Deep learning delay coordinate dynamics for chaotic attractors from partial observable data

Charles D. Young and Michael D. Graham[*]
Department of Chemical and Biological Engineering, University of Wisconsin-Madison, Madison, Wisconsin 53706, USA

(Received 22 November 2022; accepted 6 March 2023; published 30 March 2023)

A common problem in time-series analysis is to predict dynamics with only scalar or partial observations of the underlying dynamical system. For data on a smooth compact manifold, Takens' theorem proves a time-delayed embedding of the partial state is diffeomorphic to the attractor, although for chaotic and highly nonlinear systems, learning these delay coordinate mappings is challenging. We utilize deep artificial neural networks (ANNs) to learn discrete time maps and continuous time flows of the partial state. Given training data for the full state, we also learn a reconstruction map. Thus, predictions of a time series can be made from the current state and several previous observations with embedding parameters determined from time-series analysis. The state space for time evolution is of comparable dimension to reduced order manifold models. These are advantages over recurrent neural network models, which require a high-dimensional internal state or additional memory terms and hyperparameters. We demonstrate the capacity of deep ANNs to predict chaotic behavior from a scalar observation on a manifold of dimension three via the Lorenz system. We also consider multivariate observations on the Kuramoto-Sivashinsky equation, where the observation dimension required for accurately reproducing dynamics increases with the manifold dimension via the spatial extent of the system.

DOI: 10.1103/PhysRevE.107.034215

I. INTRODUCTION

Many applications require the prediction of a time series with short-term tracking and long-term statistical accuracy from only observable data, such as modeling turbulent flows, [1] weather [2], rainfall [3], protein configurations [4], and the stock market [5]. The system is often governed by underlying differential equations on a smooth compact manifold of dimension $d_{\mathcal{M}}$. For an observation of the system with dimension d_o in ambient Euclidean space, $u(t) \in \mathbb{R}^{d_o}$, Whitney's theorem proves that there is a diffeomorphic mapping to the manifold coordinates $h(t) \in \mathbb{R}^{d_{\mathcal{M}}}$ when the observation dimension satisfies $d_o > 2d_{\mathcal{M}}$ [6,7]. In this case, time prediction of the state can be performed from only the current observation, as demonstrated by data-driven approaches such as sparse regression [8] and reduced order modeling [9,10]. In other approaches, the full state is encoded with the history of the system to improve short-time predictions [11–13]. An advantage of reduced order modeling is the ability to perform time evolution at low computational expense, which is essential in control applications [14]. In particular, autoencoders discover a latent space $h = \chi(u;\theta) \in \mathbb{R}^{d_{\mathcal{M}}}$, which approximates the minimum dimension manifold with trainable parameters θ [9,10].

For a partial observable $u_p(t) \in \mathbb{R}^{d_p}$ of dimension $d_p < 2d_{\mathcal{M}}$, the information contained in the current observation is insufficient to reconstruct the manifold. An alternative approach is to embed the state and its $m-1$ time delays $u_d(t) = [u_p(t), u_p(t-\tau), \ldots, u_p(t-(m-1)\tau)] \in \mathbb{R}^{m\times d_p}$. Takens theorem proves that for an embedding dimension $m > 2d_{\mathcal{M}}$ and nearly any choice of delay spacing τ there exists a diffeomorphic delay coordinate map to the manifold [7,15]. Takens theorem was originally formulated for scalar observations $d_p = 1$, but generalizations for vector observations have been developed [16]. While Takens' theorem proves the existence of delay coordinate maps, it does not offer any guidance in determining these functions. In this paper, we use neural networks (NNs) to learn these delay coordinate dynamics. We continue the idea of a minimal data-driven model for chaotic dynamics [9,10], where the delay coordinate embedding u_d serves as the reduced order model in the absence of full state data.

Before learning delay coordinate dynamics, embedding parameters must be chosen. Progress in time series analysis has provided techniques for generating optimal embeddings to reconstruct the manifold [17,18]. The embedding dimension is generally estimated by false nearest neighbor (FNN) methods [19,20] and the delay spacing by the mutual information (MI) [21] or correlation integral [22]. Notably, these methods have primarily been applied to scalar observations of low-dimensional chaotic attractors, and the choice of m and τ are made nearly independently. Modern approaches which can account for multivariate observations and which aim to improve reconstruction of the true attractor's topology have been developed [23–27] but there has been limited testing of these methods on chaotic attractors of dimension $d_{\mathcal{M}} > 3$. Time series analysis methods have also been built into data-driven methods for automatic generation of a latent space for time evolution, where the delay coordinate embedding is encoded to mask redundant time delays and promote orthogonality [28–31]. A related challenge is observability, or whether an observed variable is capable of accurately reconstructing the underlying attractor, if at all. For example,

[*]mdgraham@wisc.edu

the invariance of the Lorenz system to the transformation $(x, y, z) \to (-x, -y, z)$ precludes z as an observable, as shown by Lu *et al.* [32] Generally, such symmetries are not apparent directly from data or short-time predictions, as we show by attempting to learn the Lorenz dynamics from the z coordinate. A promising area for future work is to preprocess data to account for these symmetries, as previous studies have shown this improves the interpretation of data and the performance of data-driven approaches [33–35].

With a suitable delay coordinate embedding, the mappings for time prediction and reconstruction can be approximated. Advancements in machine learning have motivated many data-driven approaches for time prediction of partial observables including random feature maps and data assimilation [36,37], sparse regression [38], augmented latent space embeddings [30,39], closure modeling [40,41], and neural ordinary differential equations (NODE) [31]. Delay coordinate embeddings have also been used to model nonlinear dynamics by a linear system using Koopman theory [42,43]. Many of these approaches explicitly construct delay coordinate embeddings [31,36,38]. Others invoke Takens' theorem implicitly by use of recurrent neural networks (RNNs), which contain a memory term for embedding the state history [41]. Most methods test the ability to predict chaotic dynamics on the Lorenz-63 attractor, with short-time tracking for 5–10 Lyapunov times. More recently, some methods have been applied to the Lorenz-96 attractor [37,40] and the Kuramoto-Sivashinsky equation (KSE) [41]. These approaches use RNNs, with internal memory parameters in addition to the time delay embedding. Therefore, they do not generally represent a reduced order model of the state. The state history must be parameterized into the architecture's memory, requiring memory hyperparameters in the case of LSTMs [41] and reservoirs with a high internal dimension in the case of echo state networks [32].

Another forecasting approach originating from the field of statistical mechanics is the Mori-Zwanzig formalism, in which the observation is evolved by a generalized Langevin equation with the stochastic noise term accounting for the effect of unobserved fast degrees of freedom [44]. A number of data-driven methods have been proposed which follow this approach [45–49]. In the current work, we consider only deterministic dynamical systems, although Stark *et al.* extended Takens' theorem to forecasting stochastic observations given the reconstructed dynamical noise [50]. If only the observations are known and not the noise, Hirata proposed prediction coordinates, which use nearest neighbors on a combination of scalar observation forecasts [51]. Darmon developed an information-theoretic criterion for selecting embedding parameters for stochastic systems [52], where the above mentioned methods for deterministic systems [19–27] are not appropriate. When training data is available for the full stochastic state, Ferguson *et al.* have applied delay coordinate embeddings to reconstruct protein conformations from scalar observations [4,53–55].

In addition to forecasting, data-driven methods often seek to perform reconstruction of the true attractor as a supervised learning process [32,37,38,41]. In particular, reservoir computers and closure models have successfully reconstructed the KS attractor [32]. However, reservoir computers are and require a high-dimensional internal state compared to a diffeomorphic embedding in $2d_{\mathcal{M}}$ delay coordinates. Bakarji *et al.* [38] used a SINDy (sparse identification of nonlinear dynamics) autoencoder to identify a sparse representation of the attractor from a time delay embedding and then learn the dynamics. Because the dynamics and reconstruction are trained together, the model did not generate accurate short-time predictions for the Lorenz-63 system without training data for the full state space. Additionally, learning sparse representations of chaotic dynamics on a high-dimensional manifold is a challenge, and SINDy requires time derivatives to learn the dynamics, which may not be available from data spaced widely in time. In contrast, the method we propose here learns time evolution and reconstruction separately, so short-time predictions can be made from partial observations only. Our approach does not require time derivatives because backpropagation can be performed using automatic differentiation, such that neural ODEs can be trained with high accuracy even for widely spaced data [10].

We propose a method using deep NNs to learn delay coordinate maps from partial observable data. We perform supervised learning of a discrete time map, a continuous time flow (ordinary differential equation representation), and a reconstruction map for multivariate partial observations of chaotic dynamics. Our approach finds a low state dimension, $d_p m \approx 2d_{\mathcal{M}}$, with minimal history dependence compared to recurrent approaches which require memory integral parameters which are difficult to compute [45–49] or large internal reservoir states [13,32]. Additionally, we demonstrate the scaling of our method to higher dimensional chaotic systems via the KSE. We consider domain sizes $L = 22$ and $L = 44$ with periodic boundary conditions, which lie on inertial manifolds of dimension $d_{\mathcal{M}} = 8$ and $d_{\mathcal{M}} = 18$, respectively [9,10,56,57]. The only required inputs are the embedding dimension m and the delay spacing τ, both of which can, in principle, be determined before training the model. In practice, we find some empirical testing to be required for selecting embedding parameters, although our results are insensitive to these choices. Within an appropriate range of m and τ, the autocorrelation function of the state observation and probability distribution function of state variables are quantitatively consistent. Short-time tracking exhibits some sensitivity to choice of embedding parameters because these metrics correspond closely to the training loss, which is easier to minimize for an optimal embedding.

II. METHODOLOGY

We consider a full state space observation $u(t) \in \mathbb{R}^{d_o}$ from numerical simulations or, in principle, experimental data. We project to a lower dimension $d_p < d_o$ to generate a partial observation $u_p = \mathcal{P}u \in \mathbb{R}^{d_p}$. Here the projection operator simply filters out entries from the observation vector, but more general projections are compatible with our formulation. We construct a multivariate delay coordinate embedding of the partial observable and seek to learn the discrete time map and reconstruction map proposed by Takens theorem using deep NNs. The data is available at a sampling interval Δt. We estimate the embedding dimension m using FNNs [19] and the delay spacing $\tau = n\Delta t$ using the first minimum of

TABLE I. NN architectures.

System	Function	Shape	Activation
Lorenz	G	m : 200 : 200 : 1	ReLU:ReLU:linear
	F	m : 200 : 200 : 3	ReLU:ReLU:linear
	g	m : 200 : 200 : 200 : m	ReLU:ReLU:ReLUlinear
KS $L = 22$	G	md_p : 256 : 256 : 256 : 256 : d_p	ReLU:ReLU:ReLU:ReLU:linear
	F	md_p : 256 : 256 : 256 : 256 : 64	ReLU:ReLU:ReLU:ReLU:linear
	g	md_p : 256 : 256 : 256 : 256 : md_p	ReLU:ReLU:ReLUlinear
KS $L = 44$	G	md_p : 512 : 512 : 512 : 512 : d_p	ReLU:ReLU:ReLU:ReLU:linear
	F	md_p : 512 : 512 : 512 : 512 : 64	ReLU:ReLU:ReLU:ReLU:linear
	g	md_p : 512 : 512 : 512 : 512 : md_p	ReLU:ReLU:ReLUlinear

the MI, where n is the number of samples between delay coordinates [21]. The embedding is then defined as $u_d(t) = [u_p(t), u_p(t-\tau), \ldots, u_p(t-(m-1)\tau)] \in \mathbb{R}^{m\times d_p}$.

A. Discrete time evolution

To advance the partial observable in time, we first consider a discrete time step (DTS) map,

$$\hat{u}_p(t+\tau) = G(u_d(t);\theta_G), \quad G:\mathbb{R}^{m\times d_p}\to\mathbb{R}^{d_p},$$
$$\hat{u}_d(t+\tau) = [\hat{u}_p(t+\tau), u_p(t), \ldots, u_p(t-(m-2)\tau)], \quad (1)$$

where θ_G are NN parameters. The time step is equal to the delay time such that the delay coordinate vector can be iteratively forecasted. We approximate this function by a dense feed-forward NN. Architecture details are given in Table I. NN weights are trained using stochastic gradient descent as implemented in Keras [58] to minimize the loss,

$$\mathcal{L}_G = \langle ||u_p(t+\tau) - \hat{u}_p(t+\tau)||_2^2\rangle, \quad (2)$$

where $\hat{u}_p$ is the NN output.

To generate long NN predicted trajectories, an initial delay coordinate embedding $u_d(0)$ is first integrated forward as $\hat{u}_p(\tau) = G(u_d(0);\theta_G)$. The new partial observation is used to update the delay coordinate embedding to $\hat{u}_d(\tau) = (\hat{u}_p(\tau), u_p(0), \ldots, u_p(t-(m-2)\tau)$. The next time step is then calculated as $\hat{u}_p(2\tau) = G(\hat{u}_d(\tau);\theta_G)$ and so on iteratively such that after m steps the delay coordinate embedding contains only predicted values. An advantage of this approach is that only the current state and its $m-1$ delays are required to make predictions, in comparison to reservoir networks, which require a warmup period before predictions can be made [13,32]. However, intermediate timescales between the delay spacing are not accessible without interpolation.

B. Continuous time evolution

We also consider the continuous time evolution of the delay coordinate embedding,

$$\frac{du_d}{dt} = g(u_d;\theta_g) - au_d, \quad (3)$$

where $g(u_d;\theta_g)$ is a NN with parameters θ_g, trained as described below, and the term $-au_d$ has a stabilizing effect, keeping solutions from blowing up for appropriately chosen a [10,59]. No generality is lost when including this term; the combination $g - au_d$ is learned from the data. Unless otherwise noted, here $a = 10^{-3}$. Other than the damping term, our approach is similar to that of Wang and Guet [31]. We approximate the delay coordinate dynamics g by a NN, or NODE, which we use to integrate the state in time,

$$\check{u}_d(t+N\Delta t) = u_d(t) + \int_t^{t+N\Delta t}(g(u_d(t);\theta_g) - au_d(t))dt, \quad (4)$$

and the NN for g is trained with the multistep loss over N steps of size Δt:

$$\mathcal{L}_g = \left\langle \sum_{i=1}^{N} ||u_d(t+i\Delta t) - \check{u}_d(t+i\Delta t)||_2^2 \right\rangle. \quad (5)$$

The gradient of the loss is calculated by backpropagating through the solver with automatic differentiation [60]. The ODE solver uses the fifth-order Dormand-Prince-Shapmine method implemented in TORCHDIFFEQ [60]. In contrast to the discrete time case, where the model must be trained on a fixed time step and cannot resolve intermediate timescales, the neural ODE of the continuous time model can be trained and deployed for prediction on any time interval. Thus, the DTS model is trained on the timescale of interest, here the delay spacing τ. The NODE is trained using the data sampling interval Δt because previous work has shown smaller time steps improve training [10]. However, the same study showed discrete and continuous time predictions were consistent for a data spacing $\Delta t < 0.5\tau_L$, which is the case for all models trained in this work, so it is unlikely data spacing will significantly impact model performance. The partial observable can then be evaluated at times between delay spacings using the same solver used to determine g. For comparison to the DTS models, we report a NODE loss calculated after training using only the leading delay coordinate coordinate of the embedding as in the DTS loss:

$$L_g' = \langle ||u_p(t+\tau) - \check{u}_p(t+\tau)||_2^2\rangle. \quad (6)$$

C. Reconstruction

We also reconstruct the full observation from our numerical simulation data by delay coordinate mappings. We take a supervised learning approach on the assumption that training data is available. The reconstruction map from delay coordinates is defined as

$$\tilde{u}(t) = F(u_d(t);\theta_F), \quad F:\mathbb{R}^{m\times d_p}\to\mathbb{R}^{d_o}. \quad (7)$$

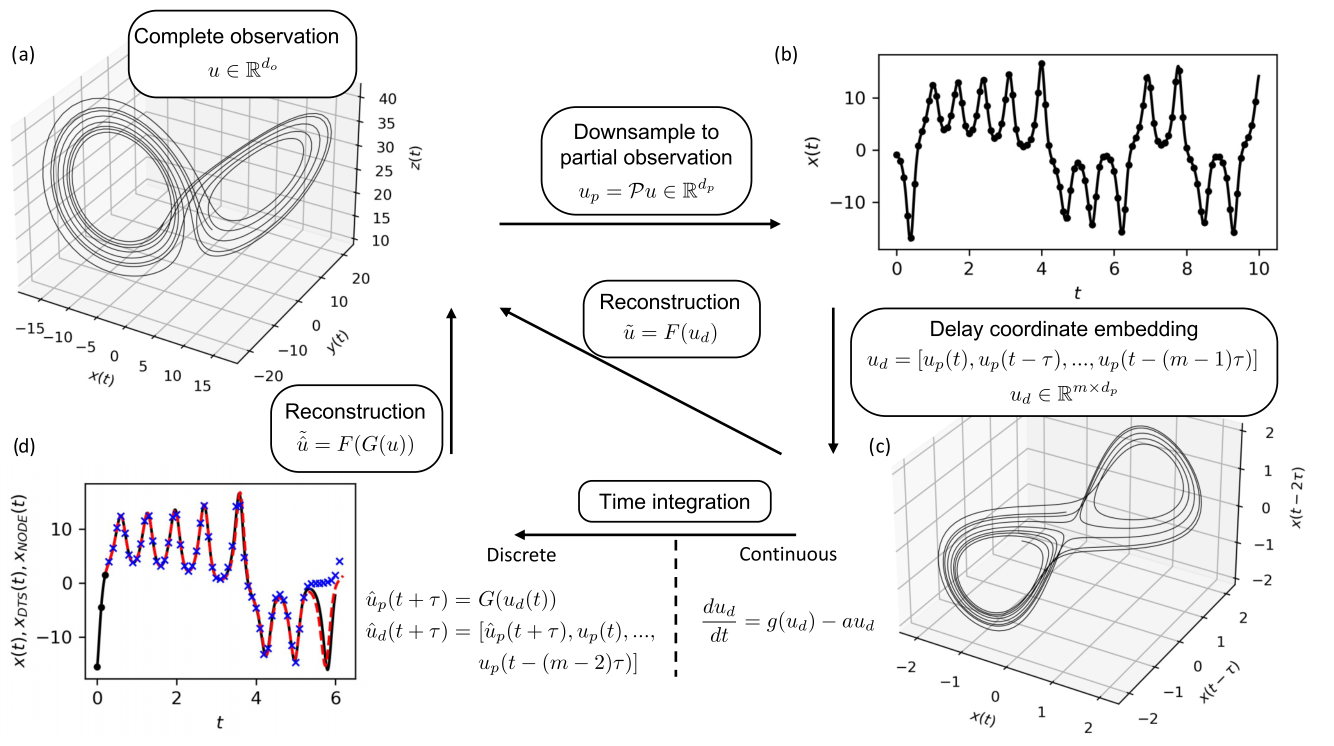
Complete observation
$u \in \mathbb{R}^{d_o}$
Downsample to partial observation
$u_p = \mathcal{P}u \in \mathbb{R}^{d_p}$
Reconstruction
$\tilde{u} = F(u_d)$
Delay coordinate embedding
$u_d = [u_p(t), u_p(t-\tau), ..., u_p(t-(m-1)\tau)]$
$u_d \in \mathbb{R}^{m\times d_p}$
Reconstruction
$\tilde{\tilde{u}} = F(G(u))$
Time integration
Discrete
Continuous
$\hat{u}_p(t+\tau) = G(u_d(t))$
$\hat{u}_d(t+\tau) = [\hat{u}_p(t+\tau), u_p(t), ..., u_p(t-(m-2)\tau)]$
$\frac{du_d}{dt} = g(u_d) - au_d$
(a)
(b)
(c)
(d)

FIG. 1. Schematic of learning delay coordinate dynamics and reconstruction. (a) Start with a complete observation of the attractor, here considering the Lorenz system. (b) Downsample to a partial observation, here taking $u_p = x$. Black points refer to the data at the delay coordinate spacing τ. (c) Construct a delay coordinate embedding. Here we select $m = 3$ delays and a delay spacing $\tau = 0.1$. (d) Learn the discrete and continuous time dynamics of the embedding. Black points refer to the delay coordinate embedding initial condition, the black solid line to the data, blue crosses to the discrete time prediction, and the red dashed line to the NODE prediction. The true or predicted delay coordinate embedding can also be reconstructed to the full state given training data.

The mapping is again approximated by dense feed-forward NNs with weights θ_F as detailed in Table I. The reconstruction loss is

$$\mathcal{L}_F = \langle ||u(t) - \tilde{u}(t)||_2^2 \rangle, \tag{8}$$

where $\tilde{u}$ is the NN output. The reconstruction training is performed independently of the time integration training, separating the error associated with the two functions. A visual example of the reconstruction process is shown in Fig. 1 for the diffeomorphism between the Lorenz attractor embedding of the partial state $u_p = x$, $m = 3$, $\tau = 0.1$ and the true attractor.

While the reconstruction training is performed only on true data, we apply the function to partial states predicted from the discrete time map. For investigating long-time dynamics we will refer to reconstructions of NN predicted partial states as $\tilde{\tilde{u}}(t) = F(\hat{u}_d(t); \theta_F)$, where predictions $\hat{u}_d(t)$ at long times t are generated as described above.

III. RESULTS

We apply our method to two common chaotic attractors to demonstrate the short term tracking and reproduction of long-time statistics in the delay coordinate embedding space, as well as reconstruction of the long-time statistics to the true attractor.

A. Lorenz system

We consider the Lorenz attractor [61]

$$\begin{aligned}\frac{dx}{dt} &= \sigma(y - x),\\ \frac{dy}{dt} &= x(\rho - z) - y,\\ \frac{dz}{dt} &= xy - \beta z,\end{aligned} \tag{9}$$

where $\sigma = 10$, $\beta = 8/3$, and $\rho = 28$. The Lyapunov time using these parameters is $\tau_L \approx 1$, and the fractal dimension estimated by the correlation integral is is $d_A \approx 2.06$ [62]. The training data is generated using a Runge-Kutta 4-5 integrator in SciPy with a sampling time $\Delta t = 0.1$. The first 10^4 data points are discarded as transients, and the next 5×10^5 data points are used for training with an 80/20 training/test split. The discrete time and reconstruction maps are trained in Keras [58] using an Adam optimizer for 1000 epochs with an initial learning rate of 0.001, which is decreased by a factor of 0.5 every 100 epochs. The NODE models are trained using TORCHDIFFEQ [60] with a batch size of 100 for 50 000 epochs. The initial learning rate is 0.001, and it is decreased by a factor of 0.5 every 10 000 epochs. The number of time steps forecasted during training is $N = 2$.

We select $\tau = 0.1$, which is the first minimum of the MI [21]. The embedding dimension determined by FNN is $m = 3$ [19]. Time-series analysis calculations are performed

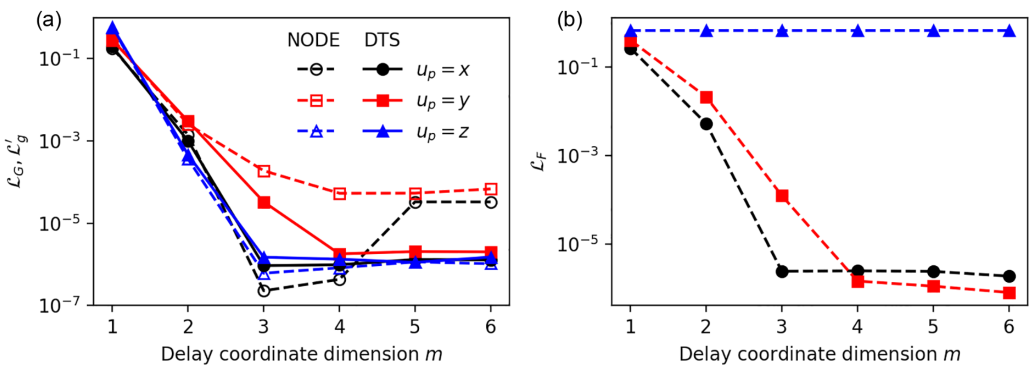

FIG. 2. Lorenz attractor test data loss for varying partial observable u_p and embedding dimension m. (a) Time integration NNs, DTS (solid lines and filled symbols), NODE (dashed lines, open symbols). (b) Reconstruction NNs. Delay spacing $\tau = 0.1$ for all models.

using the DelayEmbeddings module of the Julia package DynamicalSystems.jl. [27,63]. The embedding parameters for different observables $u_p = x, y, z$ were similar. To confirm the choice of m suggested by FNN, we fix the delay spacing and train five NNs at a varying embedding dimension $m = 1 - 6$, as shown in Fig. 2. NN maps for embeddings with delay spacings $\tau = 0.05 - 0.2$ did not qualitatively differ from the presented results. Variance of the MSE is low for both time and reconstruction NNs, although some time-stepping models fall onto periodic orbits or fixed points at long times. Therefore, in quantifying the attractor reconstruction (Figs. 3–5), we select the model which best reproduces the attractor joint probability density function (PDF) $P(\tilde{\hat{x}}, \tilde{\hat{y}})$ after time integration and reconstruction (Fig. 5). To quantify uncertainty among models which remain on the true attractor, we report in Fig. 6 the mean and standard errors for reconstruction of a long NN time-evolved trajectory. In all cases, the standard error is at least an order of magnitude smaller than the mean.

The test data mean squared error (MSE) of the DTS map plateaus at an embedding dimension $m = 3$ for an observation $u_p = x$, consistent with FNN and other data-driven approaches [30]. However, we find $m = 4$ is required for $u_p = y$. The need for an additional delay is confirmed by statistical reproduction of the attractor (Figs. 3–5). Observing the z component of the Lorenz appears to provide excellent time prediction, which is unexpected due to the invariance of the Lorenz attractor to the transformation $(x, y, z) \rightarrow (-x, -y, z)$. The low one-step error is misleading, as long-time trajectories generated by the NN fall onto periodic orbits or fixed points. Thus, we consider only the x and y observables in further detail. The MSE of the reconstruction maps are similar to the DTS models,

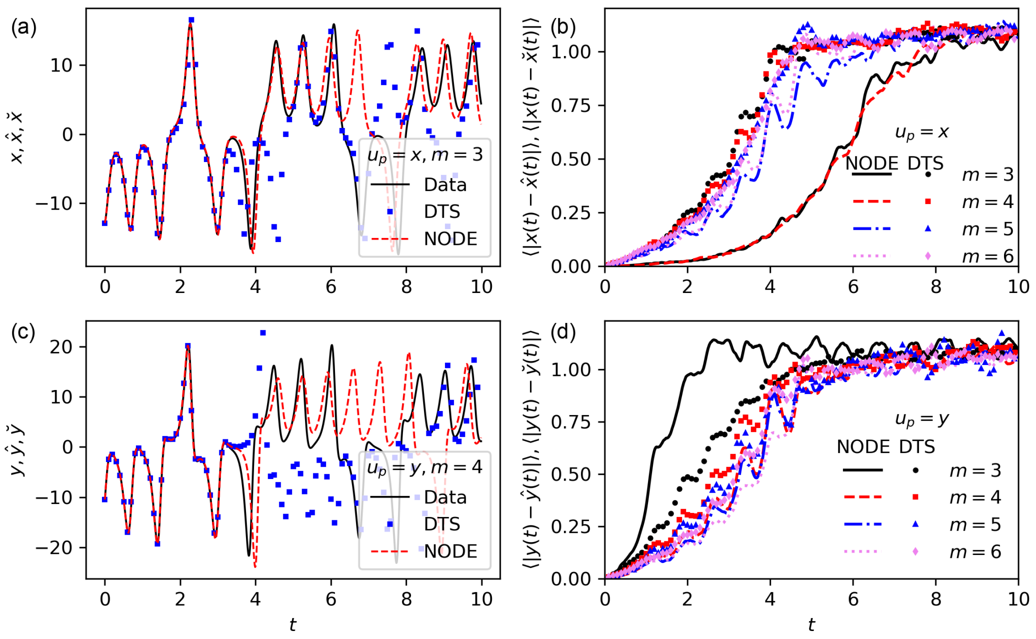

FIG. 3. Short term tracking of the Lorenz attractor with different observables: (a), (b) $u_p = x$; (c), (d) $u_p = y$. Left column: Representative trajectories generated from the same initial condition u_0. Symbols correspond to predictions from a discrete time step model and dashed lines to a neural ODE model. Both DTS and NODE models use $m = 3$, $\tau = 0.1$ for $u_p = x$ and $m = 4$, $\tau = 0.1$ for $u_p = y$. Right column: Ensemble average error of DTS models (symbols) and NODE models (lines) for increasing embedding dimension. Delay spacing $\tau = 0.1$ for all embeddings.

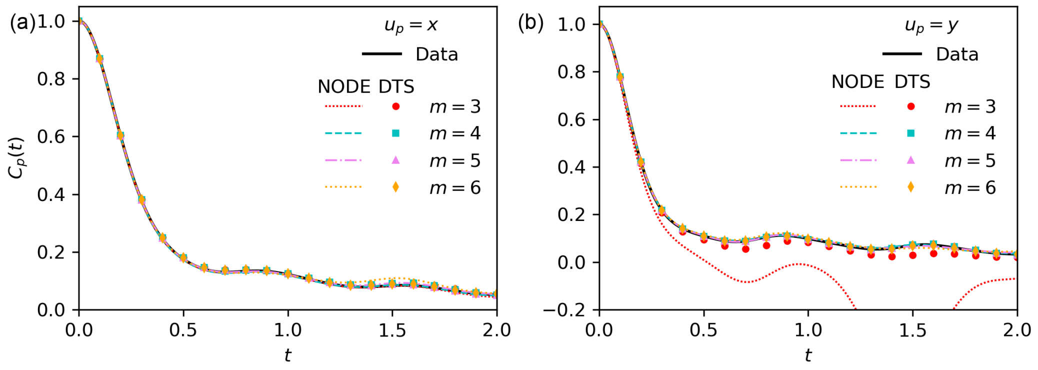

FIG. 4. Autocorrelation function of the Lorenz attractor partial observable for (a) $u_p = x$, (b) $u_p = y$. Solid lines correspond to true data, dashed lines to predictions of NODE models, and symbols to predictions of DTS model predictions. Delay spacing $\tau = 0.1$. The data and NODE predictions use a sampling interval $\Delta t = 0.01$ and DTS use $\Delta t = \tau$.

with $u_p = x$ plateauing at $m = 3$ and $u_p = y$ at $m = 4$. The embedding of $u_p = z$ fails to reconstruct the true attractor, as expected and in agreement with previous results using reservoir computing [32].

Comparing DTS and NODE models, we again find an observation $u_p = y$ plateaus at $m = 4$. The quantitative value of $\mathcal{L}'_g$ is higher than the DTS model because the NODE is trained to minimize L_g. For an observation $u_p = x$, the MSE again decreases up to $m = 3$, although the error increases for $m > 4$. This could be due to errors in predicting delay coordinate distance from the current time or simply due to the introduction of irrelevant information to the embedding. This is consistent with the results of Wang and Guet [31], who found that applying FNN in an autoencoder reduced an input $m = 6$ embedding of the Lorenz system with observation $u_p = x$ to the leading $m = 3$ delays. Further, they found NODE predictions using an autoencoder performed better in short-time tracking than training on the delay coordinate embedding with $m = 6$. However, we find that for the KSE attractor with multidimensional observations, FNN does not reliably predict the embedding dimension. Thus, performing this step automatically in an autoencoder without knowledge of the manifold dimension remains challenging.

While the test MSE for the time stepping and reconstruction NNs are low, they capture only the one-step pointwise error. To further quantify the ability of NNs to reconstruct the attractor, we consider ensemble average statistics from long NN trajectories. As expected, both DTS and NODE time integration models with an embedding dimension $m = 1$ and $m = 2$ go to fixed points and periodic orbits, respectively. Therefore, we focus on results for $m \geqslant 3$.

First, we show an example of the short-term forecasting capabilities. We generate 2000 trajectories of the partial observable for ten time units from different initial conditions using DTS and NODE models. Two representative trajectories are shown in Figs. 3(a) ($u_p = x, m = 3$) and 3(c) ($u_p = y, m = 4$), where the NNs track for several time units where $\tau_L \approx 1$, comparable to other methods [30,31,36]. We also show the ensemble average tracking error for these two embeddings in Figs. 3(b) and 3(d). Consistent with the test MSE, the tracking error converges for an embedding dimension $m \geqslant 3$ and observation $u_p = x$. With an observation $u_p = y$, tracking improves slightly with increasing embedding dimension up to $m = 6$, although the MSE plateau value $m = 4$ already provides good predictive capability and preserves the dynamics in long trajectories (Figs. 4–6).

DTS models for both observables exhibit similar tracking errors, but the NODE models for $u_p = x$ are significantly more accurate than for $u_p = y$. We speculate this is due to the sharp gradients in y that occur when the solution jumps from

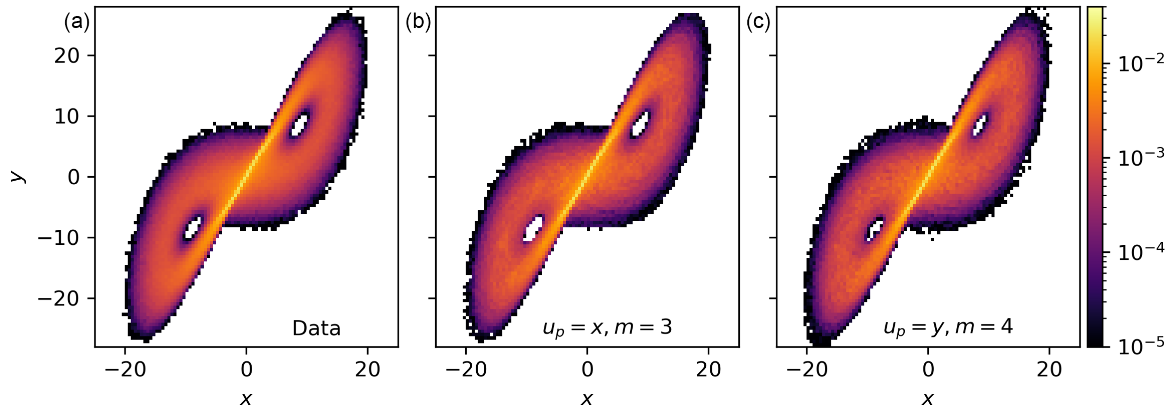

FIG. 5. Joint PDF of the Lorenz attractor from (a) data $P(x, y)$ and from a discrete time integrated and reconstructed NN trajectory from a delay coordinate initial condition, $P(\tilde{x}, \tilde{y})$; (b) $u_p = x, m = 3$; (c) $u_p = y, m = 4$.

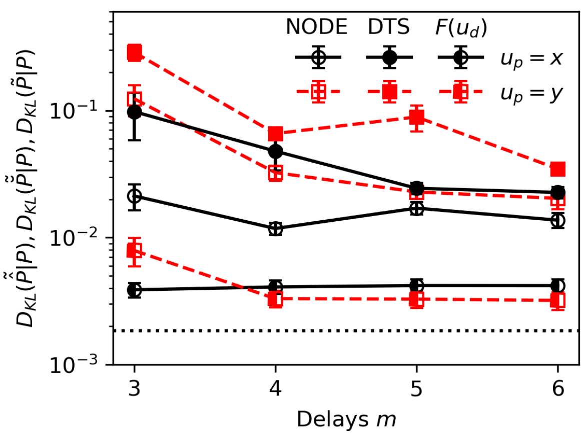

FIG. 6. KL divergence of the true and predicted Lorenz joint PDF $P(x, y)$ for increasing number of delays. The horizontal line indicates the divergence between two true data sets with different initial conditions. Open symbols and closed symbols refer to NODE and DTS time integration models respectively. Half-filled symbols refer to reconstruction of a true partial observable trajectory, $F(u_d; \theta_F)$, which excludes error from time integration. Symbols and error bars are the mean and standard error respectively of trajectories from five different models.

one wing of the attractor to the other, as seen at $t \approx 4.1$ in Fig. 3(c). Additionally, NODE models observing $u_p = x$ perform worse with $m > 4$, again due to irrelevant information in the embedding.

We compare the forecasting capabilities of DTS and NODE NN models to a model proposed by Gottwald and Reich, random feature maps and data assimilation (RAFDA) [36]. For consistency, we trained models using 4000 data points, which is the same amount used in Ref. [36]. To prevent overtraining, we used L2 regularization, such that the DTS loss is $\mathcal{L}_G^* = \mathcal{L}_G + \alpha||\theta_G||_2^2$ and the NODE loss is $\mathcal{L}_g^* = \mathcal{L}_g + \alpha||\theta_g||_2^2$, with $\alpha = 10^{-4}$. Only the results presented in Table II use this smaller data-set model. Following the definition of the relative forecast error in Ref. [36], $\mathcal{E}(t) = |u_d(t) - \breve{u}_d(t)|^2/|u_d(t)|_2^2 \leqslant \mathcal{E}_{\text{tol}}$, the reference time t_f occurs when the forecast error for a given trajectory exceeds a tolerance value $\mathcal{E}_{\text{tol}} = 40$. We compute the mean forecast time from the trajectories used to generate Fig. 3 and compare to the results reported for RAFDA [36] in Table II. We find DTS and NODE models have comparable forecasting times, and both outperform RAFDA by nearly a factor of 2. A notable contribution of RAFDA is the use of data assimilation to account for measurement errors in the observation. Wang and Guet [31] showed that a fitting NN before constructing the delay embedding had a similar effect for time-delayed NODEs.

TABLE II. Mean forecasting time for an observation $u_p = x$ and $m = 3$. RAFDA results [36] for no observational noise and delay spacing $\tau = 0.2$. DTS and NODE use a delay spacing $\tau = 0.1$.

	RAFDA [36]	DTS	NODE
$\langle\tau_f\rangle$	2.0	3.81	3.88

Next we quantify the dynamics of a long NN trajectory via the autocorrelation function of the partial observable state:

$$C_p(t) = \frac{\langle u_p(0) \cdot u_p(t)\rangle}{\langle u_p^2\rangle}. \tag{10}$$

The results shown in Fig. 4 are determined from NN trajectories run for 5×10^4 time units. For an observation $u_p = x$, the DTS model again reproduces the true data at $m = 3$ and achieves similar results with additional delays. For $u_p = y$, the NN model reproduces the data for the MSE plateau dimension $m = 4$. The slight improvement up to $m = 6$ found short-term tracking is not visible in this case. NODE and DTS model predictions agree quantitatively with the data at DTS intervals τ, but the NODE models also reproduce the true $C_p(t)$ at arbitrary timescales (sampling $\Delta t = 0.01$ shown here).

We conclude our study of the Lorenz system with the reconstruction of the 3D attractor from the long partial observable trajectories, $\tilde{u}$(t), generated by a DTS or NODE model as described in Sec. II C. We visualize the reconstruction via the joint PDF $P(x, y)$ in Fig. 5. The reconstruction results are consistent with other metrics. An observation $u_p = x$ reproduces the joint PDF at $m = 3$ and $u_p = y$ at $m = 4$. We quantify the reconstruction for increasing embedding dimension via the KL divergence in Fig. 6:

$$D_{\text{KL}}(\tilde{P}|P) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P(\tilde{x}, \tilde{y})\ln\frac{P(\tilde{x}, \tilde{y})}{P(x, y)}. \tag{11}$$

We assume the contribution to the integral from empty bins is zero as in Ref. [64]. For reference, we include the KL divergence between two true data sets with different initial conditions (dashed horizontal line). We also show the KL divergence $D_{\text{KL}}(\tilde{P}|P)$ for the case of reconstruction of a true partial observable delay coordinate embedding $\tilde{u} = F(u_d; \theta_F)$, which represents a baseline of expected performance for trajectories generated by NN time integration models. All PDFs are generated with the same trajectory length. Using the decoder only, the KL divergence plateaus at the same value as the reconstruction MSE, and the quantitative value is comparable to the divergence of two true data sets. Joint PDFs from the NN integration models have a slight dependence on the number of delays and perform quantitatively worse than only the decoder, as expected. We note the NODEs perform better than the DTS models for $m = 3 - 6$.

Thus, we have demonstrated that we can learn NN approximations to delay coordinate time integration and reconstruction maps from partial observable data for a low-dimensional ($d_\mathcal{M} = 3$) chaotic attractor.

B. Kuramoto-Sivashinky equation

Next we test our method on higher dimensional chaotic attractors with multivariate observations. In particular, we consider the KSE

$$\partial_t u = -u\partial_x u - \partial_{xx} - \partial_{xxxx} u \tag{12}$$

with periodic boundary conditions in the domain $x \in [0, L]$. We consider $L = 22, 44$ because the manifold dimension for $L = 22$ is known to be $d_\mathcal{M} = 8$ [57] and the dynamics become increasingly chaotic with L. The manifold dimension for $L = 44$ can be approximated by autoencoders [9,10] and

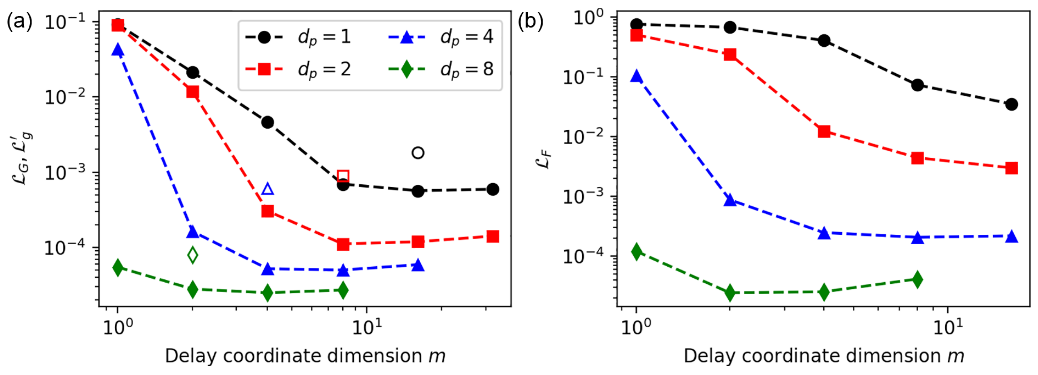

FIG. 7. KSE test loss of (a) time integration NNs, DTS (filled symbols), and NODE (open symbols). (b) Reconstruction NNs for increasing observation dimension d_p and delay coordinate dimension m. Delay spacing $\tau = 1.5$ for all results. NODE models are only trained for embedding parameters $d_pm = 16 \approx 2d_{\mathcal{M}}$.

the number of physical modes [56], which find $d_{\mathcal{M}} = 18$. Trajectories were generated using the code from Cvitanović *et al.* [65], implementing a Fourier spectral method in space and a fourth-order time integration scheme [66] on a $d_o = 64$ point grid. We generate a trajectory with 4×10^5 time steps with $\Delta t = 0.25$ with an 80/20 training test/split. DTS and reconstruction NNs are trained by the same procedure as the Lorenz models. NODE models are trained by the same procedure as the Lorenz models, but the number of epochs is increased to 200 000, the learning rate drops every 25 000 epochs, and the number of time steps forecasted during training is $N = 20$. The network depth and width is also increased (Table I) We trained five time integration and reconstruction models for each choice of embedding parameters. With a sufficient number of delays to reach the embedding dimension, the variance in KL divergence with the true data between models was smaller than the point size in Fig. 12.

First focusing on $L = 22$, we choose evenly spaced grid points as observations in the same manner as Lu *et al.* [32] with an observation dimension $d_p = 1, 2, 4, 8$. A diffeomorphism to the state without time delays is expected at $d_p = 16$ due to Whitney's theorem [6,7], so we do not consider $d_p > 8$. Additionally, we will find NNs can forecast and reconstruct the attractor at $d_p = 8$ even without time delays. We need to generate a delay coordinate embedding for each d_p. However, generating good embeddings for highly chaotic attractors and multivariate observations is challenging. Several methods have been proposed for multivariate observations, [23,24,27] but in our tests using the code available in DynamicalSystems.jl [27,63], they failed to generate an embedding for the KSE with $d_p = 1$.

Therefore, we estimate a delay spacing via the MI and embedding dimension via FNN for one grid point $d_p = 1$, which yields $\tau = 1.5$ and $m = 4$. We use these values as initial guesses and vary both parametrically in training NNs. Currently, we consider only uniform embeddings. We find $\tau = 1.0 - 4.0$ to provide the best performance in reconstruction and time stepping. For each d_p, we increase the embedding dimension from $m = 1$ to $d_pm = 2d_{\mathcal{M}} = 16$, at which dimension we expect to have a diffeomorphism to the state [16].

Figure 7 shows the test data set loss for the DTS, NODE, and reconstruction models for an increasing number of grid points in the observation and number of delays. Here we use the same $\tau = 1.5$ for quantitatively comparing the loss at different d_p because the delay spacing implicitly affects the time step loss through the step size. For later results, we will use $\tau = 4.0$ for $d_p = 4, 8$ because we find the longer embedding window improves reproduction of attractor statistics. The discrete time NN improves with increasing observation dimension, as expected. Increasing the number of delays m at a fixed d_p, we observe a dramatic improvement up to $d_pm = d_{\mathcal{M}} = 8$ for all observations d_p. Providing additional delays, the loss decreases slightly up to $d_pm = 2d_{\mathcal{M}} = 16$ and then plateaus or slightly increases.

Trends for the reconstruction map are largely similar, although the decrease in the loss with d_p is more pronounced because the error is now calculated on the full $d_o = 64$-dimensional reconstructed state, rather than the d_pm-dimensional partial state. Our results are qualitatively consistent with Lu *et al.*, [32] who observed a significant improvement in reconstruction from $d_p = 1 - 4$, and a smaller improvement from $d_p = 8$. Quantitatively, the time-delayed NNs used in this work perform better than reservoir computers (Fig. 8(b) of Ref. [32]), which implicitly embed the state history.

Based on the plateau of the DTS and reconstruction model loss at $d_pm = 16$, we train NODE models only for $d_pm = 16$.

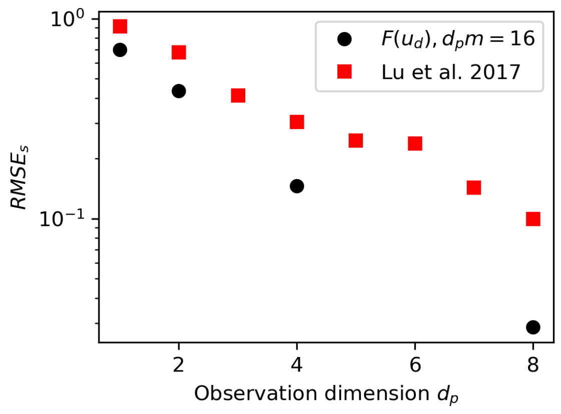

FIG. 8. Inference RMSE of the unobserved KSE state variables for increasing observation dimension d_p. Reservoir computing results reported by Lu *et al.* [32].

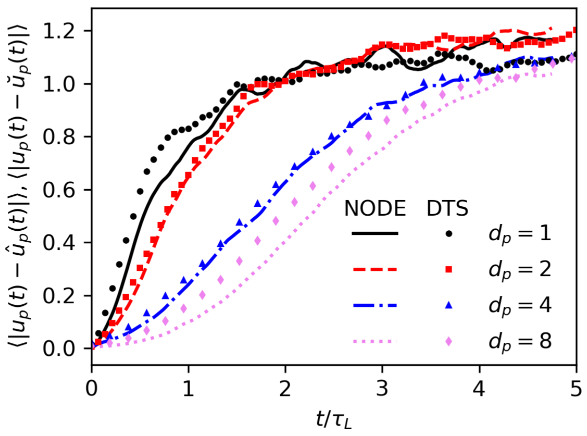

FIG. 9. Short-term ensemble average tracking of the KSE partial observable for NODE models (lines) and DTS models (points) with increasing observation dimension d_p. Delay spacing $\tau = 1.5$ for $d_p = 1, 2$ and $\tau = 4.0$ for $d_p = 4, 8$. The product of the number of time delays $m = 16, 8, 4, 2$ and $d_p = 1, 2, 4, 8$ is constant: $d_p m = 16 \approx 2d_{\mathcal{M}}$.

As noted in Sec. II B, we calculate the DTS loss term using the NODEs $\mathcal{L}_g'$ for comparison of the two models. The NODE error also decreases as the dimension of the observation d_p increases. The quantitative value of $\mathcal{L}_g'$ is larger than $\mathcal{L}_G$, again because NODEs are trained to minimize a different loss. In comparing tracking and attractor reconstruction, below we find the NODEs perform well.

We compare the reconstruction error of our models with dimension $d_p m = 16$ to those of Lu *et al.* [32] for varying observation dimension d_p, who reported the RMSE for inference unobserved states s_i from the delay coordinate embedding using reservoir computing:

$$\mathrm{RMSE}_s = \frac{\sum_{i,t}[s_i(t) - \tilde{s}_i(t)]^2}{\sum_{i,t}[s_i(t)]^2}. \tag{13}$$

We train models with the same amount of data as Lu *et al.*, who used 60 000 data points with a separation of $\Delta t = 0.25$. The embedding parameters for increasing d_p are the same as other results presented in Figs. 9–12. Only the results shown in Fig. 8 use the smaller training data set. L2 regularization was used to prevent overtraining with a reconstruction loss $\mathcal{L}_F^* = \mathcal{L}_F + \alpha||\theta_F||_2^2$ and $\alpha = 10^{-4}$. We find that our method outperforms the reservoir computers at all d_p. Notably, our method has a low state dimension and NN width (256 neurons) compared to reservoir computers used by Lu *et al.* , which have an internal reservoir dimension of 3000.

We quantify the ensemble average tracking error of the partial state for increasing observation dimension d_p in Fig. 9. Here we show only one embedding model for each d_p corresponding to $d_p m = 16$. As noted above, we find that larger

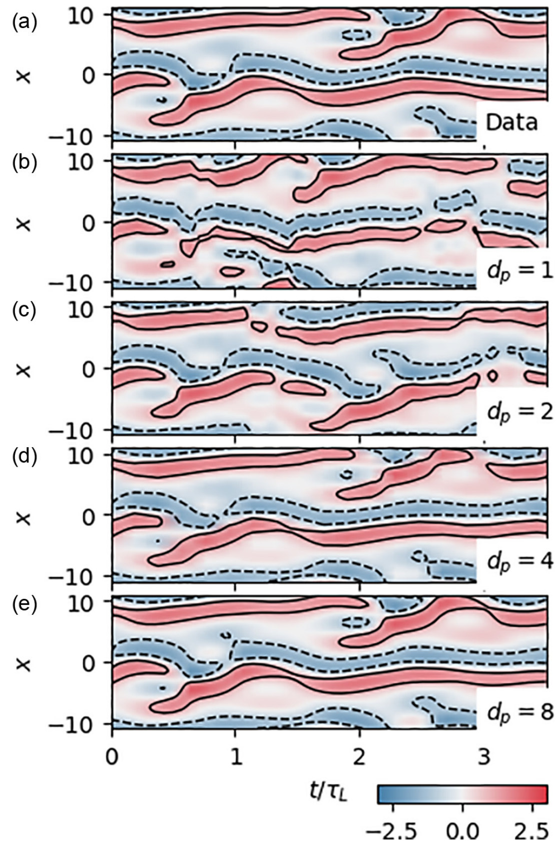

FIG. 11. Visualization of KSE reconstruction after time integration using DTS models. Color contour trajectories $u(x, t)$ with solid lines at $u = 1$ and dashed lines at $u = -1$. (a) Data. (b) $d_p = 1, m = 16, \tau = 1.5$. (c) $d_p = 2, m = 8, \tau = 1.5$. (d) $d_p = 4, m = 4, \tau = 4.0$. (e) $d_p = 8, m = 2, \tau = 4.0$.

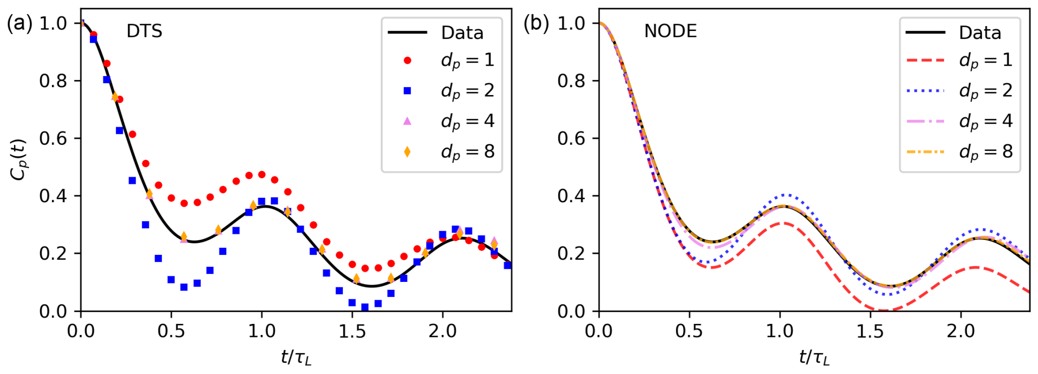

FIG. 10. Autocorrelation function of the KSE partial observable for increasing observation dimension d_p. Results from (a) DTS models and (b) NODE models. Embedding parameters are the same as Fig. 9.

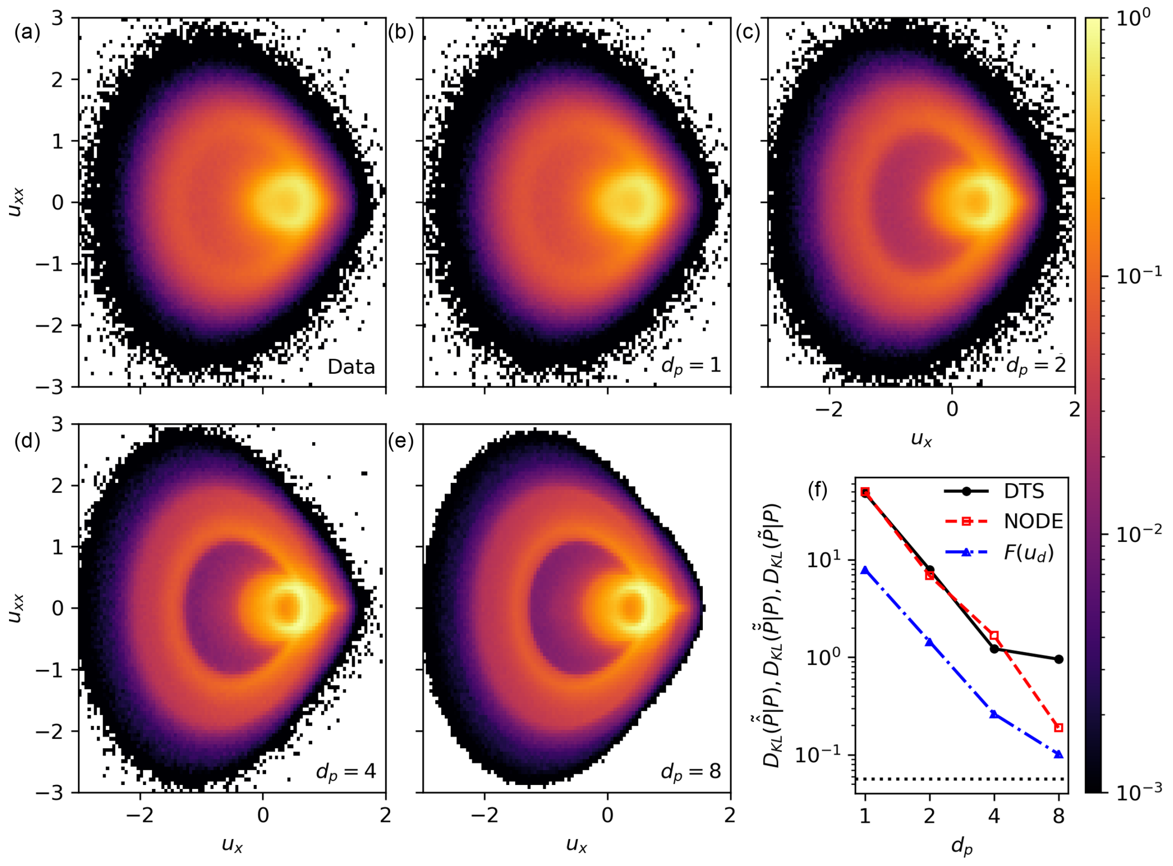

FIG. 12. Joint PDFs of the KSE for $L = 22$ generated from (a) data and discrete time integrated and reconstructed NN trajectories, $\tilde{\tilde{u}} = F(G(u))$, with observation dimensions(b) $d_p = 1$, (c) $d_p = 2$, (d) $d_p = 4$, (e) $d_p = 8$. Embedding parameters are the same as previous figures. (f) KL divergence of the true attractor PDF and NN reconstructed PDFs. Filled symbols refer to DTS forecasted data, open symbols refer to NODE forecasted data, and half-filled symbols to reconstruction of a true partial observable trajectory without time integration. The horizontal dotted line indicates the KL divergence of two PDFs from numerical simulations with different initial conditions.

delay spacings perform better for larger observation dimensions due to the increased delay window, so in these results we use $\tau = 4.0$ for $d_p = 4, 8$. We see that for a sparse observation of the state space, both DTS and NODE models diverge from the true solution relatively quickly compared to the Lyapunov time $\tau_L \approx 21$ [57]. Tracking improves dramatically from $d_p = 2$ to $d_p = 4$ and slightly more for $d_p = 8$. For $d_p = 4, 8$, there is a modest dependence on τ, but for $d_p = 1, 2$ predictions separate from the true solution quickly regardless of the choice of τ. We generally find NODE models to perform better than or equivalent to DTS models in short-term tracking. This could be related to the NODEs being trained with a data spacing $\Delta t = 0.25$ as compared to $\tau = 1.5 - 4.0$ for the DTS models, although discrete time steppers trained on ROMs of the KSE full state have shown prediction degradation does not occur until a data spacing $0.4\tau_L \approx 8.4$ [10].

Next we investigate the long-time dynamics of the NN discrete time maps by the autocorrelation in Fig. 10, again showing only one embedding for each d_p. The data is generated from a trajectories run for $t \approx 1 \times 10^5$ time units or $t/\tau_L \approx 5000$. Both DTS and NODE model predictions for $d_p = 4, 8$ reproduce the data up to 2–3 Lyapunov times. These observables also match the true autocorrelation up to at least τ_L for lower embedding dimensions $d_pm = d_{\mathcal{M}} = 8$. Shorter or longer delay windows from $\tau = 1.0 - 6.0$ also agree up to τ_L. However, model predictions using $d_p = 1, 2$ do not reproduce the data for any combination of hyperparameters and embedding parameters we tested. This suggests there is a number of observables at which learning the time map becomes significantly easier, which is consistent with the improvement from $d_p = 2$ to $d_p = 4$ seen in Ref. [32]. This could be related to the number of determining nodes as predicted by inertial manifold theory [67] and infinite dimensional versions of Takens theorem [68,69], although these works predict a significantly lower dimensional observation of $d_p = 4, m = 1$ or $d_p = 1, m = 4$ to fully describe the KS attractor.

The success of NNs with $d_p = 4, 8$ could simply be because the diffeomorphism is easier to learn, as compared to $d_p = 1, 2$, which are theoretically diffeomorphic to the attractor at $m = 16, 8$, but are found to perform significantly worse in practice. Both deviate from the true correlation function after $0.25\tau_L$, although the dynamics remain chaotic. The poor long-time performance is despite the fact that the one-step loss for $d_p = 2, m = 8$ is quantitatively comparable to $d_p = 4, m = 4$. This suggests the predicted trajectory initially stays on the true attractor for some short duration, which we visualize below after attractor reconstruction [Fig. 11(b)].

We comment that partial observation embeddings $d_p = 1, 2, 4$ with few delays perform significantly worse than the results shown here both in short-term tracking and long-time

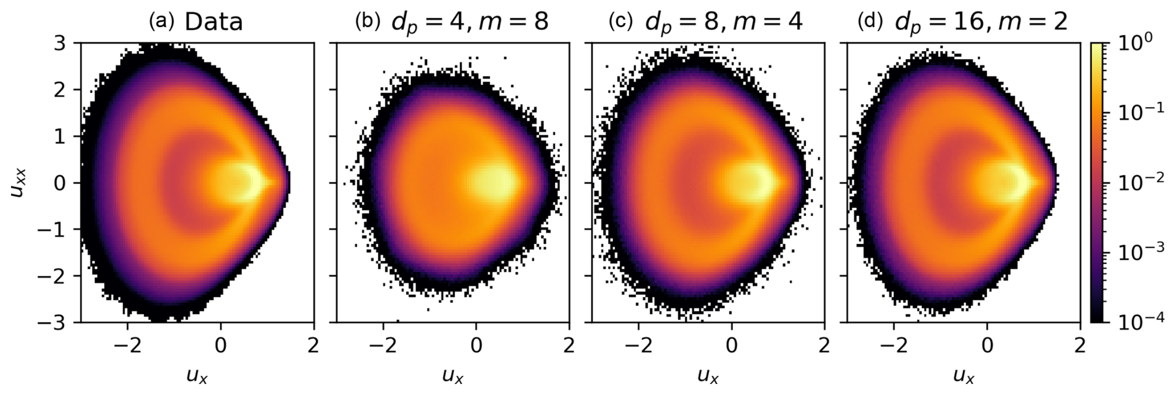

FIG. 13. Joint PDFs of the KSE for $L = 44$ as determined from (a) data, (b) $d_p = 4$, $m = 8$, $\tau = 1.25$, (c) $d_p = 8$, $m = 4$, $\tau = 2.5$, (d) $d_p = 16$, $m = 2$, $\tau = 2.5$. Time integration is performed using a DTS model (b)–(d).

dynamics. Generally, for $d_p m < d_{\mathcal{M}}$, the predicted trajectory quickly goes to a fixed point or periodic orbit. While $d_p = 1, 2$ do not quantitatively reproduce the attractor, there is a clear improvement from the delay coordinate embedding.

Finally, we investigate the quality of the reconstruction of the true KS attractor simulated on a grid from our delay coordinate embeddings. As with the Lorenz attractor, we first consider an individual trajectory to visualize the effect of increasing observation dimension. In Fig. 11, the same initial condition is used in each panel. The true data [Fig. 11(a)] is generated from numerical integration of the KSE. The panels below are generated by first filtering the initial condition to the appropriate number of equally spaced grid points d_p and embedding these observations with time delays of the partial state. The trajectories are evolved forward for $t \approx 70$ time units as detailed in Sec. II A and reconstructed to the full state as in Sec. II C. A DTS model is used here, but the NODE model predictions are visually similar.

For the trajectory shown, the NN time integrated and reconstructed trajectory for $d_p = 8, m = 2$ shows excellent agreement with the true solution up to 70 time units. The trajectory generated from $d_p = 4, m = 4$ also performs well, although visible differences emerge by $2\tau_L \approx 40$. Even a sparse observation $d_p = 2, m = 8$ generates reasonable tracking and reconstruction up to τ_L, although at intermediate times it becomes clear the predicted solution leaves the true attractor. The single grid point $d_p = 1$ time prediction separates from the true solution by $0.5\tau_L$, and error in the reconstruction are visually apparent.

Next we visualize the accuracy of reconstruction of NN time-integrated trajectories used to generate the autocorrelation function in Fig. 10. We consider the joint PDF of the first and second spatial derivatives, $P(u_x, u_{xx})$ (Fig. 12), which provides a detailed view of the attractor [9,10]. Joint PDFs generated by a DTS model are shown and NODE results are similar. Attractor reconstruction again improves with the observation dimension, with $d_p = 8$ showing excellent agreement. With $d_p = 4$, there are small and rare excursions off the true attractor, but the finer details are retained. At lower dimensions $d_p = 1, 2$, the predicted values do not capture the high density regions of the attractor accurately, in agreement with the other metrics. We quantify the difference between the predicted and true attractors with the KL divergence $D_{\text{KL}}(\tilde{\tilde{P}}|P)$, defined similarly to the Lorenz case [Eq. (11)]. The predictions improve by over an order of magntiude from $d_p = 2$ to $d_p = 4$, and then by a factor of 2 for $d_p = 8$. The KL divergence of the true data and model predictions are comparable for both DTS and NODE time integration. We include a comparison to the KL divergence $D_{\text{KL}}(\tilde{P}|P)$ of a joint PDF generated by reconstruction of a true partial trajectory without time integration, $F(u_d; \theta_F)$. As expected, it is closer to the true data due to the lack of time integration error. The dashed horizontal line indicates the KL divergence between two true solutions with different initial conditions, which is quantitatively comparable to the $d_p = 8$ prediction.

To further demonstrate the scaling of our approach to higher dimensional attractors, we consider data from the KSE with $L = 44$, which lies on a manifold of dimension $d_{\mathcal{M}} = 18$. The numerical simulation details, amount of training data, and NN training procedure are the same as for $L = 22$. The network width is increased to provide additional capacity for modeling the higher dimensional attractor. Here we show only the joint PDFs $P(u_x, u_{xx})$ in Fig. 13 and the associated KL divergence for DTS, NODE, and reconstruction models in Fig. 14.

We again find that as the observation dimension $d_p = 16$ approaches the attractor dimension, the NNs are successful even without delays, $m = 1$, although an additional delay $m = 2$ provides quantitative improvement. At $d_p = 8 \approx 1/2 d_{\mathcal{M}}$, the NN predictions stay on the attractor at an embedding dimension $m = 4$, although the predictions are quantitatively worse than the comparable case at $L = 22, d_p = 4 \approx 1/2 d_{\mathcal{M}}$. At lower dimensions $d_p = 4 \approx 1/4 d_{\mathcal{M}}$, the NNs again fail to provide accurate predictions for any delay embedding dimension. Thus, there may be practical limitations in learning global delay coordinate maps for sparse measurements on high-dimensional attractors. This could be alleviated by a multiple charts and atlases approach, in which the attractor is clustered into regions which may be locally lower dimensional and thus easier to approximate [70]. DTS models are qualitatively comparable to NODE models but more quantitatively accurate in reproducing attractor statistics (Fig. 14).

IV. CONCLUSIONS

We have presented a method for forecasting and reconstructing chaotic attractors from partial observable data. We use deep NNs to learn functions that approximate the

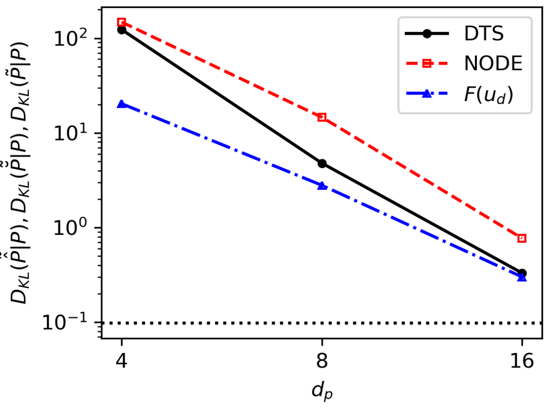

FIG. 14. KL divergence of the true and predicted KS $L = 44$ joint PDF $P(u_x, u_{xx})$ for increasing observation dimension d_p. All embedding dimensions are $d_p m = 32$. The horizontal line indicates the divergence between two true data sets with different initial conditions. Open symbols and closed symbols refer to NODE and DTS time integration models, respectively. Half-filled symbols refer reconstruction of a true partial observable trajectory, $F(u_d; \theta_F)$, which excludes error from time integration.

diffeomorphic mapping from a delay coordinate embedding to the true attractor. We have verified the approach on two common model systems: the 63 Lorenz system and the KSE. The low-dimensional Lorenz model can be accurately predicted in time and reconstructed to the true state from a scalar observation. The KSE, however, requires multivariate observations to fully reproduce the attractor. We have tested our method on short-time tracking and long-time dynamics.

The method has similarities to other data-driven approaches which reconstruct latent attractors and learn discrete time maps or continuous time flows. However, we have demonstrated the capacity of DNNs to reproduce high-dimensional attractors via the KSE at $L = 22, 44$. Currently, it has only been applied to embeddings with uniform time delays but it can also be applied to nonuniform embeddings. In this case, neural ODEs would be advantageous because the discrete time map would require interpolation or a restrictive choice of time step to update the time delays. A limitation of uniform embeddings is that it is difficult to learn the delay coordinate maps for sparse observations with many delays, as evidenced by the relatively poor performance for KSE with $d_p < 1/2 d_{\mathcal{M}}$. Information at times intermediate to the uniform delays could improve the quality delay coordinate phase space, although our attempts on the KSE using several existing methods [23,24,27] did not converge.

Our method is relevant to applications requiring forecasting of partial observations from experimental data. The reconstruction map could be trained on simulation data and applied to experimental measurements, as proposed for protein configurations [4,53–55], although it is not a focus of the current paper. We are particularly interested in applying our approach to data-driven control [71,72]. A deterministic reduced-order model would be useful for controller design by many methods, in particular, for recent deep reinforcement learning approaches [14,35,73]. With minimal modifications, a time-delayed evolution model with actuated data would replace an actuated reduced order model learned from full state data. Other data-driven time evolution models have been used for control, such as dynamic mode decomposition [74], sparse regression, [75], and Koopman theory [76–78]. These methods may be preferable for interpretability and discovery of an optimal control policy, particularly in the low-data limit [79]. However, we propose that reduced order NN models are ideal for controlling nonlinear high dimensional systems because of their accuracy and computational efficiency.

ACKNOWLEDGMENTS

This work was supported by No. FA9550-18-1-0174 and No. N00014-18-1-2865 (Vannevar Bush Faculty Fellowship).

[1] M. Wang and T. A. Zaki, J. Fluid Mech. **917**, A9 (2021).
[2] J. Elsner and A. Tsonis, Bull. Am. Meteorol. Soc. **73**, 49 (1992).
[3] B. Dodov and E. Foufoula-Georgiou, Adv. Water Resour. **28**, 711 (2005).
[4] J. Wang and A. L. Ferguson, Phys. Rev. E **93**, 032412 (2016).
[5] R. Chandra and Y. He, PLOS ONE **16**, e0253217 (2021).
[6] V. Guillemin and A. Pollack, *Differential Topology* (Englewood Springs, NJ, 2010), Vol. 370.
[7] T. Sauer, J. A. Yorke, and M. Casdagli, J. Stat. Phys. **65**, 579 (1991).
[8] S. L. Brunton, J. L. Proctor, and J. N. Kutz, Proc. Natl. Acad. Sci. USA **113**, 3932 (2016).
[9] A. J. Linot and M. D. Graham, Phys. Rev. E **101**, 062209 (2020).
[10] A. J. Linot and M. D. Graham, Chaos **32**, 073110 (2022).
[11] H. Jaeger and H. Haas, Science **304**, 78 (2004).
[12] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, Proc. R. Soc. A **474**, 20170844 (2018).
[13] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Phys. Rev. Lett. **120**, 024102 (2018).
[14] K. Zeng, A. J. Linot, and M. D. Graham, Proc. Royal Soc. A, **478**, 20220297 (2022).
[15] F. Takens, in *Dynamical Systems and Turbulence, Warwick 1980* (Springer, Berlin, Germany, 1981), pp. 366–381.
[16] E. R. Deyle and G. Sugihara, PLoS ONE **6**, e18295 (2011).
[17] H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis* (Cambridge University Press, Cambrdige, England, 2004), Vol. 7.
[18] E. Bradley and H. Kantz, Chaos **25**, 097610 (2015).
[19] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, Phys. Rev. A **45**, 3403 (1992).
[20] L. Cao, Physica D **110**, 43 (1997).
[21] A. M. Fraser and H. L. Swinney, Phys. Rev. A **33**, 1134 (1986).
[22] H. Kim, R. Eykholt, and J. Salas, Physica D **127**, 48 (1999).
[23] S. P. Garcia and J. S. Almeida, Phys. Rev. E **72**, 027205 (2005).
[24] C. Nichkawde, Phys. Rev. E **87**, 022905 (2013).

[25] M. McCullough, M. Small, T. Stemler, and H. H.-C. Iu, Chaos **25**, 053101 (2015).
[26] B. Xu, C. J. Tralie, A. Antia, M. Lin, and J. A. Perea, J. Appl. Comput. Topology **3**, 285 (2019).
[27] K.-H. Krämer, G. Datseris, J. Kurths, I. Z. Kiss, J. L. Ocampo-Espindola, and N. Marwan, New J. Phys. **23**, 033017 (2021).
[28] H. Jiang and H. He, in *2017 International Joint Conference on Neural Networks (IJCNN)* (IEEE, Anchorage, AK, 2017), pp. 3191–3198.
[29] W. Gilpin, Adv. Neural Inf. Process. Syst. **33**, 204 (2020).
[30] S. Ouala, D. Nguyen, L. Drumetz, B. Chapron, A. Pascual, F. Collard, L. Gaultier, and R. Fablet, Chaos **30**, 103121 (2020).
[31] Z. Wang and C. Guet, arXiv:2108.01862.
[32] Z. Lu, J. Pathak, B. Hunt, M. Girvan, R. Brockett, and E. Ott, Chaos **27**, 041102 (2017).
[33] P. Cvitanovic and B. Eckhardt, Nonlinearity **6**, 277 (1993).
[34] J. Ling, R. Jones, and J. Templeton, J. Comput. Phys. **318**, 22 (2016).
[35] K. Zeng and M. D. Graham, Phys. Rev. E **104**, 014210 (2021).
[36] G. A. Gottwald and S. Reich, Chaos **31**, 101103 (2021).
[37] J. Brajard, A. Carrassi, M. Bocquet, and L. Bertino, J. Comput. Sci. **44**, 101171 (2020).
[38] J. Bakarji, K. Champion, J. N. Kutz, and S. L. Brunton, arXiv:2201.05136.
[39] S. Ouala, S. L. Brunton, B. Chapron, A. Pascual, F. Collard, L. Gaultier, and R. Fablet, Phys. D: Nonlinear Phenom. **446**, 133630 (2023).
[40] S. W. Jiang and J. Harlim, Res. Math. Sci. **7**, 16 (2020).
[41] J. Harlim, S. W. Jiang, S. Liang, and H. Yang, J. Comput. Phys. **428**, 109922 (2021).
[42] S. L. Brunton, B. W. Brunton, J. L. Proctor, E. Kaiser, and J. N. Kutz, Nat. Commun. **8**, 19 (2017).
[43] S. Pan and K. Duraisamy, Chaos **30**, 073135 (2020).
[44] R. Zwanzig, *Nonequilibrium Statistical Mechanics* (Oxford University Press, Oxford, England, 2001).
[45] D. Kondrashov, M. D. Chekroun, and M. Ghil, Physica D **297**, 33 (2015).
[46] M. Ghil and V. Lucarini, Rev. Mod. Phys. **92**, 035002 (2020).
[47] H. Lei, N. A. Baker, and X. Li, Proc. Natl. Acad. Sci. **113**, 14183 (2016).
[48] Z. She, P. Ge, and H. Lei, J. Chem. Phys. (2022).
[49] J. L. Callaham, J.-C. Loiseau, G. Rigas, and S. L. Brunton, Proc. R. Soc. A **477**, 20210092 (2021).
[50] J. Stark, D. S. Broomhead, M. E. Davies, and J. Huke, J. Nonlinear Sci. **13**, 519 (2003).
[51] Y. Hirata, Chaos **28**, 033112 (2018).
[52] D. Darmon, Phys. Rev. E **97**, 032206 (2018).
[53] J. Wang and A. L. Ferguson, J. Phys. Chem. B **122**, 11931 (2018).
[54] M. Topel and A. L. Ferguson, J. Chem. Phys. **153**, 194102 (2020).
[55] M. Topel, A. Ejaz, A. Squires, and A. L. Ferguson, J. Chem. Theory Comput. (2023), doi: 10.1021/acs.jctc.2c00920.
[56] H.-l. Yang, K. A. Takeuchi, F. Ginelli, H. Chaté, and G. Radons, Phys. Rev. Lett. **102**, 074102 (2009).
[57] X. Ding, H. Chaté, P. Cvitanović, E. Siminos, and K. A. Takeuchi, Phys. Rev. Lett. **117**, 024101 (2016).
[58] F. Chollet *et al.*, *Keras* (2015), https://keras.io/getting_started/faq/#how-should-i-cite-keras.
[59] A. J. Linot, J. W. Burby, Q. Tang, P. Balaprakash, M. D. Graham, and R. Maulik, J. Comput. Phys. **474**, 111838 (2023).
[60] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, Adv. Neural Inf. Process. Syst. **31** (2018).
[61] E. N. Lorenz, J. Atmos. Sci. **20**, 130 (1963).
[62] P. Grassberger and I. Procaccia, Phys. Rev. Lett. **50**, 346 (1983).
[63] G. Datseris, J. Open Source Software **3**, 598 (2018).
[64] F. Cazáis and A. Lhéritier, in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)* (IEEE, Paris, France, 2015), pp. 1–10.
[65] P. Cvitanovic, R. Artuso, R. Mainieri, G. Tanner, G. Vattay, N. Whelan, and A. Wirzba, *Chaos: Classical and Quantum* (Niels Bohr Institute, Copenhagen, 2016).
[66] A.-K. Kassam and L. N. Trefethen, SIAM J. Sci. Comput. **26**, 1214 (2005).
[67] C. Foias and E. S. Titi, Nonlinearity **4**, 135 (1991).
[68] C. Foias and I. Kukavica, J. Dyn. Diff. Equ. **7**, 365 (1995).
[69] I. Kukavica and J. C. Robinson, Physica D **196**, 45 (2004).
[70] D. Floryan and M. D. Graham, Nat. Mach. Intell. **4**, 1113 (2021).
[71] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, Annu. Rev. Fluid Mech. **52**, 477 (2020).
[72] F. Ren, J. Rabault, and H. Tang, Phys. Fluids **33**, 037121 (2021).
[73] A. J. Linot, K. Zeng, and M. D. Graham, arXiv:2301.12098.
[74] J. L. Proctor, S. L. Brunton, and J. N. Kutz, SIAM J. Appl. Dyn. Syst. **15**, 142 (2016).
[75] S. L. Brunton, J. L. Proctor, and J. N. Kutz, IFAC-PapersOnLine **49**, 710 (2016).
[76] S. L. Brunton, B. W. Brunton, J. L. Proctor, and J. N. Kutz, PLoS ONE **11**, e0150171 (2016).
[77] J. L. Proctor, S. L. Brunton, and J. N. Kutz, SIAM J. Appl. Dyn. Syst. **17**, 909 (2018).
[78] E. Kaiser, J. N. Kutz, and S. L. Brunton, Mach. Learn.: Sci. Technol. **2**, 035023 (2021).
[79] E. Kaiser, J. N. Kutz, and S. L. Brunton, Proc. R. Soc. A **474**, 20180335 (2018).