LEARNING LATENT DYNAMICS FOR PARTIALLY-OBSERVED CHAOTIC SYSTEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper addresses the data-driven identification of latent representations of partially-observed dynamical systems, *i.e.* dynamical systems whose some components are never observed, with an emphasis on forecasting applications and long-term asymptotic patterns. Whereas state-of-the-art data-driven approaches rely on delay embeddings and linear decompositions of the underlying operators, we introduce a framework based on the data-driven identification of an augmented state-space model using a neural-network-based representation. For a given training dataset, it amounts to jointly reconstructing the latent states and learning an ODE (Ordinary Differential Equation) representation in this space. Through numerical experiments, we demonstrate the relevance of the proposed framework w.r.t. state-of-the-art approaches in terms of short-term forecasting errors and long-term behaviour. We further discuss how the proposed framework relates to Koopman operator theory and Takens' embedding theorem.

1 INTRODUCTION

Learning the underlying dynamical representation of observed variables $\mathbf{x}_t \in \mathbb{R}^n$ (where $t \in \{t_0, ..., T\}$ is the temporal sampling time and n the dimension of the observations) is a key challenge in various scientific fields, including control theory, geoscience, fluid dynamics, economics; for applications ranging from system identification to forecasting and assimilation issues Lai & Wei (1982); Abarbanel & Lall (1996); Jeong & Hussain (1995); Koopmans (1949).

For fully-observed systems, *i.e.* when the observed variables $\mathbf{x}_t$ relate to some underlying deterministic states $\mathbf{z}_t$, recent advances Brunton et al. (2016b); Fablet et al. (2018); Chen et al. (2018); Nguyen et al. (2019) have shown that one can identify the governing equations of the dynamics of $\mathbf{z}_t$ from a representative dataset of observations $\{\mathbf{x}_{t_i}\}_i$. Unfortunately, When the observed variables $\mathbf{x}_t$ only relate to some but not all the components of underlying states $\mathbf{z}_t$, these approaches can not apply since no ODE or, more generally, no one-to-one mapping defined in the observation space can represent the time evolution of the observations. In this context, Takens's theorem states the conditions under which a delay embedding, formed by lagged versions of the observed variables, guarantees the existence of governing equations in the embedded space Takens (1981).

Takens's theorem has motivated a rich literature of machine learning schemes to identify dynamical representations of partially-observed systems using a delay embedding. This comprises both non-parametric schemes based on nearest-neighbors or analogs Abarbanel (1996a) as well as parametric schemes which include polynomial representations Paduart et al. (2010), neural network models Frank et al. (2010), Support Vector Regression (SVR) models Kazem et al. (2013). For all these approaches, the identification of the appropriate delay embedding is a critical issue Abarbanel (1996b;c).

In this work, we show that we do not need to rely explicitly on a delay embedding. We address the identification of an augmented space of higher dimension than the manifold spanned by the observed variables, where the dynamics of the observations can be fully described by an ODE. Using neural-network representations for the parametrization of the dynamical model, it amounts to jointly learning the governing ODE and reconstructing the augmented latent states for a given observation dataset. We report experiments on linear and chaotic dynamics, which illustrate the relevance of the proposed framework compared to state-of-the-art approaches. We then further discuss the key

features of this framework with respect to state-of-the-art dynamical systems identification tools such as Koopman operator theory Koopman (1931).

2 BACKGROUND AND RELATED WORK

This section introduces the learning of dynamical representations for partially-observed systems and links this problem to recent advances in machine learning.

Let us consider an **unobserved** state variable $\mathbf{z}$ governed by an autonomous system of s differential equations $\dot{\mathbf{z}}_t = f(\mathbf{z}_t)$. Let us also assume that this system generates a flow $\Phi_{t_i}(\mathbf{z}_{t_0}) = \int_{t_0}^{t_i} f(\mathbf{z}_u)du \in \mathbb{R}^s$ with trajectories that are asymptotic to a limit-cycle L of dimension d contained in $\mathbb{R}^s$. We further assume that we are provided with a measurement function $\mathcal{H}$ that maps our state variable $\mathbf{z}$ to our observations $\mathbf{x}_t = \mathcal{H}(\mathbf{z}_t) \in \mathbb{R}^n$.

When considering the data-driven identification of a dynamical mapping that governs some observation data, we first need to evaluate whether the dynamics in the observation space can be described using a smooth[1] ODE. Another way to tackle this question is to find the conditions under which the deterministic properties of the unobserved limit-cycle L are preserved in the observation space in $\mathbb{R}^n$ such that one can reliably perform forecasts in the observation space. The general condition under which a mapping $\mathcal{H}$ preserves the topological properties of the initial limit-cycle involves a differential structure. Assuming that L is a smooth compact differential manifold, the topological properties of L are preserved through a mapping $\mathcal{H}$ in $\mathbb{R}^n$ if $\mathcal{H}$ is one-to-one and is an immersion of L in $\mathbb{R}^n$. Under these conditions our observation mapping is called an **embedding** Sauer et al. (1991).

The simplest example of an embedding involves an identity observation operator $\mathcal{H}$. With such embedding we have direct access to the state variable $\mathbf{z}$ which is governed by a deterministic ODE. This particular case has been widely studied in the literature. Parametric representations have been for decades the most popular models thanks to their simplicity and interpretability Paduart et al. (2010), Brunton et al. (2016b). Recently, these approaches have been enriched by neural network and deep learning schemes Wiewel et al. (2018), Raissi et al. (2018). In particular, the link between residual networks Chen et al. (2018); Ouala et al. (2019) and numerical integration schemes have opened new research avenues for learning extremely accurate dynamical models even from irregularly-sampled training data. These schemes show greater interpretability and forecasting performance for the data-driven representation of systems governed by an ODE, compared with other state-of-the-art neural networks schemes, including Recurrent Neural Networks (RNN) such as LSTM (Long-Short-Term Memory). Recent advances in dimensionality reduction algorithms such as Auto-Encoders Fabius & van Amersfoort (2014), also advocate for the data-driven identification of simple Reduced order models (ROMs) when the measurement function $\mathcal{H}$ is an embedding that maps the hidden states z_t in a higher dimensional space Champion et al. (2019).

However, for a wide range of real-world systems, we are never provided with an observation operator that forms an embedding of the latent state variables. In such situations, we do not have any guarantee on the existence of a smooth ODE that governs the temporal evolution of our observations (or, more generally, of a projection of the observations that can be computed, for example, using a dimensionality reduction algorithm). From this point of view, the question of finding an appropriate dynamical representation of some observed data may not be this straightforward. The fact that our data may come from some unobserved governing equation may restrict the use of the above-mentioned state-of-the-art algorithms. The main difficulty lies in the ability to map observation series to a latent space that provides at least a *one-to-one* mapping between two successive states. From a geometrical point of view, the time delay theorem Takens (1981) provides a way to build a latent space that preserves the topological properties of the true (unobserved) dynamics limit-cycle. A generalization of this theorem Sauer et al. (1991) shows that one can reconstruct topologically similar limit-cycles using any appropriate smooth composition map of the observations. The derivation of a dynamical system from such representations however encounters large disparities since no explicit relationships between the defined phase space and an ODE formulation have been clearly made. Classical state-of-the-art techniques such as polynomial representations Brunton et al. (2016b) and K-Nearest Neighbors (KNN) Lguensat et al. (2017) algorithms were

[1]The word smooth here stands for continuously differentiable or $\mathcal{C}^1$.

proposed but they often fail to achieve both accurate short-term forecasting performance and long-term topologically similar reconstructed limit-cycles (see experiments for an illustration). We may also point out that the limitation of ODE-based representation in deep learning architecture has also been pointed out recently in Dupont et al. (2019); Zhang et al. (2019) for classification issues. As ODE-derived trajectories do not intersect, it may limit the ability of neural ODE representations to reach relevant classification performance in a given feature space. To address this issue, Dupont et al. (2019) and Zhang et al. (2019) propose to consider an augmented state, simply by augmenting the observed state by a number of zeros to create a high-dimensional space in which an ODE representation can be identified. Such a strategy cannot apply to time series modeling as successive augmented states cannot be forced to zero for some dimension.

In this work, we address the identification of a latent embedding, associated with an ODE representation, for partially-observed systems. The core idea of this work is to replace the augmented observable vector formulated in Takens (1981) by augmented states that are the outputs of an optimization problem with respect to an ODE formulation. From a dynamical system point of view, we look for an embedding of our observations in an augmented state space that will be governed by an ODE.

3 LEARNING LATENT REPRESENTATIONS OF PARTIALLY-OBSERVED DYNAMICS

Augmented latent dynamics: Let us assume a continuous s-dimensional dynamical system $\mathbf{z}_t$ governed by an autonomous ODE $\dot{\mathbf{z}}_t = f(\mathbf{z}_t)$ with Φ_t the corresponding flow $\Phi_t(\mathbf{z}_{t_0}) = \int_{t_0}^{t} f(\mathbf{z}_u)du$.

In many applications, one cannot fully access the state z and the observations only relate to some components of this state. Formally, we can define an observation function $\mathcal{H} : \mathbb{R}^s \longrightarrow \mathbb{R}^n$ such that the observations $\mathbf{x}_t$ follow $\mathbf{x}_t = \mathcal{H}(\mathbf{z}_t)$. We can also define a bijective map $\mathcal{M}$ that maps our observations $\mathbf{x}_t$ in some low dimensional manifold $\mathbf{r}_t = \mathcal{M}(\mathbf{x}_t) \in \mathbb{R}^k$. The definition of this operator is crucial in the data driven identification of ROMs Champion et al. (2019) of real data since in this case, the provided data is usually mapped through $\mathcal{H}$ in a higher dimensional space. Finally, $\mathcal{M}$ is supposed to be bijective so the dynamics in $\mathbb{R}^n$ are completely determined by the dynamics in $\mathbb{R}^k$ so from now on, and for the sake of simplicity, we will refer to both $\mathbf{r}_t \in \mathbb{R}^k$ and $\mathbf{x}_t \in \mathbb{R}^n$ as observations since they are equivalent up to a bijective map $\mathcal{M}$.

We aim to derive an ODE representation of $\mathbf{r}_t \in \mathbb{R}^k$. However, the key question arising here is the extent to which the dynamics expressed in the observations space, reflect the true underlying dynamics in $\mathbb{R}^s$, and consequently, the conditions on $\mathcal{H}$ under which the predictable deterministic dynamical behavior of the hidden states is still predictable in the observations space. To illustrate this issue, we may consider a linear dynamical system in the complex domain governed by the following linear ODE:

$$\begin{cases} \dot{\mathbf{z}}_t = \alpha \mathbf{z}_t \\ \mathbf{z}_{t_0} = \mathbf{z}_0 \end{cases} \tag{1}$$

with $\mathbf{z} \in \mathbb{C}$ a state variable and $\alpha \in \mathbb{C}$ a complex imaginary number. The solution of this problem is

$$\mathbf{z}_t = Ke^{\alpha t} \tag{2}$$

with K a constant depending on $\mathbf{z}_0$. Let us assume now that we are only provided with the real part as direct measurements of the unobserved state *i.e.* $\mathcal{H}(.) = Real(.) : \mathbf{x}_t = Real(\mathbf{z}_t)$ so in this case $\mathcal{M} = I_1$ and $k = n$.

Proposition 1 : *The flow of an ODE cannot represent the time evolution of* $\mathbf{x}_t$.

The proof of the proposition is given in the appendix and the intuition behind it is as follows. Assuming that we are only provided the real part as direct measurements $\mathbf{x}_t \in \mathbb{R}$ of the true states $\mathbf{z}_t$, no smooth autonomous ODE model in the scalar observation space can describe the trajectories of the observations as the mapping between two observations is not one-to-one. For example, assuming that $\mathbf{z}_{t_0}$ and $\mathbf{z}_{t_1}$ correspond to two states that have the same real part but distinct imaginary parts, the associated observed states are equal $\mathbf{x}_{t_0} = \mathbf{x}_{t_1}$. However, the time evolution of the states $\mathbf{z}_{t_0}$ and $\mathbf{z}_{t_1}$ differ if they have different imaginary parts, such that the observed states $\mathbf{x}_{t_0+\delta}$ and $\mathbf{x}_{t_1+\delta}$ after any time increment δ are no longer equal. As a consequence, a given observation may have more than

one future state and this behavior cannot be represented by a smooth ODE in the observation space. And the application of an ODE mapping such as Chen et al. (2018) and Fablet et al. (2018) for such observations will lead to poor forecasting performance. From a naive Neural networks point of view, fitting such a model will most likely force the forecasting into an equilibrium point since we are iteratively matching the same inputs with different output predictions.

For a given observation operator $\mathcal{H}$ of a deterministic underlying dynamical system that governs $\mathbf{z}_t$, Takens's theorem guarantees the existence of an augmented space, defined as a delay embedding of the observations, in which a one-to-one mapping exists between successive time steps of the observation series Takens (1981). Rather than exploring such delay embedding, we aim to identify an augmented latent space, where the latent dynamics are governed by a smooth ODE and can be mapped to the observations. Let us define $\mathbf{u}_t \in \mathbb{R}^{d_E}$ a d_E-dimensional augmented latent state as follows:

$$\mathbf{u}_t{}^T = [\mathcal{M}(\mathbf{x}_t)^T, \mathbf{y}_t^T] \tag{3}$$

with $\mathbf{y}_t \in \mathbb{R}^l$ the unobserved component of latent state $\mathbf{u}_t$. The augmented latent space evolves in time according to the following state space model:

$$\begin{cases} \dot{\mathbf{u}}_t = f_\theta(\mathbf{u}_t) \\ \mathbf{x}_t = \mathcal{M}^{-1}(G(\mathbf{u}_t)) \end{cases} \tag{4}$$

where the dynamical operator f_θ belongs to a family of smooth operators (in order to guarantee uniqueness Coddington & Levinson (1955)) parametrized by θ. We typically consider a neural-network representation with Lipschitz nonlinearities and finite weights. G is a projection matrix that satisfies $\mathcal{M}(\mathbf{x}_t) = G(\mathbf{u}_t)$. As detailed in the next sections, we address the identification of the operator f_θ and of the associated latent space $\mathbf{u}$ from a dataset of observed state series $\{\mathbf{x}_0, \ldots, \mathbf{x}_T\}$ as well as the exploitation of the identified latent dynamics for the forecasting of the time evolution of the observed states, for instance unobserved future states $\{\mathbf{x}_{T+1}, \ldots, \mathbf{x}_{T+N}\}$.

Learning scheme: Given an observation time series $\{\mathbf{x}_0, \ldots, \mathbf{x}_T\}$ and the bijective map $\mathcal{M}$, we aim to identify the state-space model defined by (4), which amounts to learning the parameters θ of the dynamical operator f_θ. However, as the component $\mathbf{y}_t$ of the latent state $\mathbf{u}_t$ is never observed, this identification requires the joint optimization of the model parameters θ as well as of the hidden component $\mathbf{y}_t$. Formally, this problem is stated as the following minimization of the forecasting error on observed variables:

$$\begin{aligned} \hat{\theta} = \arg\min_\theta \min_{\{\mathbf{y}_t\}_t} \sum_{t=1}^{T} \|\mathbf{x}_t - \mathcal{M}^{-1}(G\left(\Phi_{\theta,t}\left(\mathbf{u}_{t-1}\right)\right))\|^2 \\ \text{Subject to } \begin{cases} \mathbf{u}_t &= \Phi_{\theta,t}(\mathbf{u}_{t-1}) \\ \mathcal{M}(G(\mathbf{u}_t)) &= \mathbf{x}_t \end{cases} \end{aligned} \tag{5}$$

with $\Phi_{\theta,t}$ the one-step-ahead diffeomorphic mapping associated with operator f_θ such that:

$$\Phi_{\theta,t}(\mathbf{u}_{t-1}) = \mathbf{u}_{t-1} + \int_{t-1}^{t} f_\theta(\mathbf{u}_w) dw$$

In (5), the loss to be minimized involves the one-step-ahead forecasting error for the observed variable $\mathbf{x}_t$. The constraints state that the augmented state $\mathbf{u}_t$ is composed of observed component and $G(\mathbf{u}_t)$ should be a solution of the ODE (4). Here, we numerically minimize the equivalent formulation:

$$\min_\theta \min_{\{\mathbf{y}_t\}_t} \sum_{t=1}^{T} \|\mathbf{x}_t - \mathcal{M}^{-1}(G\left(\Phi_{\theta,t}\left(\mathbf{u}_{t-1}\right)\right))\|^2 + \lambda \|\mathbf{u}_t - \Phi_{\theta,t}(\mathbf{u}_{t-1})\|^2 \tag{6}$$

where $\mathbf{u}_t^T = [\mathcal{M}(\mathbf{x}_t^T), \mathbf{y}_t^T]$ and λ a weighting parameter. The term $\|\mathbf{u}_t - \Phi_{\theta,t}(\mathbf{u}_{t-1})\|^2$ may be regarded as a regularization term such that the inference of the unobserved component $\mathbf{y}_t$ of the augmented state $\mathbf{u}_{t-1}$ is not solved independently for each time step.

Using a neural-network parametrization for the ODE operator f_θ, the corresponding forecasting operator $\Phi_{\theta,t}$ is also stated as a neural network based on a numerical integration scheme formulation

(typically a 4^{th}-order Runge-Kutta scheme). This architecture, very much similar to a ResNet He et al. (2015), allows very accurate identification of ODE models Fablet et al. (2018); Ouala et al. (2019). Hence, for a given observed state series $\{\mathbf{x}_0, \ldots, \mathbf{x}_T\}$, we minimize (6) jointly w.r.t. θ and unobserved variables $\{\mathbf{y}_0, \ldots, \mathbf{y}_T\}$. In the experiments reported in Section 4, we consider bilinear architectures Fablet et al. (2018). However, the proposed framework applies to any neural-network architecture.

Application to forecasting: We also apply the proposed framework to the forecasting of the observed states $\mathbf{x}_t$. Given a trained latent dynamical model (4), forecasting future states for $\mathbf{x}_t$ relies on the forecasting of the entire augmented latent state $\mathbf{u}_t$. The latter amounts to determining an initial condition of the unobserved component $\mathbf{y}_t$ and performing a numerical integration of the trained ODE (4).

Let us denote by $\mathbf{x}_t^n, t \in \{t_0, ..., T\}$ a new series of observed states. We aim to forecast future states $\mathbf{x}_t^n, t \in \{T+1, ..., T+\delta T\}$. Following (6), we infer the unobserved component $\widehat{\mathbf{y}}_T$ of latent state X_T^n at time T from the following minimization:

$$\hat{\mathbf{y}}_T^n = \arg\min_{\mathbf{y}_T^n} \min_{\{\mathbf{y}_t^n\}_{t<T}} \sum_{t=T+1}^{T+\delta T} \|\mathbf{x}_t^n - \mathcal{M}^{-1}(G\left(\Phi_{\theta,t}\left(\mathbf{u}_{t-1}^n\right)\right)\|^2 + \lambda \|\mathbf{u}_t^n - \Phi_{\theta,t}(\mathbf{u}_{t-1}^n)\|^2 \quad (7)$$

Here, we only minimize w.r.t. latent variables $\{\mathbf{y}_t^n\}$ given the trained forecasting operator $\Phi_{\theta,t}$. This minimization relates to a variational assimilation issue with partially-observed states and known dynamical and observation operators Lynch & Huang (2010). Similarly to the learning step, we benefit from the neural-network parameterization of operator $\Phi_{\theta,t}$ and from the associated automatic differentiation tool to compute the solution of the above minimization using a gradient descent.

We may consider different initialization strategies for this minimization problem. Besides a simple random initialization, we may benefit from the information gained on the manifold spanned by the unobserved components during the training stage. The basic idea comes to assume that the training dataset is likely to comprise state trajectories which are similar to the new one. As the training step embeds the inference of the whole latent state sequence, we may pick as initialization for minimization (7) the inferred augmented latent state in the training dataset which leads to the observed state trajectory that is the most similar (in the sense of the L2 norm) to the new observed sequence $\mathbf{x}_t^n$,. The interest of this initialization scheme is two-fold: (i) speeding-up the convergence of minimization (7) as we expect to be closer to the minimum; (ii) considering an initial condition which is in the basin of attraction of the reconstructed limit-cycle. The latter may be critical as we cannot guarantee that the learnt model does not involve other limit-cycles than the ones truly revealed by the training dataset, which may lead to a convergence to a local and poorly relevant minimum.

4 NUMERICAL EXPERIMENTS

In this section, we report numerical experiments to illustrate the key features of proposed framework. We consider three case-studies: a linear ODE case-study; a chaotic system, namely Lorenz-63 dynamics, and real upper ocean data.

Application to a linear ODE: In order to illustrate the key principles of the proposed framework, we consider the following linear ODE in the complex domain:

$$\begin{cases} \dot{\mathbf{z}}_t = \alpha \mathbf{z}_t \\ \mathbf{z}_{t_0} = \mathbf{z}_0 \end{cases} \quad (8)$$

with $\alpha = -0.1 - 0.5j$, $j^2 = -1$ and $\mathbf{z}_0 = 0.5$. As $\alpha \in \mathbb{C}$ with $Real(\alpha) < 0$ and $\mathbf{z}_0 \neq 0$, the solution of this ODE is an ellipse in the complex plane (Fig. 1).

As observation, we consider the real part of the underlying state, *i.e.* the observation function $\mathcal{H} : \mathbb{C} \longrightarrow \mathbb{R}$ is given by $\mathbf{x}_t = Real(\mathbf{z}_t)$. This is a typical example, where the mapping between two successive observations is not a one-to-one mapping since all the states that have the same real part lead to the same observation. As explained in section 3, one cannot identify an autonomous ODE model that will reproduce the dynamical behavior of the observations in the observations space.

We apply the proposed framework to this toy example. We consider a 2-dimensional augmented state $\mathbf{u}_t = [\mathbf{x}_t, \mathbf{y}_t^1]$ with $\mathcal{M} = I_1$. As neural-network parametrization for operator f_θ, we consider a neural network with a single linear fully-connected layer. We use an observation series of 10000 time steps as training data. As illustrated in Fig.1, given the same initial condition over the observable state, the inferred latent state dynamics, though different from the true ones, depicts a similar spiral pattern. This result is in agreement with the geometrical reconstruction techniques Takens (1981) of the latent dynamics up to a diffeomorphic mapping. Overall, our model learns a dynamical behavior similar to the true model represented by an elliptic transient and an equilibrium point limit-set. Furthermore, the projection of the augmented latent space and the true solution of Eq. (**??**) in the real axis illustrate the relevance of the proposed framework in forecasting the observations dynamics (mean square error $< 1E-6$).

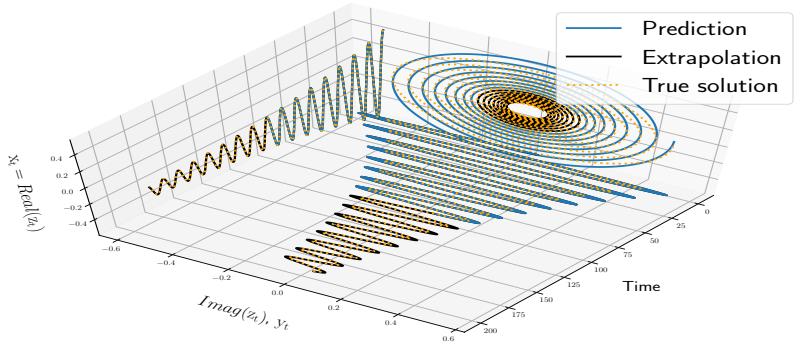


Figure 1: **Illustration for a 2-dimensional linear ODE**: (left) Forecasted augmented latent space with respect to the true states. Given the same initial condition we illustrate both the forecasted and the true trajectories. The projection of the solutions in the real plane illustrates the forecasting of the observations.

Lorenz-63 dynamics: Lorenz-63 dynamical system is a 3-dimensional model that involves, under some specific parametrizations Lorenz (1963), chaotic dynamics with a strange attractor. We simulate chaotic Lorenz-63 state sequences with the same model parameters as proposed in Lorenz (1963) using the LOSDA ODE solver Hindmarsh (1983) with an integration step of 0.01. We assume that only the first Lorenz-63 variable is observed $\mathrm{x}_t = \mathrm{z}_{t,1}$ and we pose $\mathcal{M} = I_1$. We apply the proposed framework to this experimental setting using a training sequence of 4000 time-steps.

For benchmarking purposes, we perform a quantitative comparison with state-of-the-art approaches using delay embedding representations Takens (1981). The parameters of the delay embedding representation, namely the lag τ and the dimension d_E of the augmented space were computed using state-of-the-art techniques. Specifically, the lag parameter was computed using both the mutual information and correlation techniques Abarbanel (1996b), respectively denoted as τ_{MI} and τ_{Corr}. Regarding the dimension of the embedding representation, we used the Takens embedding condition $d_E = 2d + 1$ with d the dimension of the hidden limit-cycle. The delay embedding dimension was also computed using the False Nearest Neighbors (FNN) method Abarbanel (1996c). We also tested arbitrary parameters for the delay embedding dimension. Given the delay embedding representation, we tested two state-of-the-art data-driven representations of the dynamics. The Analog Forecasting technique (AF) which is based on the nearest neighbours algorithm Lguensat et al. (2017) and the Sparse Regression (SR) method on a second order polynomial representation of the delay embedding states. Regarding deep learning models, we compare our method to a stacked Bidirectional LSTM (RNN) and to the Latent-ODE model as proposed in Chen et al. (2018). Finally, the proposed framework, referred to as Neural embedding for Dynamical Systems (NbedDyn) was tested for

different dimensions of the augmented state space, namely from 3 to 6 (please refer to the apendix for details on the neural networks based architectures)[2].

Model			$t_0 + h$	$t_0 + 4h$	λ_1
AF	τ_{MI} =16	$d_E(FNN) = 3$	$5.6E-3$	$1.3E-2$	0.85
	τ_{MI} =16	$d_E(Takens) = 6$	$9.9E-3$	$2.4E-2$	NaN
	τ_{Corr} =27	$d_E(FNN) = 3$	$8.9E-3$	$2.3E-2$	12.35
	τ_{Corr} =27	$d_E(Takens) = 6$	$8.5E-3$	$1.9E-2$	NaN
	τ = 6	$d_E = 3$	$8.0E-4$	$9.0E-4$	0.87
	τ = 10	$d_E = 3$	$2.1E-3$	$4.9E-3$	0.60
SR	τ_{MI} =16	$d_E(FNN) = 3$	$7.8E-2$	$2.5E-1$	0.12
	τ_{MI} =16	$d_E(Takens) = 6$	$4.5E-2$	$1.7E-1$	NaN
	τ_{Corr} =27	d_E(FNN) = 3	$1.4E-1$	$4.6E-1$	NaN
	τ_{Corr} =27	d_E(Takens) = 6	$2.1E-1$	$8.4E-1$	NaN
	τ = 6	$d_E = 3$	$7.6E-3$	$7.4E-3$	NaN
	τ = 10	$d_E = 3$	$2.5E-2$	$5.7E-2$	0.2535
Latent-ODE			$6.9E-2 \pm 2.9E-2$	$1.5E-1 \pm 3E-2$	NaN
RNN			$6.9E-2 \pm 4.6E-2$	$1.5E-1 \pm 1.1E-1$	-6.79 ± 0.0
NbedDyn		$d_E = 3$	$3.2E-4 \pm 1.3E-4$	$1.7E-3 \pm 7.5E-4$	0.81 ± 0.09
		$d_E = 4$	$1.3E-4 \pm 5.2E-5$	$7.3E-4 \pm 2.2E-4$	0.82 ± 0.06
		$d_E = 5$	$3.8E-4 \pm 7.4E-4$	$2.0E-3 \pm 3.4E-4$	0.80 ± 0.02
		$d_E = 6$	$3.7E-4 \pm 2.8E-4$	$2.0E-3 \pm 1.7E-3$	0.92 ± 0.02
		$d_E = 6$ (Best)	$9.1E-5$	$4.7E-4$	0.92

Table 1: ***Forecasting performance on the test set of data-driven models for Lorenz-63 dynamics where only the first variable is observed***: first two columns : mean RMSE for different forecasting time steps, third column : largest Lyapunov exponent of a predicted series of length of 10000 time-steps (The true largest Lyapunov exponent of the Lorenz 63 model is 0.91 Sprott (2003)).

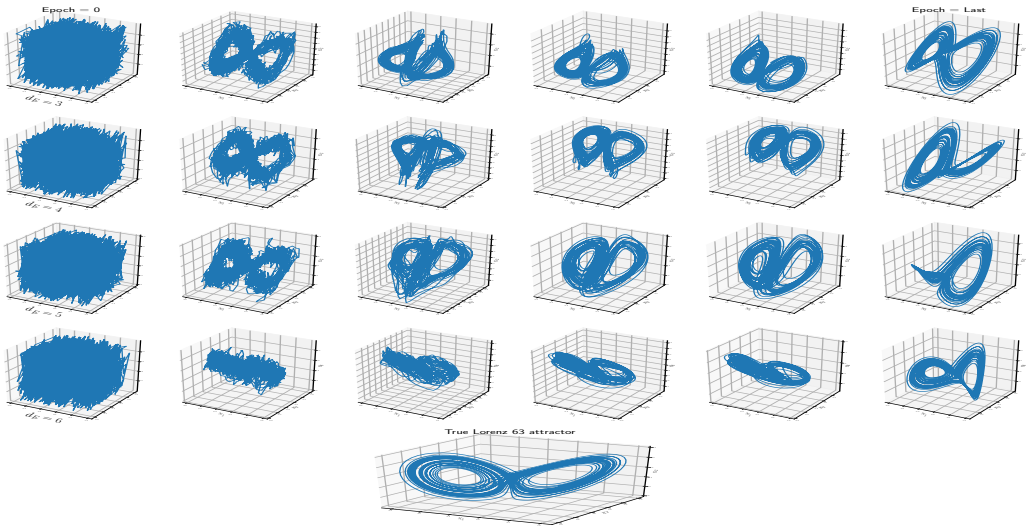


Figure 2: ***Evolution of the learnt latent space***: starting from a random initialization of the augmented states $\mathbf{y}_i$, the latent space is optimized according to thr minimization of Eq. (6) to form a limit-cycle similar to the true Lorenz 63 attractor. We depict 3-dimensional projections of the learnt latent space for the proposed model with different embedding dimensions from $d_E = 3$ to $d_E = 6$.

Fig. 2 illustrates the learning process for the latent space from the initialization to the last training epoch. We also report the analysis of short-term forecasting performance as well as the long-term asymptotic behavior characterized by the largest Lyapunov exponent of the benchmarked models in Tab 1. The proposed model leads to significant improvements in terms of short term forecasting performance with respect to the other approaches. Surprisingly, The Latent-ODE and RNN models

[2]The results of the neural networks based models were averaged over 5 runs except for NbedDyn $d_E = 4$ and $d_E = 5$ where we only launched 3 runs. We will include other runs for the final version of the manuscript

lead to the poorest performance both in terms of forecasting error and asymptotic behavior. This is mainly due, in the Latent-ODE case, to the fact that the latent space is seen as a non linear projection of the observed variables through the optimization of the ELBO loss Krishnan et al. (2016). By contrast, our latent embedding formulation optimizes the latent states to forecast the observed variables which explicitly constrain the latent space to be an embedding of the true underlying dynamics. The RNN model in the other hand converges to a periodic solution (please refer to the appendix for forecasting figures) with still a poor short term forcasting performances. Overall, this results on deep learning models suggest that one should use such tools with care to guarantee satisfying the specifications of the underlying system.The SR model seems to lead to better short term forecast (using ad hoc parameters (τ = 6, $d_E = 3$), however, it does not capture well the chaotic patterns, which are associated to a positive largest Lyapunov exponent. This may suggest the combination of the SR model and a delay embedding may require additional investigation as a good geometrical reconstruction of the phase space as stated in Takens' theorem does not guarantee the existence of a parametric ODE model based on the corresponding delay embedding variables. Better performance is reported using an analog forecasting approach. The performance however greatly varies depending on the considered definition of the delay embedding. Using ad hoc parameters (τ = 6, $d_E = 3$), one may retrieve the expected long-term chaotic behavior ($\lambda_1 = 0.87$) with a relatively low short-term forecasting error (8.0e-4 for a one-step-ahead forecast). When considering the proposed model, we report for all parametrizations, augmented space dimensions from 3 to 6, performance at least in the same range as the best analog forecasting setting. Besides, when increasing the dimension of the augmented space, we significantly decrease short-term forecasting errors (¡1.e-4 for a one-step-ahead forecast when considering the best fit for $d_E = 6$, i.e. one order of magnitude compared to the best benchmark model) while keeping an appropriate chaotic long-term pattern ($\lambda_1 = 0.92$).

Modeling Sea Level Anomaly (SLA): The data driven identification of dynamical representations of real data is an extremely difficult task especially when the underlying processes involve non stable behaviors such as chaotic attractors. This is mainly due to the fact that we do not have any exact knowledge of the closed form of the equations governing the temporal evolution of our variables. Furthermore, the measured quantity may depend on other unobserved variables which makes the exploitation of data-driven techniques highly challenging.

In this context, we report an application to SLA (Sea Level Anomaly) dynamics, which relate to upper ocean dynamics and are monitored by satellite altimeters Calmant et al. (2008). Sea surface dynamics are chaotic and clearly involve latent processes, typically subsurface and atmospheric processes. The dataset used in our experiments is a SLA time series obtained using the WMOP product Juza et al. (2016). The spatial resolution of our data is a 0.05° and the temporal resolution $h = 1$ day. We use the data from January 2009 to December 2014 as training data and we tested our approach on the last month of the year 2014. The considered region is located on south Mallorca ($2.5^\circ E$ to $4.25^\circ E$, $37.25^\circ N$ to $39.5^\circ N$). Finally, and in order to identify a ROM, we mapped our data through a bijective projection defined offline using a PCA as follow : $\mathbf{r}_t = \mathcal{M}(\mathbf{x}_t) \in \mathbb{R}^k$ with $k = 15$ which amounts to capture 92% of the total variance (here $\mathcal{M}$ is simply a linear PCA projection).

We report forecasting performance for our model and include a comparison with analog methods (AF), Sparse regression (SR), LSTM (RNN) and a neural ODE setting (Latent-ODE) in Tab. 2 (The results of the neural networks based models were averaged over 5 runs). Regarding the proposed NbedDyn model we consider an augmented latent space with $d_E = 60$. Our model clearly outperforms the three benchmarked schemes with a very significant gain for the forecasting performance at one day (relative gain greater than 90 %) and two days (relative gain greater than 90 %). For

Model		$t_0 + h$	$t_0 + 2h$	$t_0 + 4h$
AF	RMSE	0.036	0.049	0.067
	Corr	98.93%	96.97%	93.99%
SR	RMSE	0.014	0.021	xx 0.037
	Corr	99.42%	97.63%	90.91%
Latent-ODE	RMSE	0.030 ± 0.05	0.031 ± 0.031	0.040 ± 0.040
	Corr	$98.20\% \pm 0.39\%$	$97.39\% \pm 0.36\%$	$93.42\% \pm 0.55\%$
RNN	RMSE	0.026 ± 0.003	0.038 ± 0.007	0.053 ± 0.016
	Corr	$98.36\% \pm 0.40\%$	$95.29\% \pm 1.73\%$	$74.97\% \pm 5.75\%$
NbedDynZERO	RMSE	0.016 ± 0.0	0.023 ± 0.0	0.038 ± 0.0
	Corr	$99.44\% \pm 0.0\%$	$97.71\% \pm 0.0\%$	$91.18\% \pm 0.0\%$
NbedDyn	RMSE	0.002 ± 0.0003	0.006 ± 0.001	0.020 ± 0.004
	Corr	$99.99\% \pm 0.0017\%$ %	$99.91\% \pm 0.01\%$	$99.01\% \pm 0.04\%$

Table 2: ***SLA Forecasting performance on the test set of data-driven models***: RMSE and correlation coefficients for different forecasting time steps.

a 4-day-ahead forecasting, our model still outperforms the other ones though the gain is lower (relative gain of 40%). Finally, and in order to illustrate the influence of adding extra dimensions to define an augmented latent space on real data, we also tested the proposed NbedDyn model directly on the PCA space ($d_E = k = 15$) this model is referred to as NbedDynZERO and the influence of the latent components is clear from the results in Tab. 2 witch allows a relative gain up to 90 % with respect to the same model directly on the PCA space. We let the reader refer to the Supplementary Material for a more detailed analysis of these experiments, including visual comparisons of the forecasts.

5 DISCUSSION

In this work, we address the data-driven identification of latent dynamics for systems which are only partially observed, *i.e.* when some components of the system of interest are never observed. The reported forecasting performance for Lorenz-63 dynamics is in line with the forecasting performance of state-of-the-art learning-based approaches for a noise-free and fully-observed setting. This is of key interest for real-world applications, where observing systems most often monitor only some components of the underlying systems. As a typical example, the SLA forecasting experiment clearly motivates the proposed framework in the context of ocean dynamics for which neither in situ nor satellite observing systems can provide direct observations for all state variables (e.g., subsurface velocities, fine-scale sea surface currents).

We may also further discuss how the proposed framework relates to state-of-the-art dynamical system theory approaches. Most of these approaches rely on delay embedding, as Takens's theorem states the existence of a delay embedding in which the topological properties of the hidden dynamical system are equivalent to those of the true systems up to a diffeomorphic mapping. Hence, state-of-the-art approaches typically combine the selection of a delay embedding representation within classic regression models to represent the one-step-ahead mapping in the considered embedding. Here, we consider latent dynamics governed by an unknown ODE (4) but we do not explicitly state the latent space. This is however implicit in our forecasting framework. By construction, the considered forecasting model relies on the integration of the learnt ODE (4) from an initial condition given as the solution of minimization (7). Let us consider the following embedding ψ such that:

$$\psi\left(\{\mathbf{x}_t\}_{t_0:T}\right) = \arg\min_{\mathbf{u}_T} \min_{\{\mathbf{u}_t\}_{t<T}} \sum_{t=1}^{T} \|\mathbf{x}_t - \mathcal{M}^{-1}(G\left(\Phi_{\theta,t}\left(\mathbf{u}_{t-1}\right)\right))\|^2 + \lambda\|\mathbf{u}_t - \Phi_{\theta,t}(\mathbf{u}_{t-1})\|^2 \quad (9)$$

Given this embedding, the resulting one-step-ahead forecasting for the observed variable may written as:

$$\mathbf{x}_{T+1} = \mathcal{M}(G\left(\Phi_{\theta,t}\left(\psi\left(\{\mathbf{x}_t\}_{t=t_0:T}\right)\right)\right)) \quad (10)$$

Hence, ψ defines a delay embedding representation implicitly stated through minimization (7). In this embedding, the dynamics of the observed system $\mathbf{x}$ is governed by the composition of observation operator G and forecasting operator $\Phi_{\theta,t}$. Regarding the literature on Koopman operator theory, most approaches rely on the explicit identification of eigenfunctions and eigenvalues of the Koopman operator Koopman (1931); Brunton et al. (2016a); Tu et al. (2014). Our framework relates to the identification of the infinitesimal generator f_θ of the one-parameter subgroup defined by Koopman operator through the ODE representation (4). By construction, the Koopman operator associated with the identified operator $f_{\hat{\theta}}$ is also diagonalizable, such that the identification of infinitesimal generator $f_{\hat{\theta}}$ provides an implicit decomposition of the Koopman operator of the underlying and unknown dynamical system onto the eigenbasis of the learnt latent dynamics governed by ODE (4).

Future work will further explore methodological aspects, especially the application to high-dimensional and stochastic systems. In the considered framework, operator $\mathcal{M}$ is stated as an identity operator on the observed component of state $\mathbf{u}_t$ or as a simple PCA projection. Although for the geociences community, using PCA to reduce the dimensionality is motivated by the Galerkin derivation of reduced order models from complex high dimensional governing partial differential equations Holmes et al. (2012), using auto-encoders have shown promising results in discovering optimal coordinates when trained jointly with a dynamical system. The combination of the proposed framework with the variational setting considered in the Latent-ODE model Chen et al. (2018) also appears as an interesting direction for future work. The extension to stochastic systems through the identification of a Stochastic ODE is also of key interest, for instance for future applications of the

proposed framework to geophysical random flows, especially to the simulation and forecasting of ocean-atmosphere dynamics in which stochastic components naturally arise Chapron et al. (2018).

REFERENCES

Henry D. I. Abarbanel. *Modeling Chaos*, pp. 95–114. Springer New York, New York, NY, 1996a. ISBN 978-1-4612-0763-4. doi: 10.1007/978-1-4612-0763-4_6. URL https://doi.org/10.1007/978-1-4612-0763-4_6.

Henry D. I. Abarbanel. *Choosing Time Delays*, pp. 25–37. Springer New York, New York, NY, 1996b. ISBN 978-1-4612-0763-4. doi: 10.1007/978-1-4612-0763-4_3. URL https://doi.org/10.1007/978-1-4612-0763-4_3.

Henry D. I. Abarbanel. *Choosing the Dimension of Reconstructed Phase Space*, pp. 39–67. Springer New York, New York, NY, 1996c. ISBN 978-1-4612-0763-4. doi: 10.1007/978-1-4612-0763-4_4. URL https://doi.org/10.1007/978-1-4612-0763-4_4.

Henry D. I. Abarbanel and Upmanu Lall. Nonlinear dynamics of the great salt lake: system identification and prediction. *Climate Dynamics*, 12(4):287–297, Mar 1996. ISSN 1432-0894. doi: 10.1007/BF00219502. URL https://doi.org/10.1007/BF00219502.

Steven L Brunton, Bingni W Brunton, Joshua L Proctor, and J Nathan Kutz. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. *PloS one*, 11(2):e0150171, 2016a.

Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, April 2016b. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1517384113. URL http://www.pnas.org/lookup/doi/10.1073/pnas.1517384113.

Stéphane Calmant, Frédérique Seyler, and Jean François Cretaux. Monitoring continental surface waters by satellite altimetry. *Surveys in Geophysics*, 29(4):247–269, Oct 2008. ISSN 1573-0956. doi: 10.1007/s10712-008-9051-1. URL https://doi.org/10.1007/s10712-008-9051-1.

Kathleen Champion, Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Data-driven discovery of coordinates and governing equations. *arXiv preprint arXiv:1904.02107*, 2019.

Bertrand Chapron, Pierre Dérian, Etienne Mémin, and Valentin Resseguier. Large-scale flows under location uncertainty: a consistent stochastic framework. *Quarterly Journal of the Royal Meteorological Society*, 144(710):251–260, 2018.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pp. 6571–6583, 2018.

Earl A Coddington and Norman Levinson. *Theory of ordinary differential equations*. Tata McGraw-Hill Education, 1955.

Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. *arXiv preprint arXiv:1904.01681*, 2019.

Otto Fabius and Joost R van Amersfoort. Variational recurrent auto-encoders. *arXiv preprint arXiv:1412.6581*, 2014.

R. Fablet, S. Ouala, and C. Herzet. Bilinear residual neural network for the identification and forecasting of geophysical dynamics. In *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 1477–1481, Sep. 2018. doi: 10.23919/EUSIPCO.2018.8553492.

R. Fablet, J. Verron, B. Mourre, B. Chapron, and A. Pascual. Improving Mesoscale Altimetric Data From a Multitracer Convolutional Processing of Standard Satellite-Derived Products. *IEEE Transactions on Geoscience and Remote Sensing*, 56(5):2518–2525, may 2018. ISSN 0196-2892. doi: 10.1109/TGRS.2017.2750491.

Ronan Fablet, Said Ouala, and Cedric Herzet. Bilinear residual Neural Network for the identification and forecasting of dynamical systems. *SciRate*, dec 2017. URL https://scirate.com/arxiv/1712.07003.

Jordan Frank, Shie Mannor, and Doina Precup. Activity and gait recognition with time-delay embeddings. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, pp. 1581–1586. AAAI Press, 2010. URL http://dl.acm.org/citation.cfm?id=2898607.2898859.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, december 2015. URL http://arxiv.org/abs/1512.03385. arXiv: 1512.03385.

A. C. Hindmarsh. ODEPACK, a systematized collection of ODE solvers. *IMACS Transactions on Scientific Computation*, 1:55–64, 1983.

Philip Holmes, John L. Lumley, Gahl Berkooz, and Clarence W. Rowley. *Galerkin projection*, pp. 106129. Cambridge Monographs on Mechanics. Cambridge University Press, 2 edition, 2012. doi: 10.1017/CBO9780511919701.006.

Jinhee Jeong and Fazle Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285: 6994, 1995. doi: 10.1017/S0022112095000462.

M. Juza, B. Mourre, L. Renault, S. Gmara, K. Sebastin, S. Lora, J. P. Beltran, B. Frontera, B. Garau, C. Troupin, M. Torner, E. Heslop, B. Casas, R. Escudier, G. Vizoso, and J. Tintor. Socib operational ocean forecasting system and multi-platform validation in the western mediterranean sea. *Journal of Operational Oceanography*, 9(sup1):s155–s166, 2016. doi: 10.1080/1755876X.2015.1117764. URL https://doi.org/10.1080/1755876X.2015.1117764.

Ahmad Kazem, Ebrahim Sharifi, Farookh Khadeer Hussain, Morteza Saberi, and Omar Khadeer Hussain. Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Applied Soft Computing*, 13(2):947 – 958, 2013. ISSN 1568-4946. doi: https://doi.org/10.1016/j.asoc.2012.09.024. URL http://www.sciencedirect.com/science/article/pii/S1568494612004449.

B. O. Koopman. Hamiltonian systems and transformations in hilbert space. *Proceedings of the National Academy of Sciences of the United States of America*, 17(5):315–318, 1931. ISSN 00278424. URL http://www.jstor.org/stable/86114.

Tjalling C. Koopmans. Identification problems in economic model construction. *Econometrica*, 17 (2):125–144, 1949. ISSN 00129682, 14680262. URL http://www.jstor.org/stable/1905689.

Rahul G. Krishnan, Uri Shalit, and David Sontag. Structured Inference Networks for Nonlinear State Space Models. *arXiv:1609.09869 [cs, stat]*, September 2016. URL http://arxiv.org/abs/1609.09869. arXiv: 1609.09869.

Tze Leung Lai and Ching Zong Wei. Least squares estimates in stochastic regression models with applications to identification and control of dynamic systems. *The Annals of Statistics*, 10(1): 154–166, 1982. ISSN 00905364. URL http://www.jstor.org/stable/2240506.

Redouane Lguensat, Pierre Tandeo, Pierre Ailliot, Manuel Pulido, and Ronan Fablet. The Analog Data Assimilation. *Monthly Weather Review*, aug 2017. ISSN 0027-0644, 1520-0493. doi: 10.1175/MWR-D-16-0441.1. URL http://journals.ametsoc.org/doi/10.1175/MWR-D-16-0441.1.

Edward N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, 20(2): 130–141, March 1963. ISSN 0022-4928. doi: 10.1175/1520-0469(1963)020⟨0130:DNF⟩2.0.CO;2. URL http://journals.ametsoc.org/doi/abs/10.1175/1520-0469(1963)020%3C0130:DNF%3E2.0.CO;2.

Peter Lynch and Xiang-Yu Huang. *Initialization*, pp. 241–260. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. ISBN 978-3-540-74703-1. doi: 10.1007/978-3-540-74703-1_9. URL https://doi.org/10.1007/978-3-540-74703-1_9.

Duong Nguyen, Said Ouala, Lucas Drumetz, and Ronan Fablet. Em-like learning chaotic dynamics from noisy and partial observations. *SciRate*, Mar. 2019. URL https://scirate.com/arxiv/1903.10335.

S. Ouala, A. Pascual, and R. Fablet. Residual integration neural network. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3622–3626, May 2019. doi: 10.1109/ICASSP.2019.8683447.

Johan Paduart, Lieve Lauwers, Jan Swevers, Kris Smolders, Johan Schoukens, and Rik Pintelon. Identification of nonlinear systems using Polynomial Nonlinear State Space models. *Automatica*, 46(4):647–656, April 2010. ISSN 0005-1098. doi: 10.1016/j.automatica.2010.01.001. URL http://www.sciencedirect.com/science/article/pii/S000510981000021X.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Multistep neural networks for data-driven discovery of nonlinear dynamical systems. *arXiv preprint arXiv:1801.01236*, 2018.

Tim Sauer, James A. Yorke, and Martin Casdagli. Embedology. *Journal of Statistical Physics*, 65 (3):579–616, Nov 1991. ISSN 1572-9613. doi: 10.1007/BF01053745. URL https://doi.org/10.1007/BF01053745.

Houshang H Sohrab. *Basic real analysis*, volume 231. Springer, 2003.

Julien Clinton Sprott. *Chaos and Time-Series Analysis*. Oxford University Press, Inc., New York, NY, USA, 2003. ISBN 0198508409.

Floris Takens. Detecting strange attractors in turbulence. In David Rand and Lai-Sang Young (eds.), *Dynamical Systems and Turbulence, Warwick 1980*, pp. 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg. ISBN 978-3-540-38945-3.

Jonathan H Tu, Clarence W Rowley, Dirk M Luchtenburg, Steven L Brunton, and J Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1 (2):391–421, 2014.

Steffen Wiewel, Moritz Becher, and Nils Thuerey. Latent-space physics: Towards learning the temporal evolution of fluid flow, 2018.

Han Zhang, Xi Gao, Jacob Unterman, and Tom Arodz. Approximation capabilities of neural ordinary differential equations. *arXiv preprint arXiv:1907.12998*, 2019.

APPENDIX

A PROOF OF PROPOSITION 1

This proposition can be easily extended to any observation function that doesn't form an embedding of the initial unobserved ODE. However, for the sake of simplicity, we will consider the example given in Eq. (1).

Lets suppose a a smooth ODE in the observation space that governs the time evolution of $\mathbf{x}$ from Eq. (1).

$$\begin{cases} \dot{\mathbf{x}}_t = f(\mathbf{x}_t) \\ \mathbf{x}_{t_0} = \mathbf{x}_0 \end{cases} \tag{11}$$

This ODE generates a flow $\mathbf{x}_t = \Psi_t(\mathbf{x}_0)$.

Since our observation operator is not one-to-one, we can assume the existence of some $\hat{t}, t_1, t_2$ where $Real(\Phi_{\hat{t}}(\mathbf{z}_{t_1})) = Real(\Phi_{\hat{t}}(\mathbf{z}_{t_2}))$ with $Real(\mathrm{z}_{t_1}) \neq Real(\mathrm{z}_{t_2})$ (Φ is the flow generated by the unobserved ODE illustrated in Eq. (1)). Projecting this equality to the observation space leads to : $\Psi_{\hat{t}}(\mathbf{x}_{t_1}) = \Psi_{\hat{t}}(\mathbf{x}_{t_2})$ with $\mathbf{x}_{t_1} \neq \mathbf{x}_{t_2}$.

Since the above ODE is smooth (or continuously differentiable), we can show that f is locally Lipschitz on any interval containing t_0 Sohrab (2003) which garentees by Picard's Existence Theorem the existance of a unique solution Coddington & Levinson (1955). Formally, for the times $\hat{t}, t_1, t_2$, $\Psi_{\hat{t}}(\mathbf{x}_{t_1}) = \Psi_{\hat{t}}(\mathbf{x}_{t_2})$ if and only if $\mathbf{x}_{t_1} = \mathbf{x}_{t_2}$. This contradicts the assumption that $\mathbf{x}_{t_1} \neq \mathbf{x}_{t_2}$ and thus, there is no existence of a $\hat{t}$ such that $Real(\Phi_{\hat{t}}(\mathbf{z}_{t_1})) = Real(\Phi_{\hat{t}}(\mathbf{z}_{t_2}))$ with $Real(\mathrm{z}_{t_1}) \neq Real(\mathrm{z}_{t_2})$.

B DIMENSIONALITY ANALYSIS OF THE NBEDDYN MODEL

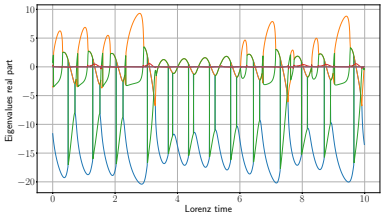
One of the Key parameters of the proposed model is the dimension of the latent space. Despite the fact that it is extremely challenging to get a prior idea of the dimension of the model in the case of real data experiments, one can analyze the spawned manifold of the learnt latent states to get an idea of the true dimension of the underlying model (true here stands for a sufficient dimension of the latent space). The idea here is to compute the modulus of the eigenvalues of the Jacobian matrix for each input of the training data. An eigenvalue does not influence the temporal evolution of the latent state if it has a modulus that tend to zero. The number of non-zero eigenvalues can then be seen as a sufficient dimension of the latent space.

Regarding the identification of an ODE model governing the first state variable of the Lorenz 63 model, Fig. 3 illustrates the eigenvalues of the Jacobian matrix and their modulus for a dimension of the latent space $d_E = 6$. Interestingly, only 3 eigenvalues have non-zero modulus and are effectively influencing the underlying dynamics. This result shows that one can use a 3 dimensional latent-space as a sufficient dimension to identify an ODE model governing the first state of the Lorenz 63 system which is the same dimension as the true Lorenz 63 model.

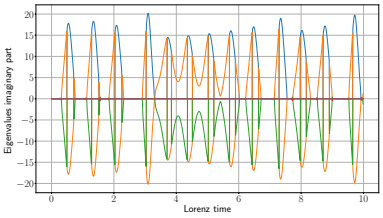
The analysis of the eigenvalues of the Sea Level Anomaly model in the other hand are not as straightforward as in the case of the Lorenz model since we do not have any idea on the analytical form of the underlying dynamical model. Fig. 4 illustrates that using a 60 dimensional latent space for the NbedDyn model, only 50 eigenvalues have non-zero modulus and thus, are effectively influencing the underlying dynamics. The conclusion in this case is that the observed SLA data evolve in a 50 dimensional latent space parametrised by the dynamical model f_θ.

C ADDITIONAL FIGURES OF THE LORENZ 63 EXPERIMENT

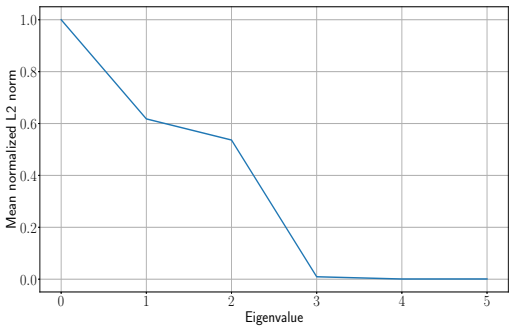
We illustrate the forecasting performance of the tested models for the Lorenz-63 experiment through an example of forecasted trajectories in Fig. 8. Our model with $d_E = 6$ leads to a trajectory similar to the true one up to 7 Lyapunov times, when the best alternative approach diverge from the true trajectory beyond 4 Lyapunov times.



(a) Eigenvalues real part.



(b) Eigenvalues imaginary part.



(c) Eigenvalues modulus.

Figure 3: ***Analysis of the eigenvalues of the NbedDyn model Jacobian matrix.***: Lorenz-63 case-study with $d_E = 6$

An other interesting experiment is to find the initial condition for new observation data. This issue is addressed as presented in section 3 as follow. Given a new noisy and partial observation sequence (Fig. 6), we first look for a potential initial condition in the inferred training latent state sequence. This initial condition is then optimized using the cost function described by equation (7) to minimize the forecasting error of the new observation sequence.

D ADDITIONAL FIGURES OF THE SEA LEVEL ANOMALY EXPERIMENT

Forecasted states of the Sea Level Anomaly are illustrated in Fig. 7 and 9. The visual analysis of the forecasted SLA states emphasize the relevance of the proposed NbedDyn model. While state of the art approaches generally overestimate the time evolution of some structures such as eddies, our model is the only one to give near perfect forecasting up to 4 days.

E NEURAL NETWORKS HYPERPARAMETERS

E.1 LORENZ 63 EXPERIMENTS HYPERPARAMETERS

E.2 SLA EXPERIMENTS HYPERPARAMETERS

F SCOOP AND LIMITATIONS

Constraining limit cycles The proposed augmented ODE formulation does not suppose any prior knowledge on the underlying dynamics responsible for the temporal evolution of the observations. This can lead in some cases (especially when working on chaotic dynamics) to output a dynamical representation that has several attracting regions in addition to the one leading to the observations limit cycle. This can lead to inappropriate results when trying to find an initial condition that forecasts a given observation sequence. The Idea of using the manifold spanned by the augmented

Parameter	Value
Number of LSTM layers	10
Hidden size	10
Sequence length	30
Learning rate	0.001
Optimizer	Adam
Training data	4000

Table 3: RNN parameters in the Lorenz 63 Experiment.

Parameter	Value
Latent dimension	4
Hidden size	15
RNN hidden size	100
Learning rate	0.01
Optimizer	Adam
Training data	4000

Table 4: Latent-ODE parameters in the Lorenz 63 Experiment, please refer to Chen et al. (2018) for more details.

Parameter	Value
Augmented Latent dimension	6
Number of bilinear layers	6
Number of linear layers	6
Integration scheme	Runge Kutta 4
Learning rate	0.001
Optimizer	Adam
Training data	4000

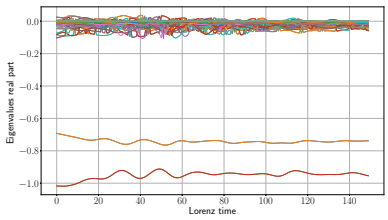
Table 5: NbedDyn parameters in the Lorenz 63 Experiment, please refer to Fablet et al. (2017) for more details.

Parameter	Value
Number of LSTM layers	5
Hidden size	20
Sequence length	40
Learning rate	0.001
Optimizer	Adam
Training data	2000

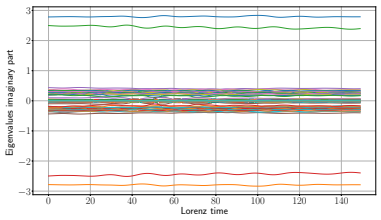
Table 6: RNN parameters in the SLA Experiment.

Parameter	Value
Latent dimension	60
Hidden size	70
RNN hidden size	200
Learning rate	0.01
Optimizer	Adam
Training data	2000

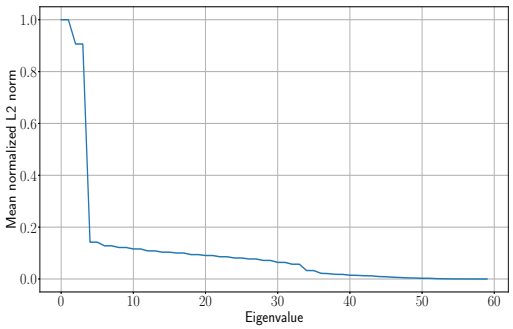
Table 7: Latent-ODE parameters in the SLA Experiment, please refer to Chen et al. (2018) for more details.



(a) Eigenvalues real part.



(b) Eigenvalues imaginary part.



(c) Eigenvalues modulus.

Figure 4: ***Analysis of the eigenvalues of the NbedDyn model Jacobian matrix.***: Sea Level Anomaly case-study with $d_E = 60$

Parameter	Value
Augmented Latent dimension	60
Number of bilinear layers	60
Number of linear layers	60
Integration scheme	Runge Kutta 4
Learning rate	0.001
Optimizer	Adam
Training data	2000

Table 8: NbedDyn parameters in the SLA Experiment, please refer to Fablet et al. (2017) for more details.

training data allows to bypass this issue but we believe that adding additional constraints (energy preserving constraints, known symmetries in the models ...etc.) can significantly improve the quality of the learnt dynamical models.

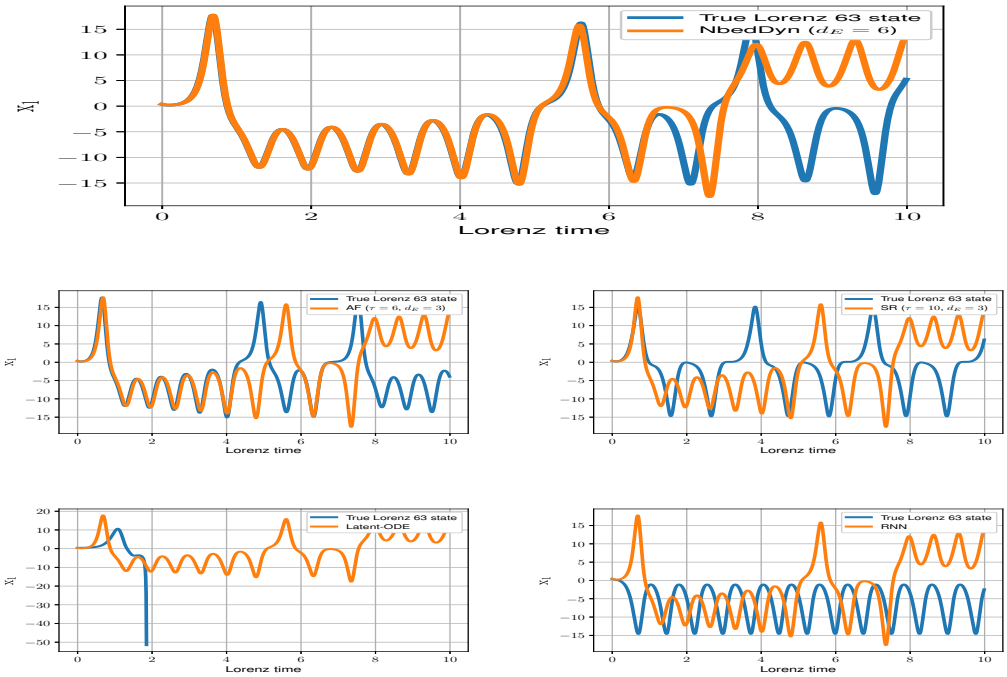


Figure 5: ***Generated time series of the proposed models.*** : Given the same initial condition, we generated a time series of 1000 time steps.

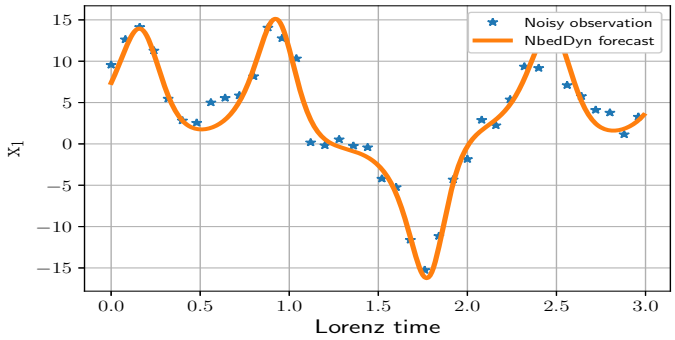
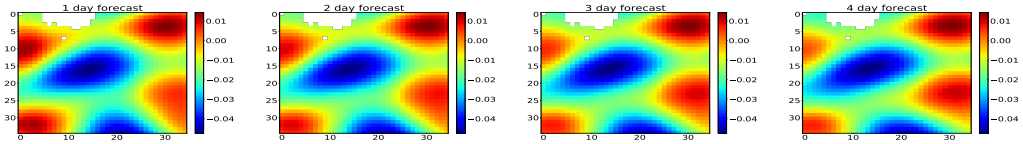


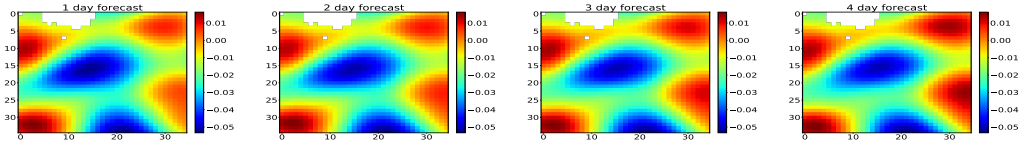
Figure 6: ***Forecasted Lorenz 63 state sequence given noisy and partial observations***: Given noisy and partial observations, our model optimizes equation (7) to infer an initial condition that minimize the forecasting of the observations.

G CODE SAMPLE

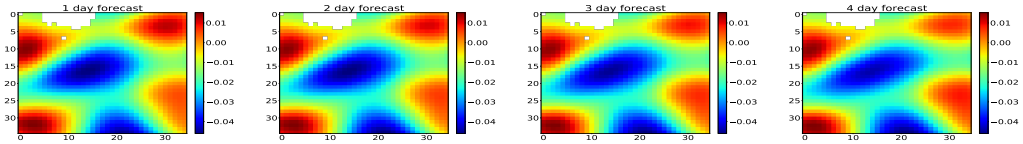
```
# import libs
from generate_data import generate_data
import numpy as np
import torch
from torch.autograd import Variable
seed = 0
np.random.seed(seed)
torch.manual_seed(seed)
#Generate data
class GD:
```



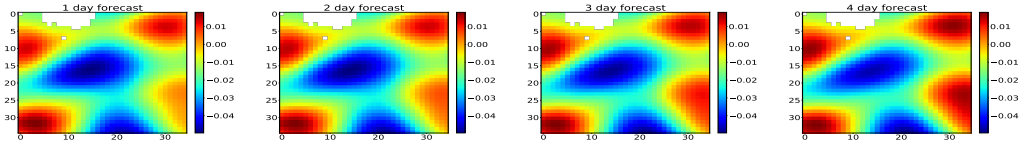
(a) Ground truth.



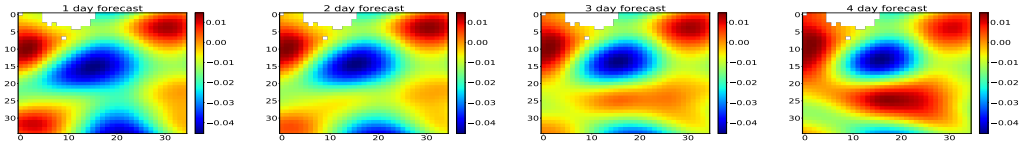
(b) Analogs forecasting.



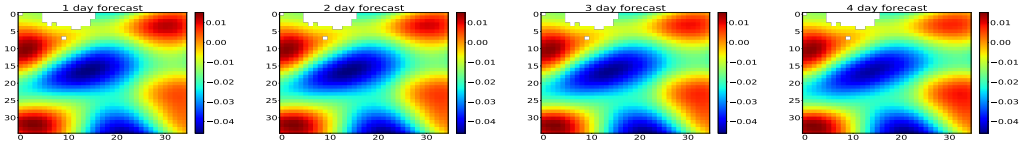
(c) Sparse regression.



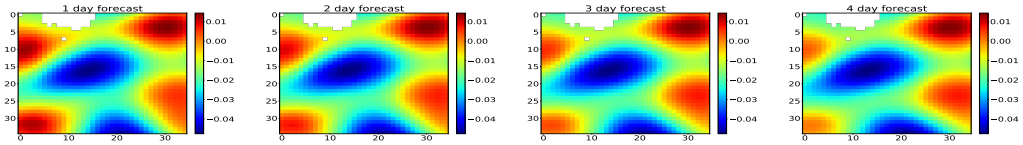
(d) Latent-ODE.



(e) RNN.



(f) NbedDynZERO.



(g) NbedDyn forecast.

Figure 7: ***Forecasted SLA states of the proposed models.***

```
model = 'Lorenz_63'
class parameters:
    sigma = 10.0
    rho = 28.0
```

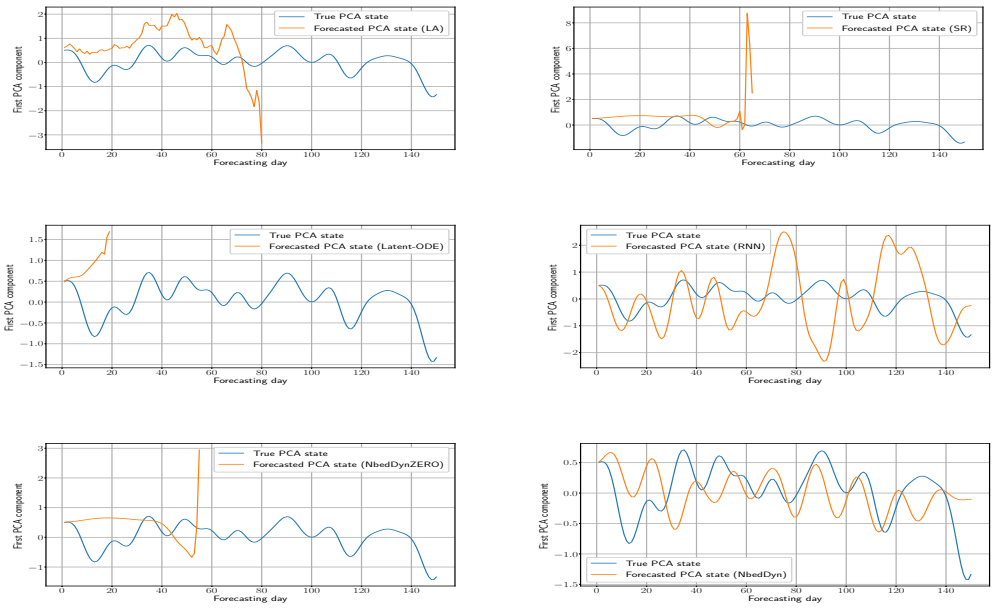
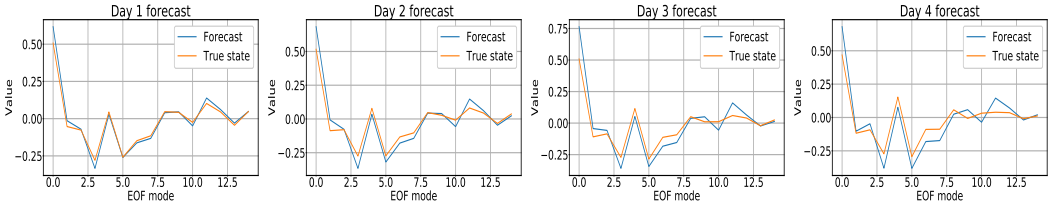


Figure 8: ***Generated time series of the proposed models for the forecasting of the SLA dynamics.*** : Given the same initial condition, we generated a time series of 150 days.

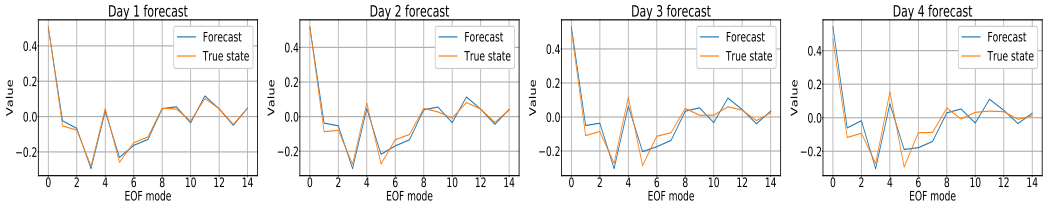
```
        beta = 8.0/3
    dt_integration = 0.01 # integration time
    nb_loop_data = 50.01 # size of the catalog
    test_samples = 1000
# run the data generation
catalog, xt, yo = generate_data(GD)
X_test    = catalog.analogs[catalog.analogs.shape[0]-GD.test_samples:,:1]
X_train   = catalog.analogs[:catalog.analogs.shape[0]-GD.test_samples,:1]
Grad_t    = np.gradient(X_train[:,0]).reshape(X_train.shape[0],1)\
/GD.dt_integration
x = Variable(torch.from_numpy(X_train).float())
z = Variable(torch.from_numpy(Grad_t).float())
data_size = X_train.shape[0]
#neural net params
params = {}
params['transition_layers']=1
params['bi_linear_layers']=6
params['dim_hidden_linear'] = 6
params['dim_input']=1
params['dim_latent']=5
params['dim_output']=params['dim_input'] + params['dim_latent']
params['dt_integration'] = GD.dt_integration
#Dynamical model
class FC_net(torch.nn.Module):
        def __init__(self, params):
            super(FC_net, self).__init__()
            y_aug = np.random.uniform(size=(data_size,params['dim_latent']))
            self.y_aug = torch.nn.Parameter(torch.from_numpy(y_aug).float())
            self.linearCell    = torch.nn.Linear(params['dim_output']\
                                                 , params['dim_hidden_linear'])
            self.BlinearCell1 = \
            torch.nn.ModuleList(\
            [torch.nn.Linear(params['dim_output'], 1)\
```

```
            for i in range(params['bi_linear_layers'])])
            self.BlinearCell2 = \
            torch.nn.ModuleList(\
            [torch.nn.Linear(params['dim_output'], 1)\
            for i in range(params['bi_linear_layers'])])
            augmented_size     = params['bi_linear_layers']\
            + params['dim_hidden_linear']
            self.transLayers = \
            torch.nn.ModuleList(\
            [torch.nn.Linear(augmented_size, params['dim_output'])])
            self.transLayers.extend(\
            [torch.nn.Linear(params['dim_output'], params['dim_output'])\
            for i in range(1, params['transition_layers'])])
            self.outputLayer  = torch.nn.Linear(params['dim_output'],\
                                                params['dim_output'])
        def forward(self, inp, dt):
            if inp.shape[-1]<params['dim_latent']+params['dim_input']:
                aug_inp = torch.cat((inp, self.y_aug), dim=1)
            else:
                aug_inp = inp
            BP_outp = Variable(torch.zeros((aug_inp.size()[0],\
                                            params['bi_linear_layers'])))
            L_outp   = self.linearCell(aug_inp)
            for i in range((params['bi_linear_layers'])):
                BP_outp[:,i]=self.BlinearCell1[i](aug_inp)[:,0]*\
                self.BlinearCell2[i](aug_inp)[:,0]
            aug_vect = torch.cat((L_outp, BP_outp), dim=1)
            for i in range((params['transition_layers'])):
                aug_vect = (self.transLayers[i](aug_vect))
            grad = self.outputLayer(aug_vect)
            return grad, aug_inp
model  = FC_net(params)
# compute flow : RK4
class INT_net(torch.nn.Module):
        def __init__(self, params):
            super(INT_net, self).__init__()
#            self.add_module('Dyn_net',FC_net(params))
            self.Dyn_net = model
        def forward(self, inp, dt):
            k1, aug_inp   = self.Dyn_net(inp,dt)
            inp_k2 = aug_inp + 0.5*params['dt_integration']*k1
            k2, tmp   = self.Dyn_net(inp_k2,dt)
            inp_k3 = aug_inp + 0.5*params['dt_integration']*k2
            k3, tmp   = self.Dyn_net(inp_k3,dt)
            inp_k4 = aug_inp + params['dt_integration']*k3
            k4, tmp   = self.Dyn_net(inp_k4,dt)
            pred = aug_inp +dt*(k1+2*k2+2*k3+k4)/6
            return pred, k1, inp, aug_inp
#Instanciate the model
modelRINN = INT_net(params)
criterion = torch.nn.MSELoss(reduction='elementwise_mean')
optimizer = torch.optim.Adam(model.parameters())
#Pretraining : fit the gradient
params['ntrain']=[300000,10000]
for t in range(params['ntrain'][0]):
        pred, grad, inp, aug_inp = modelRINN(x,params['dt_integration'])
        loss1 = criterion(grad[:,:1], z)
        loss2 = criterion(pred[:-1,:], aug_inp[1:,:])
        loss = 0.9*loss1+0.1*loss2
```

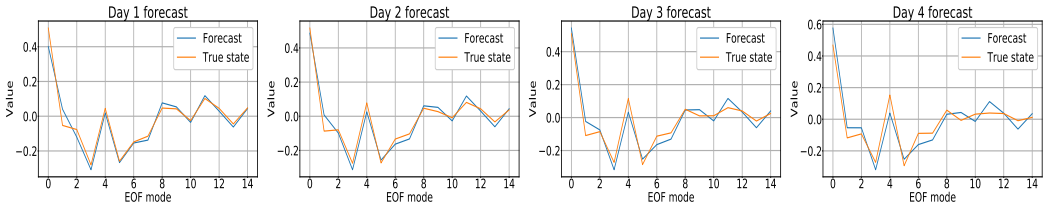
```
        print(t,loss)
        optimizer.zero_grad()
        loss.backward(retain_graph=True)
        optimizer.step()
# training
for t in range(params['ntrain'][1]):
        pred, grad, inp, aug_inp = modelRINN(x,params['dt_integration'])
        loss1 = criterion(pred[:-1,:], aug_inp[1:,:])
        loss2 = criterion(pred[:-1,1:], aug_inp[1:,1:])
        loss =10.0*loss1 + 1.0*loss2
        print(t,loss)
        optimizer.zero_grad()
        loss.backward(retain_graph=True)
        optimizer.step()
```



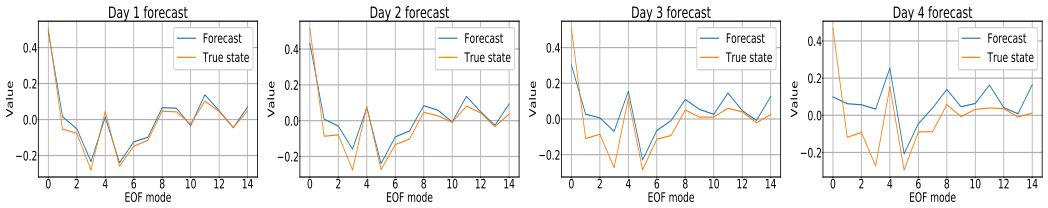
(a) Analog forecasting.



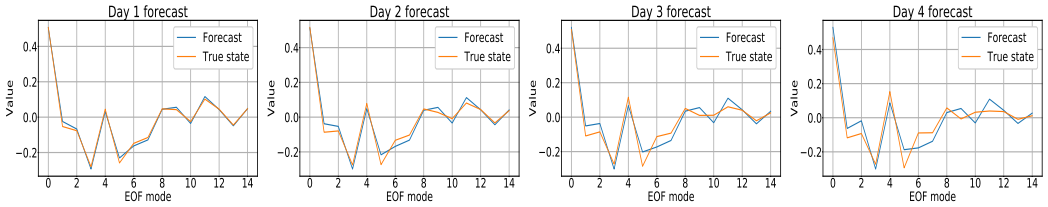
(b) Sparse regression.



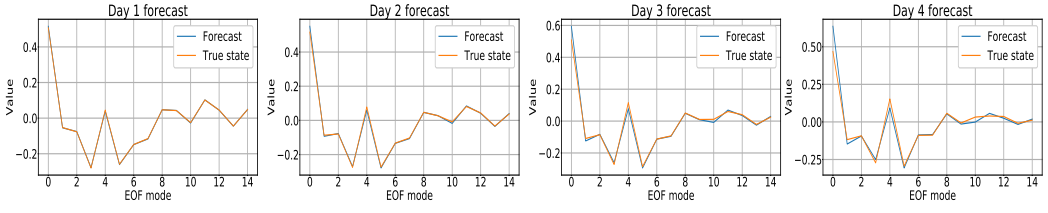
(c) Latent-ODE.



(d) RNN.



(e) NbedDynZERO.



(f) NbedDyn.

Figure 9: ***Forecasted EOF components of the proposed models.***