Interpretable learning of effective dynamics for multiscale systems

PROCEEDINGS A

royalsocietypublishing.org/journal/rspa

Research

Cite this article: Menier E, Kaltenbach S, Yagoubi M, Schoenauer M, Koumoutsakos P. 2025 Interpretable learning of effective dynamics for multiscale systems. *Proc. R. Soc. A* **481**: 20240167.
https://doi.org/10.1098/rspa.2024.0167

Received: 1 March 2024
Accepted: 24 October 2024

Subject Category:
Computer Science

Subject Areas:
computational physics, applied mathematics, artificial intelligence

Keywords:
model-order reduction, interpretability, Mori–Zwanzig formalism, neural networks, Koopman operator, multiscale systems

Author for correspondence:
Petros Koumoutsakos
e-mail: petros@seas.harvard.edu

Interpretable learning of effective dynamics for multiscale systems

Emmanuel Menier[1,2], Sebastian Kaltenbach[3,4], Mouadh Yagoubi[2], Marc Schoenauer[1] and Petros Koumoutsakos[3]

[1]Tau, Inria/LISN, Université Paris-Saclay/CNRS, 1 Rue Raimond Castaing, Gif-sur-Yvette, 91190, France
[2]IRT SystemX, 8 Avenue de la Vauve, Palaiseau, 91120, France
[3]Harvard SEAS, 29 Oxford Street, Cambridge, MA 02138, USA
[4]Computational Science and Engineering Laboratory, ETH Zürich, 8092, Switzerland

PK, 0000-0001-8337-2122

The modelling and simulation of high-dimensional multiscale systems is a critical challenge across all areas of science and engineering. It is broadly believed that even with today's computer advances resolving all spatio-temporal scales described by the governing equations remains a remote target. This realization has prompted intense efforts to develop model-order reduction techniques. In recent years, techniques based on deep recurrent neural networks (RNNs) have produced promising results for the modelling and simulation of complex spatiotemporal systems and offer large flexibility in model development as they can incorporate experimental and computational data. However, neural networks lack interpretability, which limits their utility and generalizability across complex systems. Here, we propose a novel framework of interpretable learning effective dynamics (iLED) that offers comparable accuracy to state-of-the-art RNN-based approaches while providing the added benefit of interpretability. The iLED framework is motivated by Mori–Zwanzig and Koopman operator (KO) theory, which justifies the choice of the specific architecture. We demonstrate the effectiveness of the proposed framework in simulations of three benchmark multiscale systems. Our results show that the iLED framework can generate accurate predictions and obtain interpretable dynamics, making it a promising approach for solving high-dimensional multiscale systems.

THE ROYAL SOCIETY PUBLISHING

1. Introduction

The accurate forecasting of critical phenomena, such as weather systems and epidemics, depends on the veracity and efficiency of numerical simulations. These simulations frequently rely on models governed by partial differential equations (PDEs) that describe the intricate interplay of multiphysics and multiscale processes. Notable examples include models for turbulence [1], climate systems [2], neuroscience [3] and oceanic dynamics [4]. Owing to substantial progress in numerical techniques, algorithms, software and computational hardware, we are now exploring simulation capabilities far beyond what was conceivable just a few decades ago. However, it is becoming evident that the reliability and energy costs of simulations are reaching their limits [5]. Multitudes of spatiotemporal scales in such systems present high, if not insurmountable, barriers for classical numerical methods. Multiscale methods aim to resolve this issue by judicious approximations of the various scales and the interactions between physical processes occurring over different scales.

Throughout the years, several powerful frameworks have been introduced, such as the equation-free framework (EFF) [6–8], the heterogeneous multiscale method (HMM) [9,10] and the fLow aVeraged integatoR (FLAVOR) [11]. These frameworks owe their success to the ability to separate scales within system dynamics and effectively capture the exchange of information across those scales. While it is clear that EFF, HMM and FLAVOR have transformed the field of multiscale modelling and simulation, their full potential remains limited by two key challenges: (i) the accuracy of coarse-grained dynamics propagation depends heavily on the time integrators used and (ii) inefficient information transfer, particularly from coarse to fine scales, significantly constrains their effectiveness. An alternative and very potent approach are reduced-order models (ROMS) [12,13] aim to capture the effective system dynamics, and they can also lead to interpretable descriptions allowing better understanding of the underlying system [14,15]. The success of ROMs hinges on identifying the low-dimensional sub-spaces, on which the dynamics of a system can be summarized. The existence of such low-dimensional sub-spaces has been well established for systems ranging from the one-dimensional Kuramoto–Shivashinsky (KS) equation [16] to highly complex fluid mechanics problems [17]. Linear reduction methods such as the proper orthogonal decomposition have attracted significant attention as foundations of ROMs [18] thanks to their simplicity and stability. Indeed, they can be used to build analytical ROMs-based governing equations and the chosen proper orthogonal decomposition (POD) modes [19,20]. However, they have shown limitations for the reduction of highly nonlinear dynamics [21–24], such as those encountered in turbulent flows or chemical reactions. Various approaches have been proposed to address these limitations such as online basis adaption strategies [25] or extensions to nonlinear projection-based reduced-order modelling [26,27], which are able to achieve significant speedups. More recently, the potential of deep learning methods in the context of model reduction has become a major topic of research. Neural autoencoders can be used to optimally represent the low-dimensional, nonlinear manifolds on which dynamical systems evolve [28,29]. The latent space identified by these autoencoders can then be used in combination with dynamical modelling approaches to construct reduced models of any system of interest.

More importantly, the deployment of autoencoders for identifying latent space dynamics can be combined with the ideas of EFF, HMM and FLAVOR. This is the core idea of the recently proposed framework of *learning effective dynamics* (*LED*) [30]. LED deploys neural network autoencoders for dimensionality reduction of the high-dimensional samples, thus identifying the structure of the latent space via a nonlinear mapping. A second type of neural network architecture, the LSTM [31], is then deployed to learn the dynamics of the reduced system. While this framework has shown promising results its functioning remains non-interpretable. In this work, we present interpretable LED (iLED) that employs for its latent dynamics a theoretically grounded, dynamical model that replaces the LSTM currently in use. The method of iLED is constructed around interpretable linear dynamics and completed by a physically motivated nonlinear closure. The proposed framework is closely related to the Mori–Zwanzig theory [32,33]

and Koopman operator (KO) theory [13,34,35]. We note that the latent variables are chosen not only regarding their potential to reconstruct the high-dimensional system of interest but also to have a close-to-linear dynamics that can be effectively closed with nonlinear terms.

The use of Mori–Zwanzig and Koopman theories allows our model to have a strong inductive bias, which is not the case with black-box models such as neural ordinary differential equations (ODE) and stochastic differential equations (SDE) [36,37]. Approaches inspired by dictionary learning that select the potentially nonlinear terms for the dynamics from a pre-defined set of candidate functions [38–40], are interpretable as well but suffer from poor generalization and do require problem-specific knowledge. iLED does not require the definition of a set of candidate functions or any knowledge regarding the governing equations of the system of interest.

There are a number of potent approaches involving linear models for describing the latent dynamics. Two notable cases are models based on KOs [35,41,42] and dynamic mode decomposition (DMD) [43,44]. The KO approaches rely on the assumption that the latent space dynamics are Markovian and, as such, they may not capture possible memory effects. DMD can capture certain time dependencies, but it has limitations in systems with strong nonlinearities. Other approaches that can lead to interpretable solutions [45] include modified recurrent neural network (RNN) architectures [46], incorporating physical constraints as virtual observables [40] or restricting the dynamics of the latent space [15,47]. Moreover, there are some models that employ existing ROMs and learn a closure term [20,48].

The iLED, presented in this article, employs temporal dynamics with explicit linear and nonlinear components allowing for higher flexibility and interpretability. It is based on both Mori–Zwanzig formalism [32,33] as well as KO theory [34,35]. We emphasize that the present method is *non-intrusive* and can be deployed to model systems for which the governing equations are unknown.

The remainder of this article is structured as follows: in §2, we present the general methodological framework with special attention on the connection between our novel framework and Mori–Zwanzig formalism as well as KO theory. Computational aspects related to training the framework and generating predictions are discussed in §2d. Afterwards, we present three numerical illustrations in §3 and demonstrate the accuracy and interpretability of iLED over classical deep learning modelling approaches. Section 4 summarizes the contribution of this work and suggests directions for future research.

2. Methodology

In this section, we present the iLED framework and focus on its connections with KO theory and Mori–Zwanzig formalism. We consider high-dimensional, potentially nonlinear system whose state $\Phi \in \mathbb{R}^{d_\Phi}$ evolves in time according to an operator $\boldsymbol{F}$:

$$\frac{\mathrm{d}\Phi}{\mathrm{d}t} = \boldsymbol{F}(\Phi). \tag{2.1}$$

This system can result from first principles and may represent the numerical discretization of a PDE such as the Navier–Stokes equations. In the following, we provide the motivation for the choice of neural networks to carry out dimensionality reduction (§2a), the theoretical justifications for our proposed interpretable reduced dynamics framework (§2b) and summarize the iLED architecture in §2d.

(a) Dimensionality reduction

We consider systems where the dimension d_Φ of the full-order system in equation (2.1) can be exceedingly high compared with the actual intrinsic system dimension. We assume that there exists a mapping $\mathcal{D} : \mathbb{R}^{d_\Phi} \mapsto \mathbb{R}^{d_z}$, with $d_z \ll d_\Phi$, such that $\Phi \approx \mathcal{D}(z)$. To identify and exploit this reduced dimensionality, the mapping $\mathcal{D}$ can be extracted from data using machine learning

methods. Classical reduced-order modelling methods leverage linear reduction approaches [18] to construct a basis that builds a matrix $\boldsymbol{V} \in \mathbb{R}^{d_\Phi \times d_z}$ on which both the system's state $\boldsymbol{\Phi}$ and the dynamics $\boldsymbol{F}$ can be projected:

$$\boldsymbol{V}^T \boldsymbol{\Phi} = z, \tag{2.2}$$

$$\frac{\mathrm{d}z}{\mathrm{d}t} = \boldsymbol{V}^T \boldsymbol{F}(\boldsymbol{V}z) + \epsilon, \tag{2.3}$$

where ϵ is an unknown error term. These linear reduction approaches have the important advantage of being interpretable as they are able to retain parts of the original model $\boldsymbol{F}$. Despite numerous successes with ROMs [49,50], linear reduction can be inefficient, in terms of dimensionality reduction when compared with nonlinear approaches such as neural networks. In many dynamical systems, their effective reduced-order dynamics evolve on strongly nonlinear manifolds [30]. The use of neural autoencoders to learn these reduced manifolds allows for a higher degree of reduction, as well as better reconstruction accuracy. Thus, we propose to learn two parameterized nonlinear mappings, a decoder $\mathcal{D}(\cdot;\theta_\mathcal{D})$ and an encoder $\mathcal{E}(\cdot;\theta_\mathcal{E})$, such that

$$\boldsymbol{\Phi} = \mathcal{D}(z;\theta_\mathcal{D}), \tag{2.4}$$

$$z = \mathcal{E}(\boldsymbol{\Phi};\theta_\mathcal{E}), \tag{2.5}$$

where $\theta_\mathcal{D}$ and $\theta_\mathcal{E}$ are the parameters of the decoder and encoder and are learned during training of the neural networks.

However, using a nonlinear encoder/decoder structure, the dynamics of the reduced-order system have to be learned afterward or concurrently, as nonlinear dimensionality reduction does not allow for the direct reduction of the original model $\boldsymbol{F}$. Existing works [51,52], and more recently, the *LED* framework [30] have demonstrated that these reduced dynamics could be directly learned using RNNs:

$$z_{t+1} = \mathrm{RNN}(z_t, \boldsymbol{h}_t; \theta_{RNN}), \tag{2.6}$$

where $\boldsymbol{h}$ is a memory term and θ_{RNN} are the parameters of the RNN. At the same time, these models have limited interpretability and, unlike KO or DMD, they cannot be justified by dynamical systems theory.

(b) Framing iLED within the Mori–Zwanzig formalism

The iLED framework is based on both Mori–Zwanzig formalism [32,33] as well as KO theory [34,35]. We first define the KO that acts on observable functions $\mathbf{g}$ of the state of high-dimensional systems $\boldsymbol{\Phi}$, before introducing the generalized Langevin equation (GLE) for a reduced subset of these observables. We subsequently define an appropriate closure term for the GLE and introduce a neural network architecture.

(i) KO theory and the generalized Langevin equation

This section introduces the connection between KO theory and Mori–Zwazig formalism, which is the basis for our iLED architecture. This connection has been established in previous work [53,54]. KO theory can quantify the dynamics of an observable of a high-dimensional system and has been employed extensively within ROMs [35,55,56]. For the high-dimensional system $\boldsymbol{\Phi}(t, \boldsymbol{\Phi}_0)$, a KO model can be used to represent the dynamics of the system instead of equation (2.1). More specifically, an observable $\mathrm{g} : \mathbb{R}^{d_\Phi} \mapsto \mathbb{R}$ of the system $\boldsymbol{\Phi}$ is advanced in time by the KO model, denoted below as $\mathcal{K}_t$:

$$\mathcal{K}_t \mathrm{g}(\boldsymbol{\Phi}_\mathbf{0}) = \mathrm{g}(\boldsymbol{\Phi}(t, \boldsymbol{\Phi}_\mathbf{0})). \tag{2.7}$$

Operator $\mathcal{K}_t$ is linear, and generally infinite dimensional. For practical purposes, its operating space can be separated in an observed subspace ($\mathcal{H}_\mathbf{g}$) defined as the space spanned by a chosen set

of M observables $\mathcal{M} = \{g_i\}_{i=1,\dots,M}$ and an orthogonal subspace $\mathcal{H}_g$ for which a set of basis functions $\overline{\mathcal{M}} = \{\bar{g}_i\}_{i=M+1,\dots,\infty}$ can be constructed so that $\langle g_i, \bar{g}_j \rangle = 0$ for all $i \in [1, M], j > M$. The dynamics of observables can then be expressed on the basis defined by the set $\mathcal{M} \cup \overline{\mathcal{M}}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} \mathbf{g}_{\mathcal{M}} \\ \mathbf{g}_{\overline{\mathcal{M}}} \end{bmatrix} = \mathbf{L}\begin{bmatrix} \mathbf{g}_{\mathcal{M}} \\ \mathbf{g}_{\overline{\mathcal{M}}} \end{bmatrix} = \begin{bmatrix} \mathbf{L}_{\mathcal{M}\mathcal{M}} & \mathbf{L}_{\mathcal{M}\overline{\mathcal{M}}} \\ \mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}} & \mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}} \end{bmatrix}\begin{bmatrix} \mathbf{g}_{\mathcal{M}} \\ \mathbf{g}_{\overline{\mathcal{M}}} \end{bmatrix}, \tag{2.8}$$

where $\mathbf{L}$ is a linear operator that corresponds to the infinitesimal generator of $\mathcal{K}_t$, $\mathbf{g}_{\mathcal{M}} = [g_1, g_2, \dots, g_M]$ are the chosen observables and $\mathbf{g}_{\overline{\mathcal{M}}} = [\bar{g}_{M+1}, \bar{g}_{M+2}, \dots, \bar{g}_\infty]$ are the orthogonal observables. Note that the operator $\mathbf{L}$ is separated in four parts, with $\mathbf{L}_{\mathcal{M}\mathcal{M}}$ the dynamics in the observed subspace, $\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}$ the orthogonal dynamics and $\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}$ and $\mathbf{L}_{\mathcal{M}\overline{\mathcal{M}}}$ the exchanges between the observed and orthogonal subspaces.

We note that by employing this decomposition, the orthogonal observables are still infinite dimensional, whereas the chosen observables are finite dimensional. While we would need to identify a Koopman-invariant subspace if we wanted to model the dynamics of these observables only using KO theory, we derive an equation for their dynamics based on Mori–Zwanzig formalism [53,54]. In case our chosen observables should be within a Koopman-invariant subspace, the additional terms with regard to the orthogonal observables/subspace vanish [54].

In more detail, the system in equation (2.8) can be solved for $\mathbf{g}_{\overline{\mathcal{M}}}$ as follows:

$$\mathbf{g}_{\overline{\mathcal{M}}}(t) = \int_0^t e^{(t-s)\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}\mathbf{g}_{\mathcal{M}}(s)ds + e^{t\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{g}_{\overline{\mathcal{M}}}(0), \quad t > 0. \tag{2.9}$$

Finally, using equation (2.9) in equation (2.8), an expression for the dynamics of the observables $\mathbf{g}_{\mathcal{M}}$ is obtained:

$$\frac{\mathrm{d}\mathbf{g}_{\mathcal{M}}}{\mathrm{d}t} = \mathbf{L}_{\mathcal{M}\mathcal{M}}\mathbf{g}_{\mathcal{M}} + \mathbf{L}_{\mathcal{M}\overline{\mathcal{M}}}\int_0^t e^{(t-s)\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}\mathbf{g}_{\mathcal{M}}(s)ds + \mathbf{L}_{\mathcal{M}\overline{\mathcal{M}}}e^{t\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{g}_{\overline{\mathcal{M}}}(0). \tag{2.10}$$

The above expression describes the dynamics of the partially observed state of a system and has the same form as the generalized Langevin equation derived in Mori–Zwanzig formalism. It still depends on the unobserved part of the initial condition ($\mathbf{g}_{\overline{\mathcal{M}}}(0)$) via the last term and thus is not a closed equation for $\mathbf{g}_{\mathcal{M}}$ only. However, this last term is often modelled as noise or simply ignored in several modelling approaches [53,57–59]. In the following section, the conditions under which this term can be accounted for are explained in more detail.

(ii) Closing the GLE

The last term in equation (2.10) depends on information that is unavailable as it is orthogonal to the observed subspace. However, this term vanishes if the history of the observed subspace is known, and the orthogonal dynamics ($\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}$) are dissipative. Indeed, if the history of the system is known, we can rewrite equation (2.9) for any initial condition $[\mathbf{g}_{\mathcal{M}}(-\tau), \mathbf{g}_{\overline{\mathcal{M}}}(-\tau)], \tau > 0$:

$$\mathbf{g}_{\overline{\mathcal{M}}}(t) = \int_{-\tau}^t e^{(t-s)\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}\mathbf{g}_{\mathcal{M}}(s)ds + e^{(t+\tau)\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{g}_{\overline{\mathcal{M}}}(-\tau). \tag{2.11}$$

The last term in equation (2.11) vanishes for $\tau \to \infty$, if the orthogonal dynamics $\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}$ is dissipative. This is often a reasonable assumption as, for instance, in the case of orthogonal (unobserved) subspace corresponds to the small scales of a dynamical system. Under such hypothesis, we obtain the following closed equation for the dynamics of the observed subspace:

$$\mathbf{g}_{\overline{\mathcal{M}}}(t) = \int_{-\infty}^t e^{(t-s)\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}\mathbf{g}_{\mathcal{M}}(s)ds, \tag{2.12}$$

$$\implies \frac{\mathrm{d}\mathbf{g}_{\mathcal{M}}}{\mathrm{d}t} = \mathbf{L}_{\mathcal{M}\mathcal{M}}\mathbf{g}_{\mathcal{M}} + \mathbf{L}_{\mathcal{M}\overline{\mathcal{M}}}\int_{-\infty}^t e^{(t-s)\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}}\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}\mathbf{g}_{\mathcal{M}}(s)ds. \tag{2.13}$$

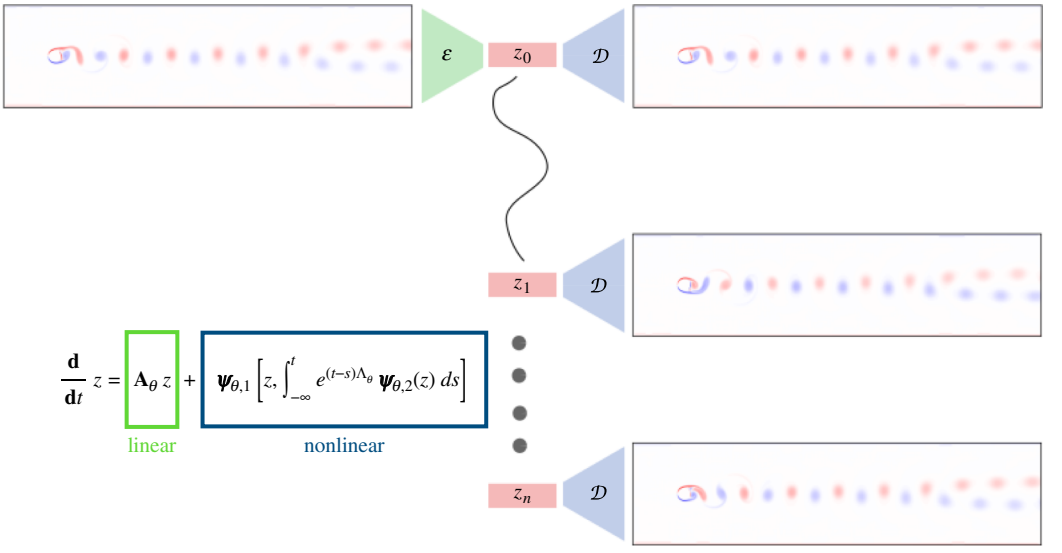

Figure 1. iLED architecture: the high-dimensional system is encoded to a lower-dimensional representation using the encoder $\mathcal{E}$. The lower-dimensional representation is propagated in time using a linear and a nonlinear part based on Mori–Zwanzig formalism. With the help of a decoder $\mathcal{D}$, the high-dimensional system is subsequently reconstructed.

In the following subsection, we show that the various operators expressed in the closed GLE (equation (2.13)) can be learned from data to derive an interpretable and theoretically sound model for the reduced dynamics of physical systems. We will model the observed subspace $\mathbf{g}$ only, whereas the orthogonal unobserved subspace $\mathbf{g}_{\overline{\mathcal{M}}}$ will not be explicitly represented.

(c) The iLED architecture

To construct the iLED architecture, we first identify the observables $\mathbf{g}_{\mathcal{M}}$ with the learned subspace of the neural encoder $\mathcal{E}$ so that $\mathbf{g}_{\mathcal{M}} \equiv z = \mathcal{E}(\Phi; \theta_{\mathcal{E}})$. We then approximate the various operators $(L_{\mathcal{M}\mathcal{M}}, L_{\overline{\mathcal{M}}\mathcal{M}}, L_{\mathcal{M}\overline{\mathcal{M}}}, L_{\overline{\mathcal{M}}\overline{\mathcal{M}}})$ that express the different parts of the Mori–Zwanzig formalism in equation (2.13). The nonlinear part of the dynamics in equation (2.13) accounts for the exchanges between the unobserved and observed subspaces when the dynamics is not closed on the observed subspace [54].

The observed dynamics $\mathbf{L}_{\mathcal{M}\mathcal{M}}$ can be directly learned as a linear operator, denoted $A_\theta \in \mathbb{R}^{d_z \times d_z}$ below. However, because both operators $\mathbf{L}_{\overline{\mathcal{M}}\mathcal{M}}$ and $\mathbf{L}_{\mathcal{M}\overline{\mathcal{M}}}$ are possibly infinite dimensional, they need to be approximated. We propose to learn these operators as nonlinear transformations of the observables z. Justifications for this choice will be detailed in §i.

We introduce two neural networks $\Psi_1(\cdot;\theta) : \mathbb{R}^{d_h+d_z} \mapsto \mathbb{R}^{d_z}$ and $\Psi_2(\cdot;\theta) : \mathbb{R}^{d_z} \mapsto \mathbb{R}^{d_h}$, where d_h is a user-defined parameter, and model the orthogonal dynamics $\mathbf{L}_{\overline{\mathcal{M}}\overline{\mathcal{M}}}$ as a negative diagonal operator $\Lambda_\theta \in \mathbb{R}_-^{d_h \times d_h}$. This choice is consistent with the assumption that the orthogonal dynamics is dissipative and significantly simplifies certain computations such as the initialization of the non-Markovian (or memory) term in the model. This leads to the iLED architecture in figure 1.

The key part of this novel architecture is the temporal dynamics of the iLED state z:

$$\frac{\mathrm{d}}{\mathrm{d}t} z = \mathbf{A}_\theta z + \Psi_{\theta,1}\left[z, \int_{-\infty}^{t} e^{(t-s)\Lambda_\theta} \Psi_{\theta,2}(z) ds\right] \tag{2.14}$$

We note that the operator Ψ_1 now also takes as argument the reduced state z itself. This choice ensures that the nonlinear part of the iLED model has access to the state at the current time. In terms of the framework derived above this corresponds to the network Ψ_2 learning an identity of the state in part of its output so that $\Psi_2(z) = [\mathbf{z}, ...]$ and the corresponding entries of the diagonal matrix Λ going to $-\infty$. Indeed, the integral in equation (2.14) is a low-pass filtering of the trajectories of each dimension of $\Psi_2(z)$ with a cutoff frequency which grows with $-\lambda_i$, the i_{th} entry in the diagonal of Λ. Thus, λ_i going to negative infinity implies that no frequency is filtered in the trajectory, which is equivalent to directly considering the state at the current time.

This proposed iLED architecture allows us to directly learn the various terms of the Mori–Zwanzig formalism from data. The details on the training strategy are given in §2d.

The idea of learning latent dynamics based on KO and Mori–Zwanzig formalism has been recently employed in another scientific publication [60], which has been posted after the release of the preprint of our work. Their approach also relies on a nonlinear autoencoder architecture to identify the latent dynamics, but the nonlinear terms in the Mori–Zwanzig formulation are approximated with a different neural network architecture.

(i) Remarks on the approximation of an infinite linear operator

Deep neural networks are universal approximators for nonlinear operators when both the given input and output of the operator are compact [61]. The universal approximation theorem states that there is a sequence of neural networks that can approximate the targeted operator but does not provide further details regarding the size of the networks involved as well as an error rate. However, even if the number of neurons and layers is finite, approaches such as the deep operator network [62,63] and Fourier neural operators [64] have shown good approximations. Here, the subspace is infinite and not necessarily compact. The KO, our starting point in equation (2.7), is generally represented by a finite-dimensional operator with reasonable accuracy. In fact, this is a key assumption for all main data-driven Koopman models [55,65,66]. We assume a similar but less strict assumption holds here, and thus are dealing with a finite-dimensional orthogonal space whose operators can be approximated by neural networks.

(d) Training the iLED architecture

A key difficulty in the present methods is the choice of the latent dimension d_z. The best choice is a dimension close to the intrinsic dimension of the problem at hand. In the common case where it is unknown, several approaches can be used to select this parameter [67–70]. One is to directly apply hyperparameter optimization approaches such as grid search to the problem, i.e. train autoencoders with increasing latent dimensions and select the dimension when the reconstruction performance of the autoencoder starts to plateau. However, this can be expensive when applied to problems that use high-dimensional representations such as fluid flows. To avoid these expensive computations, statistical analysis such as estimation of the correlation dimension [70] or of the fractal dimension [69] can be employed to approximate the dimension of the attractor. Both of these approaches were used to obtain a first guess of the latent dimension in the experiments presented in this paper.

Having chosen this latent dimension d_z, we can set up the model for the latent dynamics. We first rearrange the integro-differential equation (2.14) into a coupled system of ODEs. First, we define an intermediate term $\boldsymbol{h}$:

$$\boldsymbol{h}(t) = \int_{-\infty}^{t} e^{(t-s)\Lambda_\theta} \Psi_{\theta,2}(\mathbf{z}) ds. \tag{2.15}$$

This $\boldsymbol{h}$ term corresponds to the *memory* of the model, which can be advanced in time in parallel of the reduced-order state as follows:

$$\begin{aligned}\frac{\mathrm{d}z}{\mathrm{d}t} &= \mathbf{A}_\theta z + \Psi_{\theta,1}(z, \boldsymbol{h}),\\ \frac{\mathrm{d}\boldsymbol{h}}{\mathrm{d}t} &= \Psi_{\theta,2}(z) + \Lambda_\theta \boldsymbol{h}.\end{aligned} \tag{2.16}$$

With this time-continuous architecture, iLED can be used in combination with any standard ODE integrator. In this work, we used the semi-implicit Runge–Kutta (siRK) scheme [71] to advance the iLED state $[z, \boldsymbol{h}]$ in time. This scheme takes advantage of dynamics that efficiently separate a linear and a nonlinear part and is very efficient for the simulation of stiff dynamics,. The latter feature is important for iLED as the dynamics can be stiff and unstable before being fully trained. The corresponding equations can be found in equation (A 1) in appendix A.

The siRK integration scheme is used in combination with the adjoint scheme for back-propagating through an ODE [36] to train the iLED architecture. We train the model in an *end-to-end* fashion, that is to say, both the neural autoencoder $\{\mathcal{E}, \mathcal{D}\}$ and the dynamics are optimized simultaneously, using the combined loss:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \alpha \mathcal{L}_{\text{forecast}}, \tag{2.17}$$

where $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{forecast}}$ are, respectively, the reconstruction and forecast losses, and α controls their relative importance.

The reconstruction loss $\mathcal{L}_{\text{rec}}$ drives the autoencoder to accurately reconstruct the true full-order trajectory $\Phi^\star_{t_i}$:

$$\mathcal{L}_{\text{rec}} = \frac{1}{N_t} \sum_{i=1}^{N_t} \|\Phi^\star_{t_i} - \mathcal{D}(\mathcal{E}(\Phi^\star_{t_i}))\|_2^2. \tag{2.18}$$

We note that different loss functions for the reconstruction loss that reward the selection of observables or add additional constraints to the latent representation could be employed. We have opted for the well-known autoencoder reconstruction loss to also minimize the number of hyperparameters involved in the loss function. The forecast loss $\mathcal{L}_{\text{forecast}}$ pushes the model to accurately predict the reduced state z:

$$\mathcal{L}_{\text{forecast}} = \frac{1}{N_t} \sum_{i=1}^{N_t} \|\hat{z}_{t_i} - \mathcal{E}(\Phi^\star_{t_i})\|_2^2, \tag{2.19}$$

where $\hat{z}$ is calculated according to equation (2.16).

This aggregated loss is sufficient to train the iLED architecture. However, additional terms can be added to improve performance. Appendix B details the various additions that were used to obtain the results presented in this work. This includes a regularization term employed to penalize the magnitude of the nonlinear part of the iLED dynamics, effectively ensuring that the linear part upholds as much of the dynamics as possible. We note that this penalty-based approach is however not able to guarantee that all linear parts of the dynamics are compeletly represented by $\mathbf{A}_\theta$, but similar approaches have succesfully been used in other cases. For instance, the authors of [72] use it to enforce residuals when learning the solution to PDEs. Appendix C provides more details regarding other design choices such as the initialization for the memory term $\boldsymbol{h}$. The next section presents the numerical experiments carried out to demonstrate the abilities of the iLED method.

3. Numerical experiments

The capabilities of iLED are demonstrated on three benchmark problems: the FitzHugh–Nagomo (FHN) model, a simple one-dimensional equation with periodic dynamics; the chaotic dynamics

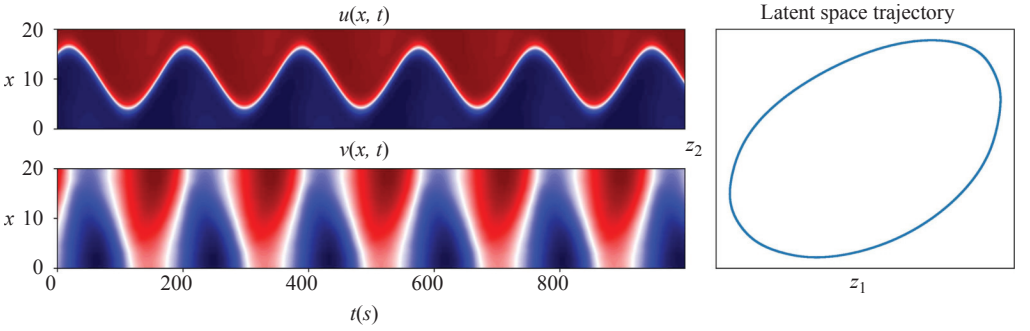

Figure 2. Visualization of the FHN model's dynamics. The evolution of the full state for a subset of the test trajectory is presented on the left. The right-hand side of the plot displays the latent manifold learned by an autoencoder using latent dimension $d_z = 2$.

presented by the KS equation; the incompressible Navier–Stokes equations describing flow around a cylinder in two different Reynolds numbers (100 and 750).[1]

(a) Example 1: the FHN model

The FHN model [73] has been extensively used in biology, physics and neuroscience for the study of the dynamics of excitable systems. The model consists of two coupled PDEs that describe the dynamics of a fast-acting variable $u(x,t) \in \mathbb{R}, x \in \Omega = [0, L], t \in [0, T]$, inhibited by a slower variable $v(x,t) \in \mathbb{R}$:

$$\frac{\partial u}{\partial t} = D_u \frac{\partial^2 u}{\partial x^2} + u - u^3 - v, \tag{3.1}$$

$$\frac{\partial v}{\partial t} = D_v \frac{\partial^2 v}{\partial x^2} + \epsilon\,(u - \alpha_1 v - \alpha_0). \tag{3.2}$$

This separation of time scales is controlled by parameter ϵ, set here to $\epsilon = 0.006$. The other model parameters are chosen as follows: $D_u = 1$, $D_v = 4$, $L = 20$, $\alpha_0 = -0.03$ and $\alpha_1 = 2$, to replicate the experiment presented in [30]. The computational domain Ω is discretized using a grid of $N = 101$ points. The problem is solved starting from five different initial conditions using the Latice-Boltzmann method [74]; its implementation is provided in [30]. The data is sampled at rate $\Delta t = 1\,s$ to obtain five trajectories of 451 s each. Two of those trajectories are set aside for validation and the others are used for training. An additional trajectory of 10^4 s is simulated for testing purposes.

By training various autoencoders to reconstruct the training trajectories described above, we determined that the optimal latent dimension was $d_z = 2$, as the reconstruction accuracy evaluated from the validation trajectories saturates for higher dimensions. This result is consistent with the oscillatory nature of the dynamics and highlights the efficiency of nonlinear dimensionality reduction. Indeed, a linear method such as principal component analysis (PCA) requires up to 16 latent dimension (see [30] figure 2A) to achieve the same level of accuracy. A smaller latent dimension is useful as it leads to faster computation in the latent space. A visualization of the system evolution, as well as the corresponding latent trajectory, is presented in figure 2.

An iLED dynamical model is also trained at the same time as the autoencoder, using the procedure described in §2d (the hyperparameters used are detailed in appendix Da). Figure 3 presents the results obtained by simulating the final model on the test trajectory. The figure shows that the iLED model is able to accurately reconstruct the full-order system state from the latent

[1]The code for the aforementioned numerical illustrations is available at https://github.com/cselab/iLED.

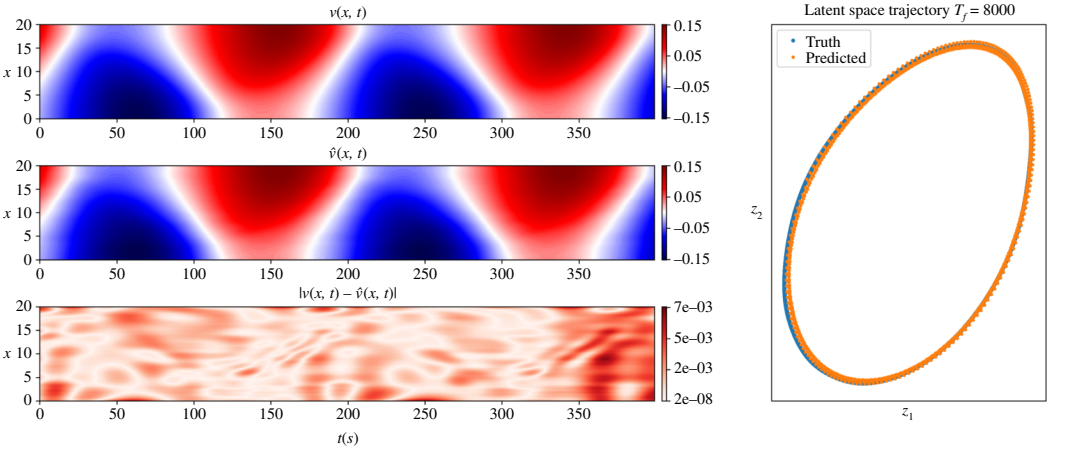

Figure 3. Forecasting performance of the iLED method on the FHN case. From top to bottom: true inhibitor field, predicted inhibitor field and absolute error between the two. The right-hand side presents the true and predicted latent trajectory for an integration period of 8000 s. *NB*: only the inhibitor *v* field is presented for clarity, as it is more difficult to predict than the activator field *u*.

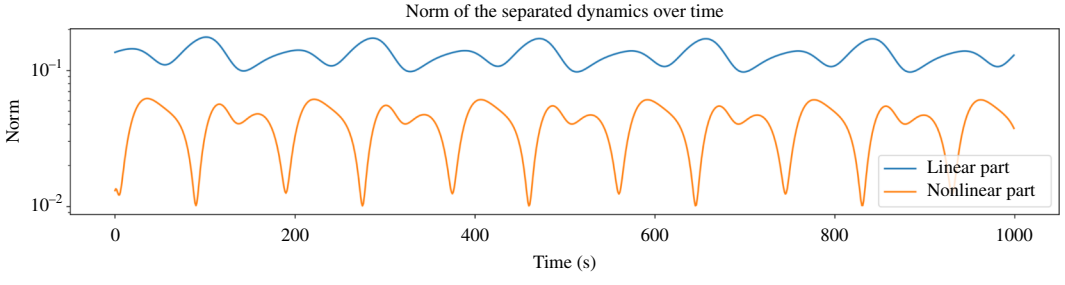

Figure 4. Norm of the different parts of the dynamics during the simulation of the iLED model of the FHN case.

state z. Moreover, the dynamics is accurately captured: the model remains on the true latent attractor even after a very long integration.

The iLED is particularly well suited for this case and highly interpretable. Owing to the optimal latent dimension $d_z = 2$, the linear part of the iLED dynamics exhibits a single natural frequency, aligning with the periodic nature of the dynamics. The learned frequency is approximately 5.74 mHz, while the primary frequency extracted from the true system data using a Fourier transform is 5.37 mHz. This comparison demonstrates that the operator has accurately captured the driving frequency of the system, allowing the linear part of the iLED model to support most of the dynamics. A close examination of the norm of the dynamics separately for the linear and nonlinear terms (figure 4) confirms this result: The figure clearly shows that the dynamics is mainly supported by the linear term, the contribution of the nonlinear term being approximately one order of magnitude smaller. It is important to note that the nonlinear term still plays a role in this case, as the learned latent attractor is not perfectly circular. We have investigated this in more detail in appendix Dd. The predictive quality of iLED is influenced by the quality of the training data and thus more data samples should be able to further increase the accuracy of the method. Moreover, adding additional nonlinear latent variables and thus increasing the dimension of $\boldsymbol{h}$ will be in our opinion positive for the predictive accuracy. We note that this, however, will increase the runtime for both training and predictions.

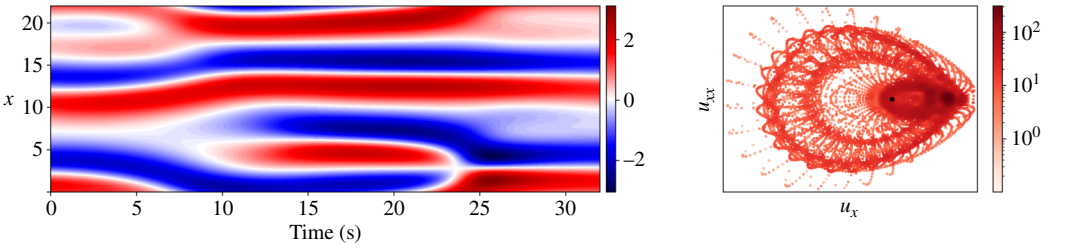

Figure 5. Two views of a training trajectory for the KS case (see text).

(b) Example 2: the KS equation

The KS equation [75] serves as a model for a broad range of physical systems. It is a prototypical example of a nonlinear PDE and exhibits a rich variety of behaviours, including the emergence of self-sustained oscillations, the formation of coherent structures and the occurrence of spatiotemporal chaos, making it an excellent test-bed for ROMs. The KS equation has been studied in various other works related to model-order reduction [15,30,67,68,76,77].

The KS equation can be expressed as

$$\begin{aligned} &\frac{\partial u}{\partial t} + \frac{\partial^2 u}{\partial x^2} + \frac{\partial^4 u}{\partial x^4} + u\frac{\partial u}{\partial x} = 0, \\ &u(x,t) \in \mathbb{R},\ x \in [0, L],\ t \in [0, T], \\ &u(0,t) = u(L,t), \end{aligned} \tag{3.3}$$

where $u(x,t)$ represents the unknown scalar field, and L is the length of the computational domain that controls the nature of the dynamics. We use here $L = 22$, a common value for the study of this problem [30,78], which yields a dynamical system that evolves on a stable attractor with a characteristic dimension approximately equal to eight (a higher dimensional attractor than the attractor of the FHN model studied in previous section). Moreover, the KS system develops chaotic dynamics under these conditions, which significantly increases the complexity of the learning problem, as small errors naturally compound over time during the simulation.

The equation is discretized on a spectral basis of $N = 64$ Fourier modes and advanced in time using a siRK scheme [71]. We generate 2048 training trajectories starting from random initial conditions and 64 others for validation. The initial conditions are all advanced in time for 3000 'warm-up' steps of length $\delta t = 0.025$ s, which are discarded as they account for the transition from the random initial conditions to the chaotic attractor. The next 1280 steps are then sub-sampled with a $\Delta t = 0.25$ s to obtain the training and validation data. Finally, one hundred new initial conditions are simulated with a longer time horizon (800 s) for testing purposes. The evolution of one of the training trajectories is presented in figure 5, as well as a visualization of the joint probability density $p\left(\frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right)$, which is a helpful way of visualizing the dynamics of the KS equation.

Applying the iLED method,[2] we find that the reconstruction performance of the autoencoder used for dimensionality reduction does not improve for latent dimensions greater than $d_z = 8$. This is in accordance with the previously discovered [16] intrinsic dimension of the KS attractor. Figure 6 presents results obtained on a test trajectory with a trained iLED model.

The results in figure 6 demonstrate that the iLED method is able to correctly capture the dynamics of the system on a previously unseen trajectory and for a time horizon at least as long as its training. As expected, the forecasting error does increase for longer integration times. The chaotic nature of the problem makes it increasingly hard for a model to accurately follow the

[2]Details on the architecture and hyperparameters used can be found in appendix Db.

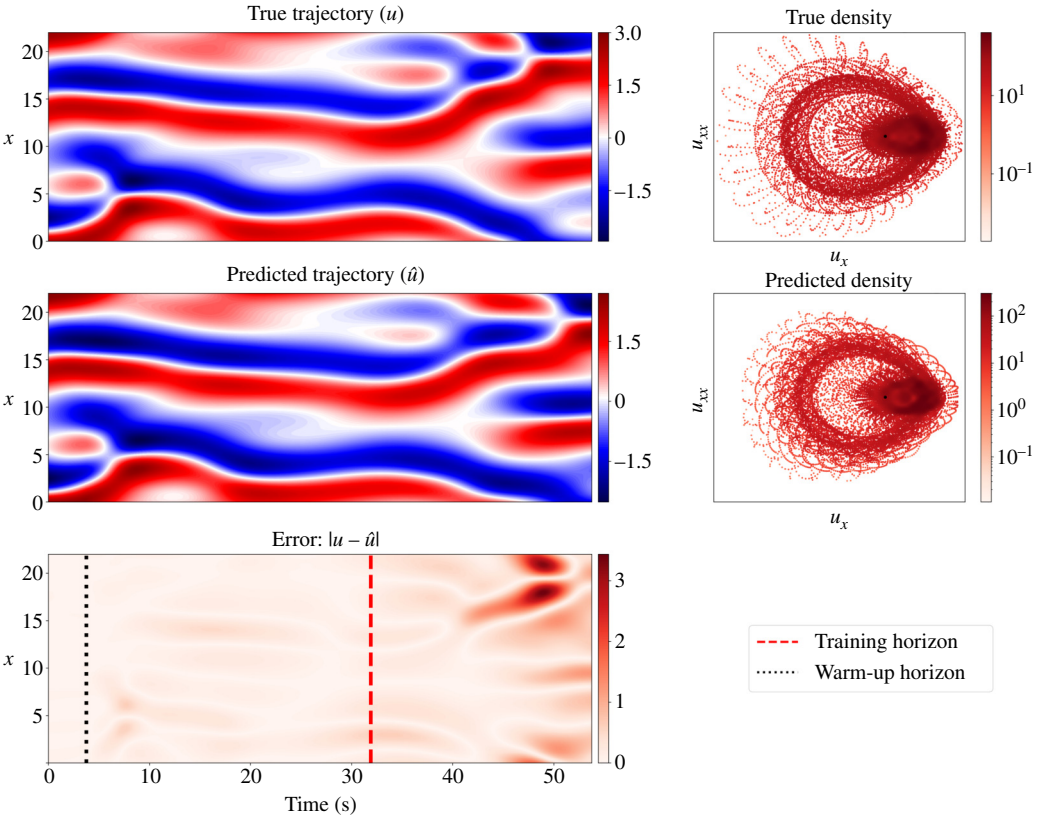

Figure 6. Results obtained with the iLED method on a test trajectory. *Dashed black line*: Horizon of the warm-up required to initialize the memory of the model *Dashed red lined*: time horizon used to train the model.

true system trajectory. Moreover, we note that despite leaving the true trajectory, the obtained attractor, visualized through the densities of the derivatives, is well captured.

We also examine the eigenvalues of the learned linear operator in iLED for the FHN case (§3a). Figure 7 shows the natural frequencies learned by the iLED models after training under ten different random seeds. As the KS system is not driven by a single main frequency, it is interesting to note that the different model initializations led to learning a similar range of frequencies. Moreover, the natural frequencies of the iLED linear operator are coherent with the frequencies observed in the data. Figure 7 displays the Fourier transform of a test trajectory, showing that a large range of frequencies is present in the data. The figure also shows that this range is covered by the various frequencies learned by the iLED linear operator, suggesting that while the chaotic attractor does not directly correspond to a periodic cycle in latent space, this cycle is still relevant to the system dynamics.

(c) Example 3: flow around a two-dimensional circular cylinder

Finally, we apply the iLED method to simulations of the uniform viscous (with viscosity ν) flow with speed U past a two-dimensional circular cylinder, with diameter D, using the Navier–Stokes equations. The complexity of this case is controlled by the Reynolds number ($Re = UD/\nu$). We consider flows with two different Reynolds numbers: $Re = 100$, a less challenging standard value often used to benchmark ROMs [79] and $Re = 750$, where the system then exhibits much more complex dynamics and is more challenging to model.

In both cases, the incompressible Navier–Stokes equations are solved using an adaptive meshing and time stepping Basilisk solver [80]. The generated data are then interpolated on

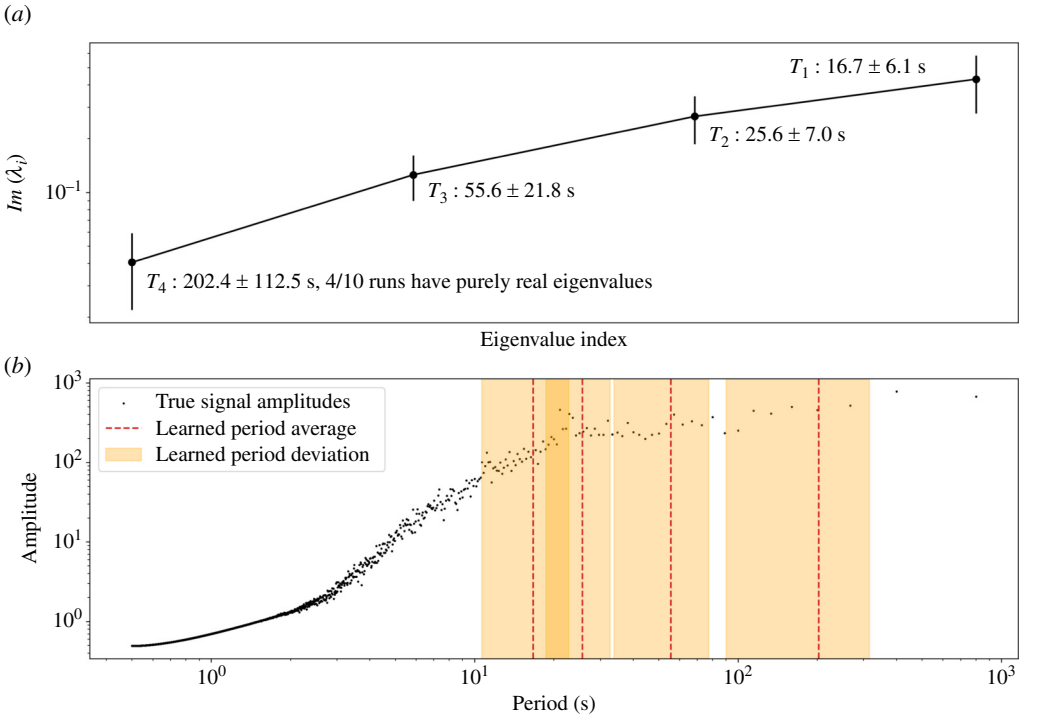

Figure 7. (*a*) Eigenvalues λ_i of the iLED linear operator (average and std. dev. over ten different training runs). (*b*) Fourier transform of a test trajectory averaged over the computational domain, the natural periods of the iLED operator are also displayed for comparison. Note that several runs learned one purely real eigenvalue, meaning that the learned period is infinite, thus not included in the computation of the largest period (T_4).

a Cartesian grid to ensure compatibility with convolutional neural networks. To construct the autoencoder, we use an approach similar to the one proposed in [81], using a higher resolution around the cylinder to capture the process of vorticity generation. Hence, we deploy separate convolutional encoders with different resolutions for the region around the cylinder and the remaining of the computational domain. These two encoders produce two intermediate latent representations z_1 and z_2, which are passed through an additional mixer multi-layer perceptron to compute the latent state z. This mixer network is used to ensure that each dimension of the latent state z can encode information for both the higher and lower resolution parts of the state, which is important as the iLED linear operator $\mathbf{A}_\theta$ acts on the full latent state z. This multi-scale architecture is illustrated in figure 8. Additional details on the architecture and hyperparameters used can be found in appendix Dc.

For both Reynolds numbers, the problem is simulated for 100 s. The first warm-up twenty seconds are discarded as they correspond to the transition from the initial condition. The rest of the trajectory is sub-sampled with a $\Delta_t = 0.02$ s yielding a trajectory of 4000 points. The first 2500 points are used for training, and the last 1500 are set aside for validation.

The results obtained by training an iLED model for the $Re = 100$ case are presented in figure 9. We used a latent dimension of $d_z = 3$, which is higher than the minimal dimension 2 required to represent the limit cycle of the system, but yielded better modelling performance according to the combined loss (equation (2.17)). This has already been discovered by Noack *et al.* [19] who showed that the third dimension essentially supports the approach to the limit cycle, so using $d_Z = 3$ stabilizes the overall system. Moreover, Noack *et al.* [19] have shown that based on this minimal dimension, nonlinear equations of motions can be derived that are able to represent the evolution of the full system. However, here we are focusing on learning such equations automatically and are aiming to obtain close-to-linear latent space dynamics. Figure 9 shows that the iLED model is

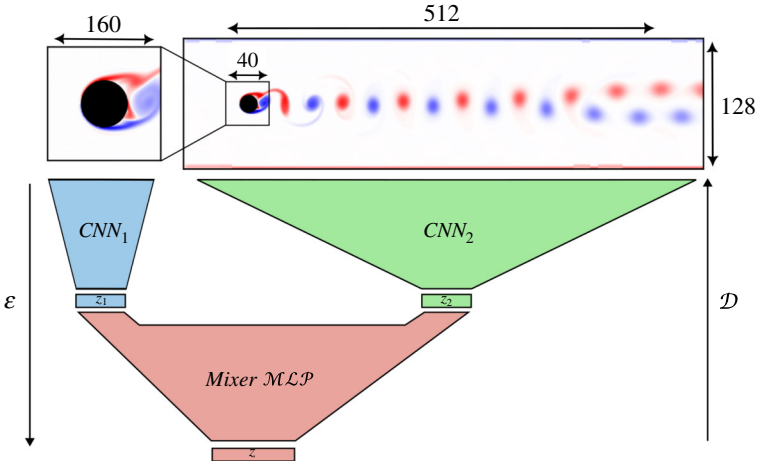

Figure 8. Multiscale architecture used to model the cylinder flow. The area around the cylinder is rendered at four times the resolution of the rest of the field, as it is where the dynamics are most complex. CNN stands for convolutional neural network and MLP stands for multi-layer perceptron.

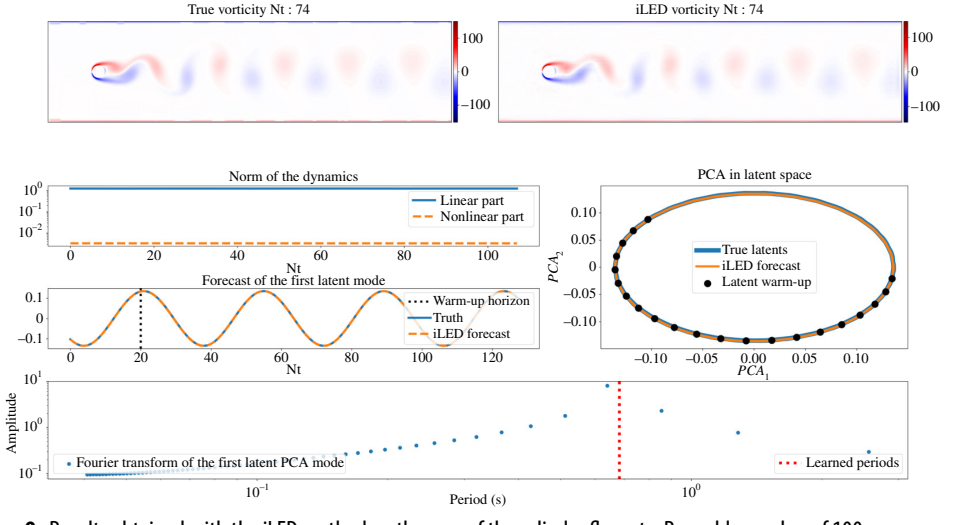

Figure 9. Results obtained with the iLED method on the case of the cylinder flow at a Reynolds number of 100.

able to accurately reconstruct the system state after multiple shedding cycles. Similarly to the FHN case (§3a), the results underline the effectiveness of the iLED architecture, as the figure shows that most of the dynamics are supported by the linear part of the model.

Finally, the natural frequency of 1.466 Hz learned by the iLED linear operator is in accordance with the system data which presents a dominant frequency of 1.562 Hz; this further confirms the validity of the model.

The results obtained on the case of the cylinder flow under a Reynolds number of 750 are presented in figure 10. This case presents more complex dynamics than the simple two-dimensional periodic limit cycle encountered for $Re = 100$. Following [81], we used a latent dimension of $d_z = 16$. Similar to the $Re = 100$ case, the iLED model is able to accurately forecast and reconstruct the system state and once again, despite the higher complexity of the case, most of the dynamics are supported by the linear operator and the neural network closure (Ψ_1 in equation (2.16)) is only used to correct the numerical imperfections in the curvature of the learned latent attractor. Figure 10 also shows that the learned frequencies are coherent with the system

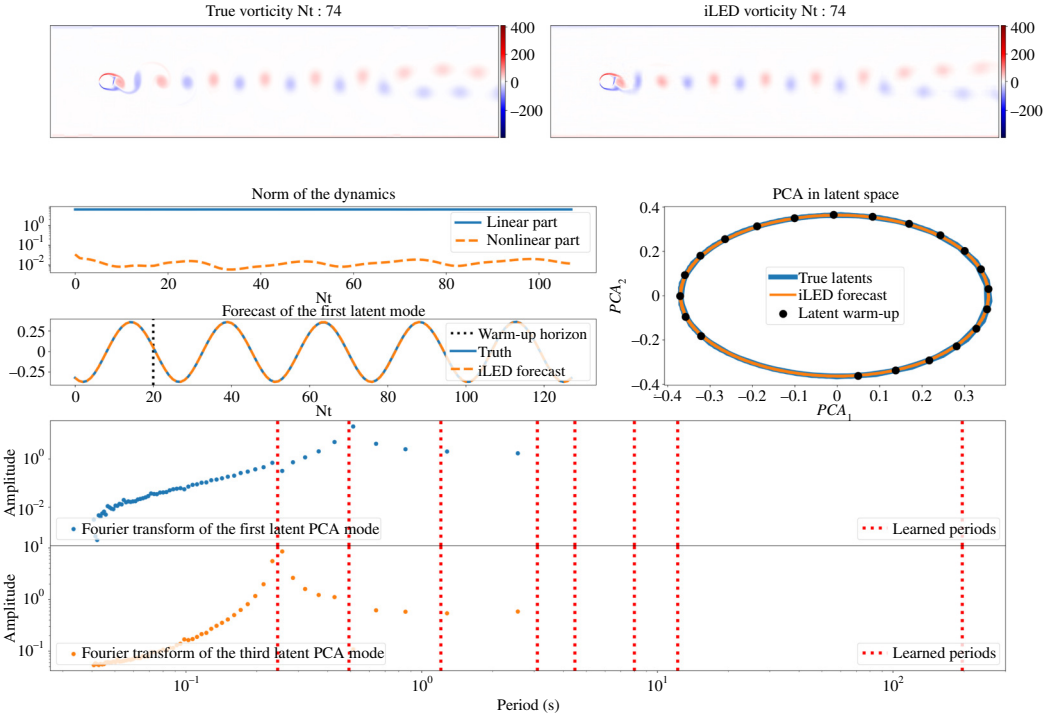

Figure 10. Results obtained with the iLED method on the case of the cylinder flow at a Reynolds number of 750.

data, as the two first natural frequencies of the linear operator are perfectly coherent with the dominant frequencies of the Fourier transform of the true latent trajectories.

These results demonstrate the ability of the iLED model to capture the dynamics of two-dimensional bluff body flows up to Re = 750. The model yields satisfying performance combined with a high degree of interpretability and stability.

(d) Remarks on the linearity of the dynamics

In two of the three numerical experiments presented above, we have shown that the iLED model was able to transform high-dimensional, nonlinear PDEs into *quasi*-linear ordinary differential equations. In case the observables learned by the encoder can be represented by a finite set of Koopman–eigenfunctions, the dynamics would be perfectly linear. This is in fact coherent with the oscillatory nature of the dynamics in both the FHN and cylinder flow cases. We underline this result as a major strength of the iLED framework as it is able to identify simple models from data without any *a priori* assumptions on the system under study.

Moreover, it is important to note that although the identified models rely almost entirely on the linear part of the dynamics, they are still completed by a nonlinear term of lesser magnitude. Indeed, the complexity involved in learning a purely linear model reaching the same degree of accuracy might be higher. This is due to the fact that the neural autoencoders used for dimensionality reduction struggle to learn perfectly *organized* latent attractors, which is critical to ensure the accuracy of purely linear dynamics. Of course, this aspect is only magnified with the increasing complexity of the application case. Thus, the nonlinear term in the framework can be looked at as a relaxation of the constraints on the shape of the latent attractor while still allowing for the extraction of a simple interpretable model, as the observed low magnitude of the nonlinear dynamics allows for accurate analysis of the model from the learned linear term.

Finally, we note that these *quasi*-linear dynamics were not observed in the case of the KS equations. This is due to the chaotic nature of the case. Because the system does not rely on a

set of clearly identified driving frequencies, the linear part of the model is not sufficient for the accurate representation of the case, and the nonlinear part then automatically learns to complete the dynamics. This once again underlines the *adaptability* of the model as no *a priori* knowledge of the nature of the dynamics is required to model the system.

(e) Remarks on computational costs

In addition to revealing salient features about the temporal evolution of a high-dimensional system, the iLED model is also able to significantly accelerate the computation of multiscale systems. The computation of the temporal dynamics using iLED is in general much easier as the latent dimension is smaller than the actual problem dimension and the evaluation of the iLED dynamics is computationally less expensive than computing an update of the high-dimensional system. We note that we incur additional computational costs owing to the training of the model.

For instance, simulating for 128 seconds and reconstructing the full-order state of the cylinder flow case was of less than a second on a single RTX2080 GPU which is significantly lower than the cost of simulating the same case in a full-order solver. On the other hand, the cost of training the iLED model on four GPUs is of the order of a day. These numbers show that our approach is relevant when repeated evaluation of a model is necessary. We note that it is complicated to give an exact measure of the computational gain, as the costs of simulating the full-order model and iLED both depend on numerous factors including software implementation, hardware availability and test case choice.

Finally, we note that there can be significant computational costs associated with the generation of the training data. While we have been able to extrapolate beyond the time regime for which training data was available, this extrapolative performance cannot be guaranteed and is often limited and thus the training data need to be representative of the evaluation conditions.

4. Conclusion

We present the iLED method to learn interpretable reduced-order dynamics for high-dimensional, multiscale systems. This method is closely related to KO theory and enhanced Mori–Zwanzig formalism. In addition to offering a high degree of interpretability, the latent dynamics of iLED is expressive enough such that the method can be applied to various problems.

We show that the approach performs well on a range of dynamics, from chaotic problems to high-dimensional two-dimensional flow cases. For each test case, the method is able to learn a linear model for the latent dynamics as well as a non-Markovian, nonlinear closure term. The high-dimensional systems are mapped with a nonlinear encoder to a latent space, in which the complex nonlinear PDEs can be reduced to very simple *quasi*-linear ODEs, thus yielding fast and stable simulations. The high-dimensional space can be reconstructed from the latent space using a decoder that is trained simultaneously with the aforementioned encoder, using an autoencoder architecture.

Currently, the latent dynamics is computed deterministically. Future work will address a probabilistic version based on either the Bayesian approach or conformal inference, to quantify the uncertainty caused by dimensionality and model reduction. Another unsolved challenge pertains to a computationally efficient choice of the latent dimension. New approaches such as [67,68,82] offer some ideas on how to address this problem. Other aspects of the approach could also be improved upon, for one, enforcing the nonlinear nature of the neural network in the dynamics would be desirable, as it is only encouraged by the loss in the current state of our proposal. A second interesting direction for future works is the analysis of the learned nonlinear encoder and decoder mapping and their comparison with the Koopman-eigenfunctions of the system.

The iLED method could be used to model more complex problems such as partially observed systems, or applied to real-world problems with unknown dynamics such as epidemic dynamics or brain activity to help derive interpretable dynamical laws from available data. An extension to

systems for which dynamics depend on the current time is also possible by adding the time as an input to our model.

Data accessibility. This paper has no additional data.

Declaration of AI use. We have not used AI-assisted technologies in creating this article.

Authors' contributions. E.M.: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing–original draft, writing—review & editing; S.K.: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing—original draft, writing—review & editing; M.Y.: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing—original draft, writing—review & editing; M.S.: conceptualization, formal analysis, investigation, methodology, software, validation, visualization, writing—review & editing; P.K.: conceptualization, formal analysis, investigation, project administration, supervision, validation, visualization, writing—original draft, writing—review & editing.

All authors gave final approval for publication and agreed to be held accountable for the work performed therein.

Conflict of interest. We declare we have no competing interests.

Funding. This material is based upon work by P.K. supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, Chemical Sciences, Geosciences and Biosciences Division under Award Number DE-SC0022199. E.M., M.Y. and M.S. acknowledge the support of the French government under the France 2030 program me as part of the SystemX Technological Research Institute. S.K. and P.K. acknowledge support by The European High Performance Computing Joint Undertaking (EuroHPC) Grant DCoMEX (956201-H2020-JTI-EuroHPC-2019-1).

Acknowledgements. We would like to thank Sergey Litvinov for his help with generating the data for the flow around a cylinder example. Moreover, we thank Hunter Heidenreich, Michele Alessandro Bucci and Lionel Mathelin for helpful discussions and feedback on the manuscript.

Appendix A. Runge–Kutta

This appendix contains the siRK3 scheme that advances the state $[z_0, h_0]$ for a time step Δt as follows:

$$\begin{aligned}\left(I - \frac{k}{6}\Delta t \mathbf{A}_\theta\right) z_{k\Delta t/3} &= z_0 + \frac{k}{6}\Delta t \mathbf{A}_\theta z_0 + \frac{k}{3}\Delta t \Psi_{\theta,1}(z_{(k-1)\Delta t/3}, \boldsymbol{h}_{(k-1)\Delta t/3}),\\ \left(I - \frac{k}{6}\Delta t \Lambda_\theta\right) \boldsymbol{h}_{k\Delta t/3} &= \boldsymbol{h}_0 + \frac{k}{6}\Delta t \Lambda_\theta \boldsymbol{h}_0 + \frac{k}{3}\Delta t \Psi_{\theta,2}(z_{(k-1)\Delta t/3}),\\ k &= 1, 2, 3.\end{aligned} \tag{A 1}$$

Here $\Lambda_\theta, \Psi_{\theta,1}, \Psi_{\theta,2}$ and $\mathbf{A}_\theta$ are based upon the iLED architecture introduced in equation (2.14).

Appendix B. Details on the loss

As mentioned in §2d, two loss terms are sufficient to train the iLED architecture:

$$\mathcal{L} = \frac{1}{N_t}\sum_{i=1}^{N_t}\left[\underbrace{\|\Phi^\star_{t_i} - \mathcal{D}(\mathcal{E}(\Phi^\star_{t_i}))\|_2^2}_{\mathcal{L}_{\text{rec}}} + \alpha_1 \underbrace{\|\hat{z}_{t_i} - \mathcal{E}(\Phi^\star_{t_i})\|_2^2}_{\mathcal{L}_{\text{forecast}}}\right], \tag{B 1}$$

where $\Phi^\star$ denotes the true system states extracted from the training trajectories N_t is the length of the trajectories, and $\hat{z}$ is the reduced states predicted by integrating the iLED model in time.

We also found that adding terms of lesser importance was beneficial and helped stabilize training. A reconstructed forecast loss was used:

$$\mathcal{L}_{\text{rec forecast}} = \frac{1}{N_t}\sum_{i=1}^{N_t}\|\Phi^\star_{t_i} - \mathcal{D}(\hat{z}_{t_i})\|_2^2. \tag{B 2}$$

And a regularization loss on the nonlinear part of the iLED dynamics:

$$\mathcal{L}_{\text{nonlinearity}} = \frac{1}{N_t}\sum_{i=1}^{N_t}\|\Psi_1(\hat{z}_{t_i}, \hat{\boldsymbol{h}}_{t_i})\|_2^2, \tag{B 3}$$

where Ψ_1 is the nonlinear part of the iLED dynamics in equation (2.14). Finally, the full loss is written as follows:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \alpha_1 \mathcal{L}_{\text{forecast}} + \alpha_2 \mathcal{L}_{\text{rec forecast}} + \alpha_3 \mathcal{L}_{\text{nonlinearity}}, \tag{B 4}$$

with the coefficients α_i adjusted to control the importance of each term. In our experiments, we generally used $\alpha_1 = 1$, and much lower values for $\alpha_2 = \alpha_3 = 10^{-3}$. We note that these values were identified by a hyperparameter search that investigated multiple different combinations for the three parameters, and that values close to the ones chosen were also explored and did not result in significantly different results.

Appendix C. Engineering details

This section discusses the choices we made during the creation of the method that should be considered to reproduce the results.

(a) Memory initialization

Thanks to the memory architecture of the iLED model, the memory term h can be initialized to an arbitrary degree of accuracy from the history of the solution. Indeed, the value of the memory at $t = 0$ is computed as follows:

$$h_0 = \int_{-\infty}^{0} \Psi_1(z(s)) e^{-\Lambda s} ds. \tag{C 1}$$

The infinite boundary of the above integral can be relaxed by computing the longest time horizon $\tau_{\max}$ of the memory from the largest entry $\lambda_{\max}$ of the negative diagonal matrix Λ:

$$\tau_{\max} = \frac{\epsilon}{\lambda_{\max}}, \tag{C 2}$$

where $\epsilon \in \mathbb{R}^+$ is a small parameter, generally chosen to be equal to 10^{-2}, that controls the relative error on the computation of h_0. After relaxing the infinite boundary in equation (C 1), the memory can be initialized as follows:

$$h_0 = \int_{\tau_{\max}}^{0} \Psi_1(\mathcal{E}(\Phi^\star(s))) e^{-\Lambda s} ds. \tag{C 3}$$

Note that the above integral can be computed from the training data as a simple trapezoidal integration, which easily is differentiated. Thus, we directly backpropagate through this operation during training.

(b) Linear parameterization

To ensure a higher degree of stability in the model. The linear operator $\mathbf{A}_\theta$ in the iLED architecture is parameterized to be stable as follows:

$$\mathbf{A}_\theta = \mathbf{W}_\theta - \mathbf{W}_\theta^T - \text{diag}(\text{abs}(\vec{w}_\theta)), \tag{C 4}$$

with $\mathbf{W} \in \mathbb{R}^{d_z \times d_z}$ a trainable weight matrix and $\vec{w}_\theta \in \mathbb{R}^{d_z}$ a trainable vector. With this formulation, the operator $\mathbf{A}_\theta$ is guaranteed to be stable, i.e. its eigenvalues have negative or zero real parts. This not only stabilizes the model but also avoids divergence of the model in the early stages of training.

Table 1. One-dimensional convolutional autoencoder used to obtain the results on the case of the FHN model presented in §3a

layer	encoder
(1)	ConstantPad1d(padding=(13, 14), value=0.0)
(2)	Conv1d(2, 8, kernel_size=(5,), stride=(1,), padding=same)
(3)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(4)	SiLU()
(5)	Conv1d(8, 16, kernel_size=(5,), stride=(1,), padding=same)
(6)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(7)	SiLU()
(8)	Conv1d(16, 32, kernel_size=(5,), stride=(1,), padding=same)
(9)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(10)	SiLU()
(11)	Conv1d(32, 4, kernel_size=(5,), stride=(1,), padding=same)
(12)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(13)	SiLU()
(14)	Flatten(start_dim=-2, end_dim=-1)
(15)	Linear(in_features=32, out_features=2, bias=True)
(16)	LatentSpaceCenteringLayer()
layer	**decoder**
(1)	Linear(in_features=2, out_features=32, bias=True)
(2)	SiLU()
(3)	Unflatten(dim=-1, unflattened_size=(4, 8))
(4)	Upsample(scale_factor=2.0, mode=linear)
(5)	ConvTranspose1d(4, 32, kernel_size=(5,), stride=(1,), padding=(2,))
(6)	SiLU()
(7)	Upsample(scale_factor=2.0, mode=linear)
(8)	ConvTranspose1d(32, 16, kernel_size=(5,), stride=(1,), padding=(2,))
(9)	SiLU()
(10)	Upsample(scale_factor=2.0, mode=linear)
(11)	ConvTranspose1d(16, 8, kernel_size=(5,), stride=(1,), padding=(2,))
(12)	SiLU()
(13)	Upsample(scale_factor=2.0, mode=linear)
(14)	ConvTranspose1d(8, 2, kernel_size=(5,), stride=(1,), padding=(2,))
(15)	1 + 0.5 Tanh()
(16)	Unpad()

(c) Latent space centring

To allow for the interpretability of the linear term in the iLED dynamics, it is important to ensure that the latent codes computed by the encoder $\mathcal{E}$ are centred. Indeed, a limit cycle arising from an unforced linear system will necessarily be centred around the origin. To do so, we define a *LatentSpaceCentering* operation $\mathbf{LC}(z)$ as follows:

$$\mathbf{LC}(z) = z - \vec{\mu}, \tag{C 5}$$

Table 2. Hyperparameters of the iLED dynamics used to obtain the results on the FHN case presented in §3a.

	iLED parameters
$\mathbf{A}_\theta$	Linear(2,2,bias=False)
Ψ_1 neurons	18 - 32 - 32 - 32 - 2
Ψ_1 activation	SiLU()
d_h	16
Ψ_2	AugmentedIdentityEncoder (see equation (C 6))
Ψ_2 neurons	2 - 5 - 8 - 11 - 14
Ψ_2 activation	SiLU()
Λ_θ	diag($\boldsymbol{w}$), $\boldsymbol{w} \in \mathbb{R}^{d_h}_-$

where $\vec{\mu}$ is a running mean of the latent code's averages that is computed during training and frozen at inference time. This approach is very similar to classical batch normalization, except the data are only centred, as unitary scaling of the latent space is not required for the model to learn efficiently. We note that other approaches can be used to treat this issue of latent space centring, such as adding a bias term to the linear part of the model.

(d) Form of the memory kernel network

The neural network Ψ_2 in equation (2.14) is used to *lift* the latent codes z to a space of arbitrary dimension. During training, it can learn to compute useful features of z that can then be integrated in time. To simplify the training, we propose a modification of the classical multi-layer perceptron, which we denote as *AugmentedIdentityEncoder* in the manuscript.

With this modification, the latent code z is added to the prediction of the neural network since the latent code already holds useful information in itself:

$$\Psi_2(z) = [z, \mathcal{MLP}(z)]. \tag{C 6}$$

where $\mathcal{MLP} : \mathbb{R}^{d_z} \mapsto \mathbb{R}^{d_h - d_z}$ denotes a standard multi-layer perceptron.

Appendix D. Network parameters

This section lists the various hyperparameters and network architectures used to obtain the results presented in §3.

(a) FHN model

Tables 1 and 2 present the architecture of both the autoencoder and iLED dynamical models used to obtain the results on the FHN case.

(b) KS

Similar to the previous paragraph, the architecture of the networks used for the KS case is presented in tables 3 and 4.

(c) Flow around a cylinder

Tables 5–7 present the architecture details and hyperparameters for the cylinder flow test case. To simplify the notation, we have defined two blacks that are both used in the autoencoder.

Table 3. One-dimensional convolutional autoencoder used to obtain the results on the case of the KS equation (§3b)

layer	encoder
(1)	Conv1d(1, 16, kernel_size=(5,), stride=(1,), padding=same)
(2)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(2), (0,))
(3)	SiLU()
(4)	Conv1d(16, 32, kernel_size=(5,), stride=(1,), padding=same)
(5)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(6)	SiLU()
(7)	Conv1d(32, 64, kernel_size=(5,), stride=(1,), padding=same)
(8)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(9)	SiLU()
(10)	Conv1d(64, 8, kernel_size=(5,), stride=(1,), padding=same)
(11)	AvgPool1d(kernel_size=(2,), stride=(2,), padding=(0,))
(12)	SiLU()
(13)	Flatten(start_dim=-2, end_dim=-1)
(14)	Linear(in_features=64, out_features=8, bias=True)
(15)	LatentSpaceCentering()
layer	**decoder**
(1)	Linear(in_features=8, out_features=64, bias=True)
(2)	Unflatten(dim=-1, unflattened_size=(8, 8))
(3)	Upsample(scale_factor=2.0, mode=linear)
(4)	ConvTranspose1d(8, 64, kernel_size=(5,), stride=(1,), padding=(2,))
(5)	SiLU()
(6)	Upsample(scale_factor=2.0, mode=linear)
(7)	ConvTranspose1d(64, 32, kernel_size=(5,), stride=(1,), padding=(2,))
(8)	SiLU()
(9)	Upsample(scale_factor=2.0, mode=linear)
(10)	ConvTranspose1d(32, 16, kernel_size=(5,), stride=(1,), padding=(2,))
(11)	SiLU()
(12)	Upsample(scale_factor=2.0, mode=linear)
(13)	ConvTranspose1d(16, 1, kernel_size=(5,), stride=(1,), padding=(2,))

Table 4. Hyperparameters of the iLED dynamics used to obtain the results on the KS case presented in §3b.

	iLED parameters
$\mathbf{A}_\theta$	$\mathbf{W} - \mathbf{W}^T - \mathrm{diag}(\boldsymbol{w})$, $\mathbf{W} \in \mathbb{R}^{d_z \times d_z}$, $\boldsymbol{w} \in \mathbb{R}_+^{d_z}$
Ψ_1 neurons	40 - 64 - 64 - 64 - 8
Ψ_1 activation	SiLU()
d_h	32
Ψ_2	AugmentedIdentityEncoder (see equation (C 6))
Ψ_2 neurons	8 - 12 - 16 - 20 - 24
Ψ_2 activation	SiLU()
Λ_θ	$\mathrm{diag}(\boldsymbol{w})$, $\boldsymbol{w} \in \mathbb{R}_-^{d_h}$

Table 5. Sub-blocks defined to help describe the CNN autoencoders

layer	downblock(in_size,out_size)
(1)	Conv2d(in_size, out_size, kernel_size=(5, 5), stride=(2, 2), padding=(2, 2), padding_mode=replicate)
(2)	SiLU()
layer	**upblock(in_size,out_size)**
(1)	Upsample(scale_factor=2.0, mode=bilinear)
(2)	Conv2d(in_size, out_size, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2), padding_mode=replicate)
(3)	SiLU()
(4)	BatchNorm2d()

Table 6. Hyperparameters of the two-dimensional convolutional autoencoder used to obtain the results on the cylinder case presented in §3c.

layer	encoder #1	encoder #2
(1)	DownBlock(2,4)	DownBlock(2,4)
(2)	DownBlock(4,16)	DownBlock(4,8)
(3)	DownBlock(16,4)	DownBlock(8,16)
(4)	DownBlock(4,2)	DownBlock(16,2)
(5)	Flatten(start=-3,end=-1)	Flatten(start=-3,end=-1)
(6)	Linear(512,20)	Linear(200,20)
(7)	$d_{z_1} = 20$	$d_{z_2} = 20$
	mixer encoder	
(8)	Concatenate(z_1, z_2)	
(9)	Linear(40,30)	
(10)	SiLU()	
(11)	Linear(30,d_z)	
layer	**mixer decoder**	
(1)	Linear(d_z,30)	
(2)	SiLU()	
(3)	Linear(30,40)	
(4)	$z_1, z_2 = z$	
	decoder #1	**decoder #2**
(5)	Linear(20,512)	Linear(20,200)
(6)	Unflatten(-1,(2,32,8))	Unflatten(-1,(2,10,10)
(7)	UpBlock(2,4)	UpBlock(2,16)
(8)	UpBlock(4,16)	UpBlock(16,8)
(9)	UpBlock(16,4)	UpBlock(8,4)
(10)	Upsample(2.0,bilinear)	Upsample(2.0,bilinear)
(11)	Conv2d(4, 1, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2), padding_mode=replicate)	Conv2d(4, 1, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2), padding_mode=replicate)
(12)	Flatten(start=-3,end=-1)	Flatten(start=-3,end=-1)
(13)	StreamFnToVelocity()	StreamFnToVelocity()

Table 7. Hyperparameters of the iLED dynamics used to obtain the results on the Cylinder cases presented in §3d.

	iLED parameters
$\mathbf{A}_\theta$	$\mathbf{W} - \mathbf{W}^T - \text{diag}(\boldsymbol{w})$, $\mathbf{W} \in \mathbb{R}^{d_z \times d_z}$, $\boldsymbol{w} \in \mathbb{R}_+^{d_z}$
Ψ_1 neurons	$d_z + d_h$ - 128 - 128 - 128 - d_z
Ψ_1 activation	SiLU()
d_h	32
Ψ_2	AugmentedIdentityEncoder (see equation (C 6))
Ψ_2 neurons ($Re = 750$)	16 - 17 - 17 - 17 - 15
Ψ_2 neurons ($Re = 100$)	3 - 10 - 16 - 22 - 28
Ψ_2 activation	SiLU()
Λ_θ	$\text{diag}(\boldsymbol{w})$, $\boldsymbol{w} \in \mathbb{R}_-^{d_h}$

The value of d_z changes depending on the Reynolds number considered. It is equal to $d_z = 3$ in the $Re = 100$ case, and $d_z = 16$ in the $Re = 750$ case. Note that we do not use a *LatentSpaceCentering* layer contrary to the other cases. This is due to the fact that because of the memory costs of the models, the batch size has to be relatively low, which has a negative effect on batch normalization approaches. To ensure that the latent space remained centred, which is critical to the accuracy and interpretability of the linear term in the dynamics, we added a term to the loss:

$$\mathcal{L}_{centring} = \left\| \frac{1}{N_T} \sum_{i=1}^{N_T} \mathcal{E}(\Phi_{t_i}) \right\|_2^2. \tag{D 1}$$

This loss effectively penalizes the average of the latent codes, ensuring that they are centred around the origin.

Also note that the decoder does not directly predict the velocity field, but the stream function ψ which is a scalar field that is used to compute the velocity components as follows:

$$u_x = \frac{\partial \psi}{\partial y}, \; u_y = -\frac{\partial \psi}{\partial x}. \tag{D 2}$$

This specific choice is inspired from previous works [81,83]. It allows for the guaranteed prediction of a divergence free field, which is a constraint of the incompressible Navier–Stokes equations.

(d) Comparison with purely linear dynamics

For the FHN case, we ran an ablation experiment to investigate if a model with a purely linear latent dynamics and no possibility of a nonlinear closure leads to a better result. By suppressing the nonlinear part of the dynamics, we obtain a purely linear model which should be able to capture the oscillatory nature of the system. However, we see in figure 11 that the long-term prediction performance suffers as the learned dynamics are dissipative.

This is due to the fact that the neural autoencoder struggles to learn a perfectly cyclical latent representation of the system as shown in figure 12. This in turn affects the quality of the dynamical model.

Finally, we point out that restricting the model to purely linear dynamics requires *a priori* assumptions on the studied system. Indeed, a purely linear model can struggle to predict complex non-oscillatory dynamics such as non-exponentially decaying and increasing transients. This is not an issue with the iLED method.

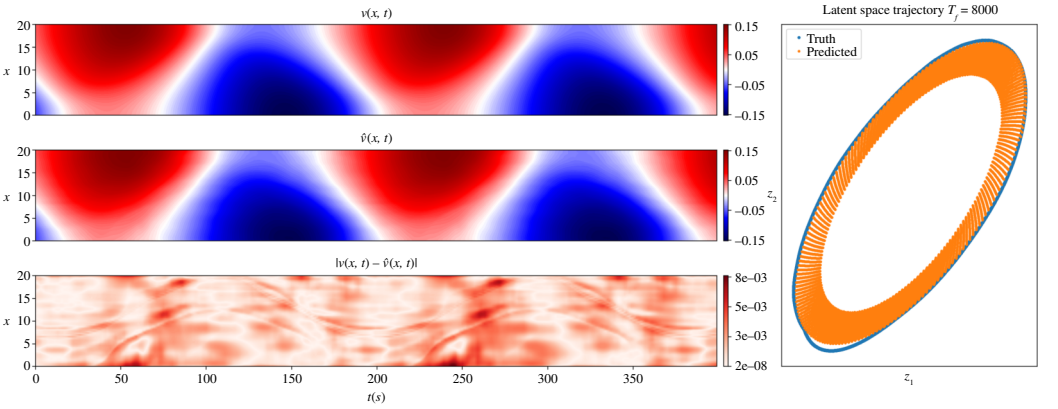

Figure 11. Forecast performance of a purely latent linear model on the case of the FHN equations

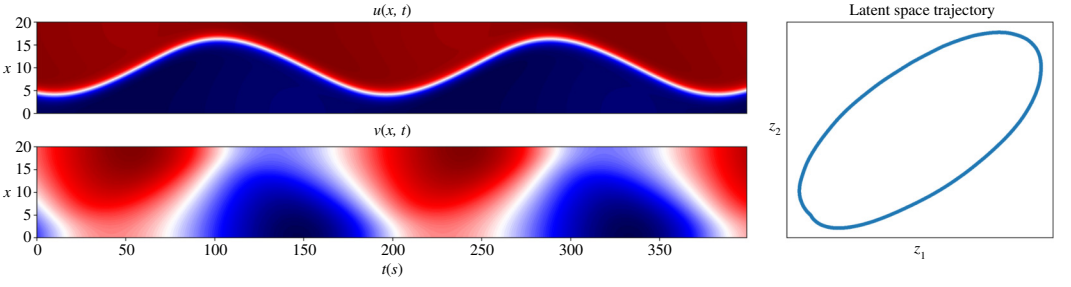

Figure 12. Latent representation of the case of the FHN equations learned through the iLED method using linear dynamics.

References

1. Wilcox DC. 1988 Multiscale model for turbulent flows. *AIAA J.* **26**, 1311–1320. (doi:10.2514/3.10042)
2. Council NR. 2012 *A Nat. Strat. for Adv. Clim. Model.* The National Academies Press.
3. Dura-Bernal S *et al.* 2019 NetPyNE, a tool for data-driven multiscale modeling of brain circuits. *Elife*. **8**, e44494. (doi:10.7554/elife.44494.016)
4. Mahadevan A. 2016 The impact of submesoscale physics on primary productivity of plankton. *Ann. Rev. of Mar. Sci.* **8**, 161–184. (doi:10.1146/annurev-marine-010814-015912)
5. Palmer T. 2015 Modelling: build imprecise supercomputers. *Nature* **526**, 32–33. (doi:10.1038/526032a)
6. Kevrekidis IG, Gear CW, Hummer G. 2004 Equation-free: the computer-aided analysis of complex multiscale systems. *AIChE J.* **50**, 1346–1355. (doi:10.1002/aic.10106)
7. Kevrekidis IG *et al.* 2003 Equation-free, coarse-grained multiscale computation: enabling microscopic simulators to perform system-level analysis. *Commun. Math. Sci.* **1**, 715–762. (doi:10.4310/cms.2003.v1.n4.a5)
8. Bar-Sinai Y, Hoyer S, Hickey J, Brenner MP. 2019 Learning data-driven discretizations for partial differential equations. *Proc. Natl Acad. Sci. USA* **116**, 15344–15349. (doi:10.1073/pnas.1814058116)
9. Weinan E, Engquist B, Huang Z. 2003 Heterogeneous multiscale method: a general methodology for multiscale modeling. *Phys. Rev. B* **67**, 092101. (doi:10.1103/physrevb.67.092101)
10. Weinan E, Engquist B, Li X, Ren W, Vanden-Eijnden E. 2007 Heterogeneous multiscale methods: a review. *Commun. in Comput. Phys.* **2**, 367–450.
11. Tao M, Owhadi H, Marsden JE. 2010 Nonintrusive and structure preserving multiscale integration of stiff ODEs, SDEs, and Hamiltonian systems with hidden slow dynamics via flow averaging. *Multi. Model. & Simul.* **8**, 1269–1324. (doi:10.1137/090771648)

12. Givon D, Kupferman R, Stuart A. 2004 Extracting macroscopic dynamics: model Problems and algorithms. *Nonlinearity* **17**, R55. (doi:10.1088/0951-7715/17/6/r01)
13. Mezić I. 2005 Spectral properties of dynamical systems, model reduction and decompositions. *Nonlin. Dynam.* **41**, 309–325. (doi:10.1007/s11071-005-2824-x)
14. Grigo C, Koutsourelakis PS. 2019 A physics-aware, probabilistic machine learning framework for coarse-graining high-dimensional systems in the small data regime. *J. Comput. Phys.* **397**, 108842. (doi:10.1016/j.jcp.2019.05.053)
15. Kaltenbach S, Koutsourelakis PS, Koumoutsakos P. 2023 Interpretable reduced-order modeling with time-scale separation. *arXiv preprint arXiv:2303.02189*.
16. Robinson JC. 1994 Inertial manifolds for the Kuramoto–Sivashinsky equation. *Phys. Lett. A* **184**, 190–193. (doi:10.1016/0375-9601(94)90775-7)
17. Rowley CW, Dawson ST. 2017 Model reduction for flow analysis and control. *Ann. Rev. of Fluid Mech.* **49**, 387–417. (doi:10.1146/annurev-fluid-010816-060042)
18. Peherstorfer B, Willcox K. 2015 Dynamic data-driven reduced-order models. *Comp. Meth. in Appl. Mech. and Eng.* **291**, 21–41. (doi:10.1016/j.cma.2015.03.018)
19. Noack BR, Afanasiev K, Morzyński M, Tadmor G, Thiele F. 2003 A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.* **497**, 335–363. (doi:10.1017/s0022112003006694)
20. Menier E, Bucci MA, Yagoubi M, Mathelin L, Schoenauer M. 2022 CD-ROM: complemented deep-reduced order model. (doi:10.48550/ARXIV.2202.10746)
21. Peherstorfer B. 2022 Breaking the Kolmogorov barrier with nonlinear model reduction. *Notices of the American Mathematical Society* **69**, 725–733. (doi:10.1090/noti2475)
22. Cohen A, DeVore R. 2016 Kolmogorov widths under holomorphic mappings. *IMA J. Num. Anal.* **36**, 1–12. (doi:10.1093/imanum/dru066)
23. Maday Y, Patera AT, Turinici G. 2002 Global a priori convergence theory for reduced-basis approximations of single-parameter symmetric coercive elliptic partial differential equations. *Comp. Rend. Math.* **335**, 289–294. (doi:10.1016/S1631-073X(02)02466-4)
24. Choi Y, Carlberg K. 2019 Space–time least-squares Petrov–Galerkin projection for nonlinear model reduction. *SIAM J. Sci.Comput.* **41**, A26–A58. (doi:10.1137/17m1120531)
25. Peherstorfer B. 2020 Model reduction for transport-dominated problems via online adaptive bases and adaptive sampling. *SIAM J. Sci. Comput.* **42**, A2803–A2836. (doi:10.1137/19m1257275)
26. Barnett J, Farhat C. 2022 Quadratic approximation manifold for mitigating the Kolmogorov barrier in nonlinear projection-based model order reduction. *J. Comput. Phys.* **464**, 111348. (doi:10.1016/j.jcp.2022.111348)
27. Geelen R, Wright S, Willcox K. 2023 Operator inference for non-intrusive model reduction with quadratic manifolds. *Comput. Meth. Appl. Mech. Eng.* **403**, 115717. (doi:10.1016/j.cma.2022.115717)
28. Fresca S, Dede' L, Manzoni A. 2021 A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs. *J. Sci. Comput.* **87**, 1–36. (doi:10.1007/s10915-021-01462-7)
29. Lee K, Carlberg KT. 2020 Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *J. Comput. Phys.* **404**, 108973. (doi:10.1016/j.jcp.2019.108973)
30. Vlachas PR, Arampatzis G, Uhler C, Koumoutsakos P. 2022 Multiscale simulations of complex systems by learning their effective dynamics. *Nat. Mach. Intel.* **4**, 359–366. (doi:10.1038/s42256-022-00464-w)
31. Hochreiter S, Schmidhuber J. 1997 Long short-term memory. *Neu. Comput.* **9**, 1735–1780. (doi:10.1162/neco.1997.9.8.1735)
32. Mori H. 1965 Transport, collective motion, and Brownian motion. *Prog. Theor. Phys.* **33**, 423–455. (doi:10.1143/ptp.33.423)
33. Zwanzig R. 1973 Nonlinear generalized Langevin equations. *J. Statis. Phys.* **9**, 215–220.
34. Koopman BO. 1931 Hamiltonian Systems and Transformations in Hilbert Space. *Proc. Natl Acad. Sci. USA* **17**, 315–318.
35. Mezić I. 2013 Analysis of fluid flows via spectral properties of the Koopman operator. *Ann. Rev. Fluid Mech.* **45**, 357–378. (doi:10.1146/annurev-fluid-011212-140652)
36. Chen TQ, Rubanova Y, Bettencourt J, Duvenaud DK. 2018 Neural ordinary differential equations. In *Adv. Neu. Inform. Procl Syst.*, pp. 6571–6583.
37. Li X, Wong TKL, Chen RT, Duvenaud D. 2020 scalable gradients for stochastic differential equations. *arXiv preprint arXiv:2001.01328*.

38. Brunton SL, Proctor JL, Kutz JN. 2016 Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl Acad. Sci. USA* **113**, 3932–3937. (doi:10.1073/pnas.1517384113)
39. Kaheman K, Kutz JN, Brunton SL. 2020 SINDy-PI: a robust algorithm for parallel implicit sparse identification of nonlinear dynamics. *Proc. R. Soc. A* **476**, 20200279. (doi:10.1098/rspa.2020.0279)
40. Kaltenbach S, Koutsourelakis PS. 2020 Incorporating physical constraints in a deep probabilistic machine learning framework for coarse-graining dynamical systems. *J. Comput. Phys.* **419**, 109673. (doi:10.1016/j.jcp.2020.109673)
41. Champion KP, Brunton SL, Kutz JN. 2019 Discovery of nonlinear multiscale systems: Sampling strategies and embeddings. *SIAM J. Appl. Dynam. Syst.* **18**, 312–333. (doi:10.1137/18m1188227)
42. Lusch B, Kutz JN, Brunton SL. 2018 Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.* **9**, 1–10. (doi:10.1038/s41467-018-07210-0)
43. Schmid PJ. 2010 Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **656**, 5–28. (doi:10.1017/s0022112010001217)
44. Williams MO, Kevrekidis IG, Rowley CW. 2015 A data–driven approximation of the koopman operator: Extending dynamic mode decomposition. *J. Nonlin. Sci.* **25**, 1307–1346. (doi:10.1007/s00332-015-9258-5)
45. Doshi-Velez F, Kim B. 2017 Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.
46. Koutnik J, Greff K, Gomez F, Schmidhuber J. 2014 A clockwork RNN. In *Internat. Conf. Mach. Learn.*, pp. 1863–1871. PMLR.
47. Kaltenbach S, Koutsourelakis PS. 2021 Physics-aware, probabilistic model order reduction with guaranteed stability. In *Internat. Conf. Learn. Rep. (ICLR)*.
48. Novati G, de Laroussilhe HL, Koumoutsakos P. 2021 Automating turbulence modelling by multi-agent reinforcement learning. *Nat. Mach. Intell.* **3**, 87–96.
49. Amsallem D, Farhat C. 2011 An online method for interpolating linear parametric reduced-order models. *SIAM J. Sci. Comput.* **33**, 2169–2198. (doi:10.1137/100813051)
50. Geelen R, Wright S, Willcox K. 2023 Operator inference for non-intrusive model reduction with quadratic manifolds. *Comput. Meth. Appl. Mech. Eng.* **403**, 115717. (doi:10.1016/j.cma.2022.115717)
51. Girin L, Leglaive S, Bie X, Diard J, Hueber T, Alameda-Pineda X. 2020 Dynamical variational autoencoders: a comprehensive review. *arXiv preprint arXiv:2008.12595*. (doi:10.1561/9781680839135)
52. Srivastava N, Mansimov E, Salakhudinov R. 2015 Unsupervised learning of video representations using lstms. In *Internat. Conf. Mach. Learn.*, pp. 843–852. PMLR.
53. Lin YT, Tian Y, Anghel M, Livescu D. 2021 Data-driven learning for the Mori–Zwanzig formalism: a generalization of the Koopman learning framework. *SIAM J. Appl. Dynam. Syst.* **20**, 2558–2601. (doi:10.1137/21m1401759)
54. Mezić I. 2020 Koopman operator, geometry, and learning. *arXiv preprint arXiv:2010.05377*.
55. Brunton SL, Budišić M, Kaiser E, Kutz JN. 2021 Modern Koopman theory for dynamical systems. *arXiv preprint arXiv:2102.12086*.
56. Budišić M, Mohr R, Mezić I. 2012 Applied koopmanism. *Chaos: An Inter. J. Nonli. Sci.* **22**, 047510.
57. Chorin A, Stinis P. 2007 Problem reduction, renormalization, and memory. *Commun. Appl. Math. Comput. Sci.* **1**, 1–27. (doi:10.2140/camcos.2006.1.1)
58. Darve E, Solomon J, Kia A. 2009 Computing generalized Langevin equations and generalized Fokker–Planck equations. *Proc. Nat. Acad. Sci.* **106**, 10884–10889. (doi:10.1073/pnas.0902633106)
59. Kondrashov D, Chekroun MD, Ghil M. 2015 Data-driven non-Markovian closure models. *Phys. D: Non. Phen.* **297**, 33–55. (doi:10.1016/j.physd.2014.12.005)
60. Gupta P, Schmid PJ, Sipp D, Sayadi T, Rigas G. 2023 Mori–Zwanzig latent space Koopman closure for nonlinear autoencoder. *arXiv preprint arXiv:2310.10745*.
61. Chen T, Chen H. 1995 Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Trans. Neu. Netw.* **6**, 911–917. (doi:10.1109/72.392253)

62. Lu L, Jin P, Pang G, Zhang Z, Karniadakis GE. 2021 Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat. Mach. Intell.* **3**, 218–229. (doi:10.1038/s42256-021-00302-5)
63. Kissas G, Seidman JH, Guilhoto LF, Preciado VM, Pappas GJ, Perdikaris P. 2022 Learning operators with coupled attention. *J. Mach. Learn. Res.* **23**, 1–63.
64. Zongyi L, Nikola BK, Kamyar A, Burigede L, Kaushik B, Andrew S, Anima A. 2021. Fourier Neural Operator for Parametric Partial Differential Equations. In *International Conference on Learning Representations*. See https://openreview.net/forum?id=c8P9NQVtmnO.
65. Li Q, Dietrich F, Bollt EM, Kevrekidis IG. 2017 Extended dynamic mode decomposition with dictionary learning: a data-driven adaptive spectral decomposition of the Koopman operator. *Chaos: An Inter. J. Nonlin. Sci.* **27**, 103111. (doi:10.1063/1.4993854)
66. Otto S, Rowley C. 2017 Linearly-recurrent autoencoder networks for learning dynamics. *SIAM J. Appl. Dyn. Sys.* **18**. (doi:10.1137/18M1177846)
67. Floryan D, Graham MD. 2022 Data-driven discovery of intrinsic dynamics. *Nat. Mach. Intell.* **4**, 1113–1120.
68. Zeng K, Graham MD. 2023 Autoencoders for discovering manifold dimension and coordinates in data from complex dynamical systems. *arXiv preprint arXiv:2305.01090*.
69. Falconer K. 2003 *Fractal Geometry: Mathematical Foundations and Applications*. (doi:10.1002/0470013850)
70. Mannattil M. 2023 NoLiTSA (NonLinear Time Series Analysis). See https://github.com/manu-mannattil/nolitsa.
71. Kar SK. 2006 A Semi-Implicit Runge–Kutta time-difference scheme for the two-dimensional shallow-water equations. *Mont. Weat. Rev.* **134**, 2916–2926. (doi:10.1175/MWR3214.1)
72. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. 2021 Physics-informed machine learning. *Nat. Rev. Phys.* **3**, 422–440. (doi:10.1038/s42254-021-00314-5)
73. Rocsoreanu C, Georgescu A, Giurgiteanu N. 2012 *The FitzHugh-Nagumo model: bifurcation and dynamics*, vol. 10. Springer Science & Business Media.
74. Karlin IV, Ansumali S, Frouzakis CE, Chikatamarla SS. 2006 Elements of the Lattice Boltzmann method I: linear advection equation. *Commun. Comput. Phys.* **1**, 616–655.
75. Hyman JM, Nicolaenko B. 1986 The Kuramoto-Sivashinsky equation: a bridge between PDE's and dynamical systems. *Phys. D: Nonlin. Phen.* **18**, 113–126. (doi:10.1016/0167-2789(86)90166-1)
76. Linot AJ, Graham MD. 2020 Deep learning to discover and predict dynamics on an inertial manifold. *Phys. Rev. E* **101**, 062209. (doi:10.1103/physreve.101.062209)
77. Linot AJ, Burby JW, Tang Q, Balaprakash P, Graham MD, Maulik R. 2023 Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems. *J. Comput. Phys.* **474**, 111838. (doi:10.2139/ssrn.4142014)
78. Bucci MA, Semeraro O, Allauzen A, Cordier L, Mathelin L. 2022 Nonlinear optimal control using deep reinforcement learning. In *IUTAM Laminar-Turbulent Transition: 9th IUTAM Symposium, London, UK, September 2–6, 2019* pp. 279–290. Springer. (doi:10.1007/978-3-030-67902-6_24)
79. Callaham JL, Maeda K, Brunton SL. 2019 Robust flow reconstruction from limited measurements via sparse representation. *Phys. Rev. Fluids* **4**, 103907. (doi:10.1103/physrevfluids.4.103907)
80. Popinet S Basilisk flow solver and PDE library. See doi:http://basilisk.fr/Front%20Pageversion=2023-06-01
81. Kičić I, Vlachas PR, Arampatzis G, Chatzimanolakis M, Guibas L, Koumoutsakos P. 2023 Adaptive learning of effective dynamics: adaptive real-time, online modeling for complex systems. *Comput. Methods Appl. Mech. Eng*. **415**, 116204. (doi:10.1016/j.cma.2023.116204)
82. Mezić I. 2020 Spectrum of the Koopman operator, spectral expansions in functional spaces, and state-space geometry. *J. Nonlin. Sci*. **30**, 2091–2145.
83. Mohan AT, Lubbers N, Chertkov M, Livescu D. 2023 Embedding hard physical constraints in neural network coarse-graining of three-dimensional turbulence. *Phys. Rev. Fluids* **8**, 014604. (doi:10.1103/PhysRevFluids.8.014604)