots, d$ and $j = 1, \ldots, N$, which corresponds to one variant of piecewise-constant velocity parameterization.

Since we are only interested in computing the gradient away from $\partial\Omega$, we can utilize the property that $w_j^{i,\pm} = v_{j+1}^{i,\pm}$. First, observe that

$$\frac{\partial M_\varepsilon}{\partial v_j^i} = (1-\varepsilon) \sum_{\ell=1}^{d} \frac{\Delta t}{\Delta x} \frac{\partial K_\ell}{\partial v_j^i} = (1-\varepsilon) \frac{\Delta t}{\Delta x} \frac{\partial K_i}{\partial v_j^i},$$

as well as

$$\frac{\partial K_i}{\partial v_j^i} = \begin{matrix} \\ S_i \left\{ \vphantom{\begin{matrix} a \\ a \\ a \end{matrix}} \right. \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{bmatrix} \ddots & & & & & & \\ & 0 & & & & & \\ \ddots & \vdots & & & & & \\ & -H(v_j^i) & -(1-H(v_j^i)) & & & & \\ \ddots & \vdots & \vdots & & & & \\ & H(v_j^i) & (1-H(v_j^i)) & 0 & & & \\ & & \vdots & \vdots & \ddots & & \\ & & 0 & 0 & & & \\ & & & \vdots & \ddots & & \\ & & & 0 & & & \\ & & & & \ddots & & \end{bmatrix}$$

$$H(x) := \begin{cases} 1, & x > 0 \\ 0, & x \le 0 \end{cases}. \tag{14}$$

In (14), $H(\cdot)$ is the Heaviside function. We remark that $\partial_{v_j^i} K_i$ can only be nonzero in the (j,j), $(j, j-S_i)$, $(j-S_i, j)$, and $(j-S_i, j-S_i)$th entries where S_i is defined in (10). After solving (12) for λ and applying (13), we deduce that

$$\begin{aligned} \frac{\partial \mathcal{J}}{\partial v_j^i} &= \lambda \cdot \frac{\partial M_\varepsilon}{\partial v_j^i} \rho = (1-\varepsilon) \frac{\Delta t}{\Delta x} \left(\lambda \cdot \frac{\partial K_i}{\partial v_j^i} \rho \right) \\ &= (1-\varepsilon) \frac{\Delta t}{\Delta x} \big(H(v_j^i) \rho_{j-S_i} \lambda_j + (1 - H(v_j^i)) \rho_j \lambda_j \\ &\quad - H(v_j^i) \rho_{j-S_i} \lambda_{j-S_i} - (1 - H(v_j^i)) \rho_j \lambda_{j-S_i} \big) \\ &= (1-\varepsilon) \frac{\Delta t}{\Delta x} \left(\lambda_j - \lambda_{j-S_i} \right) \left(H(v_j^i) \rho_{j-S_i} + (1 - H(v_j^i)) \rho_j \right). \end{aligned} \tag{15}$$

Equation (15) provides an efficient way for computing the gradient of the objective function with respect to the piecewise-constant velocity based on cells $\{C_j\}$ from our finite-volume discretization.

Alternatively, if the velocity $v = v(\mathbf{x}; \theta)$ is smoothly parameterized by the vector $\theta = [\theta_1, \ldots, \theta_k, \ldots, \theta_m]^\top \in \mathbb{R}^m$, for each θ_k, we can then evaluate

$$\frac{\partial \mathcal{J}}{\partial \theta_k} = \sum_{j=1}^{N} \sum_{i=1}^{d} \frac{\partial \mathcal{J}}{\partial v_j^i} \frac{\partial v_j^i}{\partial \theta_k} \tag{16}$$
$$\frac{\partial v_j^i}{\partial \theta_k} = \mathbf{e}_i \cdot \frac{\partial v}{\partial \theta_k}\bigg|_{(x_j - \mathbf{e}_i \Delta x_i/2; \theta)}$$

to determine the derivative $\partial_\theta \mathcal{J}$. By using a similar indexing convention to Sec. III A, we can collect the terms $\partial_{v_j^i} \mathcal{J}$ and $\partial_{\theta_k} v_j^i$ into the vectors $\partial_v \mathcal{J}$ and $\partial_{\theta_k} v$, respectively. Therefore, the double summation in (16) is achieved by the inner-product $\partial_v \mathcal{J} \cdot \partial_{\theta_k} v$. Note that

for different θ_k, we only need to change $\partial_{\theta_k} v$ as $\partial_v \mathcal{J}$ does not depend on θ_k.

B. Velocity parameterization

We now apply Eqs. (15) and (16) to evaluate the gradients of several parameterized velocity models. Specifically, we consider piecewise constant, global polynomial, and neural network parameterizations of the velocity.

1. Piecewise-constant parameterization

In the case of the piecewise-constant parameterization, we model the velocity as

$$v(\mathbf{x};\theta) = \sum_{i=1}^{d}\sum_{i=j}^{N} v_j^i \, \chi_{C_j}(\mathbf{x})\, \mathbf{e}_i, \qquad \theta = \{v_j^i\}. \tag{17}$$

Here, we again use the column-major ordering from Sec. III A to accumulate vectors of cells C_j with centers x_j and velocity components

$$v_j^i = v(x_j - \mathbf{e}_i \Delta x_i/2) \cdot \mathbf{e}_i$$

along the ith direction of the cell face located at $x_j - \mathbf{e}_i \Delta x/2$. The parameter space of the model presented in (17) is given by $\{v_j^i\}$, which has size $N \cdot d$, and the gradient of the parameters $\{v_j^i\}$ can be directly evaluated by (15).

We remark that (17) is only one variant of piecewise-constant parameterization since the parameterization mesh is the same as the discretization mesh in the finite-volume method; see Sec. III A. These two meshes do not have to be coupled together. To reduce the numerical error from the first-order scheme, it is preferable to reduce the spacing $\{\Delta x_i\}$, but we can keep the parameterization mesh fixed so that the size of the optimization problem does not change. In this case, we need to apply the chain rule (16) to obtain the final gradient after evaluating (15).

The model defined by (17) can be learned by gradient-based optimization methods. The regularity of the piecewise-constant model defined by (17) can be improved to a C^0 function by interpolating between the values v_j^i using either piecewise linear or higher-order piecewise polynomial functions, as in Ref. 63.

2. Global polynomial parameterization

Though the regularity of the piecewise-constant model given by (17) can be improved by interpolation, the inverted velocity $v(\mathbf{x};\theta)$ may still be highly oscillatory if the mesh size Δx is small. Modeling approaches, such as SINDy,[7] learn the velocity fields of dynamical systems from a polynomial basis together with sparse regression. Here, we show how the gradient derivation in (16) can be adapted to such polynomial basis parameterizations of the velocity field:

$$v(\mathbf{x};\theta) = [v^1(\mathbf{x};\theta), \ldots, v^d(\mathbf{x};\theta)]^\top = \sum_{i=1}^{d} v^i(\mathbf{x};\theta)\, \mathbf{e}_i.$$

The ith component of the velocity field $v^i(\mathbf{x};\theta)$ parameterized by a linear combination of the monomial basis of degree at most K can be written as

$$v^i(\mathbf{x};\theta) = \sum_{\ell=1}^{M} a_\ell^i (\mathbf{x}^\top \mathbf{e}_1)^{{}^1k_\ell^i} \ldots (\mathbf{x}^\top \mathbf{e}_d)^{{}^dk_\ell^i},$$
$$M = \binom{d+K}{K}, \tag{18}$$

where the powers are represented by multi-indices

$$k_\ell^i = ({}^1k_\ell^i, \ldots, {}^dk_\ell^i),$$

with $1 \le \ell \le M$, $|k_\ell^i| \le K$, and $\theta = \{a_\ell^i\}$. The size of θ in this case is $d \cdot M$. To learn the model parameterized by (18), we can use (16) to compute the gradient $\partial \mathcal{J}/\partial a_\ell^i$. Without loss of generality, we assume $\Delta x_i = \Delta x$ for all $1 \le i \le d$. The only term in (16) that explicitly depends on the velocity parameterization is

$$\frac{\partial v_j^i}{\partial a_\ell^i} = \left((x_j - \mathbf{e}_i \Delta x/2)^\top \mathbf{e}_1\right)^{{}^1k_\ell^i} \ldots \left((x_j - \mathbf{e}_i \Delta x/2)^\top \mathbf{e}_d\right)^{{}^dk_\ell^i},$$

where i, j, a_ℓ^i and the multi-index k_ℓ^i are fixed. Note that $\partial_{a_\ell^i} v_j^{i'} = 0$ if $i' \neq i$. Thus, we can again use gradient-based methods to infer proper polynomial coefficients $\{a_\ell^i\}$.

Although a global polynomial parameterization guarantees ideal C^∞ regularity of the parameterized velocity $v(\mathbf{x};\theta)$, the Runge phenomenon could be a potential downside of this approach. Specifically, as we increase the maximum degree K of the polynomial basis, we may encounter substantial interpolation errors near the boundary $\partial\Omega$.

3. Neural network parameterization

Motivated by the universal approximation theory of neural networks,[64] we may also choose to model each component of the velocity $v^i(\mathbf{x};\theta)$ as a feed-forward neural network, where the tunable parameters θ make up the network's weights and biases. We follow Ref. 65 to combine the adjoint-state method for the PDE constraints and the backpropagation technique to update the weights and biases of the neural network.

The term $\partial_{v_j^i} \mathcal{J}$ in the gradient calculation (16) can be computed by first evaluating the neural network on the mesh of cell face centers oriented in the direction of $\mathbf{e}_i$ to obtain $\{v_j^i\}$, which is then plugged into (15) to obtain $\partial_{v_j^i} \mathcal{J}$. The remaining term $\partial_\theta v$ in (16) is then computed via the backpropagation technique.[24]

For simplicity, we restrict ourselves to single-layer feed-forward networks. Moreover, by using a smooth activation function, such as the hyperbolic tangent or the sigmoid function, we can guarantee C^∞ regularity of the reconstructed velocity $v(\mathbf{x};\theta)$ on the domain Ω. To enforce the zero-flux boundary condition, we manually set $v = 0$ on $\partial\Omega$. Consequently, the neural network parameterization may lack regularity near $\partial\Omega$. However, if the domain is sufficiently large, the support of the physical measure will be very far from $\partial\Omega$, in which case, we will not observe any discontinuities originating from the boundary condition while simulating the trajectories based on (6). As we increase the number of nodes in the hidden layer of the neural network, both the approximation power and the potential difficulty of training the neural network are expected to increase.

V. NUMERICAL RESULTS

In this section, we present several numerical examples to demonstrate the utility of the proposed approach for learning dynamical systems from invariant measures with intrinsic noise. [We include a publicly available code (https://github.com/jrbotvinick/Learning-Dynamics-on-Invariant-Measures), which contains an example demonstrating the velocity inversion for the Van der Pol oscillator (19) based on a global polynomial parameterization. It can also be used to reproduce the comparison in Fig. 1 and Table II.] In Sec. V A, we study the inverse problem for the Van der Pol oscillator with piecewise constant, global polynomial, and neural network parameterizations of the velocity. In Sec. V B, we time-delay embed a signal sampled from a Hall-effect thruster and proceed to model the dynamics in delay-coordinates based upon the time-delayed invariant measure. We then illustrate that a low-dimensional embedding may increase the uncertainty of the learned model and that the choice of parameterization largely affects the regularity of the reconstructed velocity. In Sec. V C, we study rolling averages of a temperature data set and perform uncertainty quantification using the learned Fokker–Planck PDE in time-delayed coordinates. We conclude in Sec. V D by inverting a component of the Lorenz-63 system's velocity using a neural network parameterization. All experiments are conducted using an Intel i7-1165G7 CPU.

A. Van der Pol oscillator

We begin by considering the autonomous Van der Pol oscillator,[66] given by

$$\begin{cases} \dot{x} = y, \\ \dot{y} = c(1 - x^2)y - x. \end{cases} \tag{19}$$

Our results for learning a dynamical system with prescribed statistical properties given by the stochastically forced Van der Pol oscillator are shown in Fig. 3. In the top row, the first panel features the velocity of (19) for the choice of $c = 0.5$, the second panel shows the approximate occupation measure [see (4)] obtained from the simulation of a single SDE trajectory [see (6)], the third panel shows the SDE trajectory used to approximate the invariant measure, and the fourth panel shows the dynamics of the oscillator without stochastic forcing. Throughout, we color the SDE trajectories by their histogrammed density to illustrate the connection between the Lagrangian and Eulerian perspectives. We also stress that the experiment in Fig. 3 assumes the diffusion coefficient to be known *a priori*, but that Sec. V B relaxes this assumption.

In the following rows of Fig. 3, we use neural network, piecewise constant, and global polynomial parameterizations of the velocity to solve the inverse problem using the optimization framework from Secs. III A and IV. For the case of the neural network parameterization, we compare each objective function studied in Sec. IV A 1, while we only focus on the L^2 objective for the remaining two parameterizations. Across all tests, the reconstructed velocity is shown to vary significantly from the true velocity shown in the first row of Fig. 3. This is mainly due to the lack of data away from the main attracting limit cycle. In regions of the state space with no available data, we can only expect that the modeled velocity $v(\mathbf{x};\theta)$ will direct trajectories toward the attracting limit cycle on which the invariant measure is supported. Indeed, this is what we observe.

Moreover, while the learned PDE model (9) matches the observed occupation measure (4) across all tests, we find that the SDE and ODE trajectories generated using the learned velocity vector fields may vary depending on the parameterization. Table I provides a comparison of the accuracy of the learned models, as well as the required computation times. While the piecewise constant velocity is by construction discontinuous and, thus, does not naturally guarantee the existence and uniqueness of the corresponding ODE solution, the neural network parameterization based on the hyperbolic tangent activation function yields a C^∞ velocity. Moreover, while the global polynomial parameterization is also C^∞, it may suffer from the Runge phenomena and grow rapidly near the boundary of the domain. Thus, we mainly consider neural network parameterizations of the velocity for the remainder of the numerical tests.

To reduce the computational cost of the inversion in the final row of Fig. 3, we compute $\mathcal{J} = W_2^2$ on a coarsened mesh. Among the four objective functions in Fig. 3, it is worth noting that the W_2 metric does not compare the two densities pointwisely and is well-defined for comparing singular measures. The distance reflects both the local intensity differences and the global geometry mismatches.[67] It has also been shown that the Wasserstein metric is robust to noise.[68,69] Thanks to the geometric nature of the optimal transportation problem, the Wasserstein metric is primarily sensitive to global changes, such as translation and dilation, and is robust to small local perturbations, such as noisy measurements of ρ^*. The better stability also brings a downside as the optimization landscape can be relatively flat around the ground truth, which may lead to compromised accuracy in the velocity inversion.

The different velocities shown in the second column of Fig. 3 reveal that there is nonuniqueness if we only use the invariant measure as the reference data. The current modeling assumption yields dynamics reproducing the same invariant measure but does not necessarily recover the same velocity field. Depending on the concrete application, one can add regularization, time information, or focus on velocities in a particular parameterized subspace to avoid nonuniqueness. The large error for the reconstructed velocity near the origin is due to the fact that the method aims to learn the flow on or (in the case of stochastically forced dynamics) near the invariant measure. It is, therefore, unsurprising that the learned velocity does not match the ground truth where there are no data.

In Fig. 4, we show how the inversion accuracy and computation time depend on the chosen value of Δx; that is, as Δx decreases, we can learn velocities that can reproduce the statistics of the observed occupation measure more accurately, with the cost of longer computation time.

Next, we provide experimental details on the comparison of our approach with SINDy[7] and the neural ODE[4] frameworks in Fig. 1. This test uses the Van der Pol oscillator with $c = 2$. Since the SINDy and neural ODE methods are designed for modeling ODEs, the experiments in Fig. 1 use the diffusion coefficient $D = 0$. While we only plot the first eight points of the slowly sampled trajectory in Fig. 1, the full trajectory used for inference contains 2.5×10^3 observations. The quickly sampled trajectory also consists of 2.5×10^3 observations. The three approaches considered for comparison have

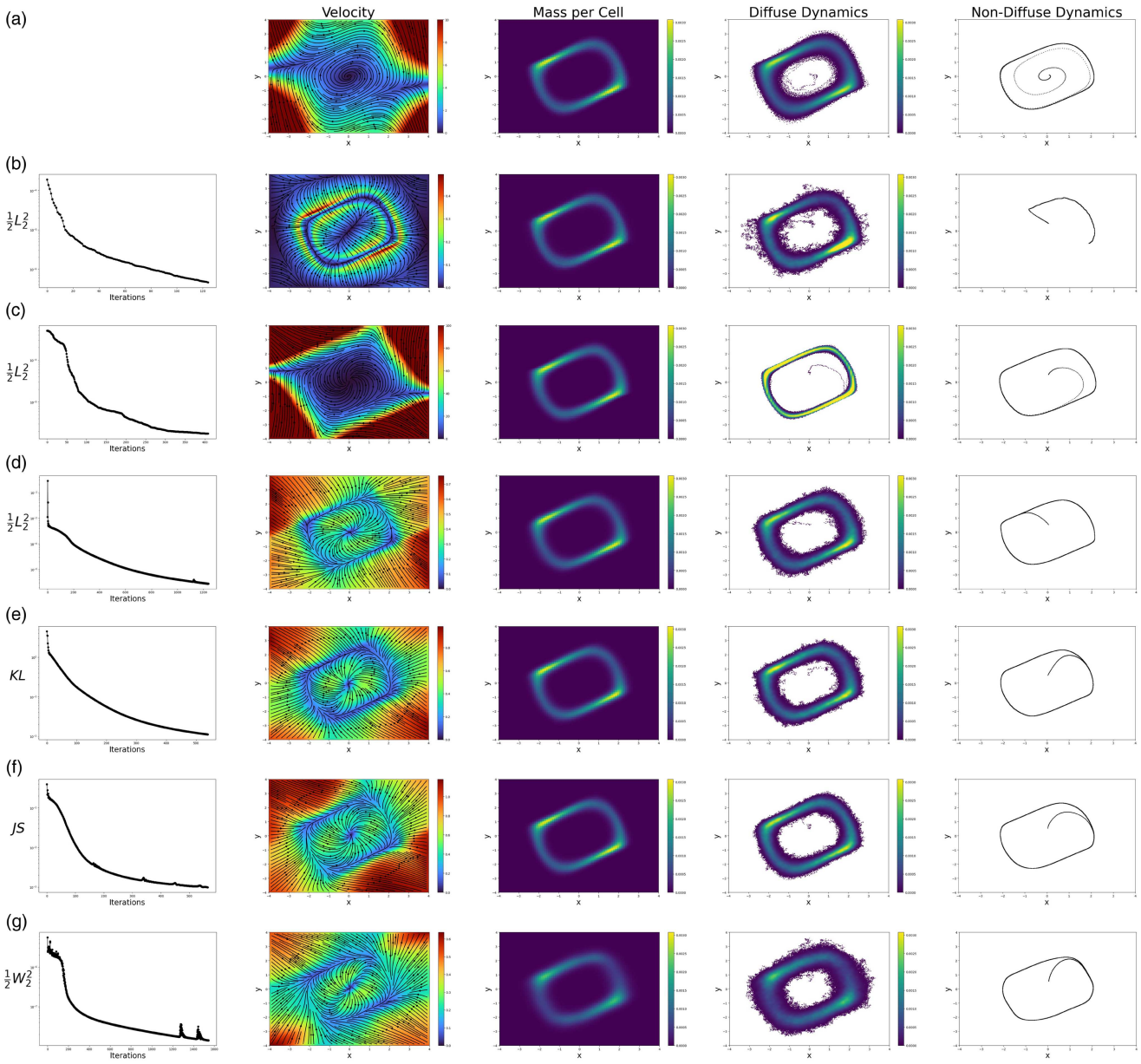

FIG. 3. Learning velocity fields to reproduce the statistics of the stochastically forced Van der Pol oscillator. The ground truth occupation measure, velocity, and dynamics are shown in (a). The results for inverting the velocity based on the occupation measure from (a) using piecewise constant, global polynomial, and neural network parameterizations are shown in (b)–(g). The first column shows the objective function, the second column shows the learned velocity vector field, the third column shows the final PDE forward model evaluation based on the learned velocity, the fourth column shows the simulation of a diffuse trajectory, and the final column shows the simulation of a trajectory without diffusion. Specifically, the "diffuse trajectories" are simulated according to the Euler–Maruyama method using the assumed diffusion coefficient $D = 0.02$, while the "non-diffuse" trajectories assume $D = 0$. The coloring of each diffuse trajectory is given by the occupation measure it generates; see (4). Across all tests, the objective function is minimized to 0.25%–0.35% of its initial value. For (b)–(c), the L-BFGS-B algorithm is used for optimization. In (d)–(g), the neural network architecture consists of a single hidden layer with the hyperbolic tangent activation function, trained by the Adam optimizer with a learning rate of 10^{-1}. (a) Ground truth velocity, occupation measure, diffuse trajectory, and non-diffuse trajectory for the Van der Pol oscillator with $c = 0.5$ and $D = 0.02$. (b) Piecewise constant parameterization (see Sec. IV B 1) with the squared L^2 objective function. (c) Degree five global polynomial parameterization (see Sec. IV B 2) with the squared L^2 objective function. (d) Neural network parameterization (see Sec. IV B 3) with the squared L^2 objective function. (e) Neural network parameterization with the KL divergence objective function. (f) Neural network parameterization with the JS divergence objective function. (g) Neural network parameterization with the squared W_2 objective function.

TABLE I. Comparison of the wall-clock computation time and the error for the experiments shown in Fig. 3. The error is quantified by the squared W_2 distance between the occupation measure of the ground truth diffuse trajectory [see the third panel of Fig. 3(a)] and the occupation measure accumulated from the simulation of a trajectory with diffusion according to the learned velocity [see the fourth panel of Figs. 3(c) and 3(d)].

Parameterization	Objective	Wall-clock time (s)	Error
Piecewise constant	L^2	3.13×10^{1}	2.36×10^{-1}
Global polynomial	L^2	1.75×10^{2}	2.90×10^{-2}
Neural network	L^2	4.14×10^{2}	9.07×10^{-3}
Neural network	KL	2.04×10^{2}	7.11×10^{-3}
Neural network	JS	2.16×10^{2}	9.48×10^{-3}
Neural network	W_2	2.16×10^{3}	1.07×10^{-2}

various hyperparameters, which can be tuned. For SINDy, we learn the models from the monomial basis up to degree three and use the sequentially thresholded least squares optimizer with a threshold of 0.025 to enforce a sparsity condition on the learned coefficients; see Ref. 7. For the neural ODE framework, the velocity is parameterized by a single-layer fully connected neural network with 100 nodes and a hyperbolic tangent activation function. The neural ODE is trained using a multiple shooting approach with the mean-squared error objective function. More specifically, rather than treating the simulation of a single long time trajectory as the forward model, we integrate $N-1$ trajectories initiated at the observed data points $\{\mathbf{x}(t_i)\}_{i=1}^{N-1}$ for a time of $\Delta t = t_{i+1} - t_i$. This approach results in greater success while modeling slowly sampled dynamics. The Adam optimizer with a learning rate of 10^{-3} is used, and the tolerance for both relative and absolute errors of the ODE solver is set as 10^{-5}.

To ensure a fair comparison with the neural ODE framework, we consider our approach based on a neural network parameterization of the velocity using the same architecture, optimizer, and learning rate. For our approach, we use the KL-divergence objective function (see Sec. IV A 1), apply additional Gaussian filtering to the occupation measure [see (4)] to simplify the resulting optimization, assume a diffusion coefficient of $D = 10^{-3}$ during training, and set $\Delta x = 0.1$. Thus, the only differences between the setup for our approach and the neural ODE framework are the forward model and the objective function.

As shown in Fig. 1, all three frameworks can learn from the quickly sampled trajectory. However, SINDy and the neural ODE frameworks are less robust to changes in the sampling frequency of the inference data than our approach. This is further demonstrated in Table II, where we quantify the error in the simulated occupation measure based on the learned velocity. We report the average error over ten trials with different random training seeds to compare our method and the neural ODE framework. When the data are sampled at a sufficiently high frequency, Table II also shows that methods, such as SINDy or the neural ODE, are preferable in terms of both computational cost and accuracy.

B. Hall-effect thruster

We now turn to the more realistic setting of experimentally sampled time-series data. Specifically, we study the cathode-Pearson

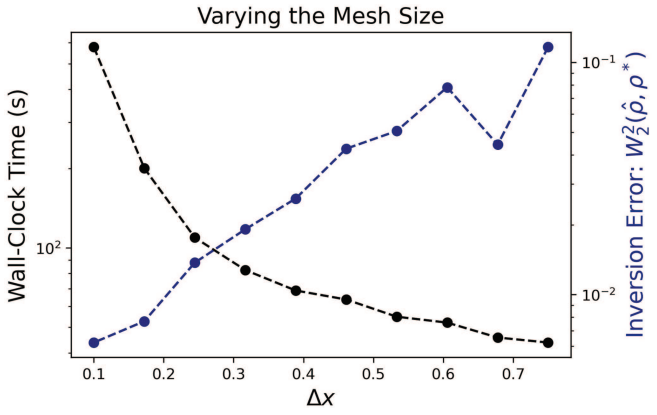

FIG. 4. We demonstrate how the computation time and inversion accuracy depend on the mesh spacing used in the first-order FVM solver. Here, we use the Van der Pol oscillator with $D = 0.05$ and learn the velocity using a neural network parameterization. The Adam optimizer is used with a learning rate of 10^{-2}. In each case, we reduce the KL divergence objective function below 0.5% of its initial value. The error is quantified in terms of the squared W_2 discrepancy between the simulated occupation measure $\hat{\rho}$ according to the learned dynamics and the observed occupation measure ρ^*.

signal sampled from a Hall-effect thruster (HET) in its breathing mode. Hall-effect thrusters are in-space propulsion devices that exhibit dynamics resembling stable limit cycles while in a breathing mode. For details about the experimental setup used to collect the data, the reader is encouraged to consult Refs. 70 and 71. In Sec. V B 1, we utilize Takens' theorem[23] to reformulate the large-scale optimization framework presented in Secs. III and IV to be compatible with scalar time-series observations, and in Sec. V B 2, we demonstrate numerical results based upon this reformulation.

1. Methods

Intrinsic physical fluctuations present in the cathode-Pearson signal indicate that the HET's dynamics may be modeled well by a Fokker–Planck equation. Motivated by this insight, we first time-delay embed the cathode-Pearson signal $C(t)$ in d-dimensions to form the trajectory $\mathbf{C}_{d,\tau}(t) := (C(t), C(t-\tau) \ldots, C(t-(d-1)\tau))$.

TABLE II. Comparison with the SINDy and neural ODE frameworks for learning from trajectories sampled at different frequencies (Hz). The wall-clock computation time is reported, and the error is quantified by $W_2^2(\hat{\rho}, \rho^*)$, where $\hat{\rho}$ is the simulated occupation measure from the learned velocity field and ρ^* is the observed occupation measure.

Method	Sampling freq.	Wall-clock time (s)	Error
SINDy	10.00	2×10^{-2}	5.6×10^{-3}
Neural ODE	10.00	5×10^{2}	5.32×10^{-3}
Ours	10.00	5×10^{2}	1.14×10^{-1}
SINDy	0.25	10^{-2}	3.52
Neural ODE	0.25	5×10^{2}	1.81
Ours	0.25	5×10^{2}	6.79×10^{-2}

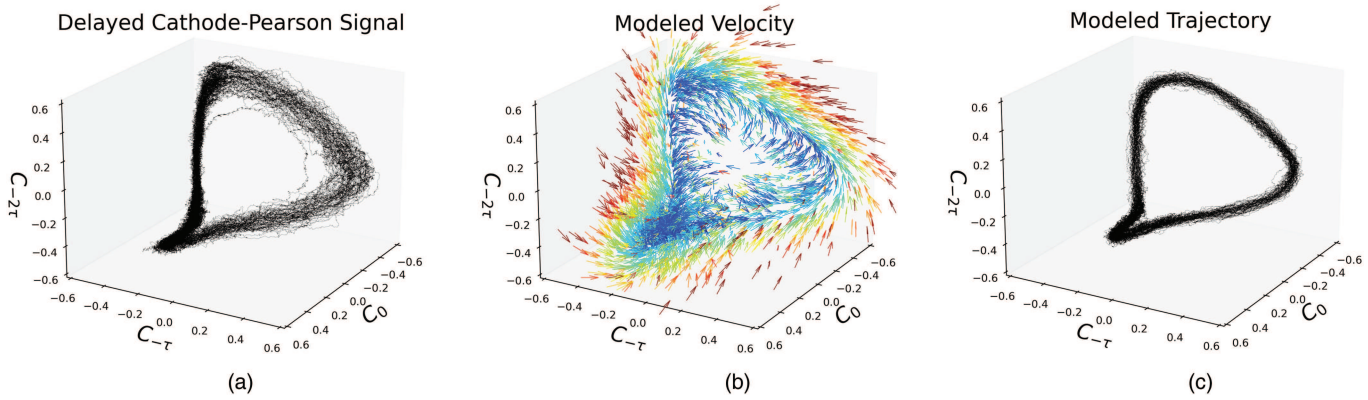

FIG. 5. Learning the velocity from the embedded cathode-Pearson signal's invariant measure. We present the time-delay embedded signal (a), the reconstructed velocity field from the embedded signal's occupation measure (b), and the trajectory simulated with the Euler–Maruyama method from the learned velocity and the diffusion coefficient (c). In (b), blue indicates slow speed and red indicates fast. The velocity was parameterized by a neural network with 500 nodes in a single hidden layer and learned using the KL divergence loss function. The three-step procedure in Sec. V B 1 is used to learn the model, and in step one, additional Gaussian filtering is applied to the occupation measure ρ^* to simplify the resulting optimization.

We then use a histogram approximation to compute the occupation measure ρ^* of $\mathbf{C}_{d,\tau}(t)$; see (4). By viewing each dimension of the coordinate system on which the measure ρ^* is supported as the independent variables $C_{-k\tau}(t) := C(t - k\tau)$ where $0 \le k \le d-1$, we then seek a solution to the optimization problem (2) for a velocity $v = v(\mathbf{C}_{d,\tau}; \theta)$. Such a velocity can then provide us with a model of the asymptotic statistics of the embedded trajectory $\mathbf{C}_{d,\tau}(t)$, provided that a suitable diffusion coefficient can be found.

We note that forming the time-delay coordinates $\mathbf{C}_{d,\tau}(t)$ does require a knowledge of measurements at uniform increments in time. However, the available data may still be sampled slowly enough such that it is impractical to seek a direct approximation of the Lagrangian velocity through the standard approaches described in Sec. I. This perspective motivates our use of the approach developed in Secs. III and IV to learn dynamical systems from invariant measures in time-delay coordinates.

There are a few additional considerations that arise when adapting the modeling framework presented in Secs. III and IV to real-world data; namely, we do not know the proper diffusion coefficient *a priori* (as was the case in Sec. V A). Moreover, the invariant measure that the model is based on does not contain any information about the time scale at which the system evolves. Toward this, we utilize the following three-step procedure as a computationally efficient means to mitigate these difficulties.

1. Bin the trajectory $\mathbf{C}_{d,\tau}(t)$ onto a d-dimensional mesh with spacing Δx along each axis to form the occupation measure ρ^*, assume a constant diffusion coefficient $D > 0$, and learn the velocity $v = v(\mathbf{C}_{d,\tau}; \theta)$, using the framework from Secs. III and IV.
2. Bin the trajectory $\mathbf{C}_{d,\tau}(t)$ onto another d-dimensional mesh with spacing $\Delta\hat{x} \le \Delta x$ to create a new occupation measure $\hat{\rho}^*$ and adjust the diffusion coefficient by solving the optimization problem
$$\tilde{D} = \underset{\hat{D}\in\mathbb{R}}{arg\,min}\, \mathcal{J}(\rho_\varepsilon(v; \hat{D}), \hat{\rho}^*), \tag{20}$$
where the term $\rho_\varepsilon(v; \hat{D})$ in (20) denotes the forward model evaluation with the diffusion coefficient $\hat{D}$.
3. Rescale both the velocity and diffusion by solving the optimization problem
$$\tilde{a} = \underset{a\in\mathbb{R}}{arg\,min} \sum_{i=1}^{N} \left\| \hat{\mathbf{C}}(t_i; a) - \mathbf{C}_{d,\tau}(t_i) \right\|_2^2, \tag{21}$$
where $\hat{\mathbf{C}}(t_i; a)$ denotes the time-t_i solution of the ODE initial value problem with velocity $av(\cdot;\theta)$ and initial condition $\mathbf{C}_{d,\tau}(t_0)$. The final velocity and diffusion are then given by $\tilde{a}v(\cdot;\theta)$ and $\tilde{a}\tilde{D}$, respectively.

The three-step approach makes repeated use of the fact that $\rho_\varepsilon(v; D) = \rho_\varepsilon(av; aD)$ for any scalar multiple $a > 0$. Indeed, if the true diffusion coefficient $D^* > 0$ is unknown *a priori*, but we instead seek a solution $v(\cdot,\theta)$ with a different diffusion $D > 0$, it is guaranteed that the velocity $v = (D/D^*)v^*$ will still provide a solution to the inverse problem. This observation motivates step one, in which an arbitrary diffusion coefficient is used to find a solution $v(\cdot;\theta)$ to the inverse problem. As the dimensionality d is increased, solving the large-scale optimization problem in step 1 on a fine mesh becomes infeasible. As such, step one is typically performed on a coarse mesh where additional Gaussian filtering is applied to the inference measure ρ^* to make the large-scale optimization more feasible.

The diffusion coefficient is then adjusted in step two on a finer mesh via (20) to mitigate the errors due to the Gaussian filtering, numerical diffusion, and histogram errors incurred during step one (see Fig. 2). Finally, in step three, the scale of both the velocity and diffusion is adjusted via (21) such that the time evolution of simulated trajectories is consistent with the inference trajectory $\mathbf{C}_{d,\tau}(t)$ in

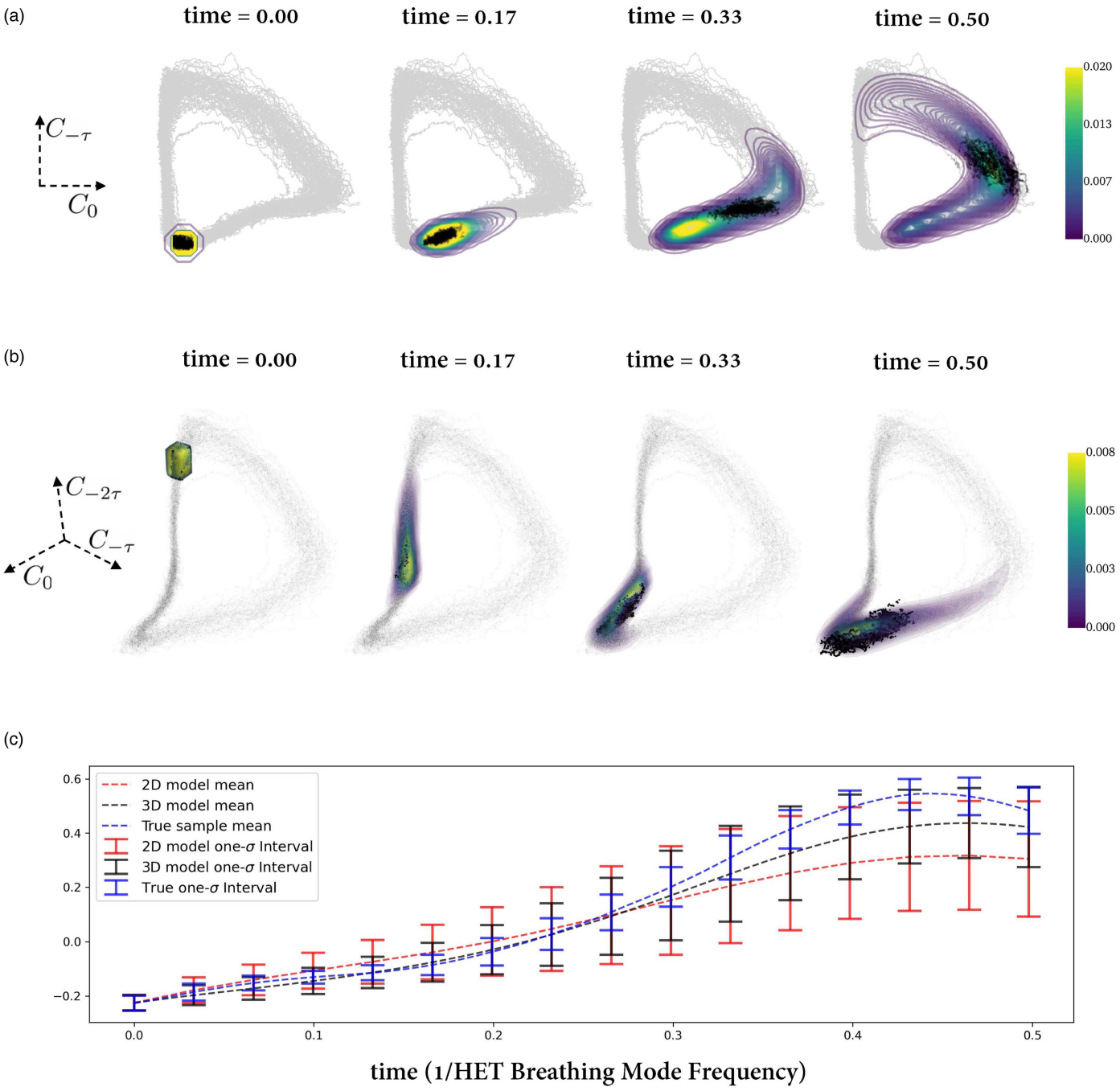

FIG. 6. Comparing the model accuracy and uncertainty for the embedded cathode-Pearson signal with 2D and 3D time delays. The time evolution of the 2D and 3D models is compared to a collection $\{\mathbf{C}_{d,\tau}(t_i)\}_{i=1}^{n}$ of samples (plotted in black) from the time-delayed cathode-Pearson signal. The plots (a) and (b) feature a qualitative comparison, whereas (c) shows a quantitative comparison of the uncertainties. Throughout, the time units are normalized to the inverse of a HET breathing mode frequency (16.6 kHz). Both the 2D and 3D models utilized neural network velocity parameterization with 500 nodes in a single hidden layer and reduced the KL divergence objective function to 0.1% of its initial value during training. As in Fig. 5, the three-step procedure in Sec. V B 1 is used to learn the models, and in step one, additional Gaussian filtering is applied to the occupation measure ρ^* to simplify the resulting optimization. The 3D visualization was plotted using Ref. 72. (a) Using the 2D model to predict the evolution of the samples $\mathbf{C}_{2,\tau}$. (b) Using the 3D model to predict the evolution of the samples $\mathbf{C}_{3,\tau}$. (c) Uncertainty comparison for the 2D and 3D model predictions.

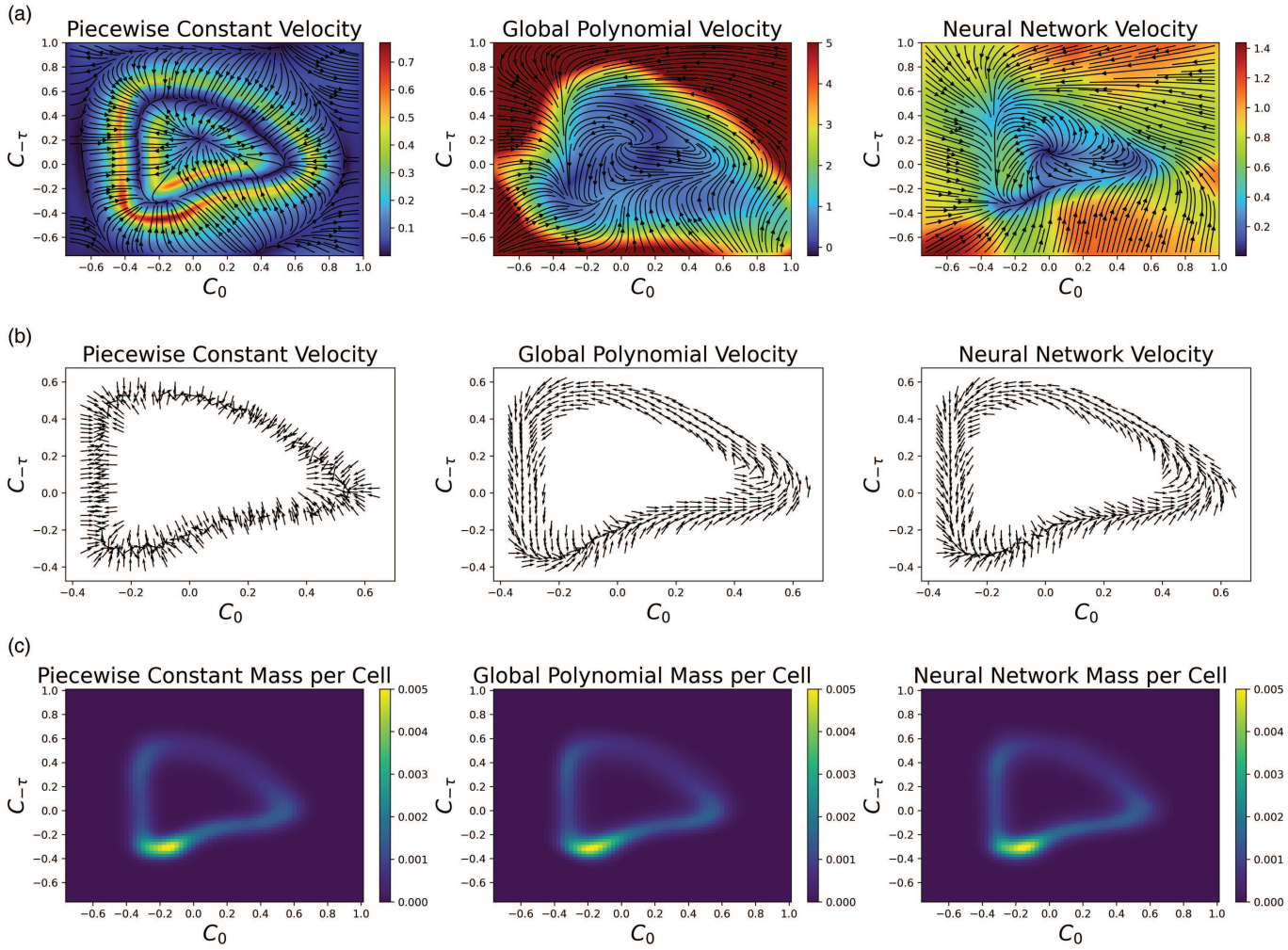

FIG. 7. Comparison between the three parameterizations detailed in Sec. IV B for learning a velocity field from the time-delayed cathode-Pearson signal's invariant measure using a diffusion coefficient $D = 0.01$. The learned velocities and densities for the piecewise constant (PC), global polynomial (GP), and neural network (NN) discretizations are shown in three columns, respectively. We show the learned velocity field on the full state space (a), a close-up of the velocity field's direction near the attracting limit cycle (b), and the forward model output $\rho_\varepsilon(v(\theta))$ for each parameterization (c). The resulting parameter spaces of these discretizations have a dimensionality of 9800 (PC), 56 (GP), and 400 (NN). The L^2 loss is reduced below 0.1% of its initial cost for the PC and NN discretizations and reduced below 0.7% of its initial value for the GP case when we stopped the optimization.

delay coordinates. Since diffusion plays a relatively small role over short time scales for the quasiperiodic HET data, we use the zero-diffusion trajectory to calibrate reasonable time scaling between our model and the available data. However, as the magnitude of the diffusion increases, the least squares fit in (21) will become less reliable, and it may be preferable to instead minimize a transport cost between a collection of model samples and a collection of data samples at each time step. While this final optimization is similar in spirit to various Lagrangian approaches for learning dynamics (see Sec. I), we remark that the parameter space in (21) has only one dimension.

2. Results

The results of the three-step procedure in Sec. V B 1 for learning the HET dynamics are shown in Fig. 5 for an embedding dimension of $d = 3$ and time-delay of $\tau = 1.4 \times 10^{-5}$ sec or rather $\tau = .23$ when normalizing the time scale to the HET breathing mode frequency (16.6 kHz). The modeled trajectory accurately reconstructs the shape of the embedded cathode-Pearson signal but cannot capture the variable diffusion present throughout the time-delayed signal. We do not expect to capture such details, as we assume a constant diffusion coefficient in our model. Nevertheless, we regard the reconstruction of the 3D globally

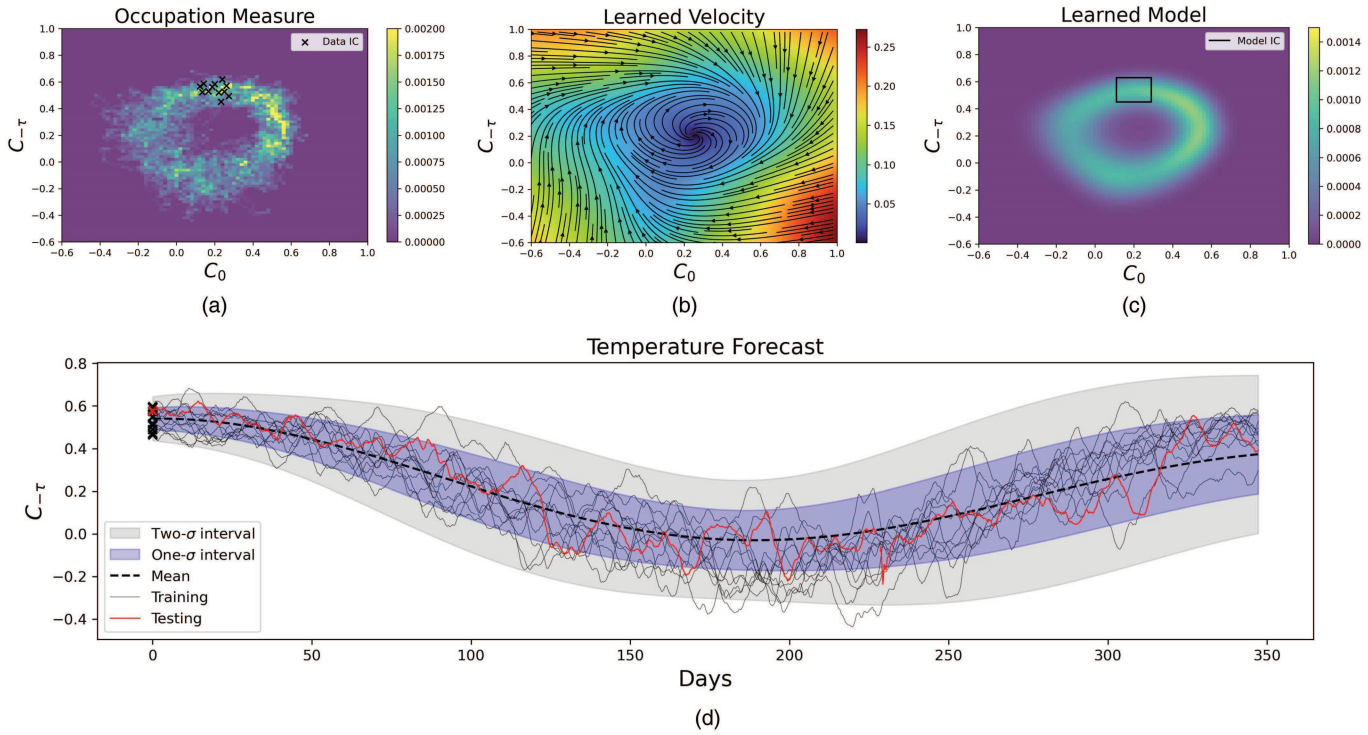

FIG. 8. Performing prediction and uncertainty quantification for Ithaca, NY's temperature in 2019. (a) The ground truth occupation measure accumulated from 13 years of weekly rolling averaged temperature observations, normalized by an affine transformation to $[-1, 1]$. (b) The learned velocity vector field. (c) The corresponding forward model output. In (d), the PDE model with a uniform initialization in the box from (c) is evolved in time and used to quantify the uncertainty in the measurements of C_0. Observed trajectories of the temperature in delay coordinates with initial conditions displayed in the top left plot are also shown to demonstrate the effectiveness of the learned model. A time delay of $\tau = 280$ days is used, and the model is trained using a neural network parameterization and the KL-divergence objective function.

attracting limit cycle as a success and leave extending the model to account for the case of a non-constant diffusion tensor to future work.

The dimensionality of the original HET dynamics is unknown, and as such, a sufficient embedding dimension for the cathode-Pearson signal is unclear, though likely very high. Interestingly, we can compare the model learned in Fig. 5 with a 2D analog to demonstrate that when the number of time delays is not sufficiently large, there is more uncertainty in modeling the time-delayed dynamics. This phenomenon is most evident when inspecting regions of the delayed cathode-Pearson signal for which the 2D embedding lacks structure readily observed in 3D.

Specifically, consider a collection of nearby samples $\{\mathbf{C}_{3,\tau}(t_i)\}_{i=1}^n$ in the 3D time-delay coordinate system $(C_0, C_{-\tau}, C_{-2\tau})$. The corresponding 2D samples $\{\mathbf{C}_{2,\tau}(t_i)\}_{i=1}^n$ will also be nearby one another in the 2D time-delay coordinate system $(C_0, C_{-\tau})$. In Fig. 6, we initiate uniform distributions centered about these samples in both 2D and 3D time-delay coordinate systems. We then evolve both the samples and initial uniform distributions forward in time. The evolution of the ground truth samples is simply determined by the time-delayed cathode-Pearson signal $\mathbf{C}_{d,\tau}(t)$, and the evolution of the uniform distributions is given by Fokker–Planck models constructed from the time-delayed cathode-Pearson signal's invariant measure. As the modeled probability densities and ground truth samples evolve in time, we observe in Fig. 6 that the mean of the 3D model matches the true sample mean more closely than the 2D model and that it has less uncertainty.

In Fig. 7, we study the three parameterizations from Sec. IV B for learning the time-delayed cathode-Pearson signal's velocity, now with an embedding dimension of two to allow for clearer visualizations. It can be seen that the density associated with each velocity parameterization indeed matches the ground truth density in Fig. 7, but that the velocity fields differ significantly from one another. The piecewise-constant velocity in Fig. 7 suffers from poor regularity with discontinuities on the attracting limit cycle. As a result, we lose the connection between the Eulerian and Lagrangian dynamics and cannot reconstruct zero-diffusion trajectories that form a stable limit cycle. On the other hand, the velocities parameterized by the global polynomial and the neural network are both C^∞. The differences among these three can clearly be seen via the zoomed-in velocity plots in the second row of Fig. 7. The global polynomial and neural network discretizations are both global parameterizations of the velocity, and as such, their values near the domain's boundary are dictated by the available data in the center of

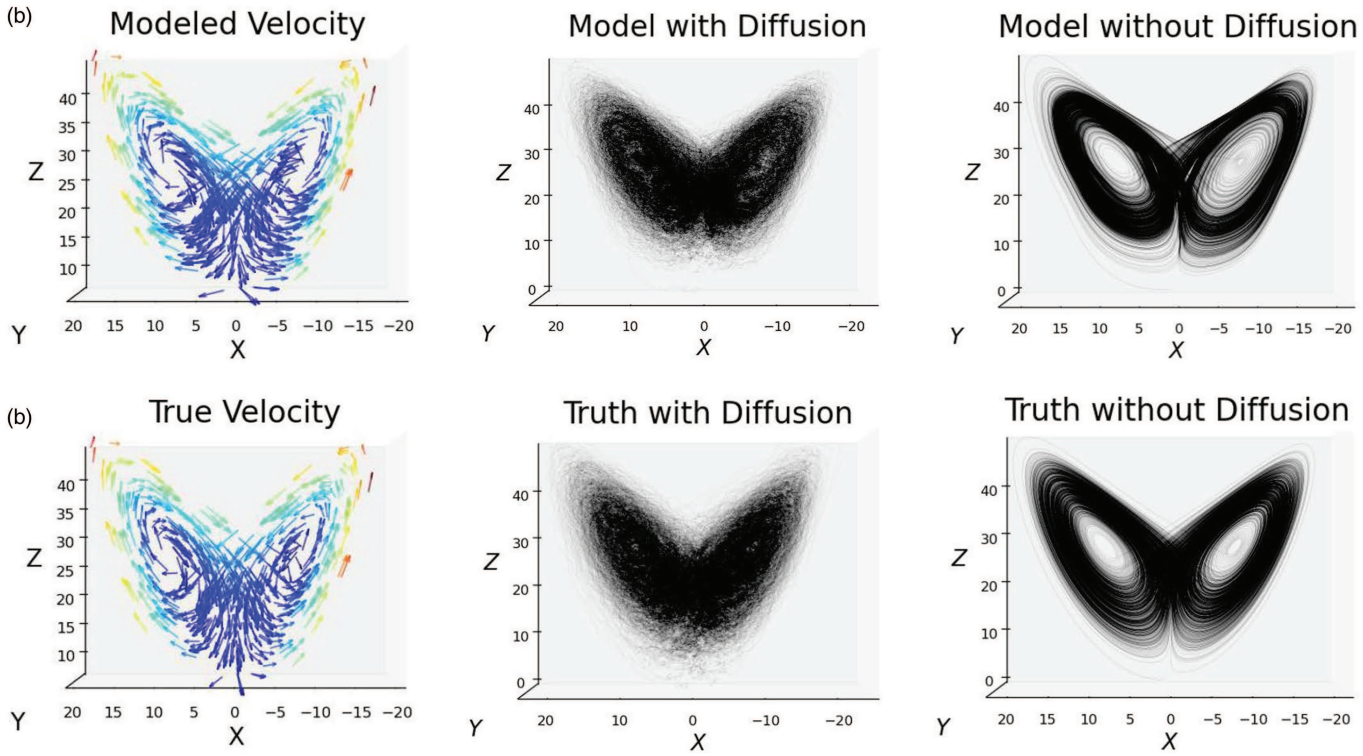

FIG. 9. Neural network parameterization of $\dot{x}$ using the Lorenz system's stochastically perturbed invariant measure with $D = 10$ and $\Delta x = 2$. For visualization of the occupation measure used to learn the model displayed in the top row, we refer to Ref. 18. (a) Learned velocity vector field (left), a simulated trajectory with diffusion (middle), and a simulated trajectory without diffusion (right). (b) True velocity (left), a ground truth SDE trajectory (middle), and a ground true ODE trajectory (right).

the domain. This causes the polynomial velocity to rapidly increase near the boundary, and a similar effect can also be seen for the neural network.

It is worth noting that the initial condition for the optimization in Fig. 7 can play a large role in the reconstructed velocity, which is related to the optimization landscape of the nonconvex optimization problem (2) we tackle. In the case of the piecewise-constant discretization, we initialize all velocities to be significantly less than the diffusion coefficient $D = 0.1$. Thus, diffusion initially dominates in the finite-volume solver, and all non-boundary cells will contain nonzero mass, which allows for accurate gradient updates everywhere. This phenomenon can also help neural network training, though it is not always necessary due to the global nature of parameterization. Moreover, we initialize our polynomial basis to form the velocity

$$(\dot{x}, \dot{y}) = \left(-y + x(0.1 - x^2 - y^2), x + y(0.1 - x^2 - y^2)\right),$$

which describes a globally attracting limit cycle. To converge to the ground truth limit cycle of the time-delayed cathode-Pearson signal, this initial velocity only needs to be translated and deformed.

C. Temperature uncertainty quantification

We now study 2D time-delay embedded data of weekly rolling averages of the temperature in Ithaca, NY, between 2006 and 2020.[73] We view temperature fluctuations over short time scales as an intrinsic diffusion process and the approximately periodic oscillation of seasonal temperatures driven by some nonzero velocity. Thus, we model the 2D data in delay coordinates as a diffuse limit cycle. We again follow the procedure in Sec. V B 1 to learn a velocity $v(\mathbf{x}; \theta)$ and diffusion coefficient D, which closely matches the occupation measure.

As in Sec. V B, we can use the trained model $v(\mathbf{x}; \theta)$ to quantify measurement uncertainties through the Fokker–Planck equation (9), whose solution is a probability density in the time-delay coordinates $(C_0, C_{-\tau})$. Specifically, if we know some initial probability distribution that captures the current state of the temperature system well, we can consider the time evolution of the distribution using our trained model to quantify the uncertainty of future temperature measurements. The process of evolving both the Fokker–Planck PDE from a uniform distribution and the ground truth sample paths from past temperature measurements is shown in Fig. 8. The uncertainty bounds from the model accurately capture fluctuations in the training data used to form the occupation

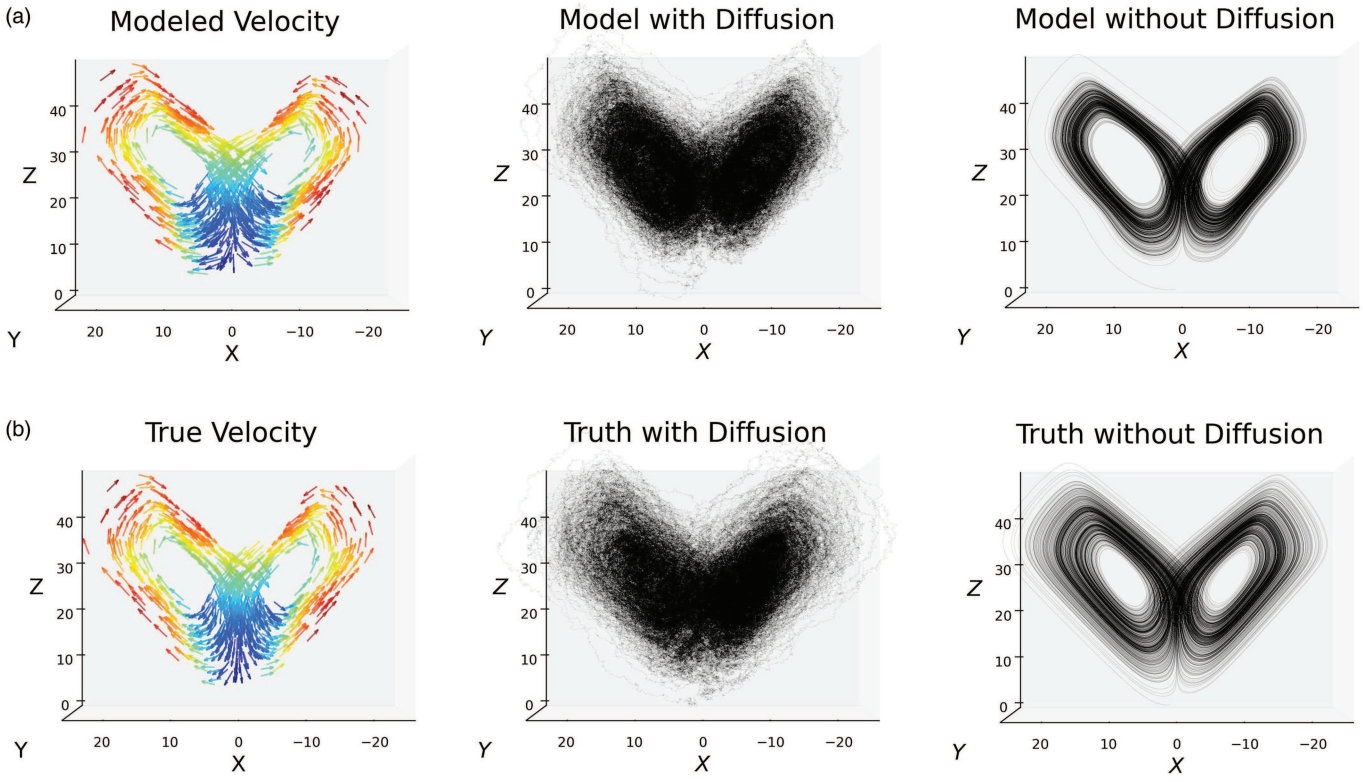

FIG. 10. Neural network parameterization of $\dot{x}$ using the arctan Lorenz system's stochastically perturbed invariant measure with $D = 10$ and $\Delta x = 2$. The neural network used to learn the velocity contains a single layer of 100 nodes with the sigmoid activation function, and the L^2 objective function is used to train the model. (a) Learned velocity vector field (left), a simulated trajectory with diffusion (middle), and a simulated trajectory without diffusion (right). (b) True velocity (left), a ground truth SDE trajectory (middle), and a ground true ODE trajectory (right).

measure (plotted in black), as well as a testing sample path previously unseen by the model (plotted in red).

It is also worth noting that the confidence intervals we construct may be larger than the actual range due to several factors, including additional extrinsic noise from filtering the data, modeling errors accumulated from the hypothesis space, numerical diffusion in the forward model, and a sub-optimal embedding dimension. Reducing such errors may result in tighter confidence intervals, and considering time delays in higher dimensions could yield better predictions of the temperature's transient behaviors.

D. Lorenz-63 system

We conclude this section by studying the Lorenz-63 system,[21] defined by

$$\begin{cases} \dot{x} = c_1(y - x), \\ \dot{y} = x(c_2 - z) - y, \\ \dot{z} = xy - c_3 z, \end{cases} \tag{22}$$

where we consider $(c_1, c_2, c_3) = (10, 28, 8/3)$. For these choices of parameters, the Lorenz-63 system exhibits chaotic behavior and admits a unique physical measure.[22] In Fig. 9, we assume that the quantities $\dot{y}$ and $\dot{z}$ are known, and we learn a model for the velocity in the x-direction, using the stochastically forced Lorenz-63 system's occupation measure. We emphasize that the data used to approximate the Lorenz system's occupation measure can be sampled slowly or even randomly in time (see Fig. 7 in Ref. 18). From the approximate occupation measure, we are able to successfully invert the first component $\dot{x}$ of the Lorenz-63 system's velocity via neural network parameterization.

We remark that when $\dot{x}$, $\dot{y}$, and $\dot{z}$ are all simultaneously inverted, the optimization is unsuccessful at reconstructing the true velocity (22). While we may be able to learn a velocity that approximately recovers the stationary state of the Lorenz-63 system in the sense of (9), the physical property (3) does not hold. Whether the difficulties of inverting all velocity components of the Lorenz-63 system are due to inherent non-uniqueness in the inverse problem or simply inconvenient local minima during training is worth further investigation in future work. To demonstrate the applicability of our approach to non-rational velocities, we also consider the arctan Lorenz-63

system,[18] given by

$$\begin{cases} \dot{x} = 50\arctan(c_1(y-x)/50), \\ \dot{y} = 50\arctan(x(c_2-z)/50 - y/50), \\ \dot{z} = 50\arctan(xy/50 - c_3 z/50), \end{cases} \tag{23}$$

where again, $(c_1, c_2, c_3) = (10, 28, 8/3)$. The results for inverting $\dot{x}$ from the occupation measure generated by (23) with additional stochastic forcing are shown in Fig. 10, assuming that the quantities $\dot{y}$ and $\dot{z}$ are known.

VI. CONCLUSION

In this paper, we introduced a PDE-constrained optimization approach to modeling trajectory data originating from stochastic dynamical systems. We first adapted the invariant measure surrogate model in Ref. 18 based upon the continuity equation to the Fokker–Planck equation. This increased our modeling capacity and prevented overfitting the reconstructed velocity while modeling intrinsically noisy trajectories. We next extended the three-coefficient learning performed in Ref. 18 to thousands of coefficients by modeling the velocity via global polynomials, piecewise polynomials, and fully connected neural networks. The efficient gradient computation presented in Sec. IV made these large-scale parameterizations of the velocity computationally tractable. We finally studied velocity inversion for invariant measures of time-delay embedded observables. The method of time-delay embedding is useful for analyzing real-world data, where in many cases, only limited observations of complex systems are available. As such, we proceeded to learn the velocity in time-delay coordinates for a Hall-effect thruster system and rolling weekly averages of temperature measurements. Using these models, we predicted future states of the systems and quantified uncertainty in forecasts by evolving the learned Fokker–Planck equation forward in time.

ACKNOWLEDGMENTS

This paper was supported in part by a fellowship award under Contract No. FA9550-21-F-0003 through the National Defense Science and Engineering Graduate (NDSEG) Fellowship Program, sponsored by the Air Force Research Laboratory (AFRL), the Office of Naval Research (ONR) and the Army Research Office (ARO). R. Martin was partially supported by AFOSR Grants FA9550-20RQCOR098 (PO: Leve) and FA9550-20RQCOR100 (PO: Fahroo). This work was done in part while Y. Yang was visiting the Simons Institute for the Theory of Computing in Fall 2021. Y. Yang acknowledges support from Dr. Max Rössler, the Walter Haefner Foundation, and the ETH Zürich Foundation. This material is based upon work supported by the National Science Foundation under Award Number DMS-1913129.

We thank Dr. Chen Li for his helpful suggestions and generosity in sharing the code for the approach of Sec. IV B 3.

We would like to thank the referees for carefully reading our manuscript and giving many constructive comments that helped improve the paper.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Jonah Botvinick-Greenhouse: Formal analysis (lead); Software (lead); Writing – original draft (equal); Writing – review & editing (equal). **Robert Martin:** Conceptualization (equal); Writing – review & editing (equal). **Yunan Yang:** Conceptualization (equal); Formal analysis (supporting); Supervision (lead); Writing – original draft (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data used in Secs. V A, V B, and V D are available from the corresponding author upon reasonable request. The data used in Sec. V B were obtained from the EPTEMPEST experimental program funded by the AFSOR grant FA9550-17QCOR497 (Program Officer: Dr. Brett Pokines). The data used in Sec. V C are openly available via Ref. 73.

REFERENCES

[1] F. J. Montáns, F. Chinesta, R. Gómez-Bombarelli, and J. N. Kutz, "Data-driven modeling and learning in science and engineering," C. R. Mécanique **347**, 845–855 (2019).
[2] E. Baake, M. Baake, H. Bock, and K. Briggs, "Fitting ordinary differential equations to chaotic data," Phys. Rev. A **45**, 5524 (1992).
[3] C. Michalik, R. Hannemann, and W. Marquardt, "Incremental single shooting—A robust method for the estimation of parameters in dynamical systems," Comput. Chem. Eng. **33**, 1298–1305 (2009).
[4] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," Adv. Neural Inf. Process. Syst. **31**, 6571–6583 (2018).
[5] J. Jia and A. R. Benson, "Neural jump stochastic differential equations," Adv. Neural Inf. Process. Syst. **32**, 9815–9826 (2019).
[6] E. Negrini, G. Citti, and L. Capogna, "System identification through Lipschitz regularized deep neural networks," J. Comput. Phys. **444**, 110549 (2021).
[7] 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).
[8] U. Fasel, J. N. Kutz, B. W. Brunton, and S. L. Brunton, "Ensemble-SINDy: Robust sparse model discovery in the low-data, high-noise limit, with active learning and control," Proc. R. Soc. A **478**, 20210904 (2022).
[9] R. Van Der Merwe and E. A. Wan, "The square-root unscented Kalman filter for state and parameter-estimation," in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 01CH37221)* (IEEE, 2001), Vol. 6, pp. 3461–3464.
[10] G. Evensen, "The ensemble Kalman filter: Theoretical formulation and practical implementation," Ocean Dyn. **53**, 343–367 (2003).
[11] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches* (John Wiley & Sons, 2006).
[12] J. Harlim, A. Mahdi, and A. J. Majda, "An ensemble Kalman filter for statistical estimation of physics constrained nonlinear regression models," J. Comput. Phys. **257**, 782–812 (2014).
[13] F. Hamilton, T. Berry, and T. Sauer, "Ensemble Kalman filtering without a model," Phys. Rev. X **6**, 011021 (2016).
[14] C. Schillings and A. M. Stuart, "Analysis of the ensemble Kalman filter for inverse problems," SIAM J. Numer. Anal. **55**, 1264–1290 (2017).
[15] B. de Silva, K. Champion, M. Quade, J.-C. Loiseau, J. Kutz, and S. Brunton, "PySINDy: A Python package for the sparse identification of nonlinear dynamical systems from data," J. Open Source Softw. **5**, 2104 (2020).

[16]A. A. Kaptanoglu, B. M. de Silva, U. Fasel, K. Kaheman, A. J. Goldschmidt, J. Callaham, C. B. Delahunt, Z. G. Nicolaou, K. Champion, J.-C. Loiseau, J. N. Kutz, and S. L. Brunton, "PySINDy: A comprehensive Python package for robust sparse system identification," J. Open Source Softw. **7**, 3994 (2022).
[17]C. Greve, K. Hara, R. Martin, D. Eckhardt, and J. Koo, "A data-driven approach to model calibration for nonlinear dynamical systems," J. Appl. Phys. **125**, 244901 (2019).
[18]Y. Yang, L. Nurbekyan, E. Negrini, R. Martin, and M. Pasha, "Optimal transport for parameter identification of chaotic dynamics via invariant measures," SIAM J. Appl. Dyn. Syst. **22**, 269–310 (2023).
[19]T. R. Bewley and A. S. Sharma, "Efficient grid-based Bayesian estimation of nonlinear low-dimensional systems with sparse non-Gaussian PDFs," Automatica **48**, 1286–1290 (2012).
[20]L.-S. Young, "What are SRB measures, and which dynamical systems have them?," J. Stat. Phys. **108**, 733–754 (2002).
[21]S. Luzzatto, I. Melbourne, and F. Paccaut, "The Lorenz attractor is mixing," Commun. Math. Phys. **260**, 393–401 (2005).
[22]W. Tucker, "The Lorenz attractor exists," C. R. Math. **328**, 1197–1202 (1999).
[23]F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980* (Springer, 1981), pp. 366–381.
[24]Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," Neural Comput. **1**, 541–551 (1989).
[25]A. M. McDonald, M. A. van Wyk, and G. Chen, "The inverse Frobenius–Perron problem: A survey of solutions to the original problem formulation," AIMS Math. **6**, 11200–11232 (2021).
[26]W. Cowieson and L.-S. Young, "SRB measures as zero-noise limits," Ergod. Theory Dyn. Syst. **25**, 1115–1138 (2005).
[27]G. Froyland, "Estimating physical invariant measures and space averages of dynamical systems indicators", Ph.D. thesis (The University of Western Australia, 1996).
[28]A. Katok and B. Hasselblatt, "Introduction to the modern theory of dynamical systems," in *Encyclopedia of Mathematics and Its Applications* (Cambridge University Press, 1995).
[29]M. Einsiedler and T. Ward, *Ergodic Theory: With a View Towards Number Theory* (Springer, London, 2011).
[30]A. Allawala and J. B. Marston, "Statistics of the stochastically forced Lorenz attractor by the Fokker-Planck equation and cumulant expansions," Phys. Rev. E **94**, 052218 (2016).
[31]A. Lasota and M. C. Mackey, *Chaos, Fractals, and Noise: Stochastic Aspects of Dynamics* (Springer Science & Business Media, 1998), Vol. 97.
[32]M. Dellnitz, G. Froyland, and O. Junge, "The algorithms behind GAIO—Set oriented numerical methods for dynamical systems," in *Ergodic Theory, Analysis, and Efficient Simulation of Dynamical Systems*, edited by B. Fiedler (Springer, Berlin, 2001), pp. 145–174.
[33]S. Klus, P. Koltai, and C. Schütte, "On the numerical approximation of the Perron-Frobenius and Koopman operator," J. Comput. Dyn. **3**, 51–77 (2016).
[34]A. Blumenthal and L.-S. Young, "Equivalence of physical and SRB measures in random dynamical systems," Nonlinearity **32**, 1494–1524 (2019).
[35]J. Hong and X. Wang, "Invariant measures for stochastic differential equations," in *Invariant Measures for Stochastic Nonlinear Schrödinger Equations* (Springer, 2019), pp. 31–61.
[36]G. A. Pavliotis, *Stochastic Processes and Applications* (Springer, New York, 2014).
[37]W. Huang, M. Ji, Z. Liu, and Y. Yi, "Steady states of Fokker–Planck equations: I. Existence," J. Dyn. Differ. Equ. **27**, 721–742 (2015).
[38]X. Chen, L. Yang, J. Duan, and G. E. Karniadakis, "Solving inverse stochastic problems from discrete particle observations using the Fokker–Planck equation and physics-informed neural networks," SIAM J. Sci. Comput. **43**, B811–B830 (2021).
[39]S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz, "Chaos as an intermittently forced linear system," Nat. Commun. **8**, 19 (2017).
[40]A. Kirtland, J. Botvinick-Greenhouse, M. DeBrito, M. Osborne, C. Johnson, R. S. Martin, S. J. Araki, and D. Q. Eckhardt, "An unstructured mesh approach to nonlinear noise reduction for coupled systems," arXiv:2209.05944 (2022).
[41]G. Sugihara, R. May, H. Ye, C.-H. Hsieh, E. Deyle, M. Fogarty, and S. Munch, "Detecting causality in complex ecosystems," Science **338**, 496–500 (2012).
[42]T. Sauer, J. A. Yorke, and M. Casdagli, "Embedology," J. Stat. Phys. **65**, 579–616 (1991).
[43]D. Chelidze, "Reliable estimation of minimum embedding dimension through statistical analysis of nearest neighbors," J. Comput. Nonlinear Dyn. **12**, 051024 (2017).
[44]H. Ma and C. Han, "Selection of embedding dimension and delay time in phase space reconstruction," Front. Electr. Electron. Eng. China **1**, 111–114 (2006).
[45]A. Maus and J. C. Sprott, "Neural network method for determining embedding dimension of a time series," Commun. Nonlinear Sci. Numer. Simul. **16**, 3294–3302 (2010).
[46]S. Wallot and D. Mønster, "Calculation of average mutual information (AMI) and false-nearest neighbors (FNN) for the estimation of embedding parameters of multidimensional time series in Matlab," Front. Psychol. **9**, 1679 (2018).
[47]X. Chen, H. Wang, and J. Duan, "Detecting stochastic governing laws with observation on stationary distributions," Phys. D: Nonlinear Phenom. **448**, 133691 (2023).
[48]X. Nie, D. Coca, J. Luo, and M. Birkin, "Solving the inverse Frobenius-Perron problem using stationary densities of dynamical systems with input perturbations," Commun. Nonlinear Sci. Numer. Simul. **90**, 105302 (2020).
[49]D. Pingel, P. Schmelcher, and F. Diakonos, "Theory and examples of the inverse Frobenius–Perron problem for complete chaotic maps," Chaos **9**, 357–366 (1999).
[50]N. Wei, "Solutions of the inverse Frobenius–Perron problem," master's thesis (Concordia University, 2015), unpublished.
[51]S. Grossmann and S. Thomae, "Invariant distributions and stationary correlation functions of one-dimensional discrete processes," Z. Naturforsch. A **32**, 1353–1363 (1977).
[52]X. Nie and D. Coca, "A matrix-based approach to solving the inverse Frobenius–Perron problem using sequences of density functions of stochastically perturbed dynamical systems," Commun. Nonlinear Sci. Numer. Simul. **54**, 248–266 (2018).
[53]C. Fox, L.-J. Hsiao, and J.-E. Lee, "Solutions of the multivariate inverse Frobenius–Perron problem," Entropy **23**, 838 (2021).
[54]A. M. McDonald and M. A. van Wyk, "A novel approach to solving the generalized inverse Frobenius–Perron problem," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)* (IEEE, 2020), pp. 1–5.
[55]R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems* (Cambridge University Press, 2002), Vol. 31.
[56]J. Hu and X. Zhang, "Positivity-preserving and energy-dissipative finite difference schemes for the Fokker–Planck and Keller–Segel equations," IMA J. Numer. Anal. **43**, 1450–1484 (2022).
[57]D. F. Gleich, "PageRank beyond the web," SIAM Rev. **57**, 321–363 (2015).
[58]D. P. Kingma and J. Ba, "A method for stochastic optimization," arXiv:1412.6980 (2014).
[59]L. Nurbekyan, W. Lei, and Y. Yang, "Efficient natural gradient descent methods for large-scale PDE-based optimization problems," arXiv:2202.06236 (2022).
[60]C. Villani, *Topics in Optimal Transportation* (American Mathematical Society, 2021), Vol. 58.
[61]M. Jacobs and F. Léger, "A fast approach to optimal transport: The back-and-forth method," Numer. Math. **146**, 513–544 (2020).
[62]T. Séjourné, F.-X. Vialard, and G. Peyré, "Faster unbalanced optimal transport: Translation invariant Sinkhorn and 1-D Frank-Wolfe," in *International Conference on Artificial Intelligence and Statistics* (PMLR, 2022), pp. 4995–5021.
[63]F. Lu, M. Maggioni, and S. Tang, "Learning interaction kernels in stochastic systems of interacting particles from multiple trajectories," Found. Comput. Math. **22**, 1–55 (2021).
[64]K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Netw. **2**, 359–366 (1989).
[65]C. Li, M. Dunlop, and G. Stadler, "Bayesian neural network priors for edge-preserving inversion," Inverse Probl. Imaging **16**, 1229–1254 (2022).
[66]J. Guckenheimer, "Dynamics of the Van der Pol equation," IEEE Trans. Circuits Syst. **27**, 983–989 (1980).

[67]B. Engquist and Y. Yang, "Optimal transport based seismic inversion: Beyond cycle skipping," Commun. Pure Appl. Math. **75**, 2201–2244 (2020).
[68]M. M. Dunlop and Y. Yang, "Stability of Gibbs posteriors from the Wasserstein loss for Bayesian full waveform inversion," SIAM/ASA J. Uncertain. Quantif. **9**, 1499–1526 (2021).
[69]B. Engquist, K. Ren, and Y. Yang, "The quadratic Wasserstein metric for inverse data matching," Inverse Probl. **36**, 055001 (2020).
[70]D. Eckhardt, J. Koo, R. Martin, M. Holmes, and K. Hara, "Spatiotemporal data fusion and manifold reconstruction in Hall thrusters," Plasma Sources Sci. Technol. **28**, 045005 (2019).
[71]N. A. MacDonald, M. A. Cappelli, and W. A. Hargus, Jr., "Time-synchronized continuous wave laser-induced fluorescence on an oscillatory xenon discharge," Rev. Sci. Instrum. **83**, 113506 (2012).
[72]P. Ramachandran and G. Varoquaux, "Mayavi: 3D visualization of scientific data," Comput. Sci. Eng. **13**, 40–51 (2011).
[73]H. J. Diamond, T. R. Karl, M. A. Palecki, C. B. Baker, J. E. Bell, R. D. Leeper, D. R. Easterling, J. H. Lawrimore, T. P. Meyers, M. R. Helfert, G. Goodge, and P. W. Thorne, "U.S. climate reference network after one decade of operations: Status and assessment," Bull. Am. Meteorol. Soc. **94**, 485–498 (2013).