AUTHOR QUERY FORM

AIP Publishing	**Journal: Chaos** **Article Number: CHA24-AR-01496**	**Please provide your responses and any corrections by annotating this PDF and uploading it to AIP's eProof website as detailed in the Welcome email.**

Dear Author,

Below are the queries associated with your article; please answer all of these queries before sending the proof back to AIP.

Article checklist: In order to ensure greater accuracy, please check the following and make all necessary corrections before returning your proof.
1. Is the title of your article accurate and spelled correctly?
2. Please check affiliations including spelling, completeness, and correct linking to authors.
3. Did you remember to include acknowledgment of funding, if required, and is it accurate?

Location in article	Query / Remark: click on the Q link to navigate to the appropriate spot in the proof. There, insert your comments as a PDF annotation.
Q1	Please check that the author names are in the proper order and spelled correctly. Also, please ensure that each author's given and surnames have been correctly identified (given names are highlighted in red and surnames appear in blue). OK
Q2	In the sentence beginning "As pointed out...," please verify that "previous section" refers to Sec. I. Sec IA
Q3	In the sentence beginning "The same network architecture...," please verify that "previous section" refers to Sec. III A. OK
Q4	Please provide page number for Ref. 19. pp 1-14
Q5	Please provide publisher's name for Refs. 23 and 42. for 23 is Curran Associates, for 42, The International Machine learning society
	Please confirm ORCIDs are accurate. If you wish to add an [illegible] may do so now. For more information on ORCID, see https://orcid.org/. F. Fainstein–0000-0002-3700-0106 G. B. Mindlin–0000-0002-7808-5708 P. Groisman–0009-0006-7970-356X OK
	Please check and confirm the Funder(s) and Grant Reference Number(s) listed: Agencia Nacional de Promoción de la Investigación, el Desarrollo Tecnológico y la Innovación, PICT-2018-00619 Secretaría de Ciencia y Técnica, Universidad de Buenos Aires Consejo Nacional de Investigaciones Científicas y Técnicas Please add any additional funding sources not stated above:

Thank you for your assistance.

Additional comment: "figures should be one-column width"

in line 89 is should say "the Rossler attractor" instead of "a Rossler attractor"
in line 110 it should say "superscripts" inestead of "subscripts"
in 285 it should say "that have no activation functions" instead of "they"
in 369, it should say "for a given footage" instead of "for given footage"
in 389, it should say "... loss function" (not "loss functions")

Reconstructing attractors with autoencoders

Cite as: Chaos 35, 000000 (2024); doi: 10.1063/5.0232584
Submitted: 8 August 2024 · Accepted: 16 December 2024 ·
Published Online: ■■ ■■ 2025

F. Fainstein,[1,2,a)] G. B. Mindlin,[1,2] and P. Groisman[3]

AFFILIATIONS

[1]Departamento de Física, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Ciudad Universitaria, 1428 Buenos Aires, Argentina
[2]CONICET—Universidad de Buenos Aires, Instituto de Física Interdisciplinaria y Aplicada (INFINA), Ciudad Universitaria, 1428 Buenos Aires, Argentina
[3]IMAS-CONICET and Departamento de Matemática, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Ciudad Universitaria, 1428 Buenos Aires, Argentina

[a)]**Author to whom correspondence should be addressed:** ffainstein@df.uba.ar

ABSTRACT

We propose a method based on autoencoders to reconstruct attractors from recorded footage, preserving the topology of the underlying phase space. We provide theoretical support and test the method with (i) footage of the temperature and stream function fields involved in the Lorenz atmospheric convection problem and (ii) a time series obtained by integrating the Rössler equations.

Unsupervised dynamical system model discovery requires extracting the appropriate dynamic variables from data. Two main challenges arise: (i) ensuring that the learned variables preserve the topology of phase space and (ii) dealing with data that involve either a subset of variables or an excessive amount, as in the case of video recordings. Here, we propose a method based on autoencoders to address these challenges. We provide a framework under which we can expect the reconstructed flow to be topologically equivalent to the dynamics underlying the problem being studied.

I. INTRODUCTION

In the natural sciences, many problems can be modeled in terms of dynamic systems. For those problems, the state of the system at each time corresponds to a point in a finite dimensional space, usually called the phase space. The temporal evolution is represented by a trajectory contained on a manifold. Elucidating the dynamics of the problem typically involves reconstructing and characterizing the attractor (which represents the long-time behavior) from the available data.[1–3]

Various approaches have been employed to characterize attractors of complex flows. One such approach is metric-based, relying on the computation of distances between points on the attractor. Within this framework, the calculation of Lyapunov exponents, dimensions, and scaling functions is commonly used.[3–6] These features help quantify the complexity of the flow but are insufficient to validate or refute a proposed dynamical system intended to model it. This limitation arises because topologically distinct systems can produce attractors that share the same metric properties. To address this issue, a topological perspective was incorporated.[2,7] In three-dimensional systems, for example, unstable orbits coexisting with a strange attractor can be reconstructed from data, and their topological organization in phase space (i.e., how they intertwine or form knots) provides a unique "fingerprint" for the data.[8] A model incapable of reproducing the topology of the orbits reconstructed from data can, therefore, be decisively refuted. Thus, the topological structure of an attractor plays a critical role in building confidence in, or rejecting, models. Furthermore, while parameter-dependent dynamical systems may exhibit different strange attractors as parameters change, the topological organization of the unstable orbits coexisting with them remains invariant as long as the orbits exist, offering a highly robust characterization of the flow—its true underlying skeleton.[2,7]

A common challenge arises on how to reconstruct the attractor when we have access to a subset of the variables or, as in a video recording, when we do have access to a large set of measurements, but the relevant variables are given by an unknown nonlinear transformation. Currently, numerous approaches leverage the breakthrough of artificial intelligence to propose algorithms aimed at overcoming this challenge (see, for instance, Refs. 9–17).

In this context, autoencoders, a prominent architecture in the realm of artificial neural networks, which learns a low-dimensional representation of input data,[18] has been a natural and potential venue to address this problem. Some autoencoder-based methods have shown good performance in determining chaotic attractor's dimension,[19–21] similar maximal Lyapunov exponents as the original system,[20,22–24] and dynamical equations have been fitted in the latent space.[10,11,25] A recent method optimized the learning task for quadratic systems,[26] and regularizations have been proposed to also learn projection fibers from data.[27] Autoencoders have also been designed for the analysis of systems governed by partial-differential equations (e.g., Refs. 13, 22, and 28–30).

Despite these advances, the topological validity of autoencoders' low-dimensional representations lacks extensive studies, and it is still under investigation.[23,31,32] In this article, we address this issue. We focus on the case in which the data have been measured by an array of sensors that provide a temporally ordered sequence of frames capturing the spatiotemporal evolution of the system. Simple extensions for other cases are discussed. Our approach differs from previous studies since through a formal mathematical argument, we propose a framework in which the reconstructed flow preserves the topology of the underlying dynamics. This is explicitly demonstrated in the classic Lorenz model for finite-amplitude atmospheric convection[33] and by analyzing the time series of one of the variables of a Rössler attractor.[34]

A. The autoencoder

An autoencoder is a type of artificial neural network specifically designed for dimensional reduction. It comprises two functions, an encoder (E_w) and a decoder (D_w). The encoder maps the input data into a lower-dimensional space, and correspondingly, the decoder reconstructs the input from this encoded representation. The mid-layer of the autoencoder, often referred to as the latent space, serves as a compressed representation of the data, where the most critical information about the input is condensed.

The autoencoder is usually implemented as a feedforward neural network.[35–37] These networks are composed of layers of units ("neurons"), each providing an output that is a nonlinear function of a linear combination of its inputs. The inputs to each layer are the outputs from the preceding layer. Specifically, the output of a layer j having n_j units, denoted as $x^{(j)} \in \mathbb{R}^{n_j}$, is obtained by applying a function $f_j : \mathbb{R}^{n_{j-1}} \to \mathbb{R}^{n_j}$ to the output of the previous layer,

$$x^{(j)} = f_j\left(x^{(j-1)}\right) = \sigma_j\left(W_j x^{(j-1)} + b_j\right). \tag{1}$$

Here, $W_j \in \mathbb{R}^{n_j \times n_{j-1}}$ and $b_j \in \mathbb{R}^{n_j}$ are the layer's trainable parameters, and $\sigma_j(\cdot)$ is the activation function, typically nonlinear, applied element-wise. The autoencoder is often defined by the composition of $N+1$ layers,

$$D_w \circ E_w = f_N^{(D)} \circ \cdots \circ f_{N/2+1}^{(D)} \circ f_{N/2}^{(E)} \circ \cdots \circ f_1^{(E)}. \tag{2}$$

The subscripts indicate which layers belong to the encoder and decoder components, respectively.

Autoencoders can be trained with segments of time traces, sets of arbitrarily chosen features, or, as in our case, snapshots of our spatiotemporal data. The training of a typical autoencoder proceeds as follows. Each frame is a set of d_i numbers (the number of pixels of each frame). This will determine the number of units in the first and final layers ($n_0 = n_N = d_i$). The parameters in the network are chosen in a way that the sum of the squared differences between the network's outputs and inputs, added over the whole dataset, is as small as possible. The interesting aspect of the method is that the number of units in the middle layer $n_{N/2} \doteq d_l < d_i$ is chosen as small as possible. If the compression is pushed too much, there will be loss of information and the decoder will not be able to assign a unique output to each input frame. In other words, the successful recovery of a unique output for each input requires a unique representation of each element of the input in the latent space.

Carrying out this type of training, it was observed both by processing synthetic movies and others from actual experiments that the encoder induces trajectories in the latent space whose topological organization is equivalent to the ones of the original phase space of the problem being filmed.[31,32] The reasons that guarantee this property to persist in general situations remain veiled, and it is the purpose of this article to propose a framework to solve this problem.

II. FORMAL PROBLEM STATEMENT AND PROPOSAL

Let us assume that the dynamics of our phenomenon is described by a dynamical system,

$$\dot{y}(t) = \phi\left(y(t)\right), \qquad y \in \mathscr{M}. \tag{3}$$

We do not have direct access to y, but our data are obtained through the application of a function $\alpha \in C^2(\mathscr{M} \times \mathscr{N}, \mathbb{R}^{d_i})$ such that for every fixed p in the manifold $\mathscr{N}$, $\alpha_p = \alpha(\cdot, p)$ is an embedding. The variable p represents all the additional data that are not relevant for the problem but is saved when recording the movie. We will assume the existence of a function β such that for every p and y, $\beta(\alpha(y,p)) = y$. This means that the video has been recorded in such a way that all the relevant variables can be recovered from the movie frame by frame, and β is precisely the function that does so. For a given fixed p, we have $\beta^{-1} = \alpha_p$. We do not have access neither to α nor β. We just observe $x(t) = \alpha_p(y(t))$ [but not $y(t)$] for a given background p and trajectory $y(t)$. In words, a video recording is related to the underlying dynamical system through an embedding, and the video has been recorded in such a way that all the information not related to the dynamical phenomena can be separated (for instance, a pendulum with a fixed background).

The flow of a vector field ϕ on $\mathscr{M}$ consists of the transformations $g_\phi : \mathscr{M} \to \mathscr{M}$ converting every initial condition y_0 of Eq. (3) at time 0 into the value $g_\phi y_0$ of the solution at time t.[38] The objective of our work is to build a low-dimensional representation of the flow from the movie (that we shall call g_φ), ensuring that it is topologically equivalent to the (unknown) flow (g_ϕ). The proposal will be the representation obtained in the latent space of a specifically trained autoencoder.

Two flows $g_\varphi : \mathscr{M}_1 \to \mathscr{M}_1$ and $g_\Phi : \mathscr{M}_2 \to \mathscr{M}_2$ are topologically equivalent if there is a homeomorphism $h : \mathscr{M}_1 \to \mathscr{M}_2$ mapping orbits of g_φ to orbits of g_Φ, preserving orientation of the orbits.[38] Note that from our assumptions, the movie and the original phenomenon are topologically equivalent so we can work directly with the recorded data. This means that if we want to ensure that $h = E_w$ provides a flow in the latent space topologically equivalent to g_φ, we need to show that (i) the encoder E_w is an homeomorphism and (ii)

the image of the trajectories in the input space is the solution of a dynamical system (ruled by a vector field Φ).

We consider that our data are dense enough; i.e., either our dynamics support a strange attractor so that we are sampling an orbit that is dense or we have samples of several trajectories that fill up $\mathscr{M}$. We will assume that we are in the first case and consider only one orbit, $x_i = x(t_i)$, $t_i \in [0, T]$, with the labels ordered in time (i.e., $t_{i+1} > t_i$ for every i). If we are in the second case, we can proceed similarly but taking several trajectories instead of only one. We will guide the training with the loss function $\mathscr{L} = \lambda_1 \mathscr{L}_1 + \lambda_2 \mathscr{L}_2$, composed of the usual mean squared error,

$$\mathscr{L}_1(w) = \frac{1}{N_T} \sum_i |(D_w \circ E_w)(x_i) - x_i|^2, \tag{4}$$

and a second term given by

$$\mathscr{L}_2(w) = \frac{1}{N_T - 1} \sum_i \left| \frac{(D_w \circ E_w)(x_{i+1}) - (D_w \circ E_w)(x_i)}{t_{i+1} - t_i} - \frac{x_{i+1} - x_i}{t_{i+1} - t_i} \right|^2. \tag{5}$$

Here, w represents the parameters of the network (weights and biases), the sum is over all the training sets (of size N_T), and λ_1 and λ_2 are order one parameters used to balance the relevance we may want to assign to each of the terms. They are interpretable. The first term $\mathscr{L}_1$ forces the network $(D_w \circ E_w)$ to be close to the identity function (i.e., the encoder and the decoder are close to be inverse functions). As pointed out in Sec. I, this prevents different inputs to be mapped to the same point in the latent space. The second term $\mathscr{L}_2$ can be seen as a regularization term that (explicitly) penalizes the lack of smoothness of the minimizer. In fact, it penalizes the lack of regularity, but it does so just in the direction of the flow. Moreover, we are requiring $D_w \circ E_w$ not just to be smooth but also ask the directional derivative to be close to the one of the identity function [see Eq. (8)]. Note that we write the loss function for the discrete dataset case. However, since the second term involves derivatives of the network and the input data, both terms are fundamentally different in the limit as the size of the network and the dataset go to infinity. To clarify the role played by $\mathscr{L}_2$, let us assume, to fix ideas, that the output of the network is one dimensional. Let us also suppose that, for given parameters, the network is defined by the function $(D_w \circ E_w)(x_i) = x_i + \varepsilon \sin(k x_i)$, with k large. If the time steps $t_{i+1} - t_i$ are small, the $\mathscr{L}_1$ loss is of order ε^2, while the $\mathscr{L}_2$ loss is of order $\varepsilon^2 k^2$. Hence, we can have small $\mathscr{L}_1$ but with a large $\mathscr{L}_2$. These parameters are good candidates to minimize $\mathscr{L}_1$, but not $\mathscr{L}_1 + \mathscr{L}_2$.

We assume that the autoencoder generalizes. That means that if through the training procedure we obtain parameters w such that $\mathscr{L}$ is small, then the same will hold for all the input space $\alpha_p(\mathscr{M}) \in \mathbb{R}^{d_1}$ (for a discussion on this mild assumption, see Refs. 39–42). This forces E_w and D_w to be (equi) continuous (i.e., it is possible to obtain a bound for the modulus of continuity independent of the size of the network). For instance, if the network was trained with finite points in a periodic orbit, the ability to generalize allows the correct prediction of intermediate points that were not used in the training process, making the mapping (equi) continuous. This means that, because $\mathscr{L}_1$ is small, E_w becomes an homeomorphism with inverse D_w as the size of the network and the training set goes to infinity.

What is not yet guaranteed is that the trajectories in the latent space are in fact solutions of a dynamical system. The trajectories formed as an image of an homeomorphism $z(t) = E_w(x(t))$ are not generically solutions of a well-behaved vector field. As an example, suppose the homeomorphism $E(x) = x^3$ and the constant vector field $\dot{x} = -1$. The solution of this problem, $x(t, x_0 = 0) = -t$, is mapped to $z(t) = -t^3$. The velocity of this trajectory is given by $\dot{z} = -3t^2$, which is zero at $t = 0$, so nonzero velocities in the domain are mapped to zero velocities in the image. Moreover, the new vector field $\dot{z} = -3z^{2/3}$ has another solution for the initial value problem, given by $z(t, z_0 = 0) = 0$. The image of the mapping is not a dynamical system since uniqueness is lost.

The second term of our loss, $\mathscr{L}_2$, has been added precisely to enforce that we have a dynamical system in the latent space. To see this, we define a vector field in the latent space by taking the directional derivative,

$$\Phi(z) \doteq \frac{\partial E_w}{\partial(\varphi(x))}(x) \approx \frac{\partial E_w}{\partial(\varphi(D_w(z)))}(D_w(z)). \tag{6}$$

Note that this is indeed the velocity field of the mapped trajectories since

$$\dot{z} = \frac{d}{dt} E_w(x(t)) = \frac{E_w(x(t+dt)) - E_w(x(t))}{dt} = \frac{E_w(x(t) + \dot{x}(t)dt) - E_w(x(t))}{dt} = \frac{\partial E_w}{\partial(\varphi(x))}(x) = \Phi(z). \tag{7}$$

The second term (5) states that

$$\frac{\partial(D_w \circ E_w)}{\partial(\varphi(x))}(x) = \frac{\partial D_w}{\partial E_w} \frac{\partial E_w}{\partial(\varphi(x))} = \frac{\partial D_w}{\partial E_w} \Phi(z) \approx \varphi(x). \tag{8}$$

This condition guarantees that nonzero velocities in the space of frames are not mapped into zero velocities in the latent space; that is, $\Phi(z) \neq 0$ if $\varphi(x) \neq 0$. Note that it also penalizes strong spatial variations of Φ in the latent space, as one would expect for a smooth-enough vector field defining a dynamical system. Therefore, in the limit as the size of the network and the training set goes to infinity, the phase flows g_φ and g_Φ are topologically equivalent: the homeomorphism E_w maps (by construction) orbits of $\dot{x} = \varphi(x)$ to orbits of $\dot{z} = \Phi(z)$ preserving orientation of the orbits.

III. NUMERICAL WORK

A. Spatiotemporal data

To numerically test our method, we generate a synthetic movie motivated by the classic model developed by Lorenz for an atmospheric convection problem.[33] Considering a layer of fluid of uniform depth H and aspect ratio a, Lorenz proposes a modal decomposition for the stream function (ψ) and the departure of temperature from the non-convective state (θ). He writes

$$\psi = c_1 X(t) \sin\left(\frac{\pi a x}{H}\right) \sin\left(\frac{\pi z}{H}\right), \tag{9}$$

$$\theta = c_2 Y(t) \cos\left(\frac{\pi a x}{H}\right) \sin\left(\frac{\pi z}{H}\right) - c_3 Z(t) \sin\left(\frac{2\pi z}{H}\right), \tag{10}$$

where X, Y, Z are functions of time alone, and c_1, c_2, and c_3 are constants. Here, x and z denote the spatial coordinates of the problem. The dynamics of the modal amplitudes is determined by the famous dynamical system,

$$\begin{aligned} \dot{X} &= \sigma\,(Y - X), \\ \dot{Y} &= rX - Y - XZ, \\ \dot{Z} &= XY - bZ. \end{aligned} \tag{11}$$

By numerically integrating this system with $\sigma = 10$, $b = 8/3$, and $r = 28$, we generate temporal series with 40 000 points (time step of 0.01). We discard the first 1000 points to avoid the transient state. Setting $a = H = 1$, $c_1 = c_2 = 2c_3 = 1/20$, and a spatial discretization of 0.025, we use these time traces and Eqs. (9) and (10) to generate a sequence of 40×40 pixel frames describing the spatiotemporal pattern for ψ and θ.

To choose the proper training data set for our autoencoder, we note that not every movie is an embedding. For instance, the movie of a moving mass attached to a spring is not. However, if we place a flag attached to the mass, it is since the velocity could be recovered from each frame. In our case, we need information of the stream function and the temperature, so we train an autoencoder with a data set composed of inputs that contain both frames [see Fig. 1(a)]. The first 30 000 frames are used for training and the remaining 9000 frames for testing.

The network architecture consists of initial and final fully connected layers of $d_i = 40 \times 40 \times 2$ units. The input is encoded into the latent space through intermediate fully connected layers with 64, 32, and 16 units. Similarly, the decoder's hidden layers consist of fully connected layers with 16, 32, and 64 units [see Fig. 1(a)]. Sine activation functions are used for all the layers except the latent and output layers, they have no activation functions. The training process is conducted with a batch size of 600 samples and 10^3 epochs. The network is implemented in Python using the PyTorch library (version 2.1.1). The Adam optimization algorithm with learning rate 10^{-4}, $\beta_1 = 0.9$, and $\beta_2 = 0.999$ is used, and scaling factors for the loss function are set to $\lambda_1 = 1$, $\lambda_2 = 1/2$ to ensure that both terms have a similar magnitude. We do not perform any systematic search in the hyperparameter space to optimize the loss on the test set.

The dimensionality of the latent space can be selected based on the network's performance as the number of units in the middle layer is varied. In Fig. 1(b), we present the test dataset loss at the optimal epoch for different numbers of units in the latent layer. Mean and standard deviation computed over 20 trained networks are shown. We observe limited predictive power of the autoencoder when the latent space dimension is one or two. However, with three or more dimensions, the autoencoder achieves a reasonable reconstruction. This indicates that the predictive performance of the autoencoder is closely tied to the dimensionality of the underlying dynamical system. In fact, this analysis allows for the determination of both the latent space dimensionality and the system's dimensionality.

In Fig. 1(c), we show the average evolution, over 20 trained networks and for a three-dimensional latent space architecture, of both terms of the cost function. In Fig. 1(d), the reconstruction obtained in the latent space for one of the networks is shown. We remark that Fig. 1(d) is not the famous attractor given by Lorenz equations but the one reconstructed by the autoencoder.

To make a topological description, we use approximations of the unstable periodic orbits coexisting with the strange attractor. The way in which these orbits are intertwined is a fingerprint of the system. The organization of the orbits can be algebraically described by their relative rotation rates and linking numbers.[8] For two orbits of period one, one of period two, and two of period three, selected from the test set, we compute the relative linking numbers.[31,43,44] In Figs. 1(e) and 1(f), we show the result in the original phase space and in the latent space, respectively. The evolution of the percentage of networks with correct topology during the training epochs is shown on the right axis of Fig. 1(c). All networks learnt the correct topology.

B. Time series data

Neural networks are flexible in processing diverse data types. Our method can be used with any data set that is the image of an embedding. A common example in the natural sciences is an experiment in which only one of the scalar variables of the system can be measured. To illustrate this and test the method, let us consider the Rössler system.[34] The equations read

$$\begin{aligned} \dot{X} &= -Y - Z, \\ \dot{Y} &= X + aY, \\ \dot{Z} &= b + Z\,(X - c). \end{aligned} \tag{12}$$

Setting $a = b = 0.1$, $c = 14$, and a time step of 0.01, we integrate the system for 310 000 steps. To avoid the transient state, we discard the first 1000 data points. We assume that the time series of the variable Y, $\{Y_1, Y_2, \ldots, Y_N\}$, is the one we have access to. We build the data set with instances given by the map $\{x_k\} \doteq \{Y_k, Y_{k+1}, \ldots, Y_{k+L}\}$, with $k = 1, \ldots, N - L$ ($L \ll N$). If the instances are long enough, each one has a unique future and this map is an embedding.[45] We set $L = 150$ [see Fig. 2(a)], and we use the first 70% of the data set to train the network, with the remaining data used for testing.

The same network architecture described in Sec. III A is used, except that now the initial and final layers have $d_i = 150$ units. Training is conducted with a batch size of 2% of the training data set size (4326 instances), for 1500 epochs, using the loss scaling factors $\lambda_1 = 1$ and $\lambda_2 = 20$. The dimensionality of the latent space is chosen to be three based on the performance of the network on the test set when the number of units in the latent layer is changed [Fig. 2(b)]. In Fig. 2(c), we show the evolution of the loss function during the training process for 20 networks.

The encoder generates a flow in the latent space remarkably similar to the one in the original phase space [Fig. 2(d)]. To compare the topology, we extracted approximations for the period one, period two, and period three orbits. Test set data points are used. We show the relative linking numbers of these orbits in the phase space [Fig. 2(e)] and in the latent space of a trained autoencoder [Fig. 2(f)].[31] The result of computing these links during the training process is shown in the right axis of Fig. 2(c). A rapid transition indicates that all properly trained networks learned the correct topology.

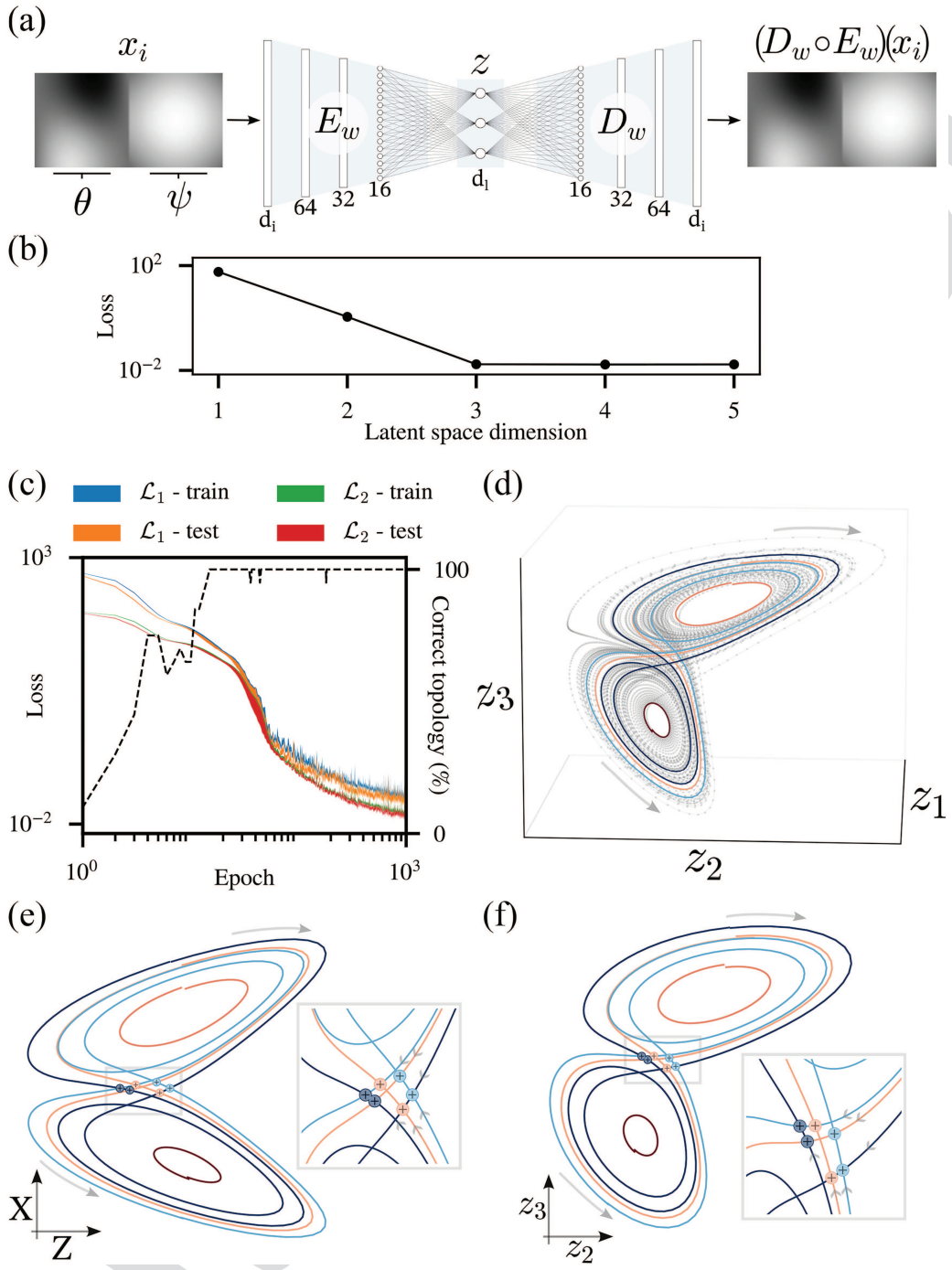

FIG. 1. (a) The autoencoder. Input data (of dimension d_i) are encoded to and decoded from the latent space (of dimension d_l). The values indicate the number of units in each layer of the feedforward artificial neural network. (b) Loss of the test data set for training sessions with different number of units in the middle layer. Mean and standard deviation over 20 trained networks, for each dimension, are shown. (c) Evolution of both terms of our loss function, measured in the train and test datasets, left axis. Mean values over 20 trained networks, with their standard error are shown. Loss values are multiplied by their scaling factors. Evolution of the percentage of autoencoders with correct topology in the latent space as a function of the training epoch is shown in a black dashed line, right axis. (d) Reconstructed flow in the latent space of an autoencoder. Five Lorenz unstable orbits and their linking numbers in phase space (e) and latent space (f). Dot color indicates which orbit is above at the crossing point.

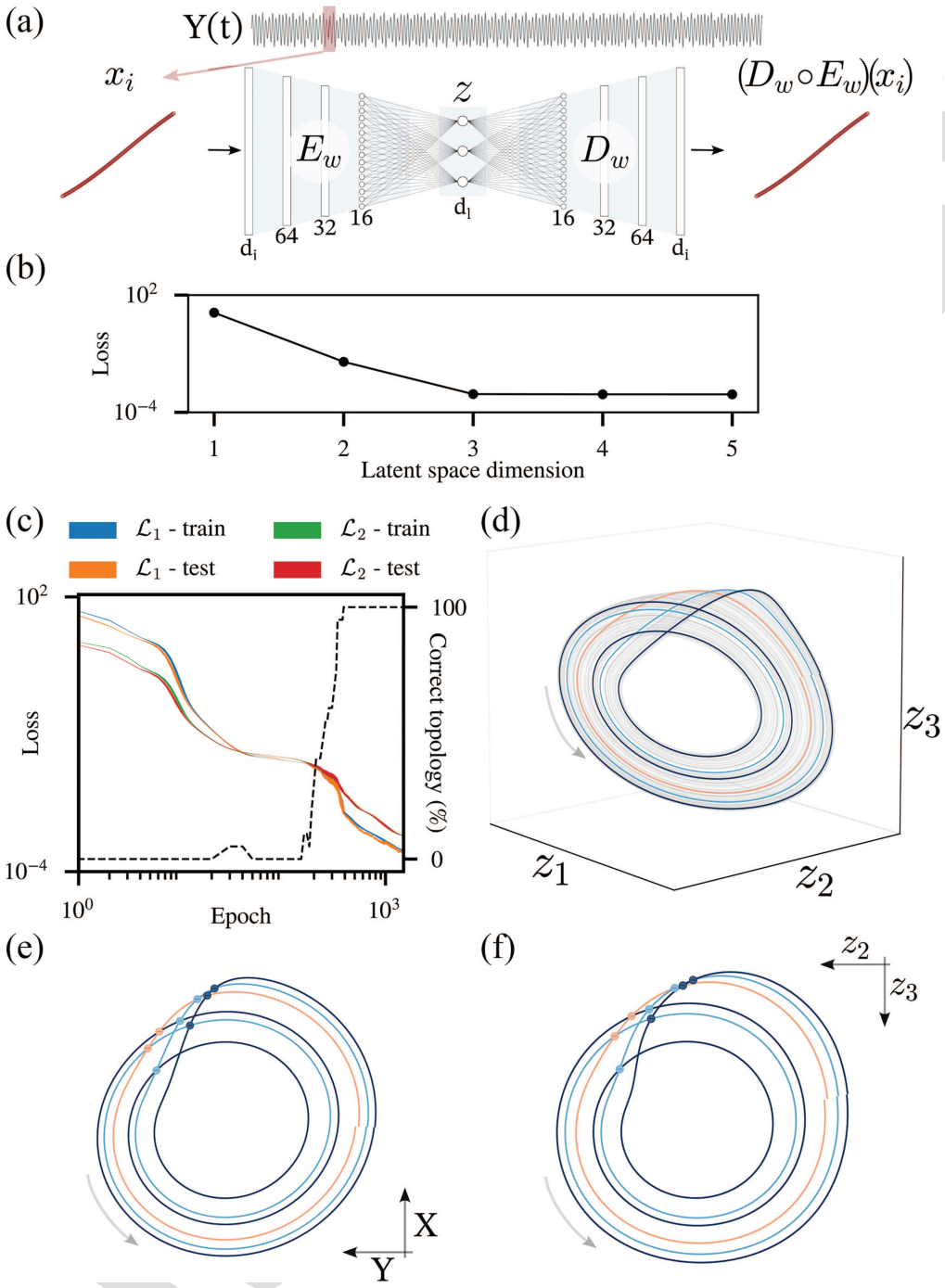

FIG. 2. (a) Segments of $Y(t)$ time series of the Rössler attractor are used as input data for the autoencoder. The values indicate the number of units in each layer of the feedforward artificial neural network. (b) Test data set loss for training sessions with different numbers of units in the middle layer. The mean and standard deviation, computed from 20 networks, are displayed for each dimension. (c) The network's loss values as a function of the training epoch, left axis. Mean values and standard errors over 20 trained networks. Loss terms are scaled by their respective factors. The evolution of the percentage of autoencoders with correct topology in the latent space as a function of the training epoch is shown in a black dashed line, right axis. (d) Reconstructed attractor in the latent space. Three periodic orbits and their linking number in the phase space (e) and the latent space (f). The dot color indicates which orbit is above at the crossing point.

IV. CONCLUSIONS

In this study, we propose a method based on autoencoders to reconstruct the dynamics of a system from recorded footage, preserving the topology of the underlying manifold. The method starts by creating a dataset that constitutes an embedding. If that is not the case, projections in the input space would remain as projections in the latent space. For instance, in our Lorenz example, input frames of the temperature field alone would never lead to a reconstructed flow without self-intersections in a latent space, regardless of its dimensionality.[32] It is worth pointing out that it is not obvious, for an unknown problem, if given footage constitutes an embedding. Our previous remark can be used, in fact, as a warning on the quality of the footage. We also remark that networks should generalize since in that case, (equi) continuity can be expected. With these remarks, the encoder becomes an homeomorphism as the size of the network and the training set goes to infinity.

We showcase a modification to the loss function commonly used to guide the parameter tuning of an autoencoder. The mapping performed by the encoder is not guaranteed to be an embedding of the data: the encoder is not tuned to have a differential that is invertible at any point but to be injective only in the direction of the flow. Our training method guarantees that nonzero velocities in the phase space are not mapped to zero velocities in the latent space. For example, limit cycles cannot be mapped to homoclinic orbits. This term also regularizes the velocity field in the latent space, as would be expected for a smooth-enough vector field defining a dynamical system.

With this setting, the vector field in the latent space gives rise to a flow that is topologically equivalent to the one in the phase space: homeomorphism maps, by construction, orbits to orbits, preserving orientation. Without this modification in the loss functions, we observe in numerical experiments that the autoencoder is able to produce a topologically equivalent dynamical system in the latent space in a significant number of fitting runs.[31,32] This might occur thanks to some kind of inducted bias/implicit regularization,[39–41] but it is not clear yet what kind of regularization might take place in particular runs. By adding the second term, we force the autoencoder to have the regularization we need to ensure topological equivalence.

We have shown that our method can be applied to both time series data and spatiotemporal problems displaying low-dimensional dynamics. In our numerical tests, we found that the reconstructed attractors in the latent space of properly trained autoencoders preserve stringent topological properties of the underlying system. This was validated by examining how unstable periodic orbits, coexisting with a strange attractor, are linked to one another. The topological analysis presented in these numerical experiments is specific to three-dimensional systems. However, we remark that our theoretical arguments do not require the dimension of the latent space to be precise or not larger than three. They hold in any dimension. For higher-dimensional attractors, other topological tools would be required for numerical testing, such as persistent homology analysis. Such an extension will be provided in future publications.

Recently, autoencoders were used to derive dynamical systems that could serve as models for data.[11] The efforts provided algorithms for reconstructing vector fields in the latent space. Our work builds confidence on those efforts; by showing that if properly trained, the flows in the latent space are topologically equivalent to the dynamics responsible for the data.

Over recent years, machine learning techniques have provided tools capable of solving extraordinarily complex problems, including the prediction of dynamics. Despite its apparent lack of necessity to illuminate the underlying mechanisms behind the learned dynamics, this work illustrates how the depth of the dynamics is embedded in some representation of the network. This study aligns with a series of observations made across various network architectures, and it would not be surprising if similar representation mechanisms could be formulated in those cases as well. This research contributes to our understanding of the nuanced interplay between artificial intelligence and dynamic systems, opening avenues for further exploration and application across diverse architectural structures.

ACKNOWLEDGMENTS

We acknowledge M. Fainstein for valuable comments on hardware and software implementation. This work was partially funded by ANPCyT-FONCyT (Argentina) under Grant No. PICT-2018-00619, Universidad de Buenos Aires (UBACyT, Argentina) and Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET, Argentina).

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

F. Fainstein: Data curation (equal); Formal analysis (equal); Investigation (equal); Methodology (equal); Software (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **G. B. Mindlin:** Conceptualization (equal); Data curation (equal); Formal analysis (equal); Funding acquisition (equal); Investigation (equal); Methodology (equal); Project administration (equal); Resources (equal); Software (equal); Supervision (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal). **P. Groisman:** Conceptualization (equal); Data curation (equal); Formal analysis (equal); Funding acquisition (equal); Investigation (equal); Methodology (equal); Project administration (equal); Resources (equal); Software (equal); Supervision (equal); Validation (equal); Visualization (equal); Writing – original draft (equal); Writing – review & editing (equal).

DATA AVAILABILITY

The data that support the findings of this study are available within the article.

REFERENCES

[1] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields* (Springer Science & Business Media, 2013), Vol. 42.

[2]H. G. Solari, M. A. Natiello, and G. B. Mindlin, *Nonlinear Dynamics: A Two-Way Trip from Physics to Math* (Taylor & Francis Group, 1996).
[3]H. D. Abarbanel, R. Brown, J. J. Sidorowich, and L. S. Tsimring, "The analysis of observed chaotic data in physical systems," Rev. Mod. Phys. **65**, 1331 (1993).
[4]P. Grassberger and I. Procaccia, "Characterization of strange attractors," Phys. Rev. Lett. **50**, 346 (1983).
[5]A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining Lyapunov exponents from a time series," Phys. D **16**, 285–317 (1985).
[6]T. C. Halsey, M. H. Jensen, L. P. Kadanoff, I. Procaccia, and B. I. Shraiman, "Fractal measures and their singularities: The characterization of strange sets," Phys. Rev. A **33**, 1141 (1986).
[7]R. Gilmore and M. Lefranc, *The Topology of Chaos: Alice in Stretch and Squeezeland* (John Wiley & Sons, 2012).
[8]G. B. Mindlin, X.-J. Hou, H. G. Solari, R. Gilmore, and N. Tufillaro, "Classification of strange attractors by integers," Phys. Rev. Lett. **64**, 2350 (1990).
[9]Z. Lu, B. R. Hunt, and E. Ott, "Attractor reconstruction by machine learning," Chaos **28**, 061104 (2018).
[10]S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control* (Cambridge University Press, Cambridge, England, 2019).
[11]K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton, "Data-driven discovery of coordinates and governing equations," Proc. Natl. Acad. Sci. U.S.A. **116**, 22445–22451 (2019).
[12]Y. Tang, J. Kurths, W. Lin, E. Ott, and L. Kocarev, "Introduction to Focus Issue: When machine learning meets complex systems: Networks, chaos, and nonlinear dynamics," Chaos **30**, 063151 (2020).
[13]L. Agostini, "Exploration and prediction of fluid dynamical systems using auto-encoder technology," Phys. Fluids **32**, 067103 (2020).
[14]Z. Liu and M. Tegmark, "Machine learning conservation laws from trajectories," Phys. Rev. Lett. **126**, 180604 (2021).
[15]D. Floryan and M. D. Graham, "Data-driven discovery of intrinsic dynamics," Nat. Mach. Intell. **4**, 1113–1120 (2022).
[16]C. D. Young and M. D. Graham, "Deep learning delay coordinate dynamics for chaotic attractors from partial observable data," Phys. Rev. E **107**, 034215 (2023).
[17]C. Pagliantini and S. Jain, "Nonlinear model reduction from equations and data," Chaos **34**, 090401 (2024).
[18]G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science **313**, 504–507 (2006).
[19]W. Chen, H. Sidky, and A. L. Ferguson, "Capabilities and limitations of time-lagged autoencoders for slow mode discovery in dynamical systems," J. Chem. Phys. **151**, ■ (2019).
[20]N. Almazova, G. D. Barmparis, and G. P. Tsironis, "Analysis of chaotic dynamical systems with autoencoders," Chaos **31**, 103109 (2021).
[21]K. Zeng, C. E. P. De Jesús, A. J. Fox, and M. D. Graham, "Autoencoders for discovering manifold dimension and coordinates in data from complex dynamical systems," Mach. Learn.: Sci. Technol. **5**, 025053 (2024).
[22]N. B. Erichson, M. Muehlebach, and M. W. Mahoney, "Physics-informed autoencoders for Lyapunov-stable fluid flow prediction," arXiv:1905.10866 (2019).
[23]W. Gilpin, "Deep reconstruction of strange attractors from time series," in *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)* (■, 2020), pp. 204–216.
[24]J. T. Quinn, "Autoencoder embeddings in dynamical systems: A comparison of current frameworks," master's thesis (University of Colorado at Boulder, 2024).
[25]J. Bakarji, K. Champion, J. Nathan Kutz, and S. L. Brunton, "Discovering governing equations from partial measurements with deep delay autoencoders," Proc. R. Soc. A **479**, 20230422 (2023).
[26]P. Goyal and P. Benner, "Generalized quadratic embeddings for nonlinear dynamics using deep learning," Phys. D **463**, 134158 (2024).
[27]S. E. Otto, G. R. Macchio, and C. W. Rowley, "Learning nonlinear projections for reduced-order modeling of dynamical systems using constrained autoencoders," Chaos **33**, 113130 (2023).
[28]K. Lee and K. T. Carlberg, "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," J. Comput. Phys. **404**, 108973 (2020).
[29]D. Sondak and P. Protopapas, "Learning a reduced basis of dynamical systems using an autoencoder," Phys. Rev. E **104**, 034202 (2021).
[30]M. A. Mendez, A. Ianiro, B. R. Noack, and S. L. Brunton, *Data-Driven Fluid Mechanics: Combining First Principles and Machine Learning* (Cambridge University Press, 2023).
[31]G. Uribarri and G. B. Mindlin, "The structure of reconstructed flows in latent spaces," Chaos **30**, 093109 (2020).
[32]F. Fainstein, J. Catoni, C. P. Elemans, and G. B. Mindlin, "The reconstruction of flows from spatiotemporal data by autoencoders," Chaos, Solitons Fractals **176**, 114115 (2023).
[33]E. N. Lorenz, "Deterministic nonperiodic flow," J. Atmos. Sci. **20**, 130–141 (1963).
[34]O. E. Rössler, "An equation for continuous chaos," Phys. Lett. A **57**, 397–398 (1976).
[35]Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature **521**, 436–444 (2015).
[36]Y. Bengio, I. Goodfellow, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2017), Vol. 1.
[37]F. Chollet, *Deep Learning with Python* (Simon & Schuster, 2021).
[38]V. I. Arnold, *Geometrical Methods in the Theory of Ordinary Differential Equations* (Springer Science & Business Media, 2012), Vol. 250.
[39]M. Belkin, D. Hsu, S. Ma, and S. Mandal, "Reconciling modern machine-learning practice and the classical bias-variance trade-off," Proc. Natl. Acad. Sci. U.S.A. **116**, 15849–15854 (2019).
[40]A. Radhakrishnan, M. Belkin, and C. Uhler, "Overparameterized neural networks implement associative memory," Proc. Natl. Acad. Sci. U.S.A. **117**, 27163 (2020).
[41]M. Belkin, "Fit without fear: Remarkable mathematical phenomena of deep learning through the prism of interpolation," Acta Numer. **30**, 203–248 (2021).
[42]A. Radhakrishnan, K. Yang, M. Belkin, and C. Uhler, "Memorization in overparameterized autoencoders," in *Deep Phenomena Workshop, International Conference on Machine Learning* (■, 2019).
[43]J. S. Birman and R. F. Williams, "Knotted periodic orbits in dynamical systems I: Lorenz's equations," Topology **22**, 47–82 (1983).
[44]For software implementation to compute linking numbers, see https://github.com/gon-uri/linking_number.
[45]F. Takens, "Detecting strange attractors in turbulence," in *Dynamical Systems and Turbulence, Warwick 1980* (Springer, 2006), pp. 366–381.