Physica D 469 (2024) 134325

Contents lists available at ScienceDirect

Physica D

journal homepage: www.elsevier.com/locate/physd

Data-driven cold starting of good reservoirs

Lyudmila Grigoryeva [a,b,*,1], Boumediene Hamzi [c,d,h], Felix P. Kemeth [d], Yannis Kevrekidis [d], G. Manjunath [e], Juan-Pablo Ortega [f,e,2], Matthys J. Steynberg [g]

[a] Universität Sankt Gallen, Faculty of Mathematics and Statistics, Bodanstrasse 6, CH-9000, Sankt Gallen, Switzerland
[b] University of Warwick, Department of Statistics, Coventry CV4 7AL, United Kingdom
[c] Department of Computing and Mathematical Sciences, Caltech, Pasadena, CA 91125, USA
[d] Department of Applied Mathematics and Statistics, Johns Hopkins University, Baltimore, MD, USA
[e] University of Pretoria, Department of Mathematics and Applied Mathematics, Pretoria 0028, South Africa
[f] Nanyang Technological University, Division of Mathematical Sciences, School of Physical and Mathematical Sciences, Singapore
[g] University of Pretoria, Department of Physics, Pretoria 0028, South Africa
[h] The Alan Turing Institute, London, United Kingdom

ARTICLE INFO

Communicated by V.M. Perez-Garcia

Keywords:
Reservoir computing
Generalized synchronization
Starting map
Forecasting
Path continuation
Dynamical systems

ABSTRACT

Using short histories of observations from a dynamical system, a workflow for the post-training initialization of reservoir computing systems is described. This strategy is called cold-starting, and it is based on a map called the starting map, which is determined by an appropriately short history of observations that maps to a unique initial condition in the reservoir space. The time series generated by the reservoir system using that initial state can be used to run the system in autonomous mode in order to produce accurate forecasts of the time series under consideration immediately. By utilizing this map, the lengthy "washouts" that are necessary to initialize reservoir systems can be eliminated, enabling the generation of forecasts using any selection of appropriately short histories of the observations.

1. Introduction

Reservoir computing (RC) [1–4] and in particular *echo state networks* (ESNs) [3,5,6] have gained much notoriety in recent years due to their excellent performance in the forecasting of dynamical systems [3,7–11] and to the ease of their implementation. RC aims at approximating nonlinear input/output systems using randomly generated state-space systems (called *reservoirs*), in which only a readout map is estimated depending on the learning task. It has been theoretically established that this is indeed possible in a variety of deterministic and stochastic contexts [12–16].

In the context of dynamical systems, it has been shown that this technique has close ties with classical embedding strategies like Takens' Theorem [17] and generalized synchronizations [18–22]. See [23–28] for recent developments in that direction. As we explain in detail later on, this connection implies, in the presence of certain hypotheses, the existence of submanifolds of the state space that are preserved by the reservoir dynamics driven by the observations of the dynamical system that we intend to model. Learning that invariant manifold proves to be beneficial in the dimension reduction of the problem and, more importantly, in the possibility of accurately initializing the reservoir just by using an initial condition of the dynamical system or, alternatively, an appropriately short history of some of its observations.

This idea has been used for the first time in [29] in the context of long-short term memory (LSTM) neural networks, and it is what we call *cold-starting* of reservoir systems. Reservoir initialization has traditionally been carried with long washout time series used in conjunction with the so-called *fading memory property* to evaluate the right initial reservoir state numerically. More specifically, there is a collection of conditions that one can impose on the reservoir system to guarantee that when the length of a time series that is fed into a reservoir tends to infinity, the dependence of the output on the value that was used to initialize the reservoir fades away; see for instance the *fading memory property* [30], the *echo state property* [1,31,32], or the *input forgetting property* [33]. Any of these properties implies that if the reservoir is fed with an input for a time sufficiently long, the

* Correspondence to: University of St. Gallen, Rosenbergstrasse 22CH-9000 St. Gallen, Switzerland.
E-mail addresses: Lyudmila.Grigoryeva@unisg.ch, Lyudmila.Grigoryeva@warwick.ac.uk (L. Grigoryeva), bhamzi@caltech.edu (B. Hamzi), FKemeth1@jh.edu (F.P. Kemeth), YannisK@jhu.edu (Y. Kevrekidis), Manjunath.Gandhi@up.ac.za (G. Manjunath), Juan-Pablo.Ortega@ntu.edu.sg (J.-P. Ortega), Thys.Steynberg@tuks.co.za (M.J. Steynberg).
[1] Honorary Associate Professor.
[2] Honorary Extraordinary Professor.

https://doi.org/10.1016/j.physd.2024.134325
Received 16 March 2024; Received in revised form 16 August 2024; Accepted 16 August 2024
Available online 23 August 2024
0167-2789/© 2024 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

output that will be obtained will approximate arbitrarily well the state value corresponding to the unique solution consistent with an input defined for all negative times, and all this, we emphasize, regardless of the value that has been used to initialize the reservoir. This input, whose only goal is finding an approximating initial state, is what we call the "washout". The length of the washout necessary for proper initialization depends on the dynamic features of the reservoir that fits the data and may be significant, which leads, in some instances, to sub-optimal data consumption.

This paper shows that under very general hypotheses, a map can be constructed (we call it the *starting map*) that associates with each state of the dynamical system or, equivalently (by Takens' Theorem), a short history of its observations, the unique initial condition in the reservoir space that is consistent with all their past history (the dynamical system is assumed to be invertible). The time series produced by the reservoir system out of that initial condition accurately mimics or path-continues those of the dynamical system that we intend to learn. The availability of this map spares the user from long washout reservoir iterations, which may prove computationally costly and challenging to carry out in the presence of small datasets and, more importantly, allows for immediate prediction.

The paper is structured as follows. In Section 2 we introduce the notion of good reservoir and present the reservoir computing forecasting framework in connection with the notion of generalized synchronizations. The reservoir cold-starting methodology is presented in Section 3 that, as we shall see, is based on the existence of what we call a starting map defined using a synchronization manifold that is obtained as the image of the generalized synchronizations introduced in the previous section. A forecasting method using the starting map is carefully spelled out in this section. Section 4 presents two methods for the learning of the synchronization manifold and the starting map, namely a diffusion maps-based methodology and a static feedforward neural network, and contains various numerical illustrations that show the pertinence of our methodology.

2. Good reservoirs and generalized synchronizations

In this section, we introduce the main tool that will be used in the construction of the starting map described in the introduction, namely, the *generalized synchronizations* (GS) between (the observations of) a dynamical system and a reservoir system.

2.1. Reservoirs and generalized synchronizations

We introduce **reservoir systems** as a state-space system (nonlinear in general) made out of two equations of the form:

$$\mathbf{x}_t = F(\mathbf{x}_{t-1}, \mathbf{z}_t), \tag{1}$$

$$\mathbf{y}_t = h(\mathbf{x}_t), \tag{2}$$

for all $t \in \mathbb{Z}_-$, where $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow D_N$ and $h : \mathbb{R}^N \longrightarrow \mathbb{R}^m$ are the ***reservoir*** (randomly generated) and the ***readout*** (trainable), respectively. The sequences $\mathbf{z} \in (\mathbb{R}^d)^{\mathbb{Z}_-}$ and $\mathbf{y} \in (\mathbb{R}^m)^{\mathbb{Z}_-}$ stand for the **input** and the ***output (target)*** of the system, respectively, and $\mathbf{x} \in (\mathbb{R}^N)^{\mathbb{Z}_-}$ are the associated ***reservoir states*** of dimension $N \in \mathbb{N}^+$, also referred to as the number of virtual neurons of the system. In this paper, we are interested in the particular setting where the reservoir (1) is driven by the (in general, partial) observations of a given dynamical system. The learning task consists in the path-continuation of the observations of this dynamical system, or, in a more general and complicated case, in the forecasting of the original dynamical system out of its available partial observations. Hence, for the rest of the paper the inputs and outputs in the system (1)–(2) will be chosen according to a particular learning task of interest. Moreover, in both considered learning scenarios only the ability of the reservoir to produce high-precision *autonomous* multi-step predictions is assessed.

Let M be a compact finite-dimensional differentiable manifold and let $\phi \in \mathrm{Diff}^1(M)$ be an invertible discrete-time differentiable dynamical system with differentiable inverse that, for any initial condition $m_0 \in M$, produces the trajectories $\{\phi^t(m_0)\}_{t\in\mathbb{Z}}$. Let $\omega \in C^1(M, \mathbb{R}^d)$, $d \in \mathbb{N}$, be a map that encodes d-dimensional observations of the dynamical system and define the (ϕ, ω)-*delay map* $S_{(\phi,\omega)} : M \longrightarrow \ell^\infty(\mathbb{R}^d)$ as $S_{(\phi,\omega)}(m) := \{\omega(\phi^t(m))\}_{t\in\mathbb{Z}}$.

Let a reservoir in (1) be a continuously differentiable state map $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ and consider the drive–response system associated to the inputs $\mathbf{z}_t = S_{(\phi,\omega)}(m)_t$, $t \in \mathbb{Z}$, that is to the ω-observations of ϕ and determined by the recursions:

$$\mathbf{x}_t = F\left(\mathbf{x}_{t-1}, S_{(\phi,\omega)}(m)_t\right), \quad t \in \mathbb{Z},\ m \in M \tag{3}$$

Definition 2.1. We say that a *generalized synchronization* (GS) occurs in this configuration when there exists a map $f : M \longrightarrow \mathbb{R}^N$ (which we call a *generalized synchronization*) such that

$$\mathbf{x}_t = f(\phi^t(m)) \quad \text{for any } \mathbf{x}_t,\ t \in \mathbb{Z}, \text{ and } m \in M \text{ as in (3).} \tag{4}$$

The existence of a *generalized synchronization* f means that the time evolution of the dynamical system in phase space (not just its observations) drives the response in (3).

2.2. Good reservoirs

The next definition specifies, in terms of the generalized synchronizations that we just introduced when a reservoir is suitable for the modeling of a given dynamical system. We refer to such systems as *good reservoirs*.

Definition 2.2. We say that $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ is a *good reservoir* for the ω-observations of the dynamical system $\phi \in \mathrm{Diff}^1(M)$ when it induces a generalized synchronization $f : M \longrightarrow \mathbb{R}^N$ that is also an embedding.

The term *embedding* in the definition means that f is an injective immersion, that is, it is a C^1 map with an injective tangent map and, additionally, the manifold topology in $f(M)$ induced by f coincides with the relative topology inherited from the standard topology in $\mathbb{R}^N$. Equivalently, this means that $f(M)$ is an *embedded* submanifold of $\mathbb{R}^N$.

We emphasize that the existence of generalized synchronizations, in general, and of good reservoirs in particular, is not something generic, and it presupposes that various dynamical constraints are satisfied. We briefly enumerate those constraints and some results in the literature that characterize situations in which they are satisfied. First of all, the definition (4) presupposes that for each $m \in M$ and the corresponding orbit of observations $S_{(\phi,\omega)}(m)$ there exists a sequence $\mathbf{x} := \{\mathbf{x}_t\}_{t\in\mathbb{Z}}$ such that (3) is satisfied. When that existence property holds and, additionally, the solution sequence $\mathbf{x}$ is unique, we say that F has the (ϕ, ω)-*Echo State Property* (ESP) (see [1,31,34] for in-depth descriptions of this property). Moreover, in the presence of the (ϕ, ω)-ESP, the state map F determines a unique causal and time-invariant filter $U^F : S_{(\phi,\omega)}(M) \longrightarrow (\mathbb{R}^N)^{\mathbb{Z}}$ that associates to each orbit $S_{(\phi,\omega)}(m)$ the unique solution sequence $\mathbf{x} \in (\mathbb{R}^N)^{\mathbb{Z}}$ of (3). It can be shown [25, Lemmas II.2 and II.3] that if $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ is a continuous reservoir map, then the map

$$\begin{array}{rccc} f_{(\phi,\omega,F)} : & M & \longrightarrow & \mathbb{R}^N \\ & m & \longmapsto & p_0\left(U^F(S_{(\phi,\omega)}(m))\right) \end{array} \tag{5}$$

is a generalized synchronization, that is, it satisfies the defining relation (4). In this expression $p_0 : (\mathbb{R}^N)^{\mathbb{Z}} \to \mathbb{R}^N$ is the projection onto the zero entry of the sequence. More generally, the following relation holds

$$U^F(S_{(\phi,\omega)}(m))_t = f_{(\phi,\omega,F)}\left(\phi^t(m)\right), \tag{6}$$

for any $t \in \mathbb{Z}$, $m \in M$. Additionally, the state synchronization map $f_{(\phi,\omega,F)}$ satisfies the identity:

$$f_{(\phi,\omega,F)}(m) = F\left(f_{(\phi,\omega,F)}(\phi^{-1}(m)), \omega(m)\right), \tag{7}$$

for all $m \in M$.

Second, the existence and differentiability of generalized synchronizations need to be addressed. GSs were introduced for the first time in [18], where it was shown that the asymptotic stability of the system response is a sufficient condition for the existence of a GS. Nevertheless, it was quickly noticed in [35,36] that the GS, whose existence is guaranteed by this theorem, might have poor regularity properties, rendering it useless as an attractor representation and reconstruction tool. This fact motivated the characterization in [36] of a first differentiability criterion for GSs. This result has been completed in [25] where it was shown that if $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ is of class C^2, ω is of class C^1, and

$$L_{F_x} < \min\left\{1, 1/\left\|T\phi^{-1}\right\|_\infty\right\}, \tag{8}$$

then the map given by (5) is a continuously differentiable GS and it is the only one that satisfies the recursion (7). The symbol L_{F_x} in (8) stands for $L_{F_x} = \sup_{(\mathbf{x},\mathbf{z})\in\mathbb{R}^N\times\omega(M)}\left\{\left\|D_x F(\mathbf{x},\mathbf{z})\right\|\right\}$ and $\|T\phi\|_\infty := \sup_{m\in M}\left\{\left\|T_m\phi\right\|\right\}$, with $T_m\phi : T_mM \longrightarrow T_{\phi(m)}M$ the tangent map of ϕ at $m \in M$. This result is a generalization of the main theorem formulated in [23] for the *echo state networks* (ESNs) that we shall introduce later on in (20). Moreover, due to the result [33, Theorem 19] and the expression (5), the synchronization $f_{(\phi,\omega,F)}$ is necessarily Lipschitz with a constant $L_{f_{(\phi,\omega,F)}}$ that satisfies

$$L_{f_{(\phi,\omega,F)}} \leq L_{F_x}/\left(1 - L_{F_x}\right). \tag{9}$$

We emphasize that the conditions that we just spelled out in order to ensure the good functional properties of the GS map are most likely sufficient but not necessary. There is solid empirical evidence that one can achieve good approximation properties using reservoirs that are, for example, non-differentiable.

Finally, there remains the embedding property, which is by far the most elusive of them all when it comes to the formulation of sufficient conditions for it to hold, and that are still not available for very popular reservoir choices like ESNs. To the best of our knowledge, only two general statements are available in this context, both of them for linear reservoirs. The first one is Takens' Theorem [17,37] since, in our language, this result shows that in the presence of certain non-resonance conditions and for generic scalar observations $\omega \in C^2(M,\mathbb{R})$ of a dynamical system $\phi \in \mathrm{Diff}^1(M)$, with M compact and q-dimensional, a $(2q+1)$-truncated version $S^{2q+1}_{(\phi,\omega)}$ of the (ϕ,ω)-delay map given by

$$S^{2q+1}_{(\phi,\omega)}(m) := \left(\omega(m), \omega(\phi^{-1}(m)), \ldots, \omega(\phi^{-2q}(m))\right) \tag{10}$$

is a continuously differentiable embedding. This map is in turn the GS corresponding to the linear state map $F(\mathbf{x}, z) := A\mathbf{x} + \mathbf{C}z$, with A the lower shift matrix in dimension $2q+1$ and $\mathbf{C} = (1, 0, \ldots, 0) \in \mathbb{R}^{2q+1}$ which, by Takens' Theorem, constitutes a differentiable GS for the scalar observations of ϕ. This statement has been generalized in [26] where it has been shown that roughly speaking, randomly generated linear systems that have the ESP generate GSs that almost surely have the same properties as Takens' delay embeddings.

2.3. Good reservoirs are indeed good

The next proposition shows that good reservoirs and their associated GS embeddings can be used to adequately represent attractor dynamics in an embedded submanifold of the reservoir space.

Proposition 2.3. *Let $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ be a good reservoir for the ω-observations of the dynamical system $\phi \in \mathrm{Diff}^1(M)$ with generalized synchronization $f : M \longrightarrow \mathbb{R}^N$. Then:*

(i) *The set $S := f(M) \subset \mathbb{R}^N$ is an embedded submanifold of the reservoir space $\mathbb{R}^N$.*

(ii) *There exists a differentiable observation map $h : S \longrightarrow \mathbb{R}^d$ that extracts the one-step-ahead prediction of the observations of the dynamical system out of the reservoir states. That is, with the notation introduced in (2) and (3):*

$$h(\mathbf{x}_t) = \omega\left(\phi^{t+1}(m)\right). \tag{11}$$

(iii) *The maps F and h determine a differentiable dynamical system $\boldsymbol{\Phi} \in C^1(S,S)$ given by*

$$\boldsymbol{\Phi}(\mathbf{s}) := F(\mathbf{s}, h(\mathbf{s})), \tag{12}$$

which is C^1-conjugate to $\phi \in \mathrm{Diff}^1(M)$ by f, that is,

$$f \circ \phi = \boldsymbol{\Phi} \circ f. \tag{13}$$

Proof. (i) is an elementary consequence of the fact that f is an embedding (see, for instance, [38] for details). **(ii)** Since the GS f is invertible (on S), we can consider the map $h := \omega \circ \phi \circ f^{-1} : f(M) \subset \mathbb{R}^N \longrightarrow \mathbb{R}^d$. Now, using the condition (4), we have that

$$h(\mathbf{x}_t) = \omega \circ \phi(f^{-1}(\mathbf{x}_t)) = \omega \circ \phi \circ \phi^t(m) = \omega\left(\phi^{t+1}(m)\right),$$

as required. Regarding **(iii)**, it is clear that the map $\boldsymbol{\Phi}$ defined in (12) is C^1. We now show that it maps into S. Let $\boldsymbol{\Phi}(\mathbf{s})$ with $\mathbf{s} \in S = f(M)$ and let $m \in M$ such that $\mathbf{s} = f(m)$. By the definition of the GS f in (3), we can write $\mathbf{s} = \mathbf{x}_0$, where $\mathbf{x}_0 \in \mathbb{R}^N$ is the zero term of the sequence $\mathbf{x} \in (\mathbb{R}^N)^{\mathbb{Z}}$ obtained as the output of the system determined by F with the sequence $S_{(\phi,\omega)}(m)$ as input. This implies that

$$\boldsymbol{\Phi}(\mathbf{s}) = F(\mathbf{x}_0, h(\mathbf{x}_0)) = F(\mathbf{x}_0, \omega(\phi(m))) = \mathbf{x}_1 = f(\phi(m)) \in S,$$

as required. Note that in the second equality, we have used (11) and that the last equality is, once again, a consequence of (3). This equality also proves the conjugation (13). ■

3. The starting map and cold-starting of reservoir systems

We now show how the tools that we just introduced can be put to work in the solution of forecasting and path continuation problems for a dynamical system given its observations. The setup of these problems is as follows: suppose that a time series $\left\{\omega(m), \omega(\phi(m)), \ldots, \omega(\phi^{T-1}(m))\right\}$ of length T of ω-observations of an invertible dynamical system $\phi \in \mathrm{Diff}^1(M)$ is provided. In the following paragraphs, we spell out the maps that need to be learned in order to solve the following two problems:

(i) The ***path-continuation*** at horizon $H \in \mathbb{N}$ of the observations. It consists of determining the values $\left\{\omega(\phi^T(m)), \omega(\phi^{T+1}(m)), \ldots, \omega(\phi^{T+H-1}(m))\right\}$.

(ii) The ***forecasting*** of the dynamical system at horizon $H \in \mathbb{N}$. It consists of determining the values $\left\{\phi^T(m), \phi^{T+1}(m), \ldots, \phi^{T+H-1}(m)\right\}$.

If the functional form of the observation ω is known, one can obviously obtain a solution for the first problem out of the solution for the second one.

The solutions to these problems are spelled out in the following theorem in which we assume that we have at our disposal a good reservoir system in the sense of Definition 2.2 with generalized synchronization $f : M \longrightarrow \mathbb{R}^N$ and that, moreover, the pair (ϕ,ω) satisfies the necessary conditions for the delay map $S^{2q+1}_{(\phi,\omega)}$ in (10) to be a continuously differentiable embedding via Takens' Theorem, with $q \in \mathbb{N}$ the dimension of M. The main ingredient of the following theorem is what we call the *starting map* defined as

$$\sigma := f \circ \left(S^{2q+1}_{(\phi,\omega)}\right)^{-1} : S^{2q+1}_{(\phi,\omega)}(M) \subset \mathbb{R}^{2q+1} \longrightarrow \mathbb{R}^N. \tag{14}$$

This terminology is justified by the fact that the starting map produces for each short $(2q+1)$-long history of observations, the unique state value that is consistent with their entire semi-infinite past. Note that if the generalized synchronization f is of the type introduced in (5) and the manifold is compact, then the combination of Takens with the inverse function theorem, together with (9) imply that the starting map σ is differentiable and globally Lipschitz.

The proof of the following theorem is a straightforward consequence of Proposition 2.3.

Theorem 3.1 (*Cold-Started Forecasting Methodology*). *Let $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ be a good reservoir for the ω-observations of the dynamical system $\phi \in \mathrm{Diff}^1(M)$. Let $f : M \longrightarrow \mathbb{R}^N$ be the corresponding embedding GS and let $h : S \longrightarrow \mathbb{R}^d$ be the predicting readout introduced in (11). Let $\{\omega(m), \omega(\phi(m)), \ldots, \omega(\phi^{T-1}(m))\}$ be a sample of ω-observations and assume that $T > 2q+1$. Then:*

(i) *The solution of the **forecasting problem** is given by the following iterations*

$$\phi^{T+j}(m) = f^{-1}\left(F\left(f\left(\phi^{T+j-1}(m)\right), h\left(f\left(\phi^{T+j-1}(m)\right)\right)\right)\right), \quad j = 0, \ldots, H-1, \tag{15}$$

that can be readily initialized at $j = 0$ if the state $\phi^{T-1}(m)$ is known. If only observations are available, then the starting map $\sigma : \mathbb{R}^{2q+1} \longrightarrow S$ defined as $\sigma := f \circ \left(S^{2q+1}_{(\phi,\omega)}\right)^{-1}$ has to be used and applied to a $(2q+1)$-long history of observations preceding the instant $T-1$, which yields:

$$\sigma\left(\omega(\phi^{T-1}(m)), \omega(\phi^{T-2}(m)), \ldots, \omega(\phi^{T-2q-1}(m))\right) = f\left(\phi^{T-1}(m)\right) \tag{16}$$

and can be used to initialize the iterations (15) at $j = 0$.

(ii) *The solution of the **path-continuation problem** is given by the following iterations*

$$\omega\left(\phi^{T+j}(m)\right) = h\left(\mathbf{x}_{T+j-1}\right), \tag{17}$$

$$\mathbf{x}_{T+j-1} = F\left(\mathbf{x}_{T+j-2}, \omega\left(\phi^{T+j-1}(m)\right)\right), \quad j = 0, \ldots, H-1, \tag{18}$$

where (17) is initialized by setting

$$\mathbf{x}_{T-2} = f\left(\phi^{T-2}(m)\right) = \sigma\left(\omega(\phi^{T-2}(m)), \omega(\phi^{T-3}(m)), \ldots, \omega(\phi^{T-2q-2}(m))\right). \tag{19}$$

3.1. The forecasting method and implementation

The forecasting approach contained in Proposition 2.3 and in Theorem 3.1 requires a few ingredients. More explicitly, first, one needs to devise a good reservoir $F : \mathbb{R}^N \times \mathbb{R}^d \longrightarrow \mathbb{R}^N$ for the ω-observations of the dynamical system $\phi \in \mathrm{Diff}^1(M)$, $\dim(M) = q$, under consideration. Second, the corresponding GS $f : M \to \mathbb{R}^N$ and a starting map $\sigma : \mathbb{R}^{2q+1} \to \mathbb{R}^N$, for initializing the states of the reservoir F out of the short, $(2q+1)$-long, histories of the dynamical system's observations, need to be obtained. Finally, in order to tackle the forecasting problem and construct autonomous multi-step predictions, a predicting readout map $h : \mathbb{R}^N \to \mathbb{R}^d$ introduced in (11) needs to be constructed. In the following paragraphs, we spell out the details of the choice of the design for our forecasting experiment in the next section.

The reservoir: We shall be using a leaking *echo state network* (ESN) given by

$$F(\mathbf{x}, \mathbf{z}) := (1-\alpha)\mathbf{x} + \alpha \tanh(A\mathbf{x} + C\mathbf{z}), \tag{20}$$

where $\alpha \in (0, 1]$ is a prespecified *leak rate*, A is a square randomly generated (often from Gaussian distribution) *connectivity matrix* of dimension $N \in \mathbb{N}$, and C is a randomly sampled (often from the uniform distribution) *input matrix* of dimension $N \times d$ that connects the d-dimensional ω-observations of the dynamical system ϕ to the reservoir given by F. The random parameters are sampled such that the sufficient condition $\|A\|_2 < 1$ ($\|\cdot\|_2$ denotes the matrix 2-norm) for the (ϕ, ω)-Echo State Property to hold (see Section 2.2) is satisfied (see, for example, [33]). In practice, since the spectral radius $\rho(A)$ it holds that $\rho(A) \leq \|A\|_2$, it suffices to take $\rho(A) < 1$, which is the most common condition used in the reservoir computing literature. The reservoir map F is parametrized by several hyperparameters such as $\rho(A)$, α, and, for example, the support of the uniform distribution of the entries of the input matrix C. These hyperparameters define the performance of the reservoir and have implications on whether it happens to be good or not in the sense of Definition 2.2. It is customary in the reservoir computing literature that one needs to find optimal reservoir hyperparameters for each particular learning task via some type of (cross-) validation procedure.

The synchronization manifold S and the starting map σ. If the reservoir devised in the previous point is good in the sense of Definition 2.2, then it has an associated GS map $f : M \longrightarrow \mathbb{R}^N$ whose image $S = f(M)$ is an embedded submanifold that is left invariant by the reservoir dynamics. Further, by Theorem 3.1, there exists a map $\sigma : \mathbb{R}^{2q+1} \longrightarrow S$. The map σ thus maps short histories of the dynamical system observations to points in the synchronization manifold S. Hence each point in S corresponds to a unique point in M and also a unique history entailed by Taken's theorem. In Sections 4.1–4.2 we adopt two techniques for the learning of the starting map, namely, (i) a diffusion maps-based methodology [39] which allows learning together with the starting map the associated synchronization manifold out of the data, and (ii) a static feedforward neural network (we refer to it as NN_1). The latter are known to be dense in the set of continuous functions with respect to the topology of uniform convergence [40], which, in particular, guarantees the learnability of the starting map σ since, as we already pointed out in the discussion after (14), this map is differentiable under very general conditions. Section 4.3 contains the robustness analysis results obtained for two techniques for the learning of the starting map.

The forecasting readout: ESNs have been shown to be universal input–output approximants with linear readouts [13,15]. This implies that one can choose the predicting readout h to be a linear map, though any choice of a higher-order polynomial or a neural network function is also possible. A geometric intuition behind the possibility of achieving universal approximation using linear readouts in reservoir computing has been provided in [41,42]. Those references show that some universal reservoir computing families that use linear readouts (the so-called state-affine systems (SAS) [12], in this case) are random projections of Volterra series expansions with semi-infinite inputs. Volterra series is an infinite-dimensional object whose universality has been proved in [30], and the Johnson–Lindenstrauss Lemma [43] can be used to show that universality is preserved under the random projections that yield (universal) SAS. We emphasize that this argument applies exclusively to SAS. An analogous result for ESNs remains an open problem.

In Section 4 we shall be presenting results that are obtained for two choices of readout maps. First, we take the forecasting readout h introduced in (11) to be of a linear functional form, that is, $h(\mathbf{x}) = W^\top \mathbf{x}$, where W is a $N \times d$ matrix. Second, we use a feedforward neural network of relatively simple architecture (referred to as NN_2) as a readout function and denote it by $h^{\mathrm{NN}_2} : \mathbb{R}^N \longrightarrow \mathbb{R}^d$. T-long sample of (partial) d-dimensional observations of a given dynamical system is used to drive the reservoir F starting from the initial state $\mathbf{x}_0 \in \mathbb{R}^N$ chosen to be either a zero vector or a randomly sampled vector. The corresponding T states are collected during this phase, sometimes called the *listening phase* in the literature [44]. To eliminate the influence of the original initialization, the readout map is estimated after discarding the first T_w observations, which is sometimes called the *washout period*. This is the most popular approach in the successful applications of reservoir computing cited in the introduction.

We point out that when working on a path-continuation problem dealing with low-dimensional observations of the dynamical system, we shall most likely be agnostic regarding the dimension q of the data-generating dynamical system ϕ. This is a classical and well-studied problem that appears when using embedding techniques in dynamical systems forecasting; some techniques for the estimation of the dimension q can be found in [45,46] and references therein.

Section 4.3 contains robustness analysis results obtained for two choices of the readout map, that is $h(\mathbf{x}) = W^\top \mathbf{x}$ and $h(\mathbf{x}) = h^{\mathrm{NN}_2}(\mathbf{x})$. This renders the forecasts to be derived as a function h of the iterates of

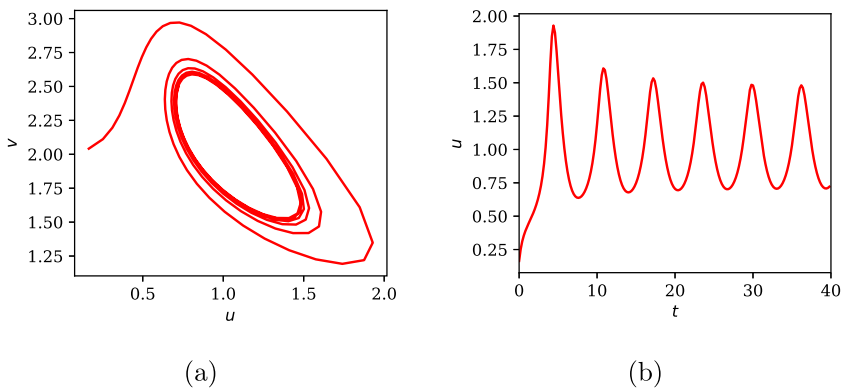


Fig. 1. Representative trajectory of the Brusselator system sampled with $\delta t = 0.2$. Initial conditions are drawn uniformly such that $u_0 \sim \mathcal{U}[0,2]$ and $v_0 \sim \mathcal{U}[0,3]$. (a) Trajectory in phase space of the Brusselator system. (b) u variable evolution of trajectory in (a).

the system in the autonomous run defined by $\mathbf{x}_{n+1} = F(\mathbf{x}_n, h(\mathbf{x}_n))$, where the starting map σ is applied only once to obtain the initial condition (for instance using $\mathbf{x}_n = \sigma\left(\omega(\phi^n(m)), \omega(\phi^{n-1}(m)), \ldots, \omega(\phi^{n-2q}(m))\right)$). Considering two different choices of the readout map, we can more reliably investigate the potential detrimental effects of an imprecise estimation of the starting map on the accuracy of ESN predictions.

The forecasted and path-continued values. They are obtained by using the recursions and the initializations spelled out in (15) and in (18)–(17), respectively. We note that, unlike in the path-continuation problem, the solution of the forecasting problem requires the learning of not only the synchronization map f but also of its inverse f^{-1}. We hence restrict our empirical analysis in Section 4 to the case of the path-continuation learning problem.

Importance of the informed cold-starting. The most important difference between the methodology that we just proposed and the one used in all the above-cited empirical contributions is in the reservoir initializations proposed in the Eqs. (16) and (19) for the forecasting and path-continuation problems, respectively. More explicitly, having obtained the readout map using some chosen loss function and solving the associated empirical risk minimization (ERM) problem (for example, for linear readouts and quadratic losses, the solutions of the corresponding ERM problems are the least squares solutions), one would traditionally reason as follows: given a history of observations for the path-continuation and the forecasting problems, one needs to initialize the reservoir state to construct the predictions. In the traditional approach, the initialization values $\mathbf{x}_{T-1} = f\left(\phi^{T-1}(m)\right)$ and $\mathbf{x}_{T-2} = f\left(\phi^{T-2}(m)\right)$ in (16) and (19), respectively, are obtained by feeding a sequence of observations $\left\{\omega(\phi^{T-n}(m)), \ldots, \omega(\phi^{T-1}(m))\right\}$ into the reservoir that is initialized at an arbitrary state $\mathbf{x}_0 \in \mathbb{R}^N$. Subsequently, the last state is processed with the trained readout map, and the output is used to autonomously run the reservoir for the desired number of future steps of the multi-step path-continuation or forecasting exercise. It is well known that for a short history sample of observations used as inputs, this traditional approach would lead to poor predicting performance of the reservoir since, in this case, the impact of the initialization of the states is very high. More explicitly, consider the iterations

$$\mathbf{x}^n_{T-j}(\mathbf{x}_0) = F\left(\mathbf{x}^n_{T-j-1}(\mathbf{x}_0), \omega(\phi^{T-j-1})\right), \quad j \in \{1, \ldots, n\}, \ \mathbf{x}^n_{T-n-1} = \mathbf{x}_0 \in \mathbb{R}^N.$$

Systems that are traditionally used in RC have the so-called fading memory property [30], and, in particular, the input forgetting property [33], which implies that:

$$\lim_{n\to\infty} \mathbf{x}^n_{T-1}(\mathbf{x}_0) = \mathbf{x}_{T-1} = f\left(\phi^{T-1}(m)\right), \quad \text{for any } \mathbf{x}_0 \in \mathbb{R}^N.$$

We find that our approach offers significant improvements compared to *traditional modus operandi*. More precisely, initializing the reservoir with the image of the learned starting map σ and hence "informing" the original state of the reservoir about the commencing point of our forecasting exercise leads to less data-intensive predictions since no washout periods are needed. Using short histories of observations of length $2q + 2$ for the path-continuation problem and $2q + 1$ for the forecasting problem, we can immediately work out what the next time series value is just by using the iterations (15) or (18)–(17). The cold-starting procedure that we propose in Theorem 3.1 based on learning the starting map σ circumvents the asymptotic traditional approach that may prove costly both from the computational and the data consumption points of view and does not allow to produce high-quality multi-step predictions based on a data of limited length ($2q+2$ and $2q + 1$ for the path-continuation and the forecasting problem, respectively).

4. Empirical results

In this section, we demonstrate the empirical forecasting improvements exhibited by our proposed cold-starting of the reservoir compared to traditional approaches. We shall use two dynamical systems: the Brusselator and the Lorenz systems. The Brusselator is a two-dimensional ($q = 2$) system exhibiting oscillatory dynamics [47] given by

$$\dot{u} = a + u^2 v - (b+1)u,$$
$$\dot{v} = bu - u^2 v,$$

and parametrized by $a = 1$ and $b = 2.1$. For this set of parameters a and b, the only stable attractor of the Brusselator is a stable limit cycle. Fig. 1 provides a representative trajectory in phase space and the temporal evolution of u over time.

The Lorenz system is a dynamical system presenting a simplified three-dimensional model ($q = 3$) for weather prediction [48] and is given by

$$\dot{u} = a(v - u),$$
$$\dot{v} = bu - uw - v,$$
$$\dot{w} = uv - cw,$$

where we use the parameters $a = 10$, $b = 28$, and $c = 8/3$. For this set of parameters, the dynamics of the Lorenz system exhibits chaotic motion. In Fig. 2, a projection of the phase space on the u-v plane and the temporal evolution of u are provided.

For these systems, we assume that only the first coordinate observations are available for the learning, and we are interested in their path continuation for H steps into the future based on $2q + 1$ past observations. The ESN reservoir systems as in (20) are implemented, and the forecasting method discussed in Section 3.1 is followed for the path continuation exercise.

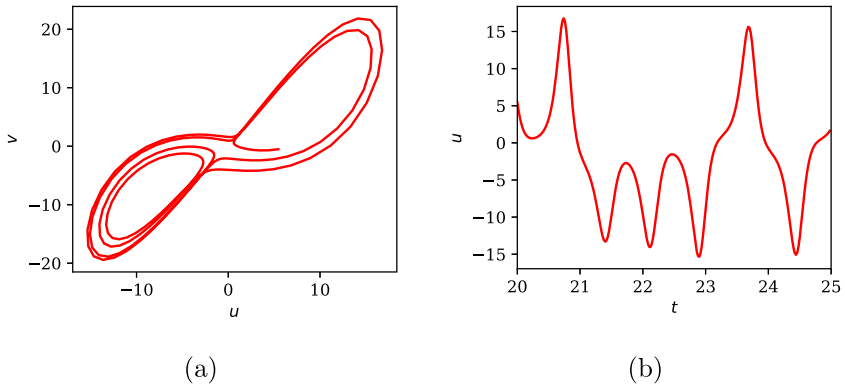


Fig. 2. Representative trajectory of the Lorenz system sampled with $\delta t = 0.2$. Initial conditions $u_0 \sim \mathcal{N}(10,1)$, $v_0 \sim \mathcal{N}(1,1)$, and $w_0 \sim \mathcal{N}(0,1)$. (a) Projection of this trajectory onto the u-v plane. (b) u variable evolution of trajectory in (a).

4.1. Diffusion maps-based learning of the synchronization manifold and the starting map

Reservoir designs. We recall that the reservoir in (20) is parametrized by several hyperparameters, namely, the spectral radius $\rho(A)$ of connectivity matrix A that is often chosen to satisfy $\rho(A) < 1$, leak rate $\alpha \in [0,1)$, and the scale of the input matrix C. We obtain reservoir hyperparameters for the path continuation of the Brusselator and the Lorenz systems with the Optuna optimization framework [49].

Brusselator: We take the reservoir of dimension $N = 1024$ and leak rate $\alpha = 0.506$. The connectivity matrix A entries are randomly sampled as standard normal, and the matrix is normalized such that its spectral radius satisfies $\rho(A) = 0.98$. The entries of the input matrix C is drawn from $\mathcal{U}[-0.5, 0.5)$ distribution and subsequently scaled by 0.128.

Lorenz: We use $N = 2048$ and $\alpha = 0.501$. The connectivity matrix A entries are standard normal, and A is normalized such that $\rho(A) = 0.80$. The entries of the input matrix C is also drawn from $\mathcal{U}[-0.5, 0.5)$ distribution and subsequently scaled by 0.179.

Collecting the reservoir states and setting the path-continuation task. For each system, for some chosen initial conditions, T input observations of the first coordinate u ($d = 1$) discretized at δt are collected. We discard the first T_w-long washout of the states and use $T_{\text{tr}} := T - T_w - 1$ to run the estimation procedure as we explain below.

Brusselator: We choose initial conditions $u_0 \sim \mathcal{U}[0,2)$ and $v_0 \sim \mathcal{U}[0,3)$ and collect 600 trajectories sampled with $\delta t = 0.2$ for 30 dimensionless time units, which results in $T = 150$. We discard the first $T_w = 1$ washout discretized steps. This results in 600 pairs of $T_{\text{tr}} = 148$-long training paths. For the testing phase, we create trajectories with initial conditions drawn from the same uniform distribution but recorded for 40 dimensionless time units (200 discrete steps). One testing trajectory is depicted in red in Fig. 3(a).

Lorenz: We sample initial conditions $u_0 \sim \mathcal{N}(10,1)$, $v_0 \sim \mathcal{N}(1,1)$, and $w_0 \sim \mathcal{N}(0,1)$. Subsequently, we sample 600 trajectories for training. For each trajectory, we sample for 2 dimensionless time units between $t_{\min} = 20$ and $t_{\max} = 22$ steps with $\delta t = 0.02$, which results in $T = 100$ discrete time observations. Out of those trajectories, we discard the first $T_w = 20$ (0.4 in the intrinsic time of the system) discretized washout steps. This results in $M = 600$ pairs of $T_{\text{tr}} = 79$ training paths (1.6 in the characteristic time of the system). For testing, we use sample trajectories with $t_{\min} = 20$ and $t_{\max} = 25$ using $\delta t = 0.02$, resulting in trajectories consisting of 250 time steps (of a duration of 5 dimensionless time units). One such trajectory is shown in red in Fig. 3(b).

For all M trajectories, we collect the associated states into $X := (\mathbf{x}_{T_w+1}^{(1)} | \cdots | \mathbf{x}_{T_w+T_{\text{tr}}}^{(1)} | \cdots | \mathbf{x}_{T_w+1}^{(M)} | \cdots | \mathbf{x}_{T_w+T_{\text{tr}}}^{(M)}) \in \mathbb{R}^{N \times MT_{\text{tr}}}$ using de-meaned states, as well as $\boldsymbol{U} := (u_{T_w+2}^{(1)} | \cdots | u_{T_w+T_{\text{tr}}+1}^{(1)} | \cdots | u_{T_w+2}^{(M)} | \cdots | u_{T_w+T_{\text{tr}}+1}^{(M)})^\top \in \mathbb{R}^{MT_{\text{tr}}}$ using de-meaned one-step ahead true observations of the first coordinate.

Training the forecasting readout h. In this empirical exercise, we assume the readout map h to be linear, that is $h(\mathbf{x}) = \mathbf{W}^\top \mathbf{x}$, for $\mathbf{W}, \mathbf{x} \in \mathbb{R}^N$. The estimated readout is given by the following solution of the ridge regression:

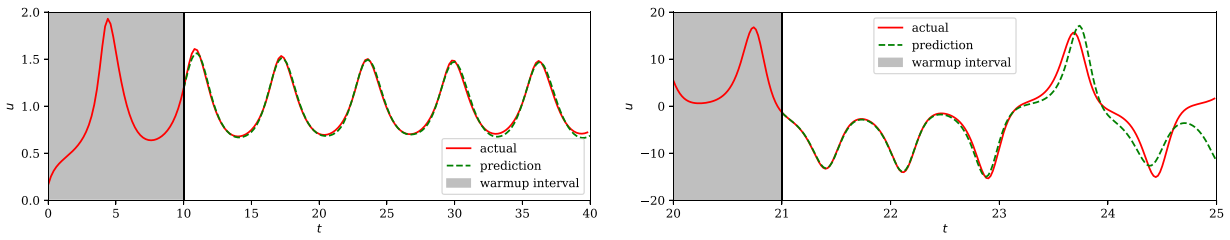
$$\widehat{\mathbf{W}}_{\text{ridge}} = \underset{\mathbf{W} \in \mathbb{R}^N}{\arg\min} \left\{ \frac{1}{MT_{\text{tr}}} \sum_{m=1}^{M} \sum_{t=T_w+1}^{T_w+T_{\text{tr}}} \left(u_{t+1}^{(m)} - \mathbf{W}^\top \mathbf{x}_t^{(m)} \right)^2 + \lambda \|\mathbf{W}\|_2^2 \right\}$$
$$= (XX^\top + \lambda \mathbb{I}_N)^{-1} X \boldsymbol{U}^\top, \tag{21}$$

with the ridge regularization penalty $\lambda > 0$. The estimated readout is hence defined by $\widehat{h}_{\text{ridge}}(\mathbf{x}) = \widehat{\mathbf{W}}_{\text{ridge}}^\top \mathbf{x}$, $\mathbf{x} \in \mathbb{R}^N$. We choose the ridge regularization penalty values $\lambda = 0.01$ and $\lambda = 0.001$ for the Brusselator and for the Lorenz system, respectively. We notice here that even though (21) provides a closed-form solution for the linear readout, still a relatively large number of readout weights needs to be estimated. Although this estimation takes place once, and the readout map is subsequently used for all path-continuation exercises for each dynamical system, one can be interested in reducing the amount of data used for obtaining $\widehat{\mathbf{W}}_{\text{ridge}}$. Some approaches have been developed in the literature to that purpose. For example, [50] hypothesize that $\widehat{\mathbf{W}}_{\text{ridge}}$ resides on a submanifold of much lower dimensionality and propose a feedforward deep autoencoder to learn it. Combining our proposed framework with techniques that allow reducing the linear readout dimensionality can be an interesting direction of research.

Once the readout map $\widehat{h}_{\text{ridge}}$ is available, we compare the performance of the autonomous multi-step path-continuation of the first coordinate history for each of the systems adopting **(i)** the traditional way of initializing the states of the reservoir for the predicting exercise, **(ii)** using the starting map proposed in this paper.

Traditional approach of initializing the states of the reservoir. We showcase the traditional way **(i)** of initializing the states of the reservoir for the case of the Brusselator system. The standard *modus operandi* consists of initializing the states with any arbitrary starting value, for example, with a randomly sampled or a zero vector (as in our case), forcing the reservoir with a warmup trajectory, collecting the last state corresponding to the last observation of the history of observations that needs to be continued and thereby autonomously iterating the reservoir system with the prior trained readout map to produce H forecasts.

We attempt to path-continue the testing trajectories of the first coordinate observations of the Brusselator system for $H = 150$ steps into the future (this corresponds to the 30 steps in the system's time) and illustrate it in Fig. 3(a). The trained reservoir is warmed up by providing an initial warmup u trajectory of length 50 steps (10 dimensionless time units) as input to the model, indicated by the gray-shaded region. We take the warmup long enough to approximately washout the influence of the initialization. The ESN is then used in an autoregressive fashion for $H = 150$ steps (30 dimensionless time units), producing forecasts



(a) Brusselator (b) Lorenz

Fig. 3. Autonomous path-continuing of the partial observations of the dynamical system produced by the ESN with the readout $\widehat{h}_{\text{ridge}}$ and with the initial zero state compared to the true trajectory. The shaded area marks the part of the path which is used as a history to drive the trained ESN and the black line shows the moment when the subsequent autonomous path-continuation starts.

for the initial input time series. The produced forecasts (dashed green curve) are shown together with the actual dynamics of u (solid red curve). The figure thus illustrates that the trained reservoir model is able to accurately continue the dynamics. In addition, a warmup length of only 50 time steps (10 dimensionless time units) is sufficient to synchronize the internal reservoir states to the input trajectory.

We repeat the same procedure for the Lorenz system. Given a testing trajectory consisting of 250 time steps (of a duration of 5 dimensionless time units) of the first coordinate, first, we use trained reservoir using a warmup length of 50 steps. This warmup period is shown in Fig. 3(b) (1 dimensionless time units) with a shaded gray area. Again, we choose a long warmup length to washout the effect of the reservoir initialization. Similarly to the case of the Brusselator system, the ESN is used autonomously for $H = 200$ steps (4 dimensionless time units) to yield the forecasts (dashed green curve) of the actual dynamics of u (solid red curve).

We emphasize that, in general, our goal is to be able to produce accurate path-continuation using only a minimally short history of these observations, for example, $2q+1$ (which corresponds to 5 for the Brusselator system and to 7 for the Lorenz one), which is possible with our cold-starting technique. For the traditional approach, though, this number of observations is insufficient to remove the influence of the arbitrary starting initialization. Fig. 5(a) and Fig. 5(b) demonstrate this scenario for the Brusselator and the Lorenz system, respectively.

Initializing the reservoir states with the diffusion maps-learned starting map. We now refer to our cold-starting technique using our proposed starting map σ. In this empirical exercise, we follow the same approach as suggested for LSTM networks in [29]. More specifically, we apply diffusion maps to input time series windows of length $2q+1$ (5 for the Brusselator system and 7 for the Lorenz case), sampled from the training trajectories to learn the data manifold as a first step [51]. We compute independent diffusion modes that span the data manifold (see Appendix A for the detailed calculation of these modes). More precisely, for the Brusselator system the data manifold is spanned by two independent diffusion modes, which is in agreement with the dimension of the original dynamical system. We hence compute modes $\boldsymbol{v}^{(1)}$ and $\boldsymbol{v}^{(2)}$ that are depicted in Fig. 4(a), where each dot corresponds to a time series window of u of length 5. Similarly, for the Lorenz system, three independent diffusion modes span the data manifold. A projection on the first two independent modes $\boldsymbol{v}^{(1)}$ and $\boldsymbol{v}^{(2)}$ is shown in Fig. 4(b), with each dot associated to 7 subsequent observations of u. For each of the training trajectories, we also produce trajectories of forced internal reservoir states. We thus obtain for each time series window also corresponding (approximately warmed-up) internal states $\mathbf{x}_i$. In Figs. 4(a)–4(b), we color each window with one hidden state variable $\mathbf{x}_0$ that corresponds to the last time step of this window.

Next, we learn a mapping from the q-dimensional data manifold to warmed-up internal states of the reservoir using geometric harmonics as it is done for the case of LSTM recurrent neural networks in [29].

Finally, we create a mapping from the q-dimensional data manifold to the corresponding internal states of the reservoir. The states are thereby obtained by forcing the reservoir with the training time series, whereas the mapping is again created by fitting geometric harmonics.

We can now use the diffusion maps-learned starting map to find the initialization of the reservoir states for any new short input time series window of length $2q+1$.

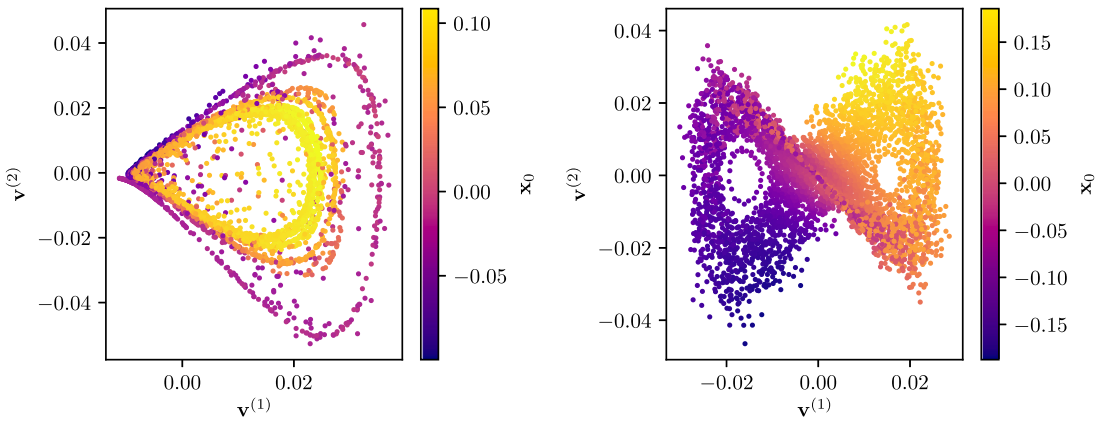
Autonomous forecasting with a diffusion maps cold-started reservoir. We notice in Fig. 5 that H steps ahead autonomous predictions produced by the reservoir, which is cold-started with our proposed starting map, are much more accurate than the one produced by the traditionally initialized one. Indeed, We can now compare the efficacy of our initialization approach versus the traditional warmup approach. For a short warmup period of just $2q+1$ steps, the prediction results are depicted in Fig. 5(a) and in Fig. 5(b), respectively. Note that the classical initialization approach leads to a fast divergence of the predicted and true dynamics since this number of steps seems to be insufficient to properly warm up the reservoir. In contrast, using our initialization approach, we obtain forecasts that stay true to the actual dynamics for a long time horizon. Note that due to the approximation errors of the trained reservoir and the starting map, as well as the chaotic nature of the dynamics, predictions will eventually diverge.

4.2. Neural network learning of the synchronization manifold and the starting map

As opposed to the previous section, where the diffusion maps procedure was used to obtain the starting map σ and the linear readout was trained using a ridge regression that admits a closed-form solution, here, instead, we consider other techniques for these steps. In particular, neural network models are employed for approximating σ and the reservoir readout h. In order to distinguish between the two neural networks used for these two different purposes, we will call them the cold-starting and the readout neural networks and will refer to them as NN_1 and NN_2, respectively.

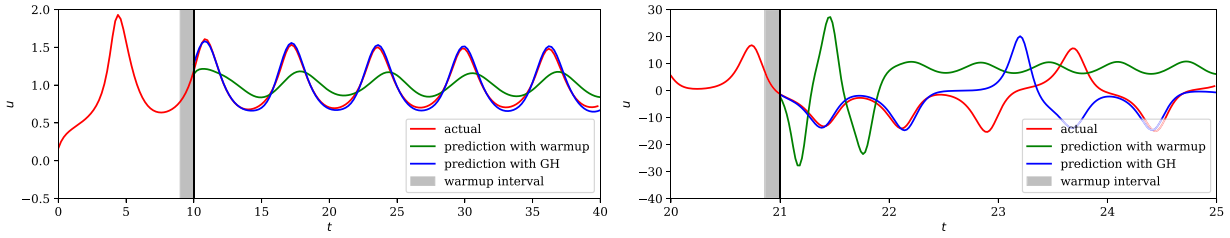
Reservoir design. For both examples of dynamical systems, we consider the same echo state network defined in (20) with the state dimension $N = 900$, and the connectivity (reservoir) matrix A and the input matrix C randomly sampled from $\mathcal{U}[0,1)$ distribution. We normalize C such that $\|C\|_2 = 1$ and, to satisfy the sufficient condition for the echo state property, we also normalize A such that its spectral radius $\rho(A) = 0.99$. We choose the leak rate to be $\alpha = 0.7$.

Collecting the reservoir states and setting the path-continuation task. To collect the training set for the readout neural network NN_2, analogously to the case of the linear readout in Section 4.1, we use a random initial condition for the given discretized dynamical system. One trajectory of length $T = 5000$ of the first coordinate is used as the input of the reservoir system $\{u_t\}_{t\in\{1,\dots,T\}}$ and to collect the corresponding T states of dimension 900. The washout period of the



(a) Brusselator (b) Lorenz

Fig. 4. Diffusion maps embedding of the system time series windows of length $2q+1$ of the training data: (a) independent diffusion maps modes $\boldsymbol{v}^{(1)}$ and $\boldsymbol{v}^{(2)}$; (b) projection onto diffusion maps modes $\boldsymbol{v}^{(1)}$ and $\boldsymbol{v}^{(2)}$. The color corresponds to one warmed-up internal state variable ($\mathbf{x}_0$, one of the internal reservoir states, $N = 1024$ for (a) and $N = 2048$ for (b)).



(a) Brusselator; $H = 150$ (30 dimensionless units) (b) Lorenz; $H = 200$ (4 dimensionless units)

Fig. 5. Representative trajectory of the test data (red). The warmup period is of length $2q+1$ (gray-shaded region). Green - H-steps ahead autonomous predictions of the ESN with the states initialized as zero vectors and warmup used. Blue - H-steps ahead of autonomous predictions of the ESN with the states initialized with the geometric harmonics (GH) method.

length $T_w = 1000$ is discarded, and only $T_{\text{tr}} = T - T_w - 1$ pairs of the states and their corresponding one-step ahead values of the trajectory are used to construct the training set of the length T_{tr}.

Training the forecasting readout h. The neural network weights $\boldsymbol{\theta}$ are estimated via minimizing the quadratic loss empirical risk on this training set, that is:

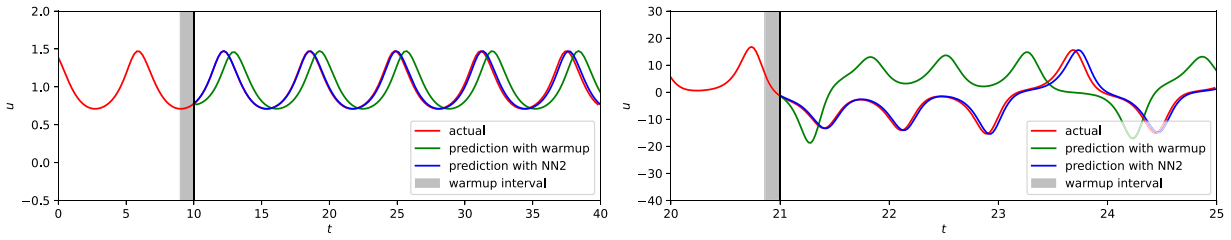
$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left\{ \frac{1}{T_{\text{tr}}} \sum_{t=T_w+1}^{T_w+T_{\text{tr}}} \left(u_{t+1} - h^{\text{NN}_2}_{\boldsymbol{\theta}}(\mathbf{x}_t) \right)^2 \right\}. \tag{22}$$

In the following we use simply h^{NN_2} to refer to the trained neural network readout map $h^{\text{NN}_2}_{\widehat{\boldsymbol{\theta}}}$.

Initializing the reservoir states with the neural network-learned starting map. In the case of the cold-starting neural network NN_1, we proceed as follows. We first observe that the co-domain of the starting map σ has a large Euclidean dimension (for example, $N = 900$ in our experiment). We hence approximate its image using the K leading principal components (K is arbitrarily chosen; for example, we use $K = 100$ in our study) obtained by the principal component analysis (PCA). We define a map $P_K : \mathbb{R}^{2q+1} \longrightarrow \mathbb{R}^K$, so that $P_K(\omega(\phi^{t-1}(m)), \omega(\phi^{t-2}(m)), \ldots, \omega(\phi^{t-2q-1}(m)))$ contains the K leading principal component values of $f(\phi^{t-1}(m))$. To construct the training set for NN_1 that approximates P_K, we collect the states of the reservoir and discard the first T_w observations (washout) in the same manner as in Section 4.1. We compute the projection of the collected data onto the K leading principal components and denote them as $\mathbf{x}_t^K \in \mathbb{R}^K$ for all t in the training set (we emphasize that the states are collected after a long enough washout). We use the same notation as above and for every state $\mathbf{x}_t$ denote by $\boldsymbol{\omega}_t := (\omega(\phi^{t-1}(m)), \omega(\phi^{t-2}(m)), \ldots, \omega(\phi^{t-2q-1}(m)))$ its corresponding history of the inputs-observations. The pairs $(\boldsymbol{\omega}_t, \mathbf{x}_t^K)$ for all t in the training set are used for the cold-starting neural network NN_1 optimization. Once NN_1 is trained, for any new short history of observations, the projection of the corresponding reservoir state onto its K leading principal components is obtained with the neural network. Thus, we derive an approximation σ^{NN_1} to the image of σ to be the inverse of the PCA transform acting on this vector in $\mathbb{R}^K$ and producing the corresponding vector of N initialized states.

Neural network architectures for the learning of σ **and** h. Throughout our experiments, we use a feedforward neural network used to approximate map P_{100} – the network is constructed with 4 hidden layers with 500 neurons each. The activation function on the input and hidden layers is the ReLU function built into Keras, whereas the output layer has no activation function. Training is accomplished using the Adam optimizer, minimizing the mean square error as the loss function. The network is trained using the ReduceLROnPlateau callback function of Keras, which monitors the value of the loss function on the validation set and reduces the learning rate when that loss reaches a plateau. The initial learning rate is set to 0.001, which is halved whenever a plateau of at least 50 epochs is reached. While learning P_{100} with NN_1, we use 500 training epochs and a batch size of 500. The readout h^{NN_2} was also obtained by training the feedforward network of the same architecture (modulo the dimensions of the inputs and outputs). While training NN_1 for P_{100} and NN_2 for h^{NN_2}, 20% of the training length was used for validation.

Autonomous forecasting with a neural networks colds-tarted reservoir. Fig. 7 reports the H steps ahead autonomous predictions produced by the reservoir which is cold-started with the neural network-



(a) Brusselator; $H = 150$ (30 dimensionless units) (b) Lorenz; $H = 200$ (4 dimensionless units)

Fig. 6. Representative trajectory of the test data (red). The warmup period is of length $2q + 1$ (gray-shaded region). Green - H-steps ahead autonomous predictions of the ESN with the states initialized as zero vectors and warmup used. Blue - H-steps ahead autonomous predictions of the ESN with the states initialized with the neural networks method.

learned starting map. Again, we notice that these predictions are much more accurate than the ones produced by the traditionally initialized reservoir. For a short warmup period of $2q + 1$ steps, the predictions are plotted in Fig. 6(a)- Fig. 6(b). Similarly to the case of Fig. 5, the classical initialization approach leads to a fast divergence of the predicted dynamics with respect to the true one. At the same time, the cold-started reservoir produces accurate forecasts for the considered forecasting horizon.

4.3. Robustness of learning the starting map

In this section, we empirically study the robustness of our proposed approach with respect to the initialization of the reservoir states using the starting map σ. More specifically, we explore the sensitivity of the forecasts produced by the reservoir systems to potential imprecision in the learning of the starting map. In the following paragraphs, we show that the results obtained with the help of cold-starting initialization of the reservoir systems do not depend much on the particular choice of the learning method and its high precision. To exemplify this claim, we conduct a series of empirical exercises for the Lorenz dynamical system. We perform the following steps:

1. *Learning the starting map* σ: (a) with a neural network of a given architecture and denote it σ^{NN_1}; (b) with a diffusion maps-based method and denote it σ^{DM}.
2. *Choosing the reservoir readout map* h: (a) via learning with another neural network trained as in (22) and denoted h^{NN_2}; (b) as a linear map in (21) and denoted h_{ridge}.
3. *Cold-starting (initializing) the reservoir with the short past history of observations.* Take a set of 10 arbitrary chosen $2q + 1 = 7$ partial subsequent observations $\boldsymbol{\omega}_k \in \mathbb{R}^{2q+1}$, $k = 1, \ldots, 10$, of the Lorenz system to construct the initial reservoir states either with the help of the neural network-learned starting map $\mathbf{x}_{k,0}^{\sigma^{\mathrm{NN}_1}} = \sigma^{\mathrm{NN}_1}(\boldsymbol{\omega}_k)$, or via taking the image of the diffusion maps-learned starting map $\mathbf{x}_{k,0}^{\sigma^{\mathrm{DM}}} = \sigma^{\mathrm{DM}}(\boldsymbol{\omega}_k)$, $k = 1, \ldots, 10$.
4. *Constructing perturbation terms.* Construct a set of 100 equally distanced values $\sigma_\eta^2 \in [0, 0.03]$. For each $\sigma_\eta^{j^2}$, $j = 1, \ldots, 100$, a sample of $K = 10$ random innovations $\{\eta_k^j\}_{k\in\{1,\ldots,K\}}$, $\eta_k^j \sim \mathcal{U}\{0, \sqrt{12}\sigma_\eta^j\}$, is drawn.
5. *Perturbing the initial states.* Each of the cold-started states is perturbed by the additive noise in (5), that is, $\widetilde{\mathbf{x}}_{k,0}^{\sigma^{\mathrm{NN}_1}} := \mathbf{x}_{k,0}^{\sigma^{\mathrm{NN}_1}} + \eta_k^j$, and $\widetilde{\mathbf{x}}_{k,0}^{\sigma^{\mathrm{DM}}} := \mathbf{x}_{k,0}^{\sigma^{\mathrm{DM}}} + \eta_k^j$, $k = 1, \ldots, 10$, $j = 1, \ldots, 100$.
6. *Autonomous run of the cold-started reservoir.* The corresponding learned readout map is applied to the perturbed initial states, and the reservoir system is run autonomously to produce $H = 50$ future steps of the path-continued trajectory. More precisely, following (18)–(17), for the case of the diffusion maps method the autonomous path-continuation is conducted as

$$\omega\left(\phi^{j+1}(m)\right) = h_{\mathrm{ridge}}(\widetilde{\mathbf{x}}_{k,j}^{\sigma^{\mathrm{DM}}}) = \widehat{\mathbf{W}}_{\mathrm{ridge}}^{\top}\widetilde{\mathbf{x}}_{k,j}^{\sigma^{\mathrm{DM}}},$$

$$\widetilde{\mathbf{x}}_{k,j+1}^{\sigma^{\mathrm{DM}}} = F\left(\widetilde{\mathbf{x}}_{k,j}^{\sigma^{\mathrm{DM}}}, \omega\left(\phi^{j+1}(m)\right)\right), \quad j = 0, \ldots, H-1,$$

while for the neural networks instance via

$$\omega\left(\phi^{j+1}(m)\right) = h^{\mathrm{NN}_2}(\widetilde{\mathbf{x}}_{k,j}^{\sigma^{\mathrm{NN}_1}}),$$

$$\widetilde{\mathbf{x}}_{k,j+1}^{\sigma^{\mathrm{NN}_1}} = F\left(\widetilde{\mathbf{x}}_{k,j}^{\sigma^{\mathrm{NN}_1}}, \omega\left(\phi^{j+1}(m)\right)\right), \quad j = 0, \ldots, H-1.$$

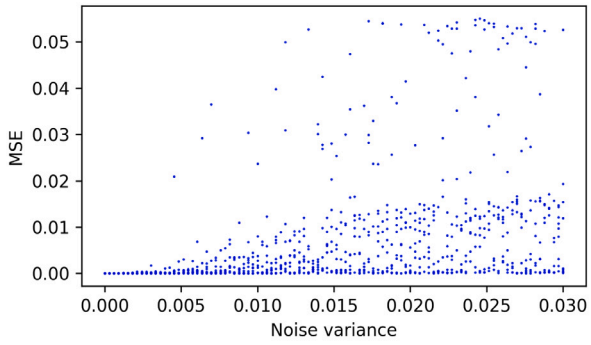
7. *Performance assessment.* The mean squared error of the $H = 50$ autonomous predictions is computed per each perturbed state and the corresponding innovation, which results in 1000 measurements which are subsequently plotted using the scatter plot versus the corresponding values of σ_η^j, $j = 1, \ldots, 100$.

Fig. 7 shows that the dependence of the mean squared forecasting errors as a function of the variance of the perturbing innovations for all the chosen sets of partial subsequent observations $\boldsymbol{\omega}_k \in \mathbb{R}^{2q+1}$, $k = 1, \ldots, 10$, is $O(\sigma_\eta^2)$ for both the techniques for the learning of the starting map proposed in the paper. We emphasize that when the readout is chosen as a neural network function, the additive perturbation of the initial cold-started states is nonlinearly transformed by the readout. Hence, one expects that the perturbations introduced with respect to the true images of the starting maps get nonlinearly amplified by the neural network readout at the time of autonomous forecasting. However, this is not observed to have detrimental consequences for the autonomous path continuation, as seen in Fig. 7(b). These empirical observations serve as evidence of the robustness of our proposed technique of reservoir initialization.

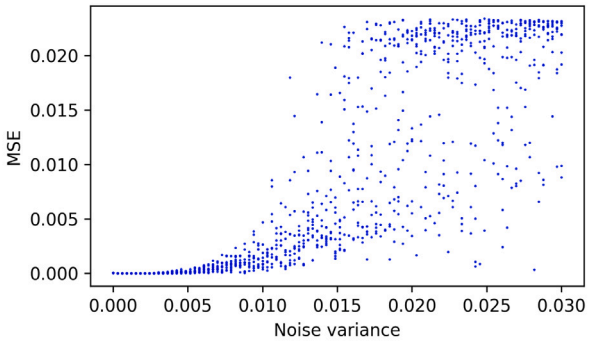
The reader may note that we have not used Lyapunov exponents of the autonomous system resulting from a cold-start to ascertain the robustness of the starting map. This is because the Lyapunov exponents, while reflecting on the magnitude of the exponent reflecting the time scale on which system dynamics become unpredictable, would depend on the error that would have incurred while learning the readout rather than the error that would have incurred in the cold-start. This contrasts the error in the short-term prediction of the learned reservoir with the long (and even infinite) time accuracy of its approximation of the original problem. Given the sensitivity to initial conditions and the lack of guarantees for the smooth dependence of the Lyapunov exponents to small system identification errors, asking for accurate Lyapunov exponent approximation lies beyond the scope of the present work.

5. Conclusion

While observing a solution of an initial value problem with an ordinary differential equation or while iterating a map on an initial condition, one can start observing the solution right away. The aforementioned amenity was not accessible for forecasting with an echo state network model since, even in their autonomous mode, they had to be driven by a not-so-short history of the very trajectory that one wanted the model to forecast. We have overcome this challenge with the notion of a cold start. By employing a small segment of the partial



(a) MSE for diffusion maps-learned starting map σ^{DM} and linear readout h_{ridge}.



(b) MSE for neural networks-learned starting map σ^{NN_1} and neural network readout h^{NN_2}.

Fig. 7. Lorenz system results: mean squared error (MSE) calculated over $H = 50$ (10 dimensionless units) versus the variance σ_η^2 of the perturbation of the state initialization suggested by the cold-start map; 10 experiments are conducted for each perturbation error and the perturbation error is varied in the interval 0 to 0.03 with a step-size of 0.03/100.

observations (enough to determine a unique state of the underlying dynamical system) as the initial condition, and using a starting map, we show that it is theoretically possible to initialize the internal state of the reservoir, enabling forecasting by iteration from that internal state when the network is run in autonomous mode. We have also pointed out the natural conditions that entail that the starting map is well-behaved in the sense that it is a Lipschitz function which also justifies the numerically observed robustness of its learning.

From the larger perspective of modeling differential equations, the "well-trained, well-initialized" reservoir is a numerical approximation of the actual dynamical system. Therefore, the notion of shadowing property would be needed to compare the trajectories of dynamical systems with their numerical approximations [52–55]. Some of the authors are currently researching this topic.

CRediT authorship contribution statement

Lyudmila Grigoryeva: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Boumediene Hamzi:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Felix P. Kemeth:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Yannis Kevrekidis:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **G. Manjunath:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Juan-Pablo Ortega:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Matthys J. Steynberg:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: G Manjunath reports financial support was provided by National Research Foundation. Boumediene Hamzi reports financial support was provided by Air Force Office of Scientific Research. Boumediene Hamzi reports financial support was provided by US Department of Energy. Yannis Kevrekidis reports financial support was provided by Defense Advanced Research Projects Agency. Yannis Kevrekidis reports financial support was provided by Air Force Office of Scientific Research. Felix P. Kemeth reports financial support was provided by Defense Advanced Research Projects Agency. Felix P. Kemeth reports financial support was provided by Air Force Office of Scientific Research. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We included the link to the GitHub folder that contains the code and data used in the paper.

Acknowledgments

LG and GM thank the hospitality of Nanyang Technological University, where part of this work was completed. GM acknowledges partial funding through an incentive grant, UID 150668 from the NRF, South Africa. JPO acknowledges partial financial support from the School of Physical and Mathematical Sciences of the Nanyang Technological University, Singapore and thanks the hospitality of the University of St. Gallen. BH acknowledges financial support from the Air Force Office of Scientific Research, USA under MURI award number FA9550-20-1-0358 (Machine Learning and Physics-Based Modeling and Simulation) and the Department of Energy, USA under the MMICCs SEA-CROGS award. The work of YK and FK was partially supported by DARPA, USA and the US Air Force Office of Scientific Research.

Appendix A. Diffusion maps

The diffusion maps parametrization technique provides a strategy for the dimensionality reduction of a finite dataset, $X = \{\mathbf{x}_i\}_{i=1}^n$, where each $\mathbf{x}_i \in \mathbb{R}^m$ is a sample from a manifold M [39]. The first step in

the diffusion maps method involves establishing a random walk across the dataset. This is facilitated by the creation of an affinity matrix $K \in \mathbb{R}^{n\times n}$, which represents the connections among the points in X. The elements of this matrix, K_{ij}, are calculated using a kernel, here a Gaussian kernel, according to:

$$K_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\epsilon}\right),$$

where $\|\cdot\|$ denotes the chosen norm for the data, in this case, the L_2 norm. The hyperparameter $\epsilon > 0$ controls the decay rate of the kernel: for smaller values of ϵ, only proximal points are considered connected in K, as K_{ij} approaches 0 for distant points.

The diffusion maps algorithm hinges on the normalized graph Laplacian of the data converging to the Laplace–Beltrami operator on the manifold M as the number of points $n \to \infty$ and $\epsilon \to 0$. However, a specific normalization is required for data obtained from non-uniformly sampled points to accurately recover the Laplace–Beltrami operator. This involves defining a diagonal matrix $D \in \mathbb{R}^{n\times n}$, with $D_{ii} = \sum_{j=1}^{n} K_{ij}$, and then calculating the normalized affinity matrix, given by

$$\widetilde{K} = D^{-\kappa} K D^{-\kappa},$$

where κ modulates the density effect. For $\kappa = 0$, the density's influence is maximal, suitable only for uniformly sampled data, whereas $\kappa = 1$ removes the density effect, enabling the recovery of the Laplace–Beltrami operator [56]. Another normalization step yields S, a Markovian matrix, by dividing each entry of $\widetilde{K}$ by the sum of its rows. The eigendecomposition of S reveals a complete set of real eigenvectors $\boldsymbol{v}^{(i)}$ and eigenvalues λ_i, facilitating a nonlinear parametrization of the dataset X in terms of these eigenvectors. Selecting the leading eigenvectors that are independent/non-harmonic generates a set of latent variables $\boldsymbol{\Phi} = \{\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(d)}\}$ that encapsulate the intrinsic geometry of the manifold from which the dataset was sampled. If the number of these selected eigenvectors d is less than the original variable dimensions m, the process effectively reduces dimensionality by presenting a more simplified representation of the dataset. For a dataset X comprising short time series windows u_t, diffusion maps enable the extraction of reduced latent variables in a data-driven manner, with $\kappa = 0$ and ϵ chosen as the median of all pairwise distances, ensuring that the choice of α does not qualitatively alter the diffusion map results.

Appendix B. Geometric harmonics

Geometric harmonics is utilized to extend a function $\mathcal{F}$, potentially vector-valued, sampled at certain points $X = \{\mathbf{x}_i\}$ on a manifold M, to a new point $\mathbf{x}_{\text{new}} \notin X$ [39]. In this context, a modified approach of geometric harmonics is employed to interpolate $\mathcal{F}$ using the reduced coordinates $\boldsymbol{\Phi}$ identified through diffusion maps. Specifically, after the dimensionality reduction phase yields non-harmonic eigenvectors, the goal is to express $\mathcal{F}$ in terms of these reduced coordinates $B = (\boldsymbol{v}^{(1)}|\boldsymbol{v}^{(2)}|\dots|\boldsymbol{v}^{(d)}) \in \mathbb{R}^{n\times d}$ with $\boldsymbol{v}^{(j)} \in \mathbb{R}^n$. Despite the exclusion of harmonic eigenvectors, a subsequent application of diffusion maps to the coordinates $\boldsymbol{\Phi}$ facilitates the creation of a functional basis connecting $\boldsymbol{\Phi}$ to any function $\mathcal{F}$ defined on the original space.

Similar to the initial diffusion maps process, the first step involves calculating an affinity matrix $C_{i,j} = C(\boldsymbol{b}_i, \boldsymbol{b}_j) = \exp\left(-\frac{\|\boldsymbol{b}_i - \boldsymbol{b}_j\|_2^2}{2\epsilon'}\right)$, where $\boldsymbol{b}_i \in \mathbb{R}^d$ denotes the ith row of the matrix B. Being symmetric and positive semidefinite, C possesses orthonormal vectors $\boldsymbol{\psi}^{(1)}, \boldsymbol{\psi}^{(2)}, \dots, \boldsymbol{\psi}^{(n)}$ and non-negative eigenvalues $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$. These eigenvectors serve as a projection basis for extending a function $\mathcal{F}$. Selecting a threshold $\delta > 0$, the set of significant eigenvalues $S_\delta = \{\alpha : \sigma_\alpha > \delta\sigma_1\}$ is determined, where δ is chosen such that $d < \text{Card}(S_\delta) < n$.

Projecting the image of $\mathcal{F}$ onto this truncated eigenvector set yields an approximation $\mathcal{F} \approx P_\delta \mathcal{F} \equiv \widetilde{\mathcal{F}} = \sum_{\alpha\in S_\delta} \boldsymbol{\psi}^{(\alpha)} (\widetilde{\mathcal{F}}^T \boldsymbol{\psi}^{(\alpha)})^\top$.

To extend $\widetilde{\mathcal{F}}$ to a new coordinate $\boldsymbol{b}_{\text{new}}$, which is not one of the rows of B, the extension is given by $\widetilde{\mathcal{F}}_{\text{new}}(\boldsymbol{b}_{\text{new}}) = \sum_{\alpha\in S_\delta} \boldsymbol{\psi}^{(\alpha)}_{\text{new}} (\widetilde{\mathcal{F}}^\top \boldsymbol{\psi}^{(\alpha)})^\top$, with $\boldsymbol{\psi}^{(\alpha)}_{\text{new}} = \sigma_\alpha^{-1} \sum_{i=1}^{n} C(\boldsymbol{b}_{\text{new}}, \boldsymbol{b}_i) \cdot \psi_i^{(\alpha)}$ and where $\psi_i^{(\alpha)}$ is the ith component of the eigenvector $\boldsymbol{\psi}^{(\alpha)}$. This approach, employing a truncated set S_δ, addresses numerical instabilities that occur when $\sigma_\alpha \to 0$.

By applying geometric harmonics in this manner, it is possible to predict the values of $\mathcal{F} = \mathbf{x}_t$ at unseen points $\boldsymbol{b}_{\text{new}} \in \mathbb{R}^d$, for $d = 2$ for the Brusselator system, $d = 3$ for the Lorenz system, is derived via Nyström extension [57] on time series windows of u_t.

Appendix C. Supplementary information

All code necessary to reproduce the numerical results presented in the paper are publicly available at https://github.com/Learning-of-Dynamic-Processes/coldstart.

References

[1] H. Jaeger, The 'echo state' approach to analysing and training recurrent neural networks with an erratum note, Tech. Rep, German National Research Center for Information Technology, 2010.

[2] W. Maass, T. Natschläger, H. Markram, Real-time computing without stable states: a new framework for neural computation based on perturbations, Neural Comput. 14 (2002) 2531–2560.

[3] H. Jaeger, H. Haas, Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication, Science 304 (5667) (2004) 78–80.

[4] W. Maass, Liquid state machines: Motivation, theory, and applications, in: S.S. Barry Cooper, A. Sorbi (Eds.), Computability in Context: Computation and Logic in the Real World, World Scientific, 2011, pp. 275–296.

[5] M.B. Matthews, On the Uniform Approximation of Nonlinear Discrete-Time Fading-Memory Systems Using Neural Network Models (Ph.D. thesis), ETH Zürich, 1992.

[6] M.B. Matthews, Approximating nonlinear fading-memory operators using neural network models, Circuits Systems Signal Process. 12 (2) (1993) 279–307.

[7] J. Pathak, Z. Lu, B.R. Hunt, M. Girvan, E. Ott, Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data, Chaos 27 (12) (2017).

[8] J. Pathak, B. Hunt, M. Girvan, Z. Lu, E. Ott, Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach, Phys. Rev. Lett. 120 (2) (2018) 24102.

[9] Z. Lu, B.R. Hunt, E. Ott, Attractor reconstruction by machine learning, Chaos 28 (6) (2018).

[10] A. Wikner, J. Pathak, B.R. Hunt, I. Szunyogh, M. Girvan, E. Ott, Using data assimilation to train a hybrid forecast system that combines machine-learning and knowledge-based components, Chaos 31 (5) (2021) 53114.

[11] T. Arcomano, I. Szunyogh, A. Wikner, J. Pathak, B.R. Hunt, E. Ott, A hybrid approach to atmospheric modeling that combines machine learning with a physics-based numerical model, J. Adv. Modelling Earth Syst. 14 (3) (2022) e2021MS002712.

[12] L. Grigoryeva, J.-P. Ortega, Universal discrete-time reservoir computers with stochastic inputs and linear readouts using non-homogeneous state-affine systems, J. Mach. Learn. Res. 19 (24) (2018) 1–40.

[13] L. Grigoryeva, J.-P. Ortega, Echo state networks are universal, Neural Netw. 108 (2018) 495–508.

[14] L. Gonon, J.-P. Ortega, Reservoir computing universality with stochastic inputs, IEEE Trans. Neural Netw. Learn. Syst. 31 (1) (2020) 100–112.

[15] L. Gonon, J.-P. Ortega, Fading memory echo state networks are universal, Neural Netw. 138 (2021) 10–13.

[16] L. Gonon, L. Grigoryeva, J.-P. Ortega, Approximation error estimates for random neural networks and reservoir systems, Ann. Appl. Probab. 33 (1) (2023) 28–69.

[17] F. Takens, Detecting strange attractors in turbulence, Springer Berlin Heidelberg, 1981, pp. 366–381.

[18] L. Kocarev, U. Parlitz, General approach for chaotic synchronization with applications to communication, Phys. Rev. Lett. 74 (25) (1995) 5028–5031.

[19] L.M. Pecora, T.L. Carroll, G.A. Johnson, D.J. Mar, J.F. Heagy, Fundamentals of synchronization in chaotic systems, concepts, and applications, Chaos 7 (4) (1997) 520–543.

[20] E. Ott, Chaos in Dynamical Systems, second ed., Cambridge University Press, 2002.

[21] S. Boccaletti, J. Kurths, G. Osipov, D.L. Valladares, C.S. Zhou, The synchronization of chaotic systems, Phys. Rep. 366 (2002) 1–101.

[22] D. Eroglu, J.S.W. Lamb, T. Pereira, Synchronisation of chaos and its applications, Contemp. Phys. 58 (3) (2017) 207–243.

[23] A.G. Hart, J.L. Hook, J.H.P. Dawes, Embedding and approximation theorems for echo state networks, Neural Netw. 128 (2020) 234–247.

[24] A.G. Hart, J.L. Hook, J.H.P. Dawes, Echo state networks trained by Tikhonov least squares are L2(μ) approximators of ergodic dynamical systems, Physica D 421 (2021) 132882.

[25] L. Grigoryeva, A.G. Hart, J.-P. Ortega, Chaos on compact manifolds: Differentiable synchronizations beyond the Takens theorem, Physical Rev. E 103 (2021) 062204.
[26] L. Grigoryeva, A.G. Hart, J.-P. Ortega, Learning strange attractors with reservoir systems, Nonlinearity 36 (2023) 4674–4708.
[27] G. Manjunath, A. de Clercq, Universal set of observables for the Koopman operator through causal embedding, 2021, arXiv preprint arXiv:2105.10759.
[28] T. Berry, S. Das, Learning theory for dynamical systems, SIAM J. Appl. Dyn. Syst. 22 (3) (2023) 2082–2122.
[29] F.P. Kemeth, T. Bertalan, N. Evangelou, T. Cui, S. Malani, I.G. Kevrekidis, Initializing LSTM internal states via manifold learning, 2021, arXiv:2104.13101.
[30] S. Boyd, L. Chua, Fading memory and the problem of approximating nonlinear operators with Volterra series, IEEE Trans. Circuits Syst. 32 (11) (1985) 1150–1161.
[31] G. Manjunath, Stability and memory-loss go hand-in-hand: Three results in dynamics & computation, Proc. R. Soc. London Ser. A Math. Phys. Eng. Sci. 476 (2242) (2020) 1–25.
[32] G. Manjunath, Embedding information onto a dynamical system, Nonlinearity 35 (3) (2022) 1131.
[33] L. Grigoryeva, J.-P. Ortega, Differentiable reservoir computing, J. Mach. Learn. Res. 20 (179) (2019) 1–62.
[34] G. Manjunath, H. Jaeger, Echo state property linked to an input: Exploring a fundamental characteristic of recurrent neural networks, Neural Comput. 25 (3) (2013) 671–696.
[35] K. Pyragas, Weak and strong synchronization of chaos, Phys. Rev. E 54 (5) (1996) 4508–4511.
[36] B.R. Hunt, E. Ott, J.A. Yorke, Differentiable generalized synchronization of chaos, Phys. Rev. E 55 (4) (1997) 4029–4034.
[37] J.P. Huke, Embedding nonlinear dynamical systems: A guide to Takens' theorem, Tech. Rep, Manchester Institute for Mathematical Sciences. The University of Manchester, 2006.
[38] R. Abraham, J.E. Marsden, T.S. Ratiu, Manifolds, Tensor Analysis, and Applications, vol. 75, Applied Mathematical Sciences. Springer-Verlag, 1988.
[39] R.R. Coifman, S. Lafon, Diffusion maps, Appl. Comput. Harmonic Anal. 21 (1) (2006) 5–30.
[40] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Systems 2 (4) (1989) 303–314.
[41] C. Cuchiero, L. Gonon, L. Grigoryeva, J.P. Ortega, J. Teichmann, Discrete-time signatures and randomness in reservoir computing, IEEE Trans. Neural Netw. Learn. Syst. 33 (11) (2022) 1–10.
[42] C. Cuchiero, L. Gonon, L. Grigoryeva, J.-P. Ortega, J. Teichmann, Expressive power of randomized signature, in: NeurIPS workshop, 2021.
[43] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, Contemp. Math. 26 (1984) 189–206.
[44] P. Verzelli, C. Alippi, L. Livi, Learn to synchronize, synchronize to learn, Chaos 31 (2021) 083119.
[45] H. Kantz, T. Schreiber, Nonlinear Time Series Analysis, second ed., Cambridge University Press, 2003.
[46] R. Martin, J. Koo, D. Eckhardt, Impact of embedding view on cross mapping convergence, 2019, arXiv preprint arXiv:1903.03069.
[47] D. Kondepudi, I. Prigogine, Dissipative structures, in: D. Kondepudi, I. Prigogine (Eds.), Modern Thermodynamics, John Wiley & Sons, Ltd, 2014, pp. 421–450.
[48] E.N. Lorenz, Deterministic nonperiodic flow, J. Atmos. Sci. 20 (1963) 130–141.
[49] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019.
[50] D. Canaday, A. Pomerance, M. Girvan, A meta-learning approach to reservoir computing: Time series prediction with limited data, 2021, arXiv preprint arXiv:2110.03722.
[51] D. Lehmberg, F. Dietrich, G. Köster, H.-J. Bungartz, Datafold: Data-driven models for point clouds and time series on manifolds, J, Open Source Softw, 5 (51) (2020) 2283.
[52] E.M. Coven, I. Kan, J.A. Yorke, Pseudo-orbit shadowing in the family of tent maps, Trans. Amer. Math. Soc. 308 (1) (1988) 227–241.
[53] C. Grebogi, L. Poon, T. Sauer, J.A. Yorke, D. Auerbach, Shadowability of chaotic dynamical systems, in: Handbook of Dynamical Systems, Elsevier, 2002, pp. 313–344.
[54] T. Sauer, C. Grebogi, J.A. Yorke, How long do numerical chaotic solutions remain valid? Phys. Rev. Lett. 79 (1) (1997) 59–62.
[55] J. Kennedy, J.A. Yorke, Shadowing in higher dimensions, in: Progress in Nonlinear Differential Equations and their Applications, Birkhauser Basel, 2007, pp. 241–246.
[56] R.R. Coifman, Y. Shkolnisky, F.J. Sigworth, A. Singer, Graph Laplacian tomography from unknown random projections, IEEE Trans. Image Process. 17 (10) (2008) 1891–1899.
[57] E.J. Nyström, Über die praktische auflösung von integralgleichungen mit anwendungen auf randwertaufgaben, Acta Math. 54 (1) (1930) 185–204.