Reconstruction, forecasting, and stability of chaotic dynamics from partial data

Chaos
An Interdisciplinary Journal of Nonlinear Science

RESEARCH ARTICLE | SEPTEMBER 06 2023

Reconstruction, forecasting, and stability of chaotic dynamics from partial data

Elise Özalp ; Georgios Margazoglou ; Luca Magri

Chaos 33, 093107 (2023)
https://doi.org/10.1063/5.0159479

Reconstruction, forecasting, and stability of chaotic dynamics from partial data

Cite as: Chaos 33, 093107 (2023); doi: 10.1063/5.0159479
Submitted: 24 May 2023 · Accepted: 10 August 2023 ·
Published Online: 6 September 2023

Elise Özalp,[1] Georgios Margazoglou,[1] and Luca Magri[1,2,a)]

AFFILIATIONS

[1]Department of Aeronautics, Imperial College London, London SW7 2BX, United Kingdom
[2]The Alan Turing Institute, London NW1 2DB, United Kingdom

[a)]**Author to whom correspondence should be addressed:** l.magri@imperial.ac.uk

ABSTRACT

The forecasting and computation of the stability of chaotic systems from partial observations are tasks for which traditional equation-based methods may not be suitable. In this computational paper, we propose data-driven methods to (i) infer the dynamics of unobserved (hidden) chaotic variables (full-state reconstruction); (ii) time forecast the evolution of the full state; and (iii) infer the stability properties of the full state. The tasks are performed with long short-term memory (LSTM) networks, which are trained with observations (data) limited to only part of the state: (i) the low-to-high resolution LSTM (LH-LSTM), which takes partial observations as training input, and requires access to the full system state when computing the loss; and (ii) the physics-informed LSTM (PI-LSTM), which is designed to combine partial observations with the integral formulation of the dynamical system's evolution equations. First, we derive the Jacobian of the LSTMs. Second, we analyze a chaotic partial differential equation, the Kuramoto–Sivashinsky, and the Lorenz-96 system. We show that the proposed networks can forecast the hidden variables, both time-accurately and statistically. The Lyapunov exponents and covariant Lyapunov vectors, which characterize the stability of the chaotic attractors, are correctly inferred from partial observations. Third, the PI-LSTM outperforms the LH-LSTM by successfully reconstructing the hidden chaotic dynamics when the input dimension is smaller or similar to the Kaplan–Yorke dimension of the attractor. The performance is also analyzed against noisy data. This work opens new opportunities for reconstructing the full state, inferring hidden variables, and computing the stability of chaotic systems from partial data.

It has been shown that the chaotic dynamics of complex systems can be inferred by machine learning from data when all the dynamics are observed, but questions remain when only partial observations are available. We design data-driven approaches that can accurately infer hidden chaotic dynamics from partial data and correctly learn the chaotic attractor, its tangent space, and its stability properties.

I. INTRODUCTION

Chaotic dynamics appear in applications in various fields, from meteorology,[1] through engineering and chemistry,[2] to propulsion.[3,4] Chaos emerges due to the system's exponential sensitivity to initial conditions, as is the case with turbulent fluid dynamics.[5] However, the accurate prediction of chaotic spatiotemporal behavior is challenging because numerical computations can be computationally expensive, and sensor measurements are often limited, providing only a partial observation of the dynamic behavior. These partial observations can lead to a misrepresentation of the unobserved (hidden) variables, or other long-term physical properties, which further challenges the modeling and prediction of the systems.

The predictability and stability of a chaotic system are characterized by its tangent space, which can be computed using the linearized dynamics provided by the Jacobian. This computation allows for the derivation of quantities, such as the Lyapunov exponents (LEs), which measure the exponential rate of separation of trajectories. A geometric characterization is provided by the covariant Lyapunov vectors (CLVs), which constitute a covariant basis of the tangent space, and point to directions of asymptotic expansion and contraction of the dynamical system.[6] Preserving these stability properties is crucial when building surrogate models from limited observations to a more comprehensive dataset.[7]

Neural Networks are expressive nonlinear representations of continuous functions, which can extract patterns from data and, once trained, provide computationally cheap predictions. Suitable

for time series and dynamical evolutions are recurrent neural networks (RNNs), which have shown promising performance in the inference of dynamical systems with multi-scale, chaotic, or turbulent behavior.[8] There are several classes of RNNs, the main classes being reservoir computing, such as echo state networks (ESNs),[9] and networks using backpropagation through time, such as long short-term memory networks (LSTMs).[10] Both networks have been extensively applied to short-term prediction tasks when trained on complete observations.[8,11–13] Additionally, ESNs have been employed for predicting from noisy, undersampled observations and for crossprediction,[14,15] whilst LSTMs have been trained on incomplete observations[16,17] with a focus on short-term time metrics, such as the root-mean-square error. In the context of learning stability properties for prototypical chaotic systems, ESNs have demonstrated accurate inference capabilities when trained on the full state.[11,18,19] This work proposes two approaches to infer the hidden chaotic dynamics from partial observations (data) with LSTMs. The first approach considers the scenario in which the full-state data are only available for a limited period, and partial data are available for the remaining time. This scenario arises when generating high-resolution data is computationally expensive. For this, the goal is to analyze how well LSTMs can infer the full-state dynamics from partial data, especially in capturing small-scale statistics in partial differential equations. The second approach analyzes the case in which only partial state data are available, and the aim is to reconstruct (i.e., infer) the hidden dynamics. To achieve this, knowledge of the underlying physical system has to be incorporated by adding constraints based on the governing equations. We propose a physics-informed loss function based on integral formulation of the dynamical system, which is versatile and suitable for the architecture of the RNN.

This paper has a threefold goal. First, we perform reconstruction of the full-state vector from partial data and forecast the entire state in the future. Second, we infer the Jacobian of the system, enabling the analysis of the tangent space both geometrically (with CLVs) and spectrally (with LEs and CLVs) from data only, i.e., without computing the Jacobian of the physical systems' governing equations. Last, both approaches are compared on partial inputs from two prototypical chaotic systems: the Lorenz-96 system[20] and the Kuramoto–Sivashinsky equation.[21,22]

This paper is structured as follows. Section II introduces the LSTM architecture to address the state reconstruction problem of a chaotic dynamical system. We propose a data-driven and physics-informed loss formulation in Sec. III. In Sec. IV, we provide an overview of the computation of LEs and CLVs, emphasizing their importance for physically accurate modeling. We derive the Jacobian of the LSTM. Section V discusses the results for the Kuramoto–Sivashinsky system, followed by the findings for the Lorenz-96 system. Finally, in Sec. VI, we summarize our work and discuss future directions.

II. LONG SHORT-TERM MEMORY

A dynamical system's solution is a time series, which provides an ordered sequence of data over time. Recurrent neural networks (RNNs) are a data-driven approach for modeling time series that utilize hidden states to encode the input's information history. Among RNNs, long short-term memory (LSTM) networks are a common tool due to their internal gating mechanisms, which mitigate the vanishing gradient problem.

LSTMs are commonly employed for time series forecasting as they embed the time-delayed inputs in their higher-dimensional hidden states. From a dynamical systems perspective, a delay embedding can provide a structure that is topologically equivalent to the attractor.[5] Given their architecture, LSTMs have the potential to be a powerful tool for representing the dynamics of a chaotic system. This enables them to be utilized for reconstructing and forecasting dynamical systems, even in cases in which we lack part of the system's full-state information, as will be demonstrated in the following.

A. State reconstruction for the inference of hidden dynamics

We consider a nonlinear autonomous dynamical system

$$\frac{d}{dt}\boldsymbol{x}(t) = f(\boldsymbol{x}(t)), \tag{1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^D$ is the state vector of the physical system and $f: \mathbb{R}^D \to \mathbb{R}^D$ is a smooth nonlinear vector function. Experimental observations often come with a limited amount of information on a system's full state; for example, if in a wind tunnel experiment $\mathcal{O}(10)$ sensors are placed in different positions, they may provide adequate but scarce spatial information of the system's evolution.[23] Mathematically, if $\boldsymbol{x}(t) = [\boldsymbol{y}(t); \boldsymbol{\xi}(t)]$ is the full state of a chaotic dynamical system, then $\boldsymbol{y}(t) \in \mathbb{R}^{D_y}$ are the scarce observations and $\boldsymbol{\xi}(t) \in \mathbb{R}^{D_\xi}$ are the unobserved (hidden) variables with $D = D_y + D_\xi$. Specifically, let us assume that $\boldsymbol{y}(t_i)$ is measured at times $t_i = i\Delta t$ with $i = 0, \ldots, N_t$ and constant time step Δt. Based on these observations, we wish to reconstruct and predict the full state $\boldsymbol{x}(t_i) = [\boldsymbol{y}(t_i), \boldsymbol{\xi}(t_i)]$.

LSTMs have been successfully applied to time series forecasting of dynamical systems when full observations are available.[8,11,24] In this work, we use the LSTM as a model of partially observed chaotic time series to both reconstruct the full state and perform accurate autonomous temporal evolution (forecasting).

B. Architecture

The long short-term memory networks are characterized by a cell state $\boldsymbol{c}_i \in \mathbb{R}^{N_h}$ and a hidden state $\boldsymbol{h}_i \in \mathbb{R}^{N_h}$ that are updated at each step. In the case of partial observations, the states are updated by using the observed variables $\boldsymbol{y}(t_i)$ as

$$\begin{aligned}
\boldsymbol{i}_{i+1} &= \sigma\left(\boldsymbol{W}^i[\boldsymbol{y}(t_i); \boldsymbol{h}_i] + \boldsymbol{b}^i\right),\\
\boldsymbol{f}_{i+1} &= \sigma\left(\boldsymbol{W}^f[\boldsymbol{y}(t_i); \boldsymbol{h}_i] + \boldsymbol{b}^f\right),\\
\boldsymbol{o}_{i+1} &= \sigma\left(\boldsymbol{W}^o[\boldsymbol{y}(t_i); \boldsymbol{h}_i] + \boldsymbol{b}^o\right),\\
\tilde{\boldsymbol{c}}_{i+1} &= \tanh\left(\boldsymbol{W}^g[\boldsymbol{y}(t_i); \boldsymbol{h}_i] + \boldsymbol{b}^g\right),\\
\boldsymbol{c}_{i+1} &= \boldsymbol{f}_{i+1} * \boldsymbol{c}_i + \boldsymbol{i}_{i+1} * \tilde{\boldsymbol{c}}_{i+1},\\
\boldsymbol{h}_{i+1} &= \tanh\left(\boldsymbol{c}_{i+1}\right) * \boldsymbol{o}_{i+1},
\end{aligned} \tag{2}$$

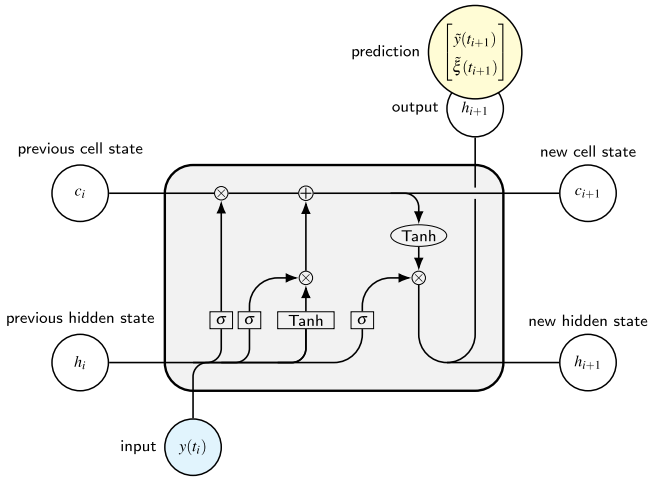

FIG. 1. Schematic representation of the LSTM cell structure.

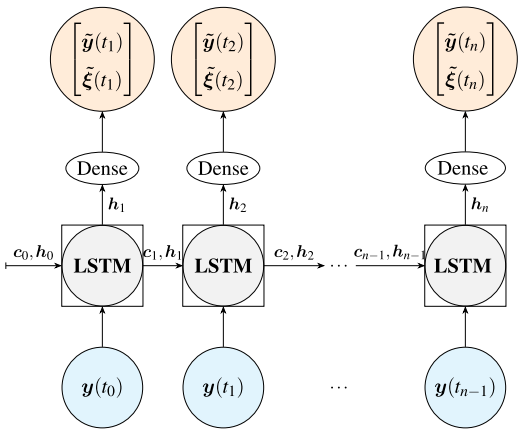

FIG. 2. LSTM in an open-loop configuration: Each cell receives input from the training data.

where $*$ denotes the elementwise multiplication and σ refers to the sigmoid activation function.[10] In the LSTM, $\boldsymbol{i}_{i+1}, \boldsymbol{f}_{i+1}, \boldsymbol{o}_{i+1} \in \mathbb{R}^{N_h}$ are the input, forget, and output gates (Fig. 1). The matrices $\boldsymbol{W}^i$, $\boldsymbol{W}^f$, $\boldsymbol{W}^o$, $\boldsymbol{W}^g \in \mathbb{R}^{N_h \times (D+N_h)}$ are the corresponding weight matrices, and $\boldsymbol{b}^i$, $\boldsymbol{b}^f$, $\boldsymbol{b}^o$, $\boldsymbol{b}^g \in \mathbb{R}^{N_h}$ are the biases. The full prediction on the next time step, $\hat{\boldsymbol{x}}(t_{i+1}) = \left[\hat{\boldsymbol{y}}(t_{i+1}), \hat{\boldsymbol{\xi}}(t_{i+1})\right]$, is obtained by applying a dense layer to the hidden state $\boldsymbol{h}_{i+1}$; i.e.,

$$\begin{bmatrix} \hat{\boldsymbol{y}}(t_{i+1}) \\ \hat{\boldsymbol{\xi}}(t_{i+1}) \end{bmatrix} = \boldsymbol{W}^{dense} \boldsymbol{h}_{i+1} + \boldsymbol{b}^{dense},$$

where $\boldsymbol{W}^{dense} \in \mathbb{R}^{D \times N_h}$ and $\boldsymbol{b}^{dense} \in \mathbb{R}^D$.

LSTMs are universal approximators for a continuous target function;[25,26] however, practically, the network's performance depends on the parameters, such as weights and biases. To determine these parameters, the available data are divided into three subsets: training, validation, and testing. The training and validation data are split into batches of fixed-length time windows. The training data are used for backpropagation through time,[27] and during the forward pass, the network output is computed and compared to a training label using a loss function. In the backward pass, the network parameters are optimized by computing the gradient of the loss function with respect to the parameters with a gradient update via the Adam optimizer.[28] The validation data are employed to optimize hyperparameters and determine parameters before training (e.g., the dimension of the hidden and cell state N^h), while test data are used to evaluate the final performance of the model.

The network works in an open-loop configuration during training and validation (Fig. 2) with the output at each time step depending on current and previous inputs within the time window, and the LSTM's states are reset to zero at the start of each input sequence. This is also known as "teacher-forced learning,"[29] which facilitates a straightforward formulation of the loss, as elaborated in Sec. III. After training, the network is evaluated on test data with fixed weights and biases, operating in a closed-loop configuration (Fig. 3). After a warm-up of one time window, the network can make long-term predictions in a closed-loop mode, even if no data are available, and the states of the LSTM are retained across time steps. The network predicts both the observed and unobserved (hidden) variables. Subsequently, the observed variables are provided as inputs for the next time step, which allows the autonomous evolution of the LSTM.

III. TASKS AND LOSS FUNCTIONS

The LSTM's internal dynamics, and, therefore, its prediction, are determined by the weights and biases of the model, which are updated during the training. The loss function plays a critical role in this process by defining the optimization problem that guides

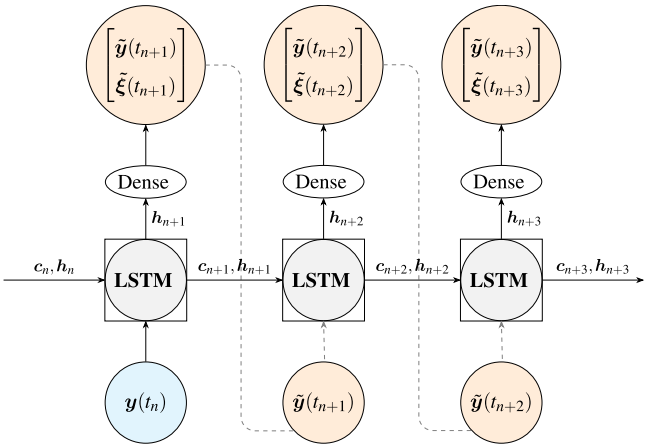

FIG. 3. LSTM in a closed-loop configuration: The network prediction is used as an input for the next cell.

the updating of these parameters. The choice of the loss function depends on the specific application and can greatly impact the accuracy and generalization performance of the model. In the following, we propose two loss functions to guide the LSTM training: (i) a data-driven loss and (ii) a physics-informed loss.

A. Low-to-high resolution loss (LH-LSTM)

The first task is to infer the high-resolution dynamics from low-resolution inputs, thereby reducing the cost of acquiring full measurement data. In fluid dynamics, in which problems display turbulent behavior, this task becomes especially difficult as the prediction is sensitive due to the chaotic nature of the data. The network predicts the full state $\hat{\boldsymbol{x}}(t_i) = [\hat{\boldsymbol{y}}(t_i); \hat{\boldsymbol{\xi}}(t_i)]$ given a partial input $\boldsymbol{y}(t_i)$. In this section, we assume that the full label $\boldsymbol{x}(t_i) = [\boldsymbol{y}(t_i); \boldsymbol{\xi}(t_i)]$ is only available for a short amount of time, which can be employed for training and validation, and partial data are available for the remaining time. Hence, we can define a data-driven loss using a mean-squared error (MSE),

$$\mathscr{L}_{dd}\left(\boldsymbol{x}, \hat{\boldsymbol{x}}\right) = \frac{1}{N_t}\sum_{i=1}^{N_t}\left\|\boldsymbol{x}(t_i) - \hat{\boldsymbol{x}}(t_i)\right\|^2, \tag{3}$$

where $||\cdot||$ is the L^2-norm. Although simple to implement, data-driven loss functions have limitations. Specifically, they are sensitive to overfitting, especially when the training data are noisy. To mitigate overfitting, we employ Tikhonov regularization by adding $\alpha_{tikh}\|\hat{\boldsymbol{x}}\|^2, \alpha_{tikh} \in \mathbb{R}^+$ to Eq. (3).[30] When data are limited, these loss functions may struggle to generalize to regions of the attractor that are sparsely sampled during training. To regularize the problem, the incorporation of the physical knowledge or constraints into the loss function is an effective means,[31,32] which favor physical solutions (in contrast to traditional Tikhonov regularization).

B. Physics-informed loss (PI-LSTM)

Physics-informed losses have shown success in feed-forward neural networks, in which automatic differentiation can be exploited to accurately compute the derivative with respect to time and space to machine precision.[33,34] Unlike feed-forward neural networks, RNNs have a dynamic temporal structure that makes it difficult to accurately differentiate the prediction with respect to time because of the recurrence and the type of inputs. As a result, the governing equations have to be incorporated in a manner that accounts for the temporal structure of the RNN architecture. This approximation is typically achieved by discretizing the time derivative in the loss function using finite differences, which allows the physics-informed loss to be computed, but it relies on the numerical differentiation scheme.[32] Hence, the autonomous dynamical system defined by Eq. (1) is reformulated with its formal solution through the integral

$$\boldsymbol{x}(t_{i+1}) = \int_{t_0}^{t_{i+1}} f(\boldsymbol{x}(t))dt = \boldsymbol{x}(t_i) + \int_{t_i}^{t_{i+1}} f(\boldsymbol{x}(t))dt,\ i \geq 0, \tag{4}$$

which enables the use of numerical quadrature methods to approximate the integral $\int_{t_i}^{t_{i+1}} f(\boldsymbol{x}(t))dt$, instead of approximating the derivative $\frac{d\boldsymbol{x}(t)}{dt}$ to construct an accurate surrogate model. This approach is naturally compatible with explicit numerical schemes of different orders of accuracy, including the explicit Runge–Kutta family or pseudo-spectral methods, thereby providing a flexible framework for modeling physical constraints into RNNs. Based on Eq. (4), we define the residual of the dynamical system

$$\mathscr{R}(\boldsymbol{x}(t_{i+1})) = \boldsymbol{x}(t_{i+1}) - \left(\boldsymbol{x}(t_i) + \int_{t_i}^{t_{i+1}} f(\boldsymbol{x}(t))dt\right). \tag{5}$$

The true solution, $\boldsymbol{x}(t)$, of the dynamical system is such that $\mathscr{R}(\boldsymbol{x}(t_i)) = 0$ for all $t_i > t_0$. In this manner, the predictions $\hat{\boldsymbol{x}}$ of the network favor physical solutions by minimizing the physics-informed loss,

$$\mathscr{L}_{pi}(\hat{\boldsymbol{x}}) = \frac{1}{N_t}\sum_{i=0}^{N_t-1}\left\|\mathscr{R}\left(\hat{\boldsymbol{x}}(t_{i+1})\right)\right\|^2. \tag{6}$$

The proposed loss function addresses a distinct task from the data-driven loss (3) by not requiring a full label, $\boldsymbol{x}(t_i)$, making it particularly advantageous in situations in which full labels are not available. Alternatively, it enables the network's prediction to be constrained based on the governing equations, which allows the reconstruction and forecasting of unobserved (hidden) variables. By providing regularization during the training of the LSTM, we also ensure that the predictions satisfy the governing equation defined in (1) within numerical tolerance. Furthermore, this regularization helps constraining the parameter space of the weights, which can mitigate overfitting and improve generalization. We also include a data-driven loss for the available data by computing the MSE between the network prediction on the partial input $\hat{\boldsymbol{y}}(t_{i+1})$ and the partial label $\boldsymbol{y}(t_{i+1})$,

$$\mathscr{L}_{dd}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N_t}\sum_{i=1}^{N_t}\left\|\boldsymbol{y}(t_i) - \hat{\boldsymbol{y}}(t_i)\right\|^2. \tag{7}$$

The loss of the physics-informed LSTM is computed by combining the data-driven loss and weighing the physics-informed loss as

$$\mathscr{L}(\boldsymbol{y}, \hat{\boldsymbol{x}}) = \mathscr{L}_{dd}(\boldsymbol{y}, \hat{\boldsymbol{y}}) + \alpha_{pi}\mathscr{L}_{pi}(\hat{\boldsymbol{x}}), \quad \alpha_{pi} \in \mathbb{R}^+, \tag{8}$$

where α_{pi} is a penalty hyperparameter, which acts as a regularization factor. In the systems under investigation, there is no obvious *a priori* choice for the selection of an appropriate regularization hyperparameter. Poor selection of α_{pi} can result in suboptimal models. If α_{pi} is too large, it may lead to overfitting of the physics while disregarding the observations. Conversely, if α_{pi} is too small, the models may overfit the measurements and fail to generalize. To select the regularization factor, α_{pi}, we employ a grid search as detailed in the Appendix.

IV. INFERRING THE STABILITY OF THE DYNAMICAL SYSTEM FROM DATA

Thanks to its recurrent nature, the LSTM can use its own prediction as input in the closed-loop mode, which allows it to predict the system's full state in time beyond the input. This autonomous evolution of LSTMs defines a dynamical system. Mathematically, we assess the LSTM stability properties by studying their tangent space.[18] By introducing small perturbations to the system's trajectory

and linearizing the system's equations, we can compute its stability in different directions. We determine the properties of its linear tangent space, such as Lyapunov exponents (LEs) and covariant Lyapunov vectors (CLVs). To gain insight into the network's dynamical behavior, we apply these concepts to the LSTM by mathematically deriving the LSTM Jacobian.

A. Lyapunov exponents

The solution of the dynamical system in Eq. (1) is $\boldsymbol{x}(t)$, which defines a trajectory in the phase space of dimension D. To analyze the properties of the dynamical system and its attractor, we impose infinitesimal perturbations to the trajectory of the system as

$$\boldsymbol{x}+\boldsymbol{u}, \quad \boldsymbol{x}\sim\mathcal{O}(1), \quad \boldsymbol{u}\sim\mathcal{O}(\varepsilon), \quad \varepsilon\to 0. \tag{9}$$

By substituting (1) and linearizing the system, the perturbation is governed by the tangent equation

$$\frac{d\boldsymbol{u}(t)}{dt}=\boldsymbol{J}(\boldsymbol{x}(t))\boldsymbol{u}(t), \tag{10}$$

where $J_{ij}=\frac{\partial f_i(\boldsymbol{x})}{\partial x_j}$ are the components of the Jacobian $\boldsymbol{J}(\boldsymbol{x}(t))\in\mathbb{R}^{D\times D}$, which is time-dependent on chaotic attractors. This shows that the perturbation $\boldsymbol{u}(t)$ evolves along the tangent space with respect to $\boldsymbol{x}(t)$ and must, therefore, be integrated alongside Eq. (1). Considering K random perturbations, we numerically integrate $K\leq D$ tangent vectors $\boldsymbol{u}_i\in\mathbb{R}^D$ as columns of $\boldsymbol{U}\in\mathbb{R}^{D\times K}$,

$$\frac{d\boldsymbol{U}}{dt}=\boldsymbol{J}(\boldsymbol{x}(t))\boldsymbol{U}. \tag{11}$$

In chaotic systems, the tangent vectors align exponentially fast with the leading Lyapunov vector, which can lead to an ill-conditioned matrix $\boldsymbol{U}$.[6,35] For practical purposes, $\boldsymbol{U}$ is periodically orthonormalized through the Gram–Schmidt procedure.[36] Hence, we decompose $\boldsymbol{U}$ into an orthogonal matrix $\boldsymbol{Q}$ and an upper triangular matrix $\boldsymbol{R}$ with QR decomposition; i.e., $\boldsymbol{U}(t)=\boldsymbol{Q}(t)\boldsymbol{R}(t)$. The columns of $\boldsymbol{Q}$ form an orthonormal basis for the tangent space and are known as Gram–Schmidt vectors (GSVs). The diagonal entries $R_{ii}(t)$ correspond to the local growth rates of the corresponding GSV $\boldsymbol{q}_i(t)$. The Lyapunov exponents (LEs) are computed by taking the time average of the logarithms of $R_{ii}(t)$,

$$\lambda_i=\lim_{T\to\infty}\frac{1}{T-t_0}\int_{t_0}^{T}\ln\left(R_{ii}(t)\right). \tag{12}$$

The Lyapunov spectrum, $\lambda_1\geq\cdots\geq\lambda_D$, provides fundamental insights into the chaotic properties and geometry of an attractor.[3,6] If the leading Lyapunov exponent $\lambda_1<0$, the perturbation decay and the attractor are a fixed point. If $\lambda_1=0$ and the remaining exponents are negative, the attractor is a periodic orbit. If $\lambda_1>0$, the perturbation grows exponentially, and typically, the attractor is chaotic. In this case, the Lyapunov time $\tau_\lambda=\frac{1}{\lambda_1}$ defines a characteristic timescale for two nearby trajectories to separate, which gives an estimate of the system's predictability horizon.[37,38] Furthermore, the LEs provide an estimate for the attractor dimension through the Kaplan–Yorke dimension[39,40]

$$D_{KY}=k+\frac{\sum_{i=1}^{k}\lambda_i}{|\lambda_{k+1}|}, \tag{13}$$

with $\sum_{i=1}^{k}\lambda_i>0$ and $\sum_{i=1}^{k+1}\lambda_i<0$.

B. Covariant Lyapunov vectors

Because LEs are scalars, they cannot describe the geometry of the tangent space. For this, we need to find a suitable basis that spans the tangent space. The GSVs $\boldsymbol{Q}(t)$ provide an orthogonal basis of the tangent space, which is not time-reversible because of the frequent orthogonalizations. On the other hand, there exists coordinate-independent, local decomposition of the phase space into subspaces spanned by the covariant Lyapunov vectors (CLVs).[41,42] Unlike the GSVs, which are orthonormal by construction, the subspaces spanned by the CLVs are generally non-orthogonal to each other, which provide information on the local geometric structure of the chaotic attractor. If the CLVs $\boldsymbol{V}=\left[\boldsymbol{v}_1,\boldsymbol{v}_2,\ldots,\boldsymbol{v}_k\right]$ are uniquely defined (i.e., nondegenerate), then each CLV $\boldsymbol{v}_i\in\mathbb{R}^D$ describes the individual expansion and contraction associated with the LE λ_i. The CLVs can be recovered from the local growth rates $\boldsymbol{R}(t)$ and the GSVs $\boldsymbol{Q}(t)$ by evolving backward in time after the forward-in-time simulation is completed. We provide a brief overview of the computation of the CLVs in Appendix A 2; for further details on the computation, we refer the reader to Refs. 6, 18, and 43.

Two key pieces of information are contained in the CLVs. First, CLVs provide information on the hyperbolicity of the chaotic attractor.[44] An attractor is hyperbolic if there is splitting of tangent space at every point of the trajectory $\boldsymbol{x}(t)$ into three subspaces: unstable $E^U_{\boldsymbol{x}}$, neutral $E^N_{\boldsymbol{x}}$, and stable $E^S_{\boldsymbol{x}}$. These subspaces contain CLVs associated with positive, zero, and negative LEs, respectively. This splitting has profound implications for the geometric structure of the attractor and in hyperbolic systems when there are no tangencies between these subspaces.[3] Therefore, the distribution of the angles between these subspaces is bounded away from zero, which is a property that can be analyzed with the CLVs. Second, CLVs provide information on the most important directions in phase space, which is useful for reduced-order modeling. CLVs can split the tangent space of spatially extended dissipative systems into a decomposition of a fixed, finite number of "physical" modes and a remaining set of "spurious" modes.[7] The physical modes contain the relevant dynamics of the phase space and define a finite-dimensional manifold in the tangent space, while the spurious modes correspond to the negative LEs and increase in number with increasing resolution. These spurious modes are hyperbolically decoupled from the physical modes because perturbations along them decay quickly and do not propagate to the physical modes. A way to characterize the transition between physical and spurious modes is by analyzing the statistics of the CLV angles and locating a clear absence of tangencies in pairs of adjacent CLVs.[7] The number of physical modes provides a lower bound for the degrees of freedom in the simulation.

C. Jacobian of the LSTM

We perform stability analysis on the dynamical system defined by the LSTM. Key to the analysis is the Jacobian of the system; therefore, the Jacobian of the LSTM is required. To compute the Jacobian, we express in a compact form the LSTM equations (2) during a closed-loop as

$$[\boldsymbol{c}_{i+1}, \boldsymbol{h}_{i+1}]^T = LSTM\left(\hat{\boldsymbol{x}}(t_i), \boldsymbol{c}_i, \boldsymbol{h}_i\right), \tag{14}$$

$$\hat{\boldsymbol{x}}(t_{i+1}) = \boldsymbol{W}^{dense}\boldsymbol{h}_{i+1} + \boldsymbol{b}^{dense}. \tag{15}$$

The Jacobian is the gradient of the internal states at a single time step,

$$\boldsymbol{J}_{LSTM}(\boldsymbol{c}_i, \boldsymbol{h}_i) = \begin{bmatrix} \frac{\partial \boldsymbol{c}_i}{\partial \boldsymbol{c}_{i-1}} & \frac{\partial \boldsymbol{c}_i}{\partial \boldsymbol{h}_{i-1}} \\ \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{c}_{i-1}} & \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{h}_{i-1}} \end{bmatrix}, \tag{16}$$

which analytically is

$$\begin{aligned} \frac{\partial \boldsymbol{c}_i}{\partial \boldsymbol{c}_{i-1}} &= \mathbf{I} * \boldsymbol{f}_i, \\ \frac{\partial \boldsymbol{c}_i}{\partial \boldsymbol{h}_{i-1}} &= \boldsymbol{c}_i * \boldsymbol{f}_i * (\mathbf{I} - \boldsymbol{f}_i)\boldsymbol{W}^f + \boldsymbol{i}_i * (\mathbf{I} - \boldsymbol{i}_i)\boldsymbol{W}^i * \tilde{\boldsymbol{c}}_i + \boldsymbol{i}_i * \left(\mathbf{I} - \tilde{\boldsymbol{C}}_i^2\right)\boldsymbol{W}^g, \\ \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{c}_{i-1}} &= \left(\mathbf{I} - \tanh^2(\boldsymbol{c}_i)\right) * \boldsymbol{o}_i * f_i, \\ \frac{\partial \boldsymbol{h}_i}{\partial \boldsymbol{h}_{i-1}} &= \boldsymbol{o}_i * (\mathbf{I} - \boldsymbol{o}_i) * \tanh(\boldsymbol{c}_i) + \left(\mathbf{I} - \tanh^2(\boldsymbol{c}_i)\right) * \boldsymbol{o}_i * \\ &\quad \times \left(\boldsymbol{c}_i * \boldsymbol{f}_i * (\mathbf{I} - \boldsymbol{f}_i)\boldsymbol{W}^f + \boldsymbol{i}_i * (\mathbf{I} - \boldsymbol{i}_i)\boldsymbol{W}^i * \tilde{\boldsymbol{c}}_i \right. \\ &\quad \left. + \boldsymbol{i}_i * \left(\mathbf{I} - \tilde{\boldsymbol{C}}_i^2\right)\boldsymbol{W}^g\right), \end{aligned} \tag{17}$$

where $\mathbf{I} \in \mathbb{R}^{N_h \times N_h}$ is the identity matrix and $\boldsymbol{J}_{LSTM} \in \mathbb{R}^{(N_h+N_h)\times(N_h+N_h)}$. In chaotic systems, the Jacobian matrix varies at each time step. To analyze the stability from partial observations, the idea is to analyze the LSTM Jacobian for the computation of the LEs and CLVs. With this, we can compute the stability properties in the network in a closed-loop mode. From Eq. (17), we expect a maximum of $2N_h > D$ Lyapunov exponents, of which the first D exponents are physically relevant, whereas the remaining are spurious (i.e., they depend on the network's architecture). We provide a pseudocode for the calculation in Algorithm 1 in Appendix A 1.

V. RESULTS

The objective is to assess the capabilities of the two LSTM architectures under investigation in the forecasting of both observed and unobserved (hidden) chaotic time series. To study the forecast performance of the proposed LSTMs, we employ the prediction horizon during the closed-loop configuration,

$$\frac{\left\|\boldsymbol{x}\left(t_{N_{PH}}\right) - \hat{\boldsymbol{x}}\left(t_{N_{PH}}\right)\right\|}{\sqrt{\frac{1}{N_{PH}}\sum_{i=0}^{N_{PH}}\|\boldsymbol{x}(t_i)\|^2}} < k, \tag{18}$$

with $k = 0.5$ as in Ref. 11. Starting from the same initial conditions, the prediction horizon quantifies how far in the future the LSTM can follow autonomously the reference chaotic solution. We then perform a long autonomous temporal evolution of the trained PI-LSTM and LH-LSTM networks and compare the statistics with the target chaotic dynamical systems. In Sec. V A, we fix the number of unobserved (hidden) variables in the example of the Kuramoto–Sivashinsky equation. After reconstructing the full state, we compute the statistics and spectrum of the kinetic energy, as well as key stability properties, such as the Lyapunov exponents, and the angles between selected CLVs. In Sec. V B, we employ the Lorenz 96 equation and test the capabilities of the LSTM networks when a different number of hidden variables is considered.

A. Kuramoto–Sivashinsky

The Kuramoto–Sivashinsky equation, also known as the KS equation, was first derived in Ref. 21 to describe diffusion-induced space-time chaos in reaction systems. In Ref. 45, the equation was independently derived to model small diffusive-thermal instabilities of laminar flame plane fronts. The KS equation is a fourth-order nonlinear partial differential equation that can be written in one spatial dimension as

$$\boldsymbol{\phi}_t(t,\mathrm{x}) + \boldsymbol{\phi}_{\mathrm{xx}}(t,\mathrm{x}) + \boldsymbol{\phi}_{\mathrm{xxxx}}(t,\mathrm{x}) + \boldsymbol{\phi}(t,\mathrm{x})\boldsymbol{\phi}_{\mathrm{x}}(t,\mathrm{x}) = 0, \tag{19}$$

with $\mathrm{x} \in [0, L)$ being the spatial direction. By assuming periodic boundary conditions $\boldsymbol{\phi}(t,0) = \boldsymbol{\phi}(t,L)$ and setting $L = 2\pi \cdot 10$, the solution of the KS equation displays chaotic behavior[22,46] with $\lambda_1 = 0.08$. We spatially discretize Eq. (19) with 128 degrees of freedom, corresponding to $\Delta \mathrm{x} = L/128$, and select a time step $\Delta t = 0.25$. The equation is then solved with a fourth-order spectral scheme for stiff PDEs[19,47] up to $T = 2.5 \times 10^4$, resulting in 10^5 samples. To eliminate transients, we disregard the initial 1000 samples

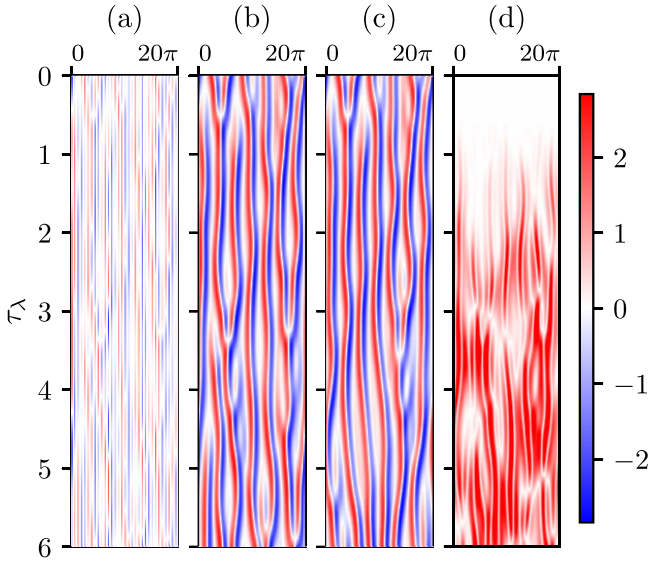

FIG. 4. Comparison of the test data's (a) input, (b) label/reference solution, (c) LH-LSTM's closed-loop prediction, and (d) the absolute error between the target and LH-LSTM prediction.

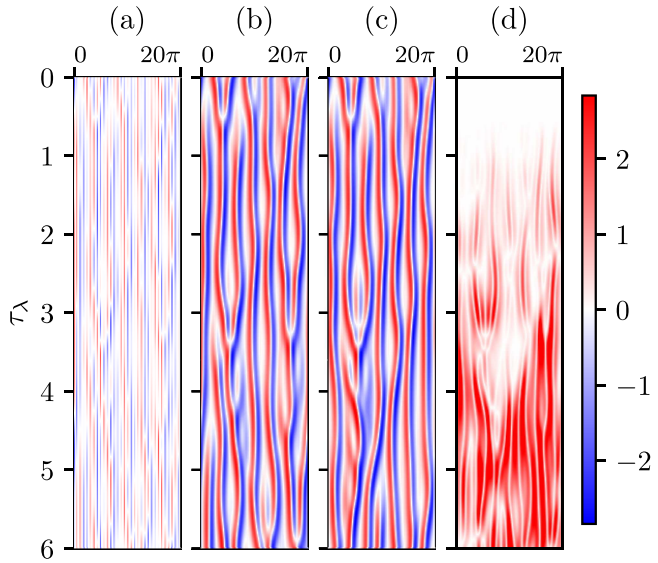

FIG. 5. Comparison of the test data's (a) input, (b) label/reference solution, (c) PI-LSTM's closed-loop prediction, and (d) the absolute error between the target and PI-LSTM prediction.

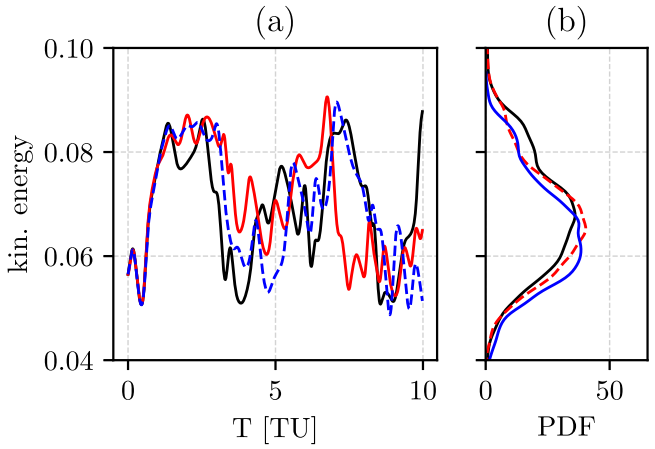

FIG. 6. Kinetic energy of the Kuramoto–Sivashinsky system: Comparison of the target (black line), PI-LSTM (red line), and LH-LSTM (blue line) for (a) short time span on the test data and (b) the statistics over $500\tau_\lambda$.

and partition the remaining data into training, validation, and testing sets with sizes 5×10^4, 2×10^4, and 5×10^4, respectively. The tangent space calculation follows Ref. 19, and the LEs and the CLVs are computed, resulting in a Kaplan–Yorke dimension equal to $D_{KY} = 15.02$. We analyze the distribution of angles between the CLVs and detect $j = 29$ physical modes for the selected parameters in the present study; hence, the modes $j \geq 30$ are spurious.

In the following, we present the results with $D_y = 32$ observed inputs, and the network is trained to predict the full state with $D = 128$. We select the window size in $\{10, 20, 50\}$ and the cell and hidden state dimensions N_h in $\{200, 500\}$ with a grid search; see Table I in Appendix A 3 for details. We do not observe significant differences in the network performance for different choices of window sizes and the cell and hidden state dimensions. For the physics-informed LSTM, the weighing α_{pi} is selected from $\{100, 10, 1\}$. We present the closed-loop prediction of the two LSTMs over $6\tau_\lambda$, with the LH-LSTM in Fig. 4 and the PI-LSTM in Fig. 5. Both networks accurately predict the KS solution over a short time horizon, with a prediction horizon of $2.6\tau_\lambda$ for the data-driven model and $2.86\tau_\lambda$ for the physics-informed model.

We investigate the long-term physical behavior of the network prediction by examining the kinetic energy $E(t) = \frac{1}{2L}\int dx|\phi(t,x)|^2$ of the reference solution and the networks' predictions in Fig. 6. Figure 6(a) shows the kinetic energy of the first $10\tau_\lambda$ of the test set, with an overlap of the energy during the prediction horizon. After the prediction horizon, the kinetic energies gradually separate, due to the chaos, but the LSTMs' closed-loop predictions have correct statistical behavior in the kinetic energy, suggesting that networks produce long-term physical solutions [Fig. 6(b)].

To investigate this further, we present in Fig. 7 the time-averaged energy spectrum, defined as $E(k) = \frac{1}{T}\int dt|\phi(t,k)|^2$, which is the distribution of energy at different wavenumbers k. High wavenumbers correspond to small-scales, which are difficult to model accurately during the numerical computation, as they involve a large range of spatial and temporal scales. Despite passing only partial spatial observations as an input, both LSTM models accurately capture the energy of the majority of wavenumbers, showing that the models accurately extrapolate to smaller scales.

Although the energy spectrum is a global quantity to analyze the distribution of energy across different scales, stability analysis provides a detailed tool to understand the chaotic and tangent dynamics of the system. In Fig. 8, we compare the reference LEs (in black squares) with the LEs extracted from the LH-LSTM (red circles) and the PI-LSTM (blue crosses), computed with Algorithm 1 and hyperparameters in Table III. (As in Ref. 11, the network does not capture two zero LEs (λ_9 and λ_{10}) and the plot is augmented, accordingly.) By reconstructing the full state, the networks effectively reconstruct the tangent space, the properties of which are encapsulated in the LEs. Both networks infer the first 29 LEs with high accuracy, with an error of 3.7% (LH-LSTM) and 0.2% (PI-LSTM) in λ_1. This means that the machine has inferred the correct physical modes based on the approach in Ref. 7. The PI-LSTM case can also infer the hidden variables without fully labeled data, which provides further evidence of the physical accuracy and correct dynamics.

We examine the principal angles between the dominant CLVs of the unstable subspace E_x^U, the neutral subspace E_x^N, and the stable subspace E_x^S in Fig. 9, where

$$\theta_{a,b} = \frac{180^\circ}{\pi}\cos^{-1}(|\boldsymbol{v}_a * \boldsymbol{v}_a|), \tag{20}$$

$\theta_{a,b} \in [0^\circ, 90^\circ]$. In Fig. 9, the agreement of the angle statistics between reference and the LSTMs is within a negligible numerical error, which reflects the robust and accurate learning of the ergodic properties from the partial input data. Further discussion on the CLVs is provided in Appendix A 5 and Fig. 17.

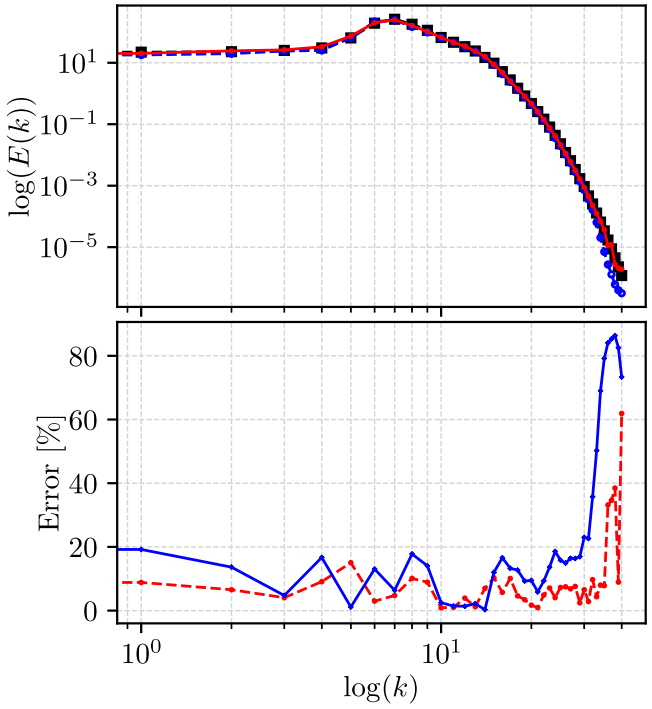

FIG. 7. Energy spectrum of the Kuramoto–Sivashinsky system: Comparison of the target (black line), PI-LSTM (red line), and LH-LSTM (blue line). The distribution shows the energy transfer across different scales, with high wavenumbers corresponding to small-scale behavior.

B. Lorenz-96

The Lorenz-96 model is a reduced-order model to describe the large-scale behavior of the mid-latitude atmosphere.[20] It consists of a set of coupled ordinary differential equations, which represent the variation of an atmospheric quantity of interest, such as temperature

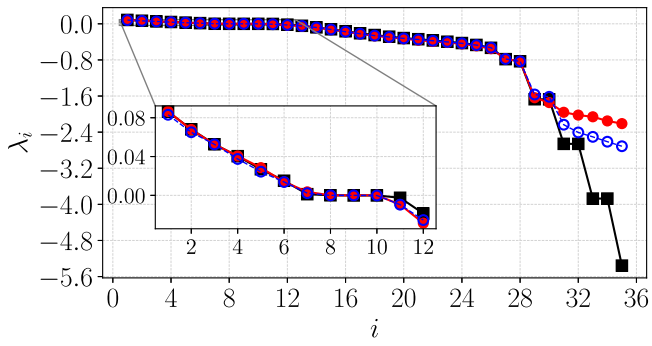

FIG. 8. The first 35 Lyapunov exponents of the Kuramoto–Sivashinsky system for the reference data (black squares), PI-LSTM (red dots), and LH-LSTM (blue crosses). Both networks infer correctly the first 30 Lyapunov exponents, which correspond to the physical behavior (i.e., physical modes[7]) of the system.

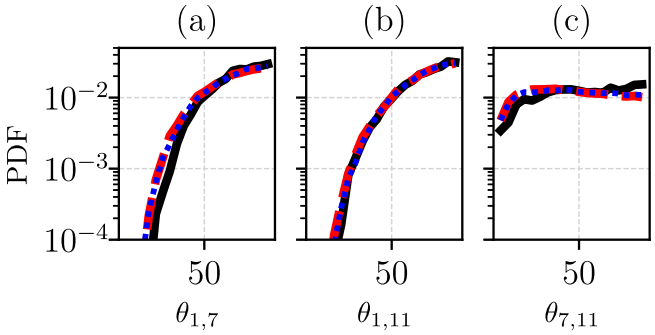

FIG. 9. The angle distribution of the Kuramoto–Sivashinsky system for the leading covariant Lyapunov vectors of three different subspaces: (a) unstable–neutral, (b) unstable–stable, and (c) neutral–stable. The black line corresponds to the reference data, and the red and blue lines indicate the results obtained from the 10 000τ_λ long autonomous evolution of the PI-LSTM and LH-LSTM models, respectively.

or vorticity, on a periodic lattice representing a latitude circle on the earth

$$\frac{d}{dt}x_i(t) = (x_{i+1}(t) - x_{i-2}(t))\,x_{i-1}(t) - x_i(t) + F, \quad i = 1, \ldots, D, \tag{21}$$

with $\mathbf{x} = [x_1, \ldots, x_D] \in \mathbb{R}^D$. The periodic boundary conditions are $x_1(t) = x_{D+1}(t)$. Assuming constant external forcing $F = 8$ and $D = 20$, the system exhibits chaotic behavior with six positive LEs.[48] Both the numerical solution and the reference LEs are computed with the fourth-order Runge–Kutta method with a time step of $\Delta t = 0.01$. The largest Lyapunov exponent is $\lambda_1 \approx 1.55$. The Kaplan–Yorke dimension is equal to $D_{KY} = 13.4$. The training set consists of $N_t = 16\,000$ points, which is equivalent to $250\tau_\lambda$. The hyperparameters can be found in Table II of Appendix A 3. We deploy the PI-LSTM to reconstruct the full state in three test cases: reconstruction of (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ variables. We choose test cases (i) and (ii) to highlight the capabilities of the PI-LSTM and select (iii) to demonstrate how the network performs with significantly fewer measured variables for training than the Kaplan–Yorke dimension. When the full state is available for training, both LH-LSTM and PI-LSTM perform equally well in predicting the long-term statistics and the LEs in a closed-loop mode, but the PI-LSTM performs well when the training contains only partial observations.

To assess the networks' short-term forecasting capabilities, we compute the mean and standard deviation of the prediction horizon for 100 time windows sampled from the test set. These time windows are used as a warm-up input to the network, after which it evolves autonomously. Figure 10 shows that in all three cases, the PI-LSTM achieves a larger mean prediction horizon and, therefore, predicts the correct solution accurately for longer.

The statistical reconstruction is based on an autonomous $1000\tau_\lambda$ long trajectory in a closed-loop mode. In Fig. 11, we show the statistics of the hidden variables, i.e., those variables that are

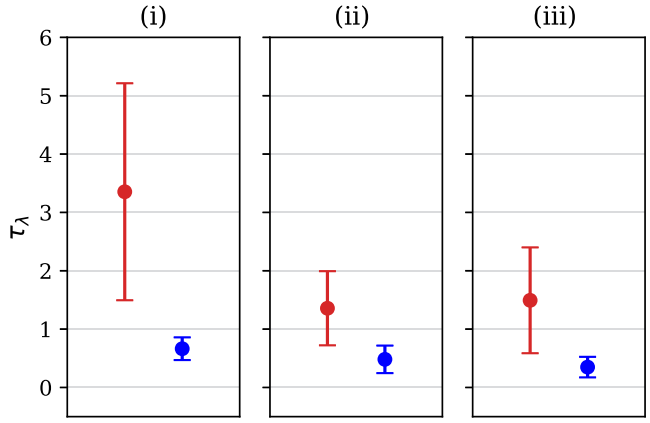

FIG. 10. Mean and standard deviation of the prediction horizon for PI-LSTM (red line) and LH-LSTM (blue line) for 100 predictions in the test set with (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ missing variables in a closed-loop configuration.

not used in the training. Despite training with a full label, the LH-LSTM fails to predict the solution. In particular, the corresponding delta-like or uniform distributions (in blue) indicate fixed-point or periodic behavior. However, the PI-LSTM (in red) provides a markedly more accurate statistical reconstruction of the target compared to the LH-LSTM, with an average Wasserstein distance[49] of (i) 0.05, (ii) 0.15, and (iii) 0.31.

In Fig. 12, we compare the reference LEs (in black squares) with LEs extracted from the data-driven LSTM (in blue circles) and PI-LSTM (in red circles) in the three test cases, computed with Algorithm 1 and hyperparameters in Table IV. By reconstructing the variables, the networks effectively reconstruct the tangent space, the properties of which are encapsulated in the LEs. In all cases, the LEs of the data-driven LSTM deviate significantly from the reference LEs, differing more from the target when fewer observations are available. In test case (i), the PI-LSTM correctly infers the LEs, with six positive LEs. In case (ii), the PI-LSTM infers five positive

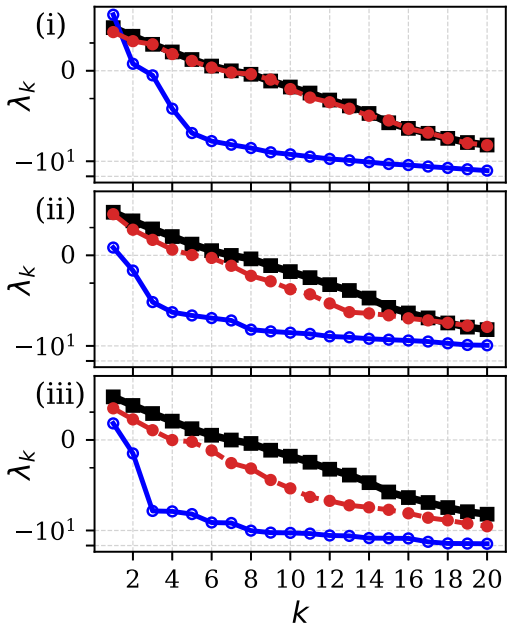

FIG. 12. Lorenz-96: Comparison of the target (black squares), PI-LSTM (red dots), and LH-LSTM (blue dots) LEs for (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ unobserved (hidden) variables.

LEs, decreasing to three positive LEs for test case (iii). These findings indicate that the Kaplan–Yorke dimension of the target chaotic system (here, $D_{KY} = 13.4$) plays an important role in the reconstruction capabilities of PI-LSTM: as a practical rule of thumb, at least $M \gtrsim D_{KY}$ independent time series are required for a sufficient reconstruction of the full chaotic attractor such that accurate LE statistics are extracted. This criterion is met in cases (i) and (ii), as well as in the example of the KS equation in Sec. V A. Still, the PI-LSTM can sufficiently reconstruct the hidden variables statistics, as seen in Fig. 11(iii), when the number of available time series is less than D_{KY}.

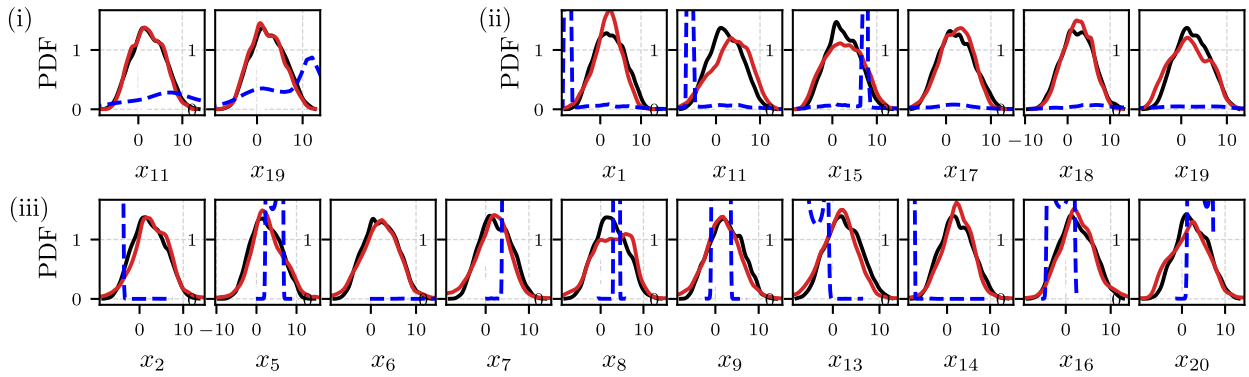

FIG. 11. Lorenz-96: Statistics of reconstructed variables. Comparison of the target (black line), PI-LSTM (red line), and LH-LSTM (blue line) probability density functions (PDFs) of (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ variables over a $1000\tau_\lambda$ trajectory in a closed-loop configuration.

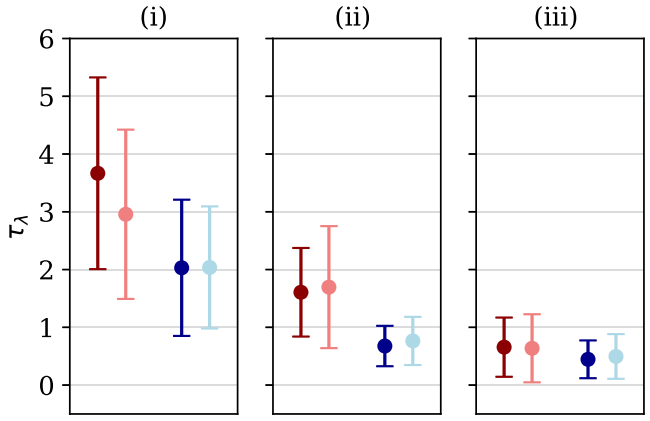

FIG. 13. Mean and standard deviation of the prediction horizon for PI-LSTM (red line) and LH-LSTM (blue line) for 100 predictions in the test set with (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ missing variables and two different noise levels, $k_n = 0.1$ (dark shade) and $k_n = 0.2$ (light shade).

1. Robustness against noise

To evaluate the robustness of the networks, we examine the impact of noise on the training data to mimic experimental measurements, which may be affected by aleatoric uncertainty. Gaussian-distributed noise, sampled from $\mathcal{N}(0, k_n\sigma_{\boldsymbol{x}})$, is added onto the training data, where $\sigma_{\boldsymbol{x}}$ is the standard deviation of the data and k_n is the noise level. Subsequently, the LH-LSTM and PI-LSTM models are tested on a time window of noisy data, and their prediction horizons are computed with respect to noise-free data. In Fig. 13, we show the mean and standard deviation of short-term predictions made by the networks trained with two different noise levels, $k_n = 0.1$ and $k_n = 0.2$. Despite the introduction of additional noise, the models' prediction horizons remain largely consistent, which signifies that the method is robust. The PI-LSTM consistently outperforms the LH-LSTM in all three test cases, highlighting the advantages of utilizing physics-informed regularization for a noisy input. A similar observation can be made with respect to the Lyapunov spectra, as shown in Fig. 14, for which the PI-LSTM Lyapunov spectra are accurate. The probability density functions of the time series can be found in Figs. 15 and 16 of Appendix A 4.

2. Discussion on the traditional methods

As a benchmark, we reviewed a selection of methods for estimation of the Lyapunov spectrum based on time-delay embeddings.[50–52] Extracting LEs from time series from high dimensional chaotic attractors is a significant numerical challenge, and it is estimated[53] that the number of required data points, N_d, grows exponentially with the attractor dimension D_{KY} as $N_d \sim \text{const}^{D_{KY}}$, where const $\sim \mathcal{O}(10)$ encapsulates the necessary parameters of those methods. As demonstrated in Ref. 54 for Lorenz-96 at $D = 6$ with $D_{KY} = 4.2$, additional care should be given to the chosen parameters, requiring further experimentation. Based on these estimates, $N_d \geq 10^{14}$ samples are required for an adequate estimation of at least the leading Lyapunov exponent in the case of Lorenz-96 at $D = 20$ used here. However, our approach achieves an accurate estimate of a large portion of the Lyapunov spectrum with significantly fewer data points [$\mathcal{O}(10^5)$].

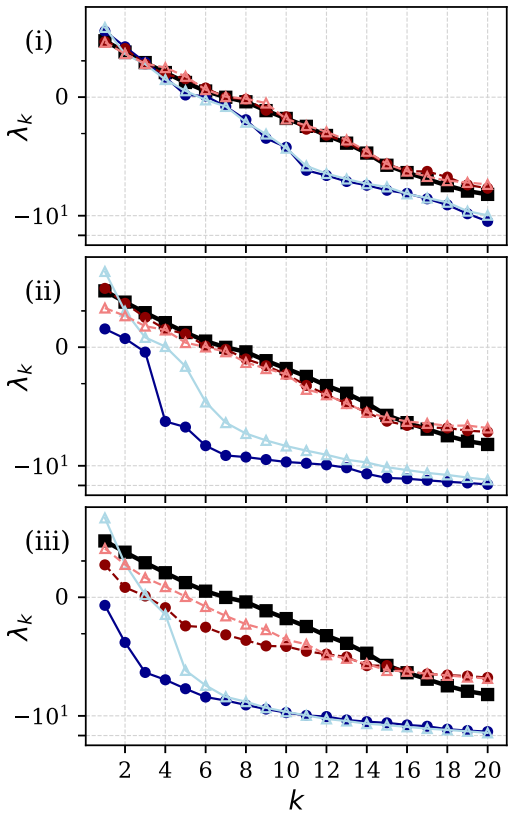

FIG. 14. Lorenz-96: Comparison of the target (black squares), PI-LSTM (red), and LH-LSTM (blue) LEs for (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ unobserved (hidden) variables for two different noise levels, $k_n = 0.1$ (dark shade) and $k_n = 0.2$ (light shade).

VI. CONCLUSION

Because of limitations with sensors, experiments typically provide information on only part of a dynamical system. With only partial observations available, equation-based methods may struggle to infer the full state. This is a particularly challenging task in chaotic systems, which are the focus of this paper because infinitesimal perturbations exponentially grow in time. We propose data-driven methods to infer the dynamics of unobserved (hidden) chaotic variables—a task that is referred to as full-state reconstruction, time forecast the evolution of the full state after the inference of the hidden variables, and compute the stability properties of the reconstructed full state. The tasks are performed from observations (data) limited to only part of the state with long short-term memory networks (LSTMs), which are versatile gated recurrent neural networks for sequential data (such as time series). First, we analyze and propose architectures. The first architecture is the low-to-high resolution LSTM (LH-LSTM), which takes partial observations as

a training input, but it requires access to the full system state (full labeled dataset). In order not to rely on the full labeled data set, we design a physics-informed LSTM (PI-LSTM), which combines data with the integral evolution equations. This allows for the use of existing numerical integration methods, such as pseudospectral time-stepping schemes. The proposed reformulation is beneficial when automatic differentiation with respect to time is not readily available and is, therefore, not limited to recurrent neural networks. Second, we mathematically derive the Jacobian of the LSTM. The Jacobian is key to computing the stability properties of the chaotic attractor, such as Lyapunov exponents and covariant Lyapunov vectors. Third, we test both the LH-LSTM and the PI-LSTM on the chaotic Kuramoto–Sivashinsky system. Our results demonstrate that both approaches correctly perform short-term predictions and energy-spectrum inference on unseen scenarios in a closed-loop mode. The machines correctly learn the covariant Lyapunov vectors (measured through the angles) and infer the Lyapunov spectrum of the attractor. Fourth, we analyze the Lorenz-96 system. Using a purely data-driven method with the LH-LSTM leads to markedly inaccurate long-term statistics and unphysical Lyapunov exponents. On the other hand, the proposed PI-LSTM is able to infer long-term statistics and chaotic properties. Fifth, the PI-LSTM outperforms the LH-LSTM by successfully reconstructing the hidden chaotic dynamics when the input dimension is smaller than the Kaplan–Yorke dimension of the attractor. This is because the missing information on the attractor is indirectly embedded in the equations. This work opens new opportunities for inferring hidden variables and computing the stability of dynamical systems by combining prior knowledge of the equations and data. Current work is focused on transferring the methods of this paper to experimental data and larger dynamical systems.

ACKNOWLEDGMENTS

Elise Özalp thanks Alberto Racca for fruitful discussions on chaos and recurrent neural networks. This research has received financial support from the ERC Starting Grant No. PhyCo 949388.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Elise Özalp: Data curation (lead); Formal analysis (equal); Investigation (equal); Methodology (equal); Validation (lead); Writing – original draft (lead); Writing – review & editing (supporting). **Georgios Margazoglou:** Conceptualization (supporting); Formal analysis (equal); Investigation (equal); Methodology (equal); Supervision (supporting); Writing – original draft (supporting); Writing – review & editing (supporting). **Luca Magri:** Conceptualization (lead); Formal analysis (equal); Methodology (equal); Project administration (lead); Resources (lead); Supervision (lead); Validation (equal); Writing – original draft (equal); Writing – review & editing (lead).

ALGORITHM 1. Algorithm to compute the Lyapunov exponents of an LSTM.

Input: Start state x_t,
Number of time steps to compute Lyapunov times N^{lyap},
Number of transient initial steps to skip for warm-up N^{w},
Number of steps until QR decomposition is performed again N^{norm}
Initialize:
$U \leftarrow random \in \mathbb{R}^{(N_h+N_h)\times d}$ ▷ Initialize d Gram–Schmidt vectors
$Q, R \leftarrow QR(U)$ ▷ Orthonormalize GSVs
$U \leftarrow Q$
$N^{QR} \leftarrow (N^{lyap} - N^{w})/N^{Norm}$
$\lambda \leftarrow 0 \in \mathbb{R}^{d\times N^{QR}}$
Evolve the LSTM with its cell and hidden state and GSV simultaneously for N^{w} steps.
for $i = N^{w} : N^{lyap}$ **do**
  $x(t_{i+1}), c_{i+1}, h_{i+1} = LSTM(x(t_i), c_i, h_i)$ ▷ Next LSTM step
  $J \leftarrow Jac_{LSTM}(c_{i+1}, h_{i+1})$ ▷Update Jacobian
  $U \leftarrow JU$ ▷ The variational equation
  if $mod(i, N^{norm}) = 0$ **then** ▷ Orthonormalize every N^{norm} steps
    $Q, R \leftarrow QR(U)$
    $U \leftarrow Q$
    $\lambda[:, i/N^{norm}] \leftarrow log(diag(R))/dt$
  end if
end for
Output: $\lambda_j \leftarrow \sum_{i=0}^{N_{QR}} \lambda[j, i]/T_{lyap}$ ▷jth Lyapunov exponent

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

APPENDIX: LYAPUNOV ANALYSIS AND HYPERPARAMETERS

1. Computation of Lyapunov exponents

2. Computation of covariant Lyapunov vectors

Let $\boldsymbol{M}(t, \Delta t) = \exp\left(\int_t^{t+\Delta t} \boldsymbol{J}(x(\tau))d\tau\right)$ (the exponential should be considered as a path-ordered exponential[55]) be the system's tangent evolution operator. Each bounded non-zero CLV $\mathbf{v}_i$ is evolved by the tangent dynamics $\boldsymbol{J}(x(t))$,[3]

$$\frac{d\mathbf{v}_i}{dt} = \boldsymbol{J}(x(t))\mathbf{v}_i - \lambda_i \mathbf{v}_i, \tag{A1}$$

where the extra term $-\lambda_i \mathbf{v}_i$ ensures that the norm is bounded. The vectors $\mathbf{v}_i$ are covariant, meaning the ith CLV at time t_1, $\mathbf{v}_i(t_1)$, maps to the ith CLV at time t_2, $\mathbf{v}_i(t_2)$ and vice versa. This translates to $\boldsymbol{M}(t, \Delta t)\mathbf{v}_i(t) = \mathbf{v}_i(t + \Delta t)$, and time-invariance follows directly as $\boldsymbol{M}(t, -\Delta t)\mathbf{v}_i(t + \Delta t) = \mathbf{v}_i(t)$. The CLVs can be recovered from the local growth rates $\boldsymbol{R}(t)$ and the GSVs, $\boldsymbol{Q}(t)$, by evolving backward in time after the forward-in-time simulation is completed. We provide a brief overview on the computation of the CLVs; for further details on the computation, we refer the reader to Refs. 6 and 18. Recall that the GSV $\boldsymbol{Q}(t)$ and growth rates $\boldsymbol{R}(t)$ are computed by solving (1) and (11) simultaneously with periodic orthonormalization. After Δt, the GSV evolution is given by

$$\boldsymbol{M}(t, \Delta t)\boldsymbol{Q}(t) = \boldsymbol{Q}(t + \Delta t)\boldsymbol{R}(t, \Delta t), \tag{A2}$$

and the corresponding CLVs $\boldsymbol{V}(t)$ can be written as

$$\boldsymbol{V}(t) = \boldsymbol{Q}(t)\boldsymbol{C}(t), \tag{A3}$$

where $\boldsymbol{C}(t)$ is an upper triangular matrix containing the CLV expansion coefficients. From the CLV evolution equation

$$\boldsymbol{M}(t, \Delta t)\boldsymbol{V}(t) = \boldsymbol{V}(t + \Delta t)\mathbf{D}(t, \Delta t), \tag{A4}$$

we replace $\boldsymbol{V}(t)$ using (A3) and then rearrange. It then follows that the CLV expansion coefficients can be computed by inverting the upper triangular matrix $\boldsymbol{R}(t, \Delta t)$ as

$$\boldsymbol{C}(t) = \boldsymbol{R}^{-1}(t, \Delta t)\boldsymbol{C}(t + \Delta t)\mathbf{D}(t, \Delta t) \tag{A5}$$

after the forward-in-time computation of the GSVs is completed. This equation is evolved backward in time starting from the end of the forward-in-time simulation. We employ the `solve_triangular` routine of SciPy[56] to invert $\boldsymbol{R}(t, \Delta t)$ and solve with respect to $\boldsymbol{C}(t)$. The diagonal matrix $\mathbf{D}(t, \Delta t)$ contains the CLV local growth factors, similar to $\boldsymbol{R}(t, \Delta t)$. The $\boldsymbol{C}$ and $\mathbf{D}$ matrices are initialized to the identity matrix $\mathbb{I}$. We leave sufficient spin-up and spin-down transient time at the beginning and end of our total time window, before we compute the CLVs via Eq. (A3), to ensure that they are converged.

3. Hyperparameters

All experiments were run on a single NVIDIA Quadro RTX 8000.

TABLE I. Kuramoto–Sivashinsky equation: LSTM hyperparameters.

Hidden and cell state N^h	200, 500
Batch size	128
Physics-informed weighing α_{pi}	1, 10, 100 (PI-LSTM) 0 (LH-LSTM)
Tikhonov regularization	10^{-8}, 10^{-9}
Window size	25
Epochs (early stopping)	2000
Learning rate (Adam optimizer)	0.001

TABLE II. Lorenz-96 system: LSTM hyperparameters.

Hidden and cell state N^h	100, 500
Batch size	128
Physics-informed weighing α_{pi}	1, 10, 100 (PI-LSTM) 0 (LH-LSTM)
Tikhonov regularization	10^{-8}, 10^{-9}
Window size	20, 50
Epochs (early stopping)	2000
Learning rate (Adam optimizer)	0.001

TABLE III. Kuramoto–Sivashinsky equation: Algorithm 1 input parameters (LEs and CLVs).

N^{lyap}	10 000 τ_λ
N^w	$100\tau_\lambda$
N^{norm}	$0.2\tau_\lambda$

TABLE IV. Lorenz-96 system: Algorithm 1 input parameters (LEs).

N^{lyap}	100 τ_λ
N^w	$10\tau_\lambda$
N^{norm}	$0.015\tau_\lambda$

4. Probability density functions of Lorenz-96 for different noise levels

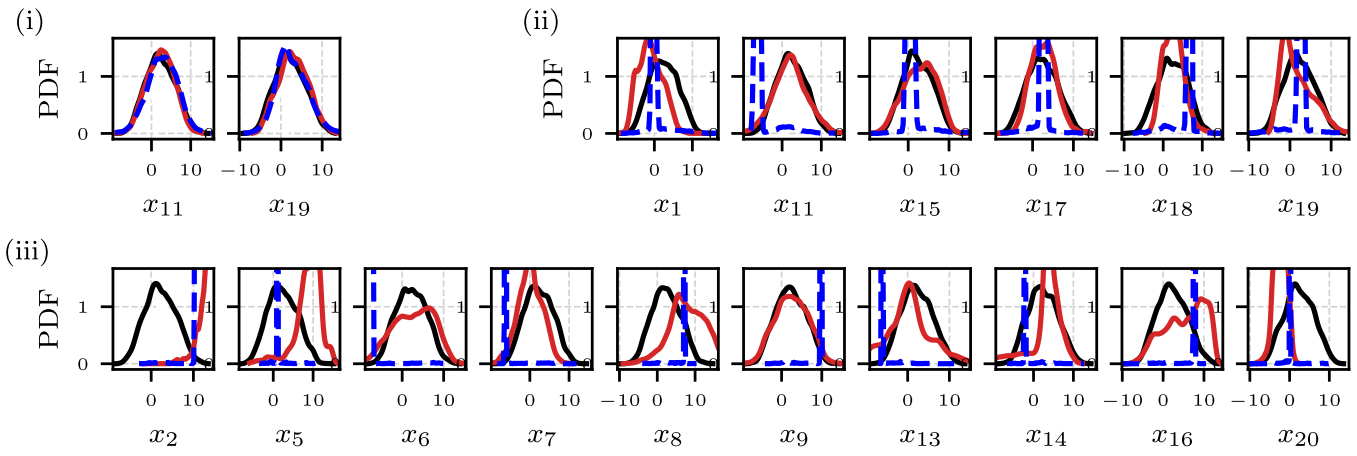

FIG. 15. Lorenz-96: Statistics of reconstructed variables for $k_n = 0.1$. Comparison of the target (black line), PI-LSTM (red line), and LH-LSTM (blue line) probability density functions (PDFs) of (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ variables over a $1000\tau_\lambda$ trajectory in a closed-loop configuration.

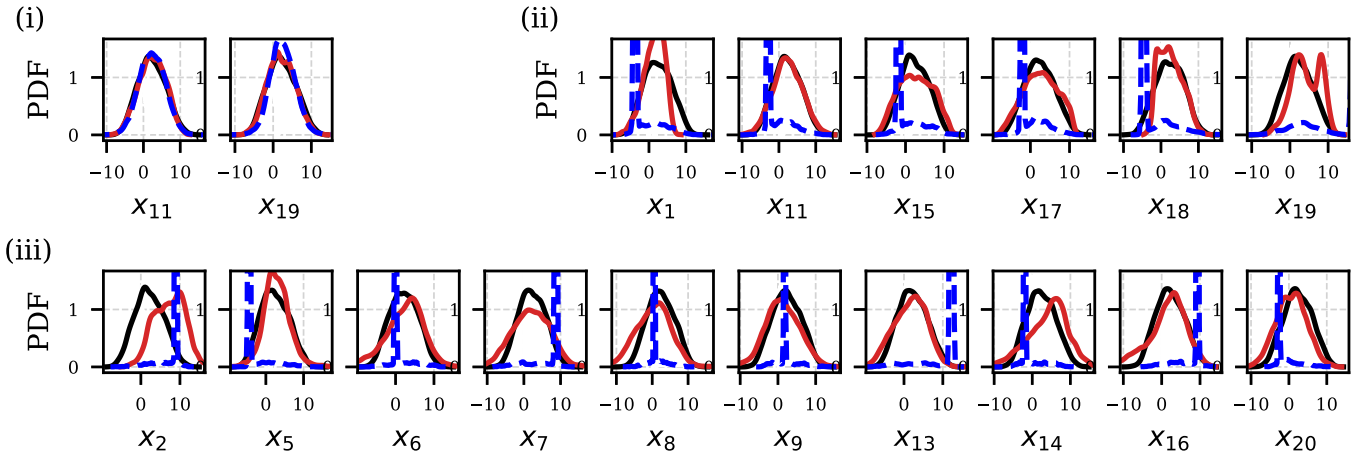

FIG. 16. Lorenz-96: Statistics of reconstructed variables for $k_n = 0.2$. Comparison of the target (black line), PI-LSTM (red line), and LH-LSTM (blue line) probability density functions (PDF) of (i) $D_\xi = 2$, (ii) $D_\xi = 6$, and (iii) $D_\xi = 10$ variables over a $1000\tau_\lambda$ trajectory in a closed-loop configuration.

5. CLVs of the Kuramoto–Sivashinsky equation

In Sec. V A, we have demonstrated that both the LH-LSTM and PI-LSTM accurately infer the angle statistics measured between the leading CLV for the unstable E_x^U, neutral E_x^N, and stable E_x^S subspaces of the Kuramoto–Sivashinksy equation. Specifically, as depicted in Figs. 17(a)–17(c) and 17(e), the angle distributions exhibit a pronounced peak at $\pi/2$ and then rapidly decline near 0 and π. A change of this behavior can be found in Fig. 17(d), in which the angle distribution spans the whole $[0, \pi]$ interval. This shows the transition between physical and spurious modes, as discussed in Ref. 7.

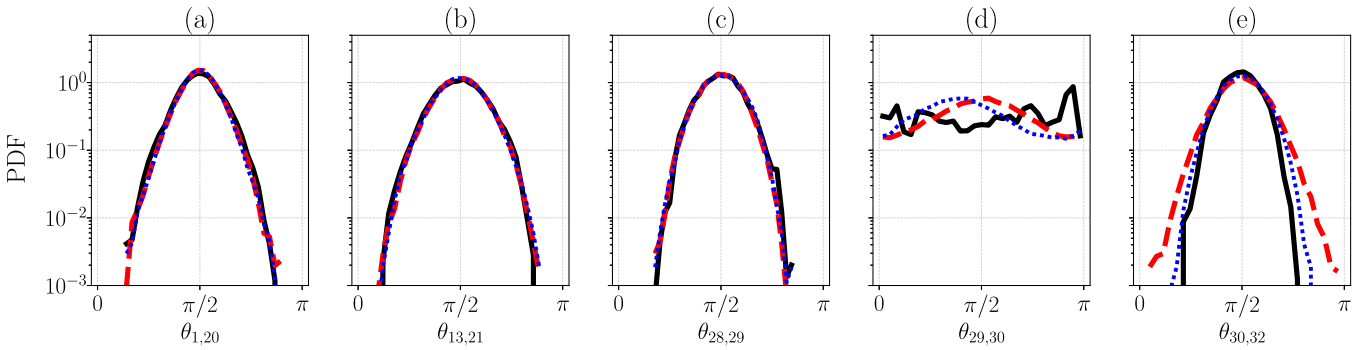

FIG. 17. The angle distribution of the Kuramoto–Sivashinsky system for the CLV pairs: (a) $\mathbf{v}_1, \mathbf{v}_{20}$; (b) $\mathbf{v}_{13}, \mathbf{v}_{21}$; (c) $\mathbf{v}_{28}, \mathbf{v}_{29}$; (d) $\mathbf{v}_{29}, \mathbf{v}_{30}$; and (e) $\mathbf{v}_{30}, \mathbf{v}_{32}$.

REFERENCES

[1] E. N. Lorenz, "Deterministic nonperiodic flow," J. Atmos. Sci. **20**, 130–148 (1963).
[2] S. H. Strogatz, *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering* (Westview Press, 1994).
[3] F. Huhn and L. Magri, "Stability, sensitivity and optimisation of chaotic acoustic oscillations," J. Fluid Mech. **882**, A24 (2020).
[4] L. Magri and N. A. K. Doan, "Physics-informed data-driven prediction of turbulent reacting flows with Lyapunov analysis and sequential data assimilation," in *Data Analysis for Direct Numerical Simulations of Turbulent Combustion: From Equation-Based Analysis to Machine Learning* (Springer, Cham, 2020), pp. 177–196.
[5] F. Takens, ""Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980*, Lecture Notes in Mathematics (Springer, 1981), Vol. 898, pp. 366–381.
[6] F. Ginelli, H. Chaté, R. Livi, and A. Politi, "Covariant Lyapunov vectors," J. Phys. A: Math. Theor. **46**, 254005 (2013).
[7] K. A. Takeuchi, H.-L. Yang, F. Ginelli, G. Radons, and H. Chaté, "Hyperbolic decoupling of tangent space and effective dimension of dissipative systems," Phys. Rev. E **84**, 046214 (2011).
[8] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, "Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks," Proc. R. Soc. A: Math. Phys. Eng. Sci. **474**, 20170844 (2018).
[9] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks," German National Research Institute for Computer Science (GMD) Report No. 148, 2001, pp. 34–49.
[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput. **9**, 1735–1780 (1997).
[11] P. Vlachas, J. Pathak, B. Hunt, T. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, "Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics," Neural Netw. **126**, 191–217 (2020).
[12] A. Racca and L. Magri, "Robust optimization and validation of echo state networks for learning chaotic dynamics," Neural Netw. **142**, 252–268 (2021).
[13] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, "Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach," Phys. Rev. Lett. **120**, 024102 (2018).
[14] S. Herzog, R. S. Zimmermann, J. Abele, S. Luther, and U. Parlitz, "Reconstructing complex cardiac excitation waves from incomplete data using echo state networks and convolutional autoencoders," Front. Appl. Math. Stat. **6**, 616584 (2021).
[15] R. S. Zimmermann and U. Parlitz, "Observing spatio-temporal dynamics of excitable media using reservoir computing," Chaos **28**, 043118 (2018).
[16] S. Ouala, D. Nguyen, L. Drumetz, B. Chapron, A. Pascual, F. Collard, L. Gaultier, and R. Fablet, "Learning latent dynamics for partially observed chaotic systems," Chaos **30**, 103121 (2020).
[17] J. Brajard, A. Carrassi, M. Bocquet, and L. Bertino, "Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: A case study with the Lorenz 96 model," J. Comput. Sci. **44**, 101171 (2020).
[18] G. Margazoglou and L. Magri, "Stability analysis of chaotic systems from data," Nonlinear Dyn. **111**, 8799–8819 (2023).
[19] J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data," Chaos **27**, 121102 (2017).
[20] E. N. Lorenz, "Predictability: A problem partly solved," in *Seminar on Predictability, Shinfield Park, Reading, 4–8 September 1995* (ECMWF, 1996), Vol. 1, pp. 1–18.
[21] Y. Kuramoto, "Diffusion-induced chaos in reaction systems," Prog. Theor. Phys. Suppl. **64**, 346–367 (1978).
[22] J. M. Hyman and B. Nicolaenko, "The Kuramoto–Sivashinsky equation: A bridge between PDE's and dynamical systems," Phys. D: Nonlinear Phenom. **18**, 113–126 (1986).
[23] R. J. Adrian, "Particle-imaging techniques for experimental fluid mechanics," Annu. Rev. Fluid Mech. **23**, 261–304 (1991).
[24] M. Sangiorgio, F. Dercole, and G. Guariso, "Forecasting of noisy chaotic systems with deep neural networks," Chaos, Solitons Fractals **153**, 111570 (2021).
[25] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," Neural Netw. **2**, 359–366 (1989).
[26] A. M. Schäfer and H.-G. Zimmermann, "Recurrent neural networks are universal approximators," Int. J. Neural Syst. **17**, 253–263 (2007).
[27] P. J. Werbos, "Backpropagation through time: What it does and how to do it," Proc. IEEE **78**, 1550–1560 (1990).
[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations* (ICLR, 2015).
[29] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," J. Mach. Learn. Res. **3**, 115–143 (2002).
[30] A. N. Tikhonov, A. Goncharsky, V. V. Stepanov, and A. Yagola, *Numerical Methods for the Approximate Solution of Ill-Posed Problems on Compact Sets* (Springer, 1995).
[31] N. Doan, W. Polifke, and L. Magri, "Short-and long-term predictions of chaotic flows and extreme events: A physics-constrained reservoir computing approach," Proc. R. Soc. A **477**, 20210135 (2021).
[32] E. Özalp, G. Margazoglou, and L. Magri, "Physics-informed long short-term memory for forecasting and reconstruction of chaos," in *Computational Science—ICCS 2023* Lecture Notes, edited by J. Mikyška, C. de Mulatier, M. Paszynski, V. V. Krzhizhanovskaya, J. J. Dongarra, and P. M. Sloot (Springer, 2023), Vol. 10476.

[33]I. Lagaris, A. Likas, and D. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," IEEE Trans. Neural Netw. **9**, 987–1000 (1998).
[34]M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," J. Comput. Phys. **378**, 686–707 (2019).
[35]M. Sandri, "Numerical calculation of Lyapunov exponents," Math. J. **6**, 78–84 (1996).
[36]G. Benettin, L. Galgani, A. Giorgilli, and J.-M. Strelcyn, "Lyapunov characteristic exponents for smooth dynamical systems and for Hamiltonian systems; a method for computing all of them. Part 2: Numerical application," Meccanica **15**, 21–30 (1980).
[37]G. Boffetta, M. Cencini, M. Falcioni, and A. Vulpiani, "Predictability: A way to characterize complexity," Phys. Rep. **356**, 367–474 (2002).
[38]G. Nastac, J. W. Labahn, L. Magri, and M. Ihme, "Lyapunov exponent as a metric for assessing the dynamic content and predictability of large-eddy simulations," Phys. Rev. Fluids **2**, 094606 (2017).
[39]P. Frederickson, J. L. Kaplan, E. D. Yorke, and J. A. Yorke, "The Liapunov dimension of strange attractors," J. Differ. Equ. **49**, 185–207 (1983).
[40]H. Kantz and T. Schreiber, *Nonlinear Time Series Analysis*, 2nd ed. (Cambridge University Press, 2003).
[41]V. Oseledec, "A multiplicative ergodic theorem. Lyapunov characteristic numbers for dynamical systems," Trans. Moscow Math. Soc. **19**, 197–231 (1968).
[42]D. Ruelle, "4Ergodic theory of differentiable dynamical systems," Publ. Math. IHES **50**, 27–58 (1979).
[43]F. Ginelli, P. Poggi, A. Turchi, H. Chaté, R. Livi, and A. Politi, "Characterizing dynamics with covariant Lyapunov vectors," Phys. Rev. Lett. **99**, 130601 (2007).
[44]J. P. Eckmann and D. Ruelle, "Ergodic theory of chaos and strange attractors," Rev. Mod. Phys. **57**, 617–656 (1985).
[45]G. Sivashinsky, "Nonlinear analysis of hydrodynamic instability in laminar flames—I. Derivation of basic equations," Acta Astronaut. **4**, 1177–1206 (1977).
[46]B. Nicolaenko, B. Scheurer, and R. Temam, "Some global dynamical properties of the Kuramoto-Sivashinsky equations: Nonlinear stability and attractors," Phys. D: Nonlinear Phenom. **16**, 155–183 (1985).
[47]A.-K. Kassam and L. N. Trefethen, "Fourth-order time-stepping for stiff PDEs," SIAM J. Sci. Comput. **26**, 1214–1233 (2005).
[48]A. Karimi and M. R. Paul, "Extensive chaos in the Lorenz-96 model," Chaos **20**, 043105 (2010).
[49]S. Vallender, "Calculation of the Wasserstein distance between probability distributions on the line," Theory Probab. Appl. **18**, 784–786 (1974).
[50]A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining Lyapunov exponents from a time series," Phys. D: Nonlinear Phenom. **16**, 285–317 (1985).
[51]M. T. Rosenstein, J. J. Collins, and C. J. De Luca, "A practical method for calculating largest Lyapunov exponents from small data sets," Phys. D: Nonlinear Phenom. **65**, 117–134 (1993).
[52]J.-P. Eckmann, S. O. Kamphorst, D. Ruelle, and S. Ciliberto, "Liapunov exponents from time series," Phys. Rev. A **34**, 4971 (1986).
[53]J.-P. Eckmann and D. Ruelle, "Fundamental limitations for estimating dimensions and Lyapunov exponents in dynamical systems," Phys. D: Nonlinear Phenom. **56**, 185–187 (1992).
[54]U. Parlitz, "Estimating Lyapunov exponents from time series," in *Chaos Detection and Predictability* (Springer, Berlin, 2016), pp. 1–34.
[55]L. Magri, P. J. Schmid, and J. P. Moeck, "Linear flow analysis inspired by mathematical methods from quantum mechanics," Annu. Rev. Fluid Mech. **55**, 541–574 (2023).
[56]P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental algorithms for scientific computing in Python," Nat. Methods **17**, 261–272 (2020).