Piecewise integrable neural network: An interpretable chaos identification framework

Piecewise integrable neural network: an interpretable chaos identification framework

Nico Novelli*,[1] Pierpaolo Belardinelli,[1] and Stefano Lenci[1]
Department of Construction, Civil Engineering and Architecture
Polytechnic University of Marche, Ancona 60131, Italy

(*Electronic mail: n.novelli@pm.univpm.it)

(Dated: 28 February 2023)

Artificial Neural Networks (ANNs) are an effective data-driven approach to model chaotic dynamics. Although ANNs are universal approximators which easily incorporate mathematical structure, physical information and constrains, they are scarcely interpretable. Here we develop a neural network framework in which the chaotic dynamics is reframed into piecewise models. The discontinuous formulation defines switching laws representative of the bifurcations mechanisms, providing to recover the system of differential equations and its primitive (or integral) which describe the chaotic regime.

Interpretation of chaotic dynamics from data is an important task in engineering[1], meteorology[2] and other fields of science. Among various aspects of measure and description of chaotic dynamics, much interest is posed in the integrability[3], i.e. "... integrable systems ... a formula could be found for all time describing a system's future state." as stated by Moore[4]. Studying and proving the existence of a chaotic attractor from a data-drive perspective is a demanding regression problem in which machine learning techniques offer a valuable assistance. The synthesis of a piecewise-smooth modeling through neural networks gives a valid and rigorous reconstruction of the dynamical structure and bifurcations with the option of providing an integrable representation. In this work, a minimal mathematically biased artificial neural network is introduced to extract analytically tractable piecewise-smooth dynamical systems that directly account for the integrability condition.

NOTATION

- $\sigma : \mathbb{R} \to \mathbb{R} :=$ a fixed generic continuous function
- $d, D \in \mathbb{Z}^+$
- $NN^{\sigma}_{d,D} :=$ the set of all FFNNs from $\mathbb{R}^d \to \mathbb{R}^D$
- $PCNN^{\sigma}_{d,D} :=$ the set of all PCNNs from $\mathbb{R}^d \to \mathbb{R}^D$
- $\sigma_{sigmoid}(x) := \dfrac{e^x}{1+e^x}$
- $\#A :=$ the cardinality of a set A
- $\mathbf{X} \in \mathbb{R}^d$: input space
- $\dot{\mathbf{X}} \in \mathbb{R}^D$: output space
- $I_K :=$ Indicator function of the generic subset $K \in \mathbf{X}$ (i.e. I_K is a function such that , $I_K = 1$ if $\mathbf{x} \in K$, and $I_K = 0$ otherwise)
- $\xi_i :=$ coordinates transformation representing the Koopman eigenfunction of the i-th sub-pattern
- $\psi_i := \xi_i^{-1}$
- Integrable system := a formula of the primitive ($\mathbf{x}(t) = \mathbf{F}(t)$) of the studied system of differential equation ($\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t)$) could be found for all time describing a future state of the system.

I. INTRODUCTION AND BACKGROUND

Machine Learning (ML) techniques are tools for clustering, classification and regression problems. In this realm, artificial neural networks (ANNs) become prominent by providing extreme flexibility for a variety of problems[5,6]. They easily incorporate mathematical structure and physical constraints, outperforming other ML techniques, such as support vector machine and dynamic mode decomposition, in term of accuracy and performance in the field of big data[7].

ANNs are universal function approximators[8–10]. Given sufficient training data, a minimal (wide and/or deep enough) ANN architecture guarantees the approximation of a function of choice with a desired precision. ANNs have been proved to be Turing complete, i.e. they could be used to solve any computation problem[11]. Yet, an approximation does not suffice to fathom the underlying physics behind the data. Analysis of dynamical system demands interpretability even at the cost of architectural flexibility. A suitable approach is to specify the architecture by adding regularization terms in the optimization problem[12–14]. For instance, a mere ANN is unaware of what lies behind chaos or strange attractors, in which exponentially separating trajectories bounded by finite energy repeatedly stretch and fold into complicated self-similar fractals. Although a structured ANN architecture succeeds to capture the dynamics, how do we efficiently constrain the framework to maintain interpretability? Structured approximations include assimilation of graph structure into the learning model[15,16], manifold-valued neural networks[17–19], encoded symmetries into the trained model[20,21], or encode inevitability[22–25]. When trying to describe the state of a system and its dynamics[26], the

chimera is a meaningful description with an architecture that does not lose its generality as approximator while requiring minimal knowledge of the studied physical system. A further, yet useful, request for chaotic systems is integrability, as it allows to analytically analyze the attractors and predict the future state of the system without the need of numerical methods[27,28]. This is possible by using specific classes of piecewise models[28] to construct an hybrid dynamics composed of multiple manifolds.

In this paper we employ a piecewise continuous structure (PCNNs hereafter) characterized by a randomized and parallelizable training meta-algorithm[29]. This PCNN framework uses integrable ANNs, which permits to interpret each manifold model (as these ANNs can be described by far fewer parameters), while managing efficiently the switch between different vector fields. The interpretability constraint is linked to piecewise nature of the approach, which by subdividing the dynamics of the system into sub-parts (manifolds) allows to simplify the description of every recovered manifold. The architecture is more universal than a common feedforward neural networks (FFNNs) where continuous activation functions are limited in their interpretability and approximation capabilities[29]. A classical FFNNs with continuous function σ successfully applies to a phenomenon governed by some continuous target function f, managing the worst-case approximation of error to arbitrary precision. However, if f is discontinuous, as is for instance in many multi-stables systems, or whenever a bifurcation or an impact dynamics occurs[30], the uniform limit theorem from classical topology[31] guarantees that the worst-case approximation error of f by FFNNs is not secured[32–34]. In this case there exists a bounded portion of the (non-empty) input space $\mathbf{X} \subseteq \mathbb{R}^d$ (where d describes a general input dimension) where the approximation becomes arbitrarily poor.

In a general piecewise continuous function $f : \mathbf{X} \to \mathbb{R}^D$ (where D describes a general output dimension), built by N sub-patterns f_n,

$$f = \sum_{n=1}^{N} I_{K_n} f_n, \quad f_n : \mathbb{R}^d \to \mathbb{R}^D \text{ continuous function}, \tag{1}$$

the poor esteem is precisely at the boundaries of the regions $\{K_n\}_{n=1}^N$, that are some non-empty compact sub-sets over which the $\{f_n\}_{n=1}^N$ are smooth. A workaround for uniform approximation is deep neural networks[35] with discontinuous activation functions. However, they do not comply with common used (stochastic) gradient descent-type algorithms. Nevertheless, a few methods for training such discontinuous models have been suggested, e.g. a heuristic approach[36], but its empirical performance and theoretical guarantees, i.e. guarantees about the convergence of the algorithm, have not been explored. A linear programming approach to train shallow feedforward networks uses threshold activation functions with fixed hidden weights and biases[37]. This avoids back-propagating through the non-differentiable threshold activation functions, yet it constructs a too specific method for shallow architectures. Similarly, an extreme learning machine approach randomizes all but the final linear network layer[38]. As a result, the training task is reduced to that of a classical linear regression problem. Approximation results are strictly weaker (i.e. a larger and deeper ANN architecture is require to reach the same accuracy) than the known guarantees for classical feedforward networks with a continuous activation function[39].

The PCNN framework implemented in this paper follows the structure of Eq. (1), parameterized by

- the sub-patters $\{f_n\}_{n=1}^N$ by independent FFNNs
- the parts $\{K_n\}_{n=1}^N$ by zero-sets of FFNNs

The two independent parts are combined via a single discontinuous unit. Each FFNN component is independent, they are only regrouped at their final outputs by the discontinuous unit as illustrated in Fig. 1 (subplot (a)). This structure decouples the training of each PCNN model component and re-combine them at the end to produce predictions. The decoupled training procedure avoids passing any gradients through the discontinuous unit, and it allows scalability via parallelizzation of the training process. This architecture possess the approximation capabilities of the discontinuous models (guaranteed by the universal approximation theorems for piecewise continuous functions[29]) while being pragmatically trainable.

In what follows, we will make use of PCNNs to regress integrable chaotic dynamical models from data, to provide a meaningful and mathematical tractable way (in a close form) study of chaotic attractors and bifurcations.

II. FORMULATION OF THE PIECEWISE MODEL

The general form of a dynamical system which can exhibit chaos reads

$$\dot{\mathbf{X}} = \mathscr{N}(\mathbf{X}, t, \Theta, \Omega) \tag{2}$$

in which $\mathbf{X}, \dot{\mathbf{X}}$ are the state of the system and its time derivative, respectively, $\mathscr{N}$ is a general nonlinear operator, t is the time variable, Θ represents all the parameters which describe the system and Ω represents the stochastic effects. In order to find a discontinuous representation, the dynamics is to be split in N manifolds. As a matter of fact, attractors can be described in terms of bounding tori and be organized around the fixed points surrounded by the flow[40]. One possible approach is to use the fixed points of the system to reference the clustering of the phase space, it reduces the cardinality of clusters and it prevents overfitting. Around fixed points the nonlinear dynamics simplifies, which it means to look for easier models to reconstruct the behaviour. Moreover, bifurcation of fixed points, e.g. saddle, could be seen as switching events (or mechanisms) that separate and link domains of influence of other fixed points determining when and how the trajectory passes the transition. The switching can be described by:

- an hypersurface which performs as codimension of the bifurcation;
- a domain associated to the respective bifurcation point (in this case the hypersurfaces that define the boundary of each subsystems play the role of connector mechanisms and not of a real bifurcation event).

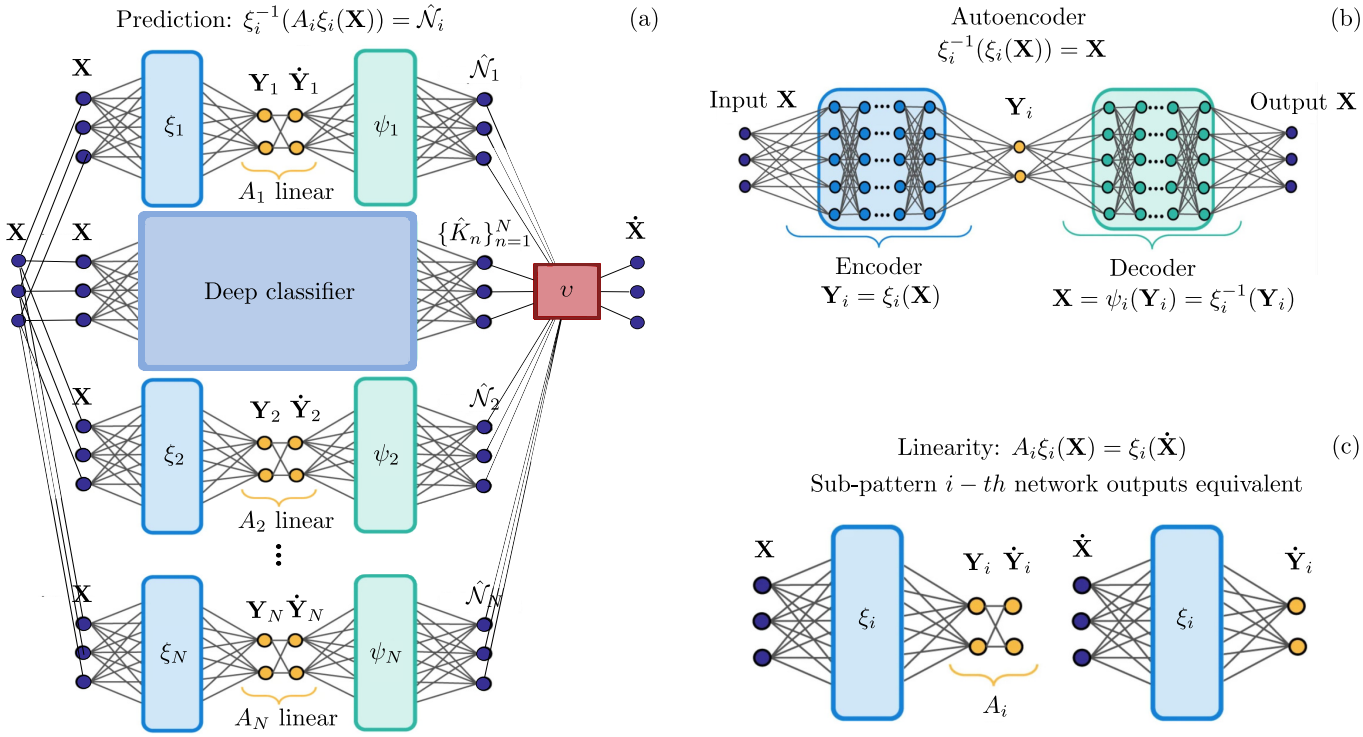

FIG. 1. PCNN architecture showing in (a) a PCNN $\hat{\mathcal{N}}$ approximating the nonlinear map $\mathcal{N}:\mathbf{X}\to\dot{\mathbf{X}}$ with N sub-patterns $\hat{\mathcal{N}}_1$, $\hat{\mathcal{N}}_2$, …, $\hat{\mathcal{N}}_N$ and N deep zero sets $\{\hat{K}_n\}_{n=1}^N$ and accordingly, υ is the discontinuous unit. Each $i-th$ sub-pattern is approximate by a FFNN $\hat{\mathcal{N}}$ to identify Koopman eigenfunctions ξ_i, which is described in (b) by a deep auto-encoder, able to identify intrinsic coordinates $\mathbf{y}=\xi_i(\mathbf{x})$ and decode these coordinates to recover $\mathbf{x}=\xi_i^{-1}(\mathbf{y})$. In (a), (c) are described additional loss function terms, relatively to each $i-th$ sub-pattern, to identify a linear Koopman model A_i that projects the intrinsic variables space $\mathbf{Y}$ into $\dot{\mathbf{Y}}$. In practice, we enforce agreement with the trajectory data. In (a), the loss function in each sub-pattern is evaluated on the state variable space $\mathbf{X}$ and in (c) it is evaluated on $\mathbf{Y}$.

Therefore, the evolution of the system in the phase space is interpreted as a continuously switching among influence domains of fixed points. Hence, we rewrite the Eq. (2) as

$$\dot{\mathbf{X}} = \sum_{i=1}^{N} f_i[s(\mathbf{X})]\mathcal{N}_i(\mathbf{X}) \tag{3}$$

$$f_i[s(\mathbf{X})] = \begin{cases} 1 & \text{if } s(\mathbf{X}) \geq 0 \\ 0 & \text{if } s(\mathbf{X}) < 0 \end{cases} \tag{4}$$

in which the explicit dependency of the system from t, Θ and Ω is omitted. The number of subsystems is related to the number of fixed points and when building the piecewise model we additionally require the integrability condition. The problem reduces in determining:

- fixed points (location and cardinality)
- the #N of integrable subsystem (space partitions)
- dynamic integrable maps $\mathcal{N}_i$ that characterize each $i-th$ manifold
- switching hypersurface $s(\mathbf{X})$
- f_i, i-th Boolean functions designed in order that, at any time, only one subsystem is active.

In order to comply with the integrability condition, a coordinates transformation is needed. We perform a linear embedding of the dynamics of the $i-th$ manifold following the Koopman theory[41]. Equation (3) is recasted by introducing the ξ_i and ψ_i coordinates transformation(see Fig. 1 subplot (b)) , representing respectively the Koopman eigenfunction of the i-th sub-pattern and its inverse ($\psi_i = \xi_i^{-1} \quad \forall i \in \{1,2,\dots N\}$), so that

$$\dot{\mathbf{X}} = \sum_{i=1}^{N} f_i[s(\mathbf{X})]\frac{d\psi_i[\xi_i(\mathbf{X})]}{dt}\frac{d\xi_i(\mathbf{X})}{dt}. \tag{5}$$

That leads to a transformed state space $\mathbf{Y}$ where the dynamics is linear and described as

$$\dot{\mathbf{Y}} = \sum_{i=1}^{N} f_i[s(\mathbf{X})]A_i\xi_i(\mathbf{X}) \tag{6}$$

in which A_i is a constant matrix (representative of the Jacobian matrix evaluated at the fixed point p_i) defining the linear dynamics of the $i-th$ subsystem in the transformed state space,

$$\mathbf{Y} = \sum_{i=1}^{N} f_i[s(\mathbf{X})]\mathbf{Y}_i. \tag{7}$$

The linear representation (in the space of $\mathbf{Y}$, with $\mathbf{Y}_i = \xi_i(\mathbf{X})$) hides the information of the position of the center of each submodel. The integrability condition for the N subsystems is

guaranteed by the existence of the ψ_i (given that each manifold is linear in its domain of $\mathbf{Y}$). In this case an exact solution for Eqs. (5)-(6) is

$$\mathbf{X}(t) = \sum_{i=1}^{N} \left\{ f_i[s(\mathbf{X})] \psi_i [\sum_{j=1}^{M} b_{ij}\phi_{ij} \exp(\lambda_{ij} t)] \right\} \tag{8}$$

$$\mathbf{Y}(t) = \sum_{i=1}^{N} \left\{ f_i[s(\mathbf{X})] \sum_{j=1}^{M} b_{ij}\phi_{ij} \exp(\lambda_{ij} t) \right\} \tag{9}$$

in which b_{ij} relates to the initial condition of the initial value problem and ϕ_{ij} is the j-th eigenmode of the i-th manifold related to the eigenvalue λ_{ij}. We remark that the integrable piecewise model is valid for fixed parameters, since the dependency from Θ has been dropped in Eq. (3). The switching surfaces s affect the topological properties of the attractor[27]. Both position and geometry of the switching surfaces operate as bifurcation parameters as those surfaces allow the transition from different dynamics governed by different fixed points. By tuning the switching surfaces it is possible to recover any desired population of periodic orbits and also multimodal attractor. For a generic chaotic system it is of difficult optimization and it incentives the use of ANNs. When the switching s occurs, the nonlinearity is activated in the embedded space $\mathbf{Y}$. From Eq. 6, the only non linear term is given by the discontinuity $f_i[s(\mathbf{X})]$.

III. ALGORITHM ARCHITECTURE AND PROPERTIES: PIECEWISE CONTINUOUS APPROXIMATOR

The piecewise formulation involves discontinuities that must be included in a piecewise continuous neural network[29]. The main point is to split the learning process among the different manifolds which compose the dynamics of the system. The implementation requires to approximate Eq. (3) via the PCNN $\hat{\mathscr{N}} : \mathbf{X} \to \dot{\mathbf{X}}$, i.e.

$$\hat{\mathscr{N}} = \sum_{n=1}^{N} \hat{\mathscr{N}}_n(\mathbf{x}) I_{\hat{K}_n}(\mathbf{x}), \tag{10}$$

where the partition $\{\hat{K}_n\}_{n=1}^{N}$ is given by the deep zero sets

$$\hat{K}_n := \{\mathbf{x} \in \mathbf{X} : 1 - I_{(\gamma,1]} \circ \sigma_{sigmoid}(\hat{c}(\mathbf{x})_n) \neq 0\}. \tag{11}$$

Here $\mathbf{x}$ is the vector defined in the space $\mathbf{X}$ describing the state of the system. Each $\hat{\mathscr{N}}_1, \hat{\mathscr{N}}_2, \ldots, \hat{\mathscr{N}}_N$ is a sub-pattern of the PCNN and it is a FFNN $\in NN^{\sigma}_{d,D}$, $\hat{c} \in NN^{\sigma}_{d,D}$ is the deep classifier, $0 < \gamma \leq 1$, $N \in \mathbb{N}_+$.

The framework is unbiased, but requires in input the system dimension. Since it could be cumbersome to guess the dimension, the dynamic mode decomposition (DMD) is employed as it relates to the Koopman theory[42] and it augments the mathematical formulation. Figure 1 illustrates the PCNN architecture. The deep zero sets are built by feeding the deep classifier $\hat{c}$ into the discontinuous unit $\mathbf{X} \mapsto \mathbf{X} \cdot I_{(\gamma,1]} \circ \sigma_{sigmoid}$. Each trainable part of the architecture $\hat{\mathscr{N}}_1, \hat{\mathscr{N}}_2, \ldots, \hat{\mathscr{N}}_N$ and the classifier $\hat{c}$, processes input data $\mathbf{X}$ independently and it can be therefore parallelizable. The outputs $\hat{\mathscr{N}}_1(\mathbf{X}), \hat{\mathscr{N}}_2(\mathbf{X}), \ldots, \hat{\mathscr{N}}_N(\mathbf{X})$ and $\hat{C} := \hat{c}(\mathbf{X})$ of each parallel sub-patterns enter the discontinuous unit $\upsilon := I_{(\gamma,1]} \circ \sigma_{sigmoid}$:

$$\begin{aligned}(\hat{\mathscr{N}}_1(\mathbf{X}), \hat{\mathscr{N}}_2(\mathbf{X}), \ldots, \hat{\mathscr{N}}_N(\mathbf{X}), \hat{C}) \mapsto \\ (\hat{\mathscr{N}}_1(\mathbf{X}) I_{(\gamma,1]} \circ \sigma_{sigmoid_1}, \hat{\mathscr{N}}_2(\mathbf{X}) I_{(\gamma,1]} \circ \sigma_{sigmoid_2}, \\ \ldots, \hat{\mathscr{N}}_N(\mathbf{X}) I_{(\gamma,1]} \circ \sigma_{sigmoid_N}).\end{aligned} \tag{12}$$

The discontinuous unit defines the deep zero sets and decides which sub-pattern ($\hat{\mathscr{N}}_i$) is active. To solve the clustering problem we choose to penalize the cardinality of the manifolds with respect the cardinality of the switches in time. The penalization acts on the initialization of the training process making a first inference on the partition of $\mathbf{X}$. This favours meaningful dynamical model spurred by the recurrent jumps of the chaotic dynamics between manifolds.

1. *Identification of the domains of the manifolds*

The clustering procedure to determine the different manifolds is approached as a binary classification problem. We aim at learning which $\mathbf{x} \in \mathbf{X}$ belongs to a fixed subset $K \subseteq \mathbf{X}$. Thus a classifier $\hat{c} : \mathbf{X} \to \{0,1\}$ is trained to approximate point-wise with high probability the indicator function I_K[43]. However, even if the approximation guarantees and $\hat{c}$, deterministically, approximate I_K point-wise, some points could be misclassified, i.e. wrong approximation of K. This occurs if the points contained in K are limits of some points sequence in the approximating sets K_k. Therefore we base the convergence of the sets by qualifying limits of all sub-sequence $\{\mathbf{x}_{k_j}\}_{j=1}^{\infty}$ where $\mathbf{x}_k \in K_k \quad \forall k \in \mathbb{N}^+$. The convergence is satisfied according to the Hausdorff distance d_H defined for any pair of subsets $A,B \subseteq \mathbf{X}$ via:

$$d_{H|\mathbf{X}}(A,B) := max\{sup_{a\in A}\|a-B\|, sup_{b\in B}\|A-b\|\}, \tag{13}$$

where $\|a-B\| := inf_{b\in B}\|a-b\|$ (similarly $\|a-B\|$). Finally, we remark that the partition of the input space does not have to satisfy the condition

$$int(K_n) \bigcap int(K_m) = 0. \quad \text{for} \quad n \neq m \tag{14}$$

From the physical point of view it slacks the recovery of different manifolds characterized by different dynamical time scales which work in conjointly.

2. *PCNN Algorithm*

The set of all PCNNs ($PCNN^{\sigma}_{d,D}$) are implemented via the algorithm Meta-Algorithm 1. The map $I_{(\gamma,1]}$ projects each $\hat{K}_n$ into a discontinuous function, by extension $\hat{\mathscr{N}}$ is not differentiable. However, by decoupling the training of $\{\hat{\mathscr{N}}_n\}_{n=1}^{N}$ and $\{\hat{K}_n\}_{n=1}^{N}$ we solve the non-differentiability issues as the passing gradient updates do not enter the indicator function $I_{(\gamma,1]}$ in $\hat{K}_n$. The presented data-driven architecture is trainable in a two-steps procedure with

Meta-Algorithm 1. PCNN pseudo-code.

```
1  input: X measurement data, N # of parts, J1,J2 ∈ ℕ+ ANN hyperparameters, GET − FFNN FFNN training subroutine,
2         GET − PARTITION Partition subroutine
3  output: 𝒩̂ := ∑_{n≤N} 𝒩̂_n(·) I_{K̂_n}(·) Trained PCNN model, boundaries of the regions {K̂_n}_{n=1}^N discontinuity surfaces
4  begin
5      X_T ∈ ℛ^{m*n}, X_V ∈ ℛ^{v*n} ⋘ TrainSeparation(Y, m, v) #Construct training and validation data set
6      {X_n}_{n=1}^N ⋘ GET − PARTITION(X_T) Initializes partitions of X
7      for w ≤ W do #Loop over the interaction index w
8          for n ≤ N do #For each manifold in the data training set
9              𝒩̂_n ⋘ GET − FFNN(X_n, ℒ̂_tot, J1) #optimize the networks 𝒩̂_n, evaluate the loss function (Eq. (15), Eqs. (16))
10         end for
11         for x ∈ X do #For each training data
12             l_{n,x} := I(ℒ̂_tot(𝒩̂_n(x)) ≤ min_{m≤N} ℒ̂_tot(𝒩̂_m(x))) #Identifies which optimized 𝒩̂_n best performs on any given input and adjusts the partitions
13         end for
14         ĉ ⋘ GET − FFNN(X, H(·|l_{n,·}), J2)
15         for n ∈ N do #For each manifold in the data training set
16             K̂_n := {x ∈ ℝ^n : ĉ_n(x) ≤ 2^{-1}} #Define deep zero-sets
17         end for
18     end for
19 end begin
```

- the clustering (identification of the domains of manifolds);
- the regression (identification of the models that describe the dynamics of data in every identified manifolds).

The Meta-Algorithm 1 is initialized by building a first partition of data domain $\mathbf{X}$ (pseudo-code line 6), with the sub-routine $GET-PARTITION$ (see pseudo algorithm GET-PARTITION). This first partition defines the first rough estimation of the switching surfaces. The sub-routine $GET-PARTITION$ implements a geometric prior so that nearby points accumulate on the same $\hat{K}_n$. In other words, we penalize the cardinality N of the manifolds getting $\{\hat{K}_n\}_{n=1}^N$ partitions $\mathbf{X}$ with the smallest boundaries. If $\mathbf{X}$ is representative of $\{\hat{K}_n\}_{n=1}^N$, $GET-PARTITION$ seeks a $\mathbf{X}$ composed of N parts $\{\mathbf{X}_n\}_{n=1}^N$ while minimizing the distance between every pair of data points in each $\mathbf{X}_n$. The sub-routine implements a randomized-polynomial time algorithm which approximately solves the above mentioned min-cut problem with high probability[44] (see in Appendix A). In the sub-routine $GET-PARTITION$, the variable q represents the minimum portion of the data required in each partition, $\Delta(\mathbf{X}) := min_{\mathbf{x},\mathbf{y}\in\mathbf{X};\mathbf{x}\neq\mathbf{y}} \frac{1}{2}\|\mathbf{x}-\mathbf{y}\|$ and it denotes the minimum distance between any pair of training data points, and $\bar{\Delta}(\mathbf{X}) := \frac{1}{\#\{(\mathbf{x},\mathbf{y})\in\mathbf{X}^2:\mathbf{x}\neq\mathbf{y}\}} \sum_{(\mathbf{x},\mathbf{y})\in\mathbf{X}^2:\mathbf{x}\neq\mathbf{y}} \|\mathbf{x}-\mathbf{y}\|$ is the mean distance between distinct training data. $GET-PARTITION$ initializes by defining a random parameter α ($GET-PARTITION$ pseudo code line 4) that defines the random radius $\alpha\bar{\Delta}(\mathbf{X})$ ($GET-PARTITION$ pseudo code line 9). Line 5 shuffles the training data. The random radius is then used to form (line 9) and extend to forms parts of the input space (line 10) the partitions of the data. Lines from 12 to 16 ensure each part is not too large relative to the others. $GET-PARTITION$ performs a first guess of the number of manifold N that is not a required input.

A second sub-routine follows (pseudo-code starting from line 8 to line 10), a FFNN $\hat{\mathcal{N}}_n \in NN_{d,D}^{\sigma}$ is trained on each partition of the input data $\mathbf{X}$ in order to get the integrable models described in Eqs. (8)-(9). The integrability condition guaranteed by the existence of the ψ_i functions is introduce in the definition of the loss function $\hat{\mathscr{L}}_{\text{tot}}$, which is built by three components

$$\hat{\mathscr{L}}_{\text{tot}} = \alpha_1 \hat{\mathscr{L}}_{\text{recon}} + \alpha_2 \hat{\mathscr{L}}_{\text{pred}} + \hat{\mathscr{L}}_{\text{lin}} \tag{15}$$

where $\hat{\mathscr{L}}_{\text{recon}}$ ensures the condition $\psi_i = \xi_i^{-1} \quad \forall i \in \{1,2,\dots N\}$ (see Fig. 1 subplot (b)), $\hat{\mathscr{L}}_{\text{pred}}$ predicts system future dynamics (see Fig. 1 subplot (a)) and $\hat{\mathscr{L}}_{\text{lin}}$ forces the model identified in $\mathbf{Y}$ to be linear (see Fig. 1 subplot (c)). Their expressions are

$$\begin{aligned}
\hat{\mathscr{L}}_{\text{recon}} &= \|\dot{\mathbf{X}}(t_0) - \xi_i(\psi_i(\mathbf{X}(t_0)))\|_{\text{MSE}} \\
\hat{\mathscr{L}}_{\text{pred}} &= \frac{1}{S_p}\sum_{m=1}^{S_p} \|\dot{\mathbf{X}}(t_0+(m-1)\Delta t) - \xi_i(A_i^m \psi_i(\mathbf{X}(t_0)))\|_{\text{MSE}} \\
\hat{\mathscr{L}}_{\text{lin}} &= \frac{1}{T-1}\sum_{m=1}^{T-1} \|\psi_i(\dot{\mathbf{X}}(t_0+(m-1)\Delta t)) - A_i^m \psi_i(\mathbf{X}(t_0))\|_{\text{MSE}},
\end{aligned} \tag{16}$$

where MSE is mean squared error, t_0 is the time related to the first recorded state $\mathbf{X}$, Δt is the time step characteristic of the measured data, T is the number of time steps in each trajectory and $S_p = 30$ the number of time steps at which the system is projected forward into the future. The weights $\alpha_1 = 0.1$ and $\alpha_2 = 10^{-7}$ are hyperparameters. The order of $\alpha_1 \gg \alpha_2$ is linked to constraints on the loss function. In particular we aim at determining a suitable coordinates transformation by penalizing the accuracy of the prediction.

The sub-routine is implemented in two ways: *i)* by using a specific Autoencoder architecture[14] (used in the Lorentz example of Sec. IVA); *ii)* by using the DMD[45] (employed in the Kuramoto example of Sec. IVB). Moreover, despite

the linear formulation of the DMD, it works excellently on nonlinear systems and it can be considered a numerical approximation to Koopman spectral analysis[42]. Finally, the Meta-Algorithm 1 from line 14 to line 17 interprets the $\{l_{n,x}\} \in \{0,1\}$ as labels, it trains $\hat{c}_n$ by defining $\hat{K}_n$ as a classifier and predicting when $\hat{\mathscr{N}}_n$ offers the best cross-entropy, i.e. $H(y|l_{n,x}) := l_{n,x} ln(y)(1-l_{n,x} ln(1-y)$, amongst the $\{\hat{\mathscr{N}}_n\}_{n=1}^{N}$. This interaction between the information get from solving the regression problem in each identify manifold and the Deep classifier enable the identification of the switching surfaces. The most external loop (pseudo-code lines 7 to 18) over the index w improves the interaction between the training of the models $\hat{\mathscr{N}}$ and the identification of the domain of each manifold.

IV. RESULTS

The presented data-driven framework is applied on two archetypal systems which exhibit chaotic behaviours, with the objective to recover interpretable and piecewise integrable dynamical models from data. The two analyzed systems are the Lorenz system (ODE) and the one-dimensional Kuramoto–Sivashinsky equation (PDE).

A. Lorenz System

The Lorenz system is a well-known simplified mathematical model for atmospheric convection[46] , it reads

$$\begin{aligned} \dot{x}(t) &= \sigma(y-x), \\ \dot{y}(t) &= x(\rho - z) - y, \\ \dot{z}(t) &= xy - \beta z. \end{aligned} \tag{17}$$

The three-dimensional system (17) in the x, y and z variables is representative of a dissipative hydrodynamic flow, such as the Rayleigh–Beńard convection problem in a plane fluid motion[47]. Here, σ is the Prandtl number, whereas ρ and β are the Rayleigh and wave number, respectively. In this model x is proportional to the intensity of the convective motion, y to the temperature difference between the ascending and descending currents. Similar sign of x and y denotes warm fluid rises and cold fluid descends. The variable z is proportional to the distortion of the vertical temperature profile from linearity, a positive values means that the strongest gradients occur near the boundaries. The dynamics is characterize by three fixed points; two unstable foci and one saddle. The dynamics around the fixed points is characterized two merged spirals composing the attractors. The role of the saddle is to separate the transition from the domain of influence of one spiral to the other, the saddle point in the Lorenz system is a switching surface. Thus we are in presence of three *mechanisms* defining the dynamics, the saddle is a topological tearing operator, the attractors fold the flow[27].

Data-sets are created numerically sampling time series of 800 time steps for 60 initial conditions $\mathbf{X}(0)$, parameters are kept constants as $(\sigma,\rho,\beta) = (10,28,8/3)$. Data is divided in 70% training and 30% validation. The PCNN elaborates the training data, divide them in inputs $\mathbf{x}$ and outputs $\dot{\mathbf{x}}$ as shown in Fig. 2. Ones training data is acquired, the PCNN starts the training applying classical stochastic gradient descent and cross-validation, recovering the model from data. The identified model

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} &= f_1[s(\mathbf{x})] \begin{bmatrix} -9.6046 & -2.2987 & 3.4145 \\ 9.6258 & 3.4156 & 9.3772 \\ -0.2154 & -5.1252 & -2.3166 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \dots \\ & f_1[s(\mathbf{x})] \begin{bmatrix} 55.4811 \\ 62.7195 \\ 59.0578 \end{bmatrix} + \dots \\ & f_2[s(\mathbf{x})] \begin{bmatrix} -9.6632 & -3.1871 & -4.8519 \\ 9.6735 & 2.1216 & -9.3355 \\ 0.2787 & 5.2920 & -2.6382 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \dots \\ & f_2[s(\mathbf{x})] \begin{bmatrix} -27.9902 \\ -122.2507 \\ -80.8086 \end{bmatrix} \end{aligned} \tag{18}$$

is piecewise integrable and the coordinates transformation recovered ξ_i and ψ_i is equal to the identity transformation (i.e. the identity matrix). Therefore the piecewise alternative of the continuous model (17) is piecewise linear and it verifies the analytical results present in literature[27]. This framework is applied using different number epochs for the learning process of each sub-model: 4 epochs (Fig. 3) and 100 epochs (Fig. 4 and Fig. 5). A few epochs suffice to capture the manifolds associated to the two foci points as shown in Fig. 3. The outcome of the algorithm is presented in Fig. 4 for training (subplots (a)) and test data (subplots (b)). In black/coloured lines we plot the true/reconstructed trajectories. Figure 5 showcases the phase space with the ability of the framework to reconstruct and predict the state of the system with 100 epochs.

B. Kuramoto–Sivashinsky equation (Flame equation)

The one dimensional Kuramoto-Sivashinsky equation models pattern formations on unstable flame fronts and thin hydrodynamic films[48]. It is described by a fourth-order nonlinear partial differential equation (PDE)

$$\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial z^2} + \frac{\partial^4 u}{\partial z^4} + u\frac{\partial u}{\partial z} = 0, \tag{19}$$

and it is characterized by the coexistence of coherent spatial structures with temporal chaos. In Eq. (19) u represents the velocity of a flow in fluid dynamics, z is the spatial coordinate, t is time. Firstly, the system is simulated for 1000 time steps ($\Delta t = 0.01[s]$), with free boundary conditions and a random standard normal distribution as initial conditions. The states of the system $u = u(z,t)$ at each position z and time t, compose the data matrix $\mathbf{X}$ shown in Fig. 6(a). The Meta-algorithm 1 labels training and validation samples accordingly to the belonging manifold with the imposition of Eq. (14). The clustering problem returns two manifolds, one for the non chaotic

Sub-routine: GET-PARTITION pseudo-code.

input: $\mathbf{X}$ measurement training data, $q \in (0,1)$ minimum portion of data required in each partition
output: $\{\mathbf{X}_n\} := \{\mathbf{X}_n \neq 0\}$, $N := \#\{\mathbf{X}_n\}$, $\{\hat{K}_n^X\} := \{\hat{K}_n^X \neq 0\}$
begin
  Sample α uniformly from $(2^{-1}, 4^{-1}]$ #Introduce randomness
  Pick a bijection $\pi : \{1,\ldots,\#\mathbf{X}\} \to \mathbf{X}$ shuffles the training data
  Compute $\Delta(\mathbf{X})$ and $\overline{\Delta}(\mathbf{X})$
  $\mathbf{X}' \leftarrow \mathbf{X}$
  for $n \leq N$ **do** #For each epochs of training
    $\mathbf{X}_n := \{\mathbf{z} \in \mathbf{X}' : \|\mathbf{z} - \mathbf{X}'_0\| < \alpha\overline{\Delta}(\mathbf{X})\}$
    $\hat{K}_n^{\mathbf{X}} := \{\mathbf{z} \in \mathbb{R}^d : (\exists \mathbf{x} \in \mathbf{X}_n)\|\mathbf{x} - \mathbf{z}\| \leq \Delta(\mathbf{X})\}$ #extend the partitions of the data to form parts of the input space
    $\mathbf{X}' \leftarrow \mathbf{X}' - \mathbf{X}_n$
    if $\frac{\#\mathbf{X}'}{\#\mathbf{X}} \leq q$ **then** this if ensure that each part is not too large relative to the others
      $\mathbf{X}_n := \mathbf{X}'$
      $\hat{K}_n^{\mathbf{X}} := \{\mathbf{z} \in \mathbb{R}^d : (\exists \mathbf{x} \in \mathbf{X}_n)\|\mathbf{x} - \mathbf{z}\| \leq \Delta(\mathbf{X})\}$
      break
    end if
  end for
end begin

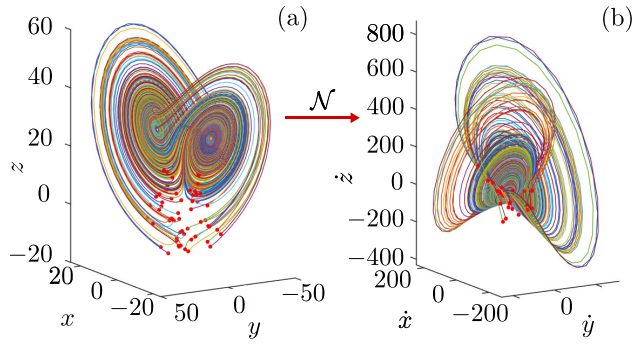

FIG. 2. Input (a) and output (b) training data, which are the state of the system $\mathbf{x}(t) = [x(t), y(t), z(t)]$ (a) and its time derivative $\dot{\mathbf{x}}(t) = [\dot{x}(t), \dot{y}(t), \dot{z}(t)]$ (b) for 42 trajectories of 800 time steps, characterized by a $\Delta t = 0.01$ and different initial conditions (red dots).

and one for the chaotic regime. We detect the transition to the chaotic regime in a completely unbiased way as shown in Figs. 6(b). Each $i-th$ region is govern by a linear map, represented by a constant matrix $\mathbf{A}_i$ such that

$$\begin{aligned} \dot{\mathbf{U}}(t) &= \sum_{i=1}^{2} \{f_i[s(\mathbf{U})]\mathbf{A}_i\mathbf{U}(t)\} \\ \mathbf{U}(t) &= \sum_{i=1}^{2} \left\{ f_i[s(\mathbf{U})] \sum_{j=1}^{M} b_{ij}\phi_{ij}\exp(\lambda_{ij}t) \right\}, \end{aligned} \tag{20}$$

and in which b_{ij}, ϕ_{ij} and λ_{ij} come from the dynamic mode decomposition of each $i-th$ linear matrix $\mathbf{A}_i$. Precisely b_{ij} and ϕ_{ij} are respectively, a constant representative of the system initial conditions and the $j-th$ eigen-mode, both associated to the $j-th$ eigen-value λ_{ij} The hyper-surface $s(\mathbf{U})$ that divides the domains of the two manifold identified model acts as a trigger mechanism to ignite chaos (dot-dashed line in Fig. 6(b)). It defines the onset of a drastic metamorphosis of the phase portrait. Regarding the computational performance, by splitting the data set in different manifolds, each model is fast to train by using simpler architecture (small ANNs). In turn less epochs for the training of the model. The drawback is to solve a clustering problem that requires the training of additional algorithms and ANNs. The algorithm spends $10[s]$ to train the deep classifier and $4[s]$ to train the two manifolds with the parameters given in Tab. I. The accuracy reported in Tab. II is not satisfactory, with consistent deviation in the time dynamics. Further tuning of hyperparameters is needed. Nevertheless, the algorithm succeeds in capturing the structure of the attractors.

PCNN architecture

Network architecture	Width of each hidden layer			
Lorenz manifolds	input layer	layer 1	layer 2	layer 3
	3	30	30	3
	layer 4	layer 5	layer 6	output layer
	3	30	30	3
Lorenz deep classifier	layer 1	layer 2	layer 3	layer 4
	10	10	10	10
Kuramoto S. deep classifier	layer 1	layer 2	layer 3	layer 4
	10	10	10	10

TABLE I. PCNN architecture for the Lorenz system and the deep classifier architecture used for Kuramoto Sivashinsky model.

Accuracy performance of the PCNN framework

		MAE	MSE	SMAPE
Lorez (100 epochs training)	training	0.1809	0.1190	0.9748
	test	0.2079	0.1059	1.0952
Lorez (4 epochs training)	training	12.0606	867.6464	41.8317
	test	13.2316	880.7967	51.6065
Kuramoto Sivashinky	training	6.1709	65.04075	141.1444
	test	10.4402	211.6761	146.42087

TABLE II. The error is evaluated using difference metrics, i.e. MAE (mean absolute error), MSE (mean squared error) and SMAPE (symmetric mean absolute percentage error).

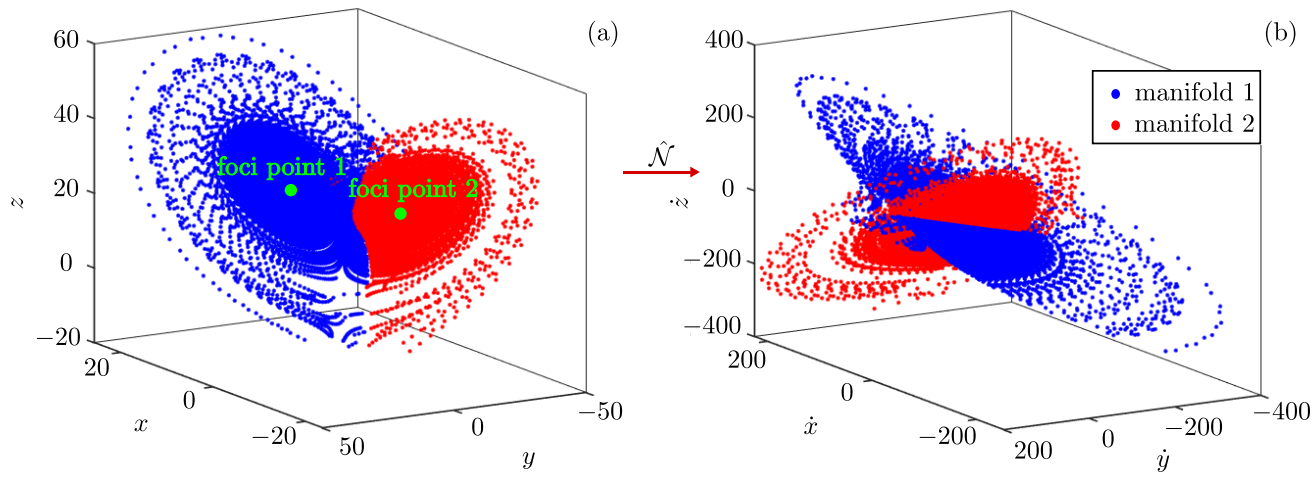

FIG. 3. Plot of the reconstructed $\dot{\mathbf{X}}$ state (b) from the $\mathbf{X}$ input state (a) obtained from the recovered Lorenz model $\hat{\mathcal{N}}$ of Sec. IVA. With only four epochs the PCNN is able to capture the cardinality of the manifolds which build the system's dynamics. The blue and red clusters reconstruct the two manifolds around the stable foci.

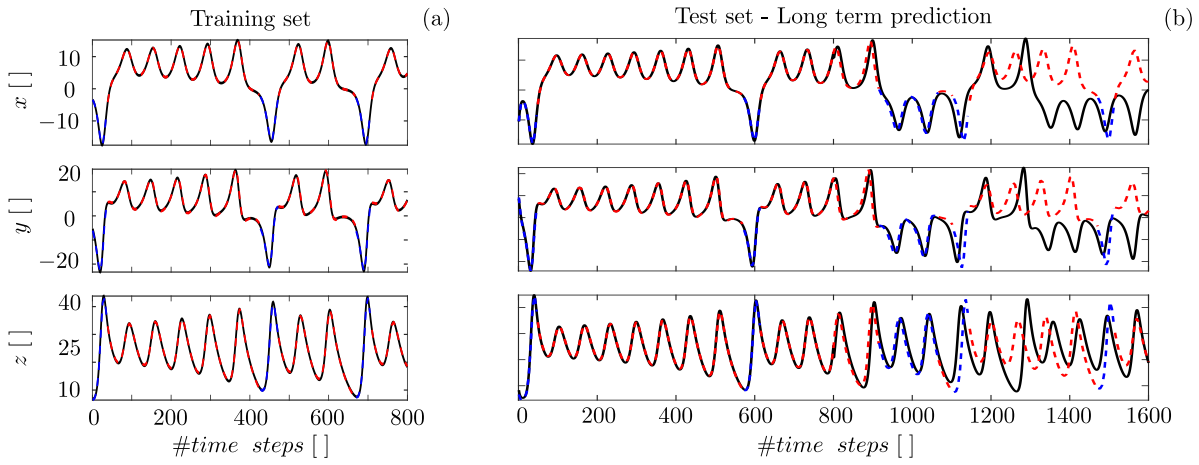

FIG. 4. Comparison between true (black line) and reconstructed (colored dashed line) dynamics for the training (a) and test set (b). The training and test set are built from trajectories of 800 and 1600 time steps respectively. The recovered model dynamics is characterize by 2 manifolds (blue and red lines) related to the 2 foci points of the Lorenz system. The model is trained with 100 epochs for each manifold and 150 epochs for the deep classifier.

V. SUMMARY AND CONCLUSIONS

This paper presents a new data-driven approach to construct piece-wise integrable models governed by switching mechanisms, suited for the identification of dynamical systems which exhibits chaotic behaviours. The method exploit the flexibility and the powerful approximation capacity of discontinuous ANN architectures (while remains trainable with (stochastic) gradient-descent algorithms like feed-forward networks with differentiable activation functions). It preserves the ability to produce physical interpretable model from data. In the paper we test two archetypal chaotic systems, Lorenz and Kuramoto-Sivashinsky systems, described by a set of ordinary and a partial differentials equation, respectively. The method is purely data-driven and does not require any physical knowledge about the system. The chaotic nature of the collected observables is encoded by the discontinuous architecture of the PCNN. The framework recovers models which are the result of the composition of N distinct sub-models defined within specific manifolds. They represent partitions of the state space domain in which the dynamics can be simplified with a great reduction of the ANNs parameters. The manifolds are related to specific fix points and they are mutually connected by discontinuous hyper-surfaces that replace the bifurcation mechanisms underlying the chaotic dynamics. The presented approach:

- identifies the switching surfaces between manifolds, it recovers the number and location of fixed points around

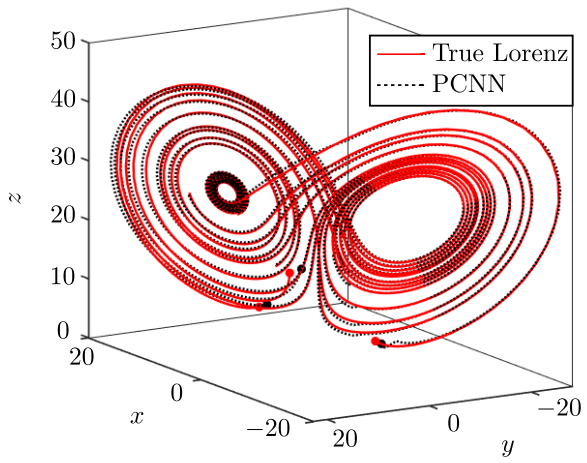

FIG. 5. Phase space with three trajectories from the test set (800 time-steps). In black/red the reconstructed/true data. The model correctly identify the structure of the attractor.

which the dynamics is organized;

- approximates in general integrable models the dynamics and it obtains the Jacobian matrix of the linear embedded system, at each of the fixed points identifying a specific domain of influence (manifold).

When accounting for stochastic effects and the possibility to introducing parametrization, during the learning process, this tool will support study and synthesize of chaotic systems in applications as weather and emergency events forecasting.

ACKNOWLEDGMENTS

The authors wish to acknowledge the GNFM (Gruppo Nazionale per la Fisica Matematica).

DATA AVAILABILITY STATEMENT

All data presented in the manuscript is available upon request. Codes are available at github.com/nicontech/PCNNchaos.

Appendix A: Meta-algorithm's 1 properties

Supplementary definitions

- $\mathbb{P} :=$ fixed Borel probability measure on $\mathbf{X}$
- $C(\mathbb{R}^d, \mathbb{R}^D) :=$ the space of continuous functions from $\mathbf{X} \in \mathscr{R}^d$ to $\mathscr{R}^D$
- $\mathscr{B}(\mathbf{X}, \mathscr{R}^D) :=$ Banach space of all bounded functions from $\mathbf{X}$ to $\mathscr{R}^D$ equipped with the L_∞-norm
- $D_{PC}(f|g)$ is the piecewise divergence between any couple of functions f and $g \in \mathscr{B}(\mathbf{X}, \mathscr{R}^D)$[29]

1. Subroutine: $GET - PARTITION$

Proposition *With high probability* $\mathbb{P}$*, two nearby data points in* $\mathbf{X}$ *are mapped to the same manifold. Here* Let $\mathbf{X}$, $q = 1$ and $\{\hat{K}_n^{\mathbf{X}}\}_{n=1}^N$ be as in the sub-routine $GET - PARTITION$, and fix $\hat{\mathscr{N}}_1(\mathbf{X})$, $\hat{\mathscr{N}}_2(\mathbf{X})$, …, $\hat{\mathscr{N}}_N(\mathbf{X}) \in NN_{n=1}^{\sigma}$. Let $\hat{\mathscr{N}} = \sum_{n=1}^N \hat{\mathscr{N}}_n I_{\hat{K}_n^{\mathbf{X}}}$. Then the following hold:

- For $\mathbf{x}_1, \mathbf{x}_2 \in \mathbf{X}$, $\mathbb{P}(min_{n=1,\dots,N} max_{i=1,2} \|\hat{\mathscr{N}}(\mathbf{x}_i) - \hat{\mathscr{N}}_n(\mathbf{x}_i)\| = 0) \geq 1 - \frac{8[ln(\#\mathbf{X})+1]}{\bar{\Delta}(\mathbf{X})}$, where $\bar{\Delta}(\mathbf{X}) := \frac{1}{\#\{(\mathbf{x}_1,\mathbf{x}_2) \in \mathbf{X}^2 : \mathbf{x}_1 \neq \mathbf{x}_2\}} \sum_{(\mathbf{x}_1,\mathbf{x}_2)\in\mathbf{X}^2:\mathbf{x}_1\neq\mathbf{x}_2} |\mathbf{x}_1 - \mathbf{x}_2|$,
- Sub-routine $GET - PARTITION$ terminates in polynomial time,
- For $n \leq N$, $int(\hat{K}_n^{\mathbf{X}}) \neq 0$
- If $n \neq m \Rightarrow int(\hat{K}_n^{\mathbf{X}}) \bigcap int(\hat{K}_m^{\mathbf{X}}) = 0$

2. Effect of the initial partition on the performance of PCNN

The effect of the size of the minimal ball radius $min(\alpha\bar{\Delta}(\mathbf{X}))$ and of the FFNNs neural networks on the number of identified manifolds is showcased in Fig. 7. The analyses is performed on the Lorenz system of Sec. IVA. On the top corner of the figure, a large yellow region shows the convergence of the algorithm to only one manifold. By increasing the # of neurons the algorithm tends to use a smaller number of manifolds for describing the dynamics. Moreover, in this analysed region of parameters space in Fig. 7 most of the regions point to descriptions with 2 and 3 manifolds. This is inline with concept that the description of the dynamics of the nonlinear system could be simplified around fixed points. When 2 manifolds are identified, these relate with the 2 stable fixed points typical of the Lorenz system and the switching surface links to the saddle. In the case of 3 manifolds the solution of the approximation problem describes the dynamics in the neighborhood of the saddle by employing a third manifold.

3. Existence: Performance Optimizing Partition[29]

Theorem 1 Fix $\{\hat{\mathscr{N}}_n\}_{n=1}^N \in NN_{d,D}^{\sigma}$ and $\hat{\mathscr{L}}_{tot} \in C(\mathbb{R}^D, [0,\infty))$ for which $\hat{\mathscr{L}}_{tot}(0) = 0$. There exists a compact subset $\mathbf{X}_{\delta,\mathbb{P}} \subseteq \mathbf{X}$ and a partition of $\mathbf{X}_{\delta,\mathbb{P}}$ satisfying:

- $\mathbb{P}(\mathbf{X}_{\delta,\mathbb{P}}) \geq 1 - \delta$

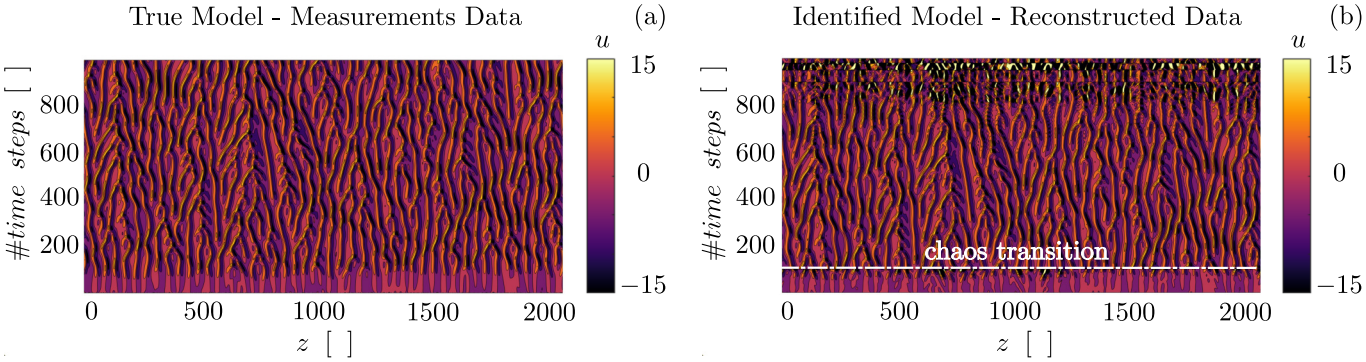

FIG. 6. One dimensional Kuramoto Sivashinsky model discovery. Here, u represent the state of the system, whereas z and t are the continuous space and time domain, respectively. In the subplot (b) the white dash lines describes the hyper-surface dividing the space domain of the system in two manifolds. The dynamics unrolls before and after that the chaos onset. In subplot (a) it is reported the training and validation data (sequentially stored), (b) shows the reconstruction of both training and validation sets. The data is decomposed into 90% for training and 10% for test (free boundary conditions).

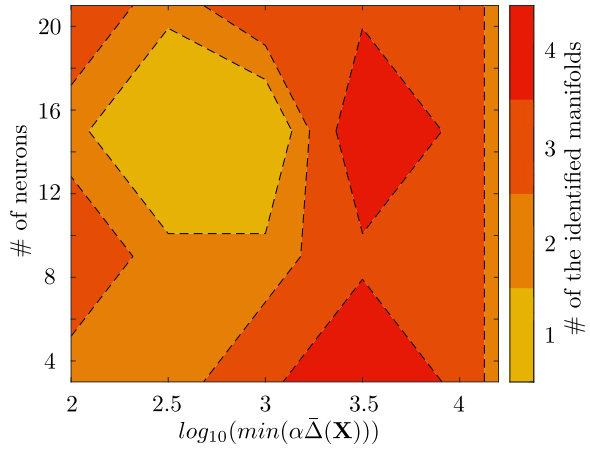

FIG. 7. The effect of the minimal ball radius $min(\alpha\bar{\Delta}(\mathbf{X}))$, related to the number of initial manifolds (clusters), and NN-width on the estimated manifolds.

- For $n \leq N$ and every $\mathbf{x} \in \hat{K}_n, \hat{\mathscr{L}}_{tot}(\hat{\mathscr{N}}_n(\mathbf{x})) = min_{m \leq N} \hat{\mathscr{L}}_{tot}(\hat{\mathscr{N}}_m(\mathbf{x}))$

Moreover, if $\mathbf{X} \subseteq \mathbf{X}_{\delta,\mathbb{P}}$ and $L(\hat{\mathscr{N}}_n(\mathbf{x})) < min_{n \neq \tilde{n}, \tilde{n} \leq N} \hat{\mathscr{L}}_{tot}(\hat{\mathscr{N}}_{\tilde{n}}(\mathbf{x}))\ \forall n \leq N, \mathbf{x} \in \mathbf{X}_n$ then, $\mathbf{X}_n \subseteq \hat{K}_n$

From the Theorem 1 exposed above is possible to conclude that $\forall \varepsilon > 0\ \exists$ a PCNN $\hat{\mathscr{N}} \backslash D_{PC}(\sum_{n=1}^{N} \hat{\mathscr{N}}_n I_{\hat{K}_n} | \hat{\mathscr{N}}) < \varepsilon$, where $\hat{\mathscr{N}}_1(\mathbf{X})$, $\hat{\mathscr{N}}_2(\mathbf{X})$, ..., $\hat{\mathscr{N}}_N(\mathbf{X})$ are from Meta-Algorithm 1. Theorem 1 guarantees that given trained models $\{\hat{\mathscr{N}}_n\}$ there exists $\{\hat{K}_n\} \subset Comp(\mathbf{X})$ which optimizes the PCNN performance with a arbitrarily high-probability.

REFERENCES

[1] G. Rega, V. Settimi, and L. Stefano, "Chaos in one-dimensional structural mechanics," Nonlinear Dynamics **102**, 785–834 (2020).

[2] A. A. Tsonis and J. B. Elsner, "Chaos, strange attractors, and weather," Bulletin of the American Meteorological Society **70**, 14 – 23 (1989).

[3] J. L. McCauley, "Nonintegrability, chaos, and complexity," Physica A: Statistical Mechanics and its Applications **237**, 387–404 (1997).

[4] C. Moore, "Unpredictability and undecidability in dynamical systems," Phys. Rev. Lett. **64**, 2354–2357 (1990).

[5] F. Huhn and L. Magri, "Gradient-free optimization of chaotic acoustics with reservoir computing," Phys. Rev. Fluids **7**, 014402 (2022).

[6] P. D. Dueben and P. Bauer, "Challenges and design choices for global weather and climate models based on machine learning," Geoscientific Model Development **11**, 3999–4009 (2018).

[7] M. Zhang, Y. Chen, Y. Shen, and J. Ma, "Comparative study on the performances of ann and svm and their application in the identification of dmd disease," **38**, 346–351 (2016).

[8] G. Cybenko, "Approximation by superpositions of a sigmoidal function," Mathematics of Control, Signals and Systems **2** (1989), 10.1007/BF02551274.

[9] K. Funahashi, "On the approximate realization of continuous mappings by neural networks," Neural Netw. **2**, 183–192 (1989).

[10] K. Hornik, "Approximation capabilities of multilayer feedforward networks," Neural Networks **4**, 251–257 (1991).

[11] H. T. Siegelmann and E. D. Sontag, "Turing computability with neural nets," Applied Mathematics Letters **4**, 77–80 (1991).

[12] K. Champion, P. Zheng, A. Y. Aravkin, S. L. Brunton, and J. N. Kutz, "A unified sparse optimization framework to learn parsimonious physics-informed models from data," IEEE Access **8**, 169259–169271 (2020).

[13] P. Belardinelli, A. Chandrashekar, R. Wiebe, F. Alijani, and S. Lenci, "Machine learning to probe modal interaction in dynamic atomic force microscopy," Mechanical Systems and Signal Processing **179**, 109312 (2022).

[14] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," Nature Communications **9** (2018), 10.1038/s41467-018-07210-0.

[15] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," AI Open **1**, 57–81 (2020).

[16] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković, "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges," (2021).

[17] O. Ganea, G. Becigneul, and T. Hofmann, "Hyperbolic neural networks," in Advances in Neural Information Processing Systems, Vol. 31, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates, Inc., 2018).

[18] A. Kratsios and E. Bilokopytov, "Non-euclidean universal approximation," in Proc. 34th Int. Conf. Neural Information Processing Systems (Curran Associates Inc., Red Hook, NY, USA, 2020).

[19] A. Kratsios, B. Zamanlooy, T. Liu, and I. Dokmanić, "Universal approximation under constraints is possible with transformers," in International Conference on Learning Representations (2022).

[20] P. Petersen and F. Voigtlaender, "Equivalence of approximation by convolutional neural networks and fully-connected networks," Proc. Amer. Math. Soc. (2018), 10.48550/ARXIV.1809.00973.

[21] D. Yarotsky, "Universal approximations of invariant maps by neural networks," Constructive Approximation **55**, 1–68 (2022).

[22] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," in Advances in Neural Information Processing Systems, Vol. 32, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Inc., 2019).

[23] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in Advances in Neural Information Processing Systems, Vol. 31, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Curran Associates, Inc., 2018).

[24] W. Grathwohl, R. T. Q. Chen, J. Bettencourt, and D. Duvenaud, "Scalable reversible generative models with free-form continuous dynamics," in International Conference on Learning Representations (2019).

[25] A. Kratsios and C. Hyndman, "Neu: A meta-algorithm for universal uap-invariant feature representation," Journal of Machine Learning Research **22**, 1–51 (2021).

[26] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," Science **324**, 81–85 (2009).

[27] G. F. V. Amaral, C. Letellier, and L. A. Aguirre, "Piecewise affine models of chaotic attractors: The rössler and lorenz systems," Chaos: An Interdisciplinary Journal of Nonlinear Science **16**, 013115 (2006).

[28] V. N. Belykh, N. V. Barabash, and I. V. Belykh, "A lorenz-type attractor in a piecewise-smooth system: Rigorous results," Chaos: An Interdisciplinary Journal of Nonlinear Science **29**, 103108 (2019).

[29] A. Kratsios and B. Zamanlooy, "Learning sub-patterns in piecewise continuous functions," (2020).

[30] N. Novelli, S. Lenci, and P. Belardinelli, "Boosting the model discovery of hybrid dynamical systems in an informed sparse regression approach," Journal of Computational and Nonlinear Dynamics **17** (2022), 10.1115/1.4053324, 051007.

[31] J. R. Munkres, Topology /, 2nd ed. (Prentice Hall,, New Delhi,, c207.).

[32] A. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," IEEE Transactions on Information Theory **39**, 930–945 (1993).

[33] R. Gribonval, G. Kutyniok, M. Nielsen, and F. Voigtlaender, "Approximation spaces of deep neural networks," Constructive Approximation **55**, 259–367 (2022).

[34] J. W. Siegel and J. Xu, "Approximation rates for neural networks with general activation functions," Neural networks : the official journal of the International Neural Network Society **128**, 313–321 (2020).

[35] A. Kratsios, "The universal approximation property," Annals of Mathematics and Artificial Intelligence **89** (2021), 10.1007/s10472-020-09723-1.

[36] D. Findlay, "Training networks with discontinuous activation functions," in 1989 First IEE Int. Conf. Artificial Neural Networks, (Conf. Publ. No. 313) (1989) pp. 361–363.

[37] L. Ferreira, E. Kaszkurewicz, and A. Bhaya, "Solving systems of linear equations via gradient systems with discontinuous righthand sides: application to ls-svm," IEEE Transactions on Neural Networks **16**, 501–505 (2005).

[38] G.-B. Huang, Q.-Y. Zhu, K. Mao, C. Siew, P. Saratchandran, and N. Sundararajan, "Can threshold networks be trained directly?. ieee trans circuits syst ii," Circuits and Systems II: Express Briefs, IEEE Transactions on **53**, 187 – 191 (2006).

[39] P. Kidger and T. Lyons, "Universal approximation with deep narrow networks," in Proceedings of Thirty Third Conference on Learning Theory, Proceedings of Machine Learning Research, Vol. 125, edited by J. Abernethy and S. Agarwal (PMLR, 2020) pp. 2306–2327.

[40] T. D. Tsankov and R. Gilmore, "Topological aspects of the structure of chaotic attractors in r 3," Physical Review E **69**, 056206 (2004).

[41] B. O. Koopman, "Hamiltonian systems and transformation in hilbert space," Proceedings of the National Academy of Sciences **17**, 315–318 (1931).

[42] I. Mezić, "Analysis of fluid flows via spectral properties of the koopman operator," Annual Review of Fluid Mechanics **45**, 357–378 (2013).

[43] A. Farago and G. Lugosi, "Strong universal consistency of neural network classifiers," IEEE Transactions on Information Theory **39**, 1146–1151 (1993).

[44] Y. Bartal, "On approximating arbitrary metrices by tree metrics," in Proc. Thirtieth Annual ACM Symposium on Theory of Computing, STOC '98 (Association for Computing Machinery, New York, NY, USA, 1998) p. 161–168.

[45] J. H. Tu, , C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. K. and, "On dynamic mode decomposition: Theory and applications," Journal of Computational Dynamics **1**, 391–421 (2014).

[46] I. Stewart, "The lorenz attractor exists," Nature **406**, 948–949 (2000).

[47] Z.-M. Chen and W. Price, "On the relation between rayleigh–beńard convection and lorenz system," Chaos, Solitons & Fractals **28**, 571–578 (2006).

[48] A. Kalogirou, E. E. Keaveny, and D. T. Papageorgiou, "An in-depth numerical study of the two-dimensional kuramoto–sivashinsky equation," (2015) p. 20140932.