# Low-Order Continuous-Time Koopman Operator Learning via Physics-Informed Neural Networks

M. Zodros[1,2] , A. Colotti[2], M. Yagoubi[2], P. Chevrel[2]

***Abstract*— One of the main objectives in control theory is to obtain a linear representation of inherently nonlinear systems in order to leverage the analytical and theoretical tools developed for linear systems. In this context, the Koopman operator has attracted increasing interest in recent years.**

**Koopman operator theory provides a framework in which nonlinear dynamical systems are represented by a linear operator acting on an infinite-dimensional Hilbert space. Since such an infinite-dimensional representation is not numerically tractable, numerous finite-dimensional approximation methods have been proposed. These approaches typically rely on time-series data and include extended dynamic mode decomposition as well as deep learning–based variants. In this paper, we propose an original machine-learning-based approach for the synthesis of a fixed-dimensional Koopman approximant (lifting) of continuous-time nonlinear systems. A differential state-space representation of the system (as opposed to a recurrent state model) is assumed to be available through its vector field (f). The proposed encoder departs from conventional approaches in that it does not directly output the current latent state, but instead generates samples of the latent trajectory evaluated at user-defined time instants (temporal discretization). This formulation enables the integration into the learning process of Physical & Latent Continuous Losses, enforcing consistency between the physical dynamics and the Koopman dynamics, as well as Physical & Latent Boundary Losses, ensuring consistency with the prescribed initial conditions. In parallel, we introduce a structural stability constraint on the Koopman operator. The effectiveness of the proposed methodology is demonstrated through the analysis and simulation of two polynomial dynamical systems.**

## I. INTRODUCTION

Nonlinear systems theory aims to represent complex dynamics through simplified models that preserve the essential behavior of the system. In this context, several approaches such as model linearization, order reduction [1], and state-space lifting techniques [2] have been developed to obtain representations more suitable for analysis and control. However, these methods often suffer from limitations including local validity and the difficulty of constructing finite-dimensional linear representations of nonlinear dynamics.

In the context of Koopman theory, Koopman showed in 1931 [3] that nonlinear dynamical systems can be represented linearly in an infinite-dimensional function space. Nevertheless, this result remains mainly theoretical and does not directly provide a practical way to compute or approximate the Koopman operator from data.

*This work was not supported by any organization

[1]M.Zodros mickael.zodros@imt-atlantique.fr

[2] Authors are all with IMT Atlantique, LS2N, UMR CNRS 6004, Nantes, France.

To address this limitation, data-driven approaches have been developed. Among them, Dynamic Mode Decomposition (DMD), introduced by Schmid [6], was initially proposed as a numerical method for extracting coherent structures and dynamical features from fluid flow data. Subsequently, DMD was shown to be closely related to the Koopman operator, providing a finite-dimensional approximation of it [7], [8]. With the emergence of neural networks, new methods to approximate the Koopman operator have been proposed. Compared to approaches relying on the manual selection of basis functions to construct a dictionary of observables, deep learning methods offer a significant advantage by automatically learning appropriate representations directly from data. The choice of basis functions in dictionary-based methods is often problem-dependent, requires expert knowledge, and may fail to capture the relevant nonlinear features of the dynamics. In contrast, neural network architectures, such as autoencoders, are capable of discovering rich and expressive latent representations without the need for explicit feature engineering. This data-driven representation learning allows the model to adapt to complex nonlinear behaviors, improves generalization across operating regimes, and avoids the combinatorial growth of manually designed dictionaries. As a result, deep learning provides a flexible and scalable framework for identifying lifted representations that are better suited to approximate the underlying dynamics.

Recent work such as [5] proposes to learn continuous-time Koopman embeddings using neural networks and exploit them as priors for data assimilation. In this approach, the latent dynamics are modeled as a linear continuous-time system, enabling interpolation and forecasting from irregularly sampled data. However, the method relies on trajectory data and does not explicitly enforce physical constraints such as stability or dissipativity. Moreover, it typically relies on linking discrete- and continuous-time dynamics through the matrix exponential/logarithm relationship, which implicitly assumes the existence and proper definition of a matrix logarithm.

In contrast, the proposed approach directly learns a structured continuous generator $\mathbf{L}$, without requiring the existence or computation of a matrix logarithm. This avoids the ambiguities and numerical issues associated with the logarithm operator. As a result, the system can be simulated analytically through the matrix exponential $\exp(\mathbf{L}t)$, providing a continuous-time solution that is not restricted to the time horizons observed during training. This enables reliable extrapolation to arbitrary time scales. Furthermore, by

enforcing structural constraints on $\mathbf{L}$, the proposed method ensures physically consistent dynamics, such as stability and dissipativity, while retaining the advantages of a continuous-time formulation. In this work, the objective is to compare the proposed approach, which relies on a reduced-order representation of the dynamics, with the analytical finite-dimensional Koopman formulation introduced in [11]. In particular, we aim to assess whether a learned low-dimensional operator can capture the essential dynamics of the system while maintaining accuracy and stability properties comparable to those of the analytical construction.

The contributions of this work are summarized as follows:

- We propose a neural network architecture based on autoencoder and Physics-Informed Neural Network (PINN) [14] structures to construct a finite-dimensional continuous-time Koopman model.
- We benchmark the proposed method against an analytical finite-dimensional Koopman formulation using a lower dimension.
- We evaluate the performance of the model on different classes of dynamical systems to assess its generalization capability.

The notation adopted in this paper is standard and will be introduced as needed throughout the text.

## II. Background on Continuous-time Koopman theory

In this section we provide details on the continuous-time operators which will be used later in the construction of the autoencoder-PINN network leading to a fixed order continuous-time Koopman model.

Let us define an autonomous system as a system of ordinary differential equations on a finite-dimensional space state denoted $\mathcal{X} \subseteq \mathbb{R}^n$ of the form:

$$\frac{dx(t)}{dt} = \dot{x}(t) = f(x(t)) \tag{1}$$

Classically, Koopman theory give rise to a discrete-time operator, whose role is to push measurements forward in time [4]. Formally, let us note $\mathcal{K}$ the Koopman Operator and $\mathbf{F}^t : \mathcal{X} \to \mathcal{X}$ the flow map operator that advances initial condition $x(0)$ forward along the trajectory by a time $t$ such that:

$$x(t) = \mathbf{F}^t(x(0)) \tag{2}$$

The Koopman operator acts on measurement functions $\varphi : \mathcal{X} \to \mathbb{R}$ such that:

$$\mathcal{K}^\tau \varphi(x(t)) = \varphi(\mathbf{F}^\tau(x(t))) \tag{3}$$

Thus, in continuous time, we introduce the Lie operator as the infinitesimal generator of the Koopman operator family such that:

$$\mathcal{L}\varphi(t) = \lim_{\tau \to 0} \frac{\mathcal{K}^\tau \varphi(t) - \varphi(t)}{\tau} \tag{4}$$

An essential property of the Koopman operator $\mathcal{K}^\tau$ is that it acts linearly on the space of measurement functions. Consequently, its infinitesimal generator is also linear on this function space. By taking the limit as $\tau \to 0$, the discrete Koopman evolution yields the following continuous-time relation $\frac{d}{dt}\varphi(x) = \mathcal{L}\varphi(x)$ By applying the chain rule to equation (5), we obtain $\nabla_x \varphi . f(x) = \mathcal{L}\varphi(x)$

## III. New Hybrid and analytical neural architecture

In this section, we present the architecture of the learning process designed to identify a finite-dimensional approximation of the Lie operator and learn suitable lifted representations of the system dynamics. We consider that our data is the graph of the vector field $f$ defined as:

$$Graph(f) := \{(x_0, f(x_0)) | \, x_0 \in \mathcal{X}\} \tag{5}$$

and a time vector $T := [t_0, t_1, ...t_N], \quad t_0 < t_1 < \cdots < t_N$ with $N+1$ the number of samples considered.

This architecture is inspired by the autoencoder-based neural network used for Koopman operator learning in the discrete-time setting, as described in [9] and [10] in the autoencoder networks for learning dynamics. In this framework, the autoencoder consists of an encoder that maps the state into a latent (lifted) representation space of higher dimension where the linear model is identified, a layer that approximates the Koopman operator in this latent space, and a decoder that reconstructs the original state from the latent representation.

The autoencoder structure considered in this work is illustrated in (Fig. 1). Unlike conventional methods, our encoder $\Phi$ does not merely project the state-space into a higher-dimensional latent space: it explicitly incorporates time as a physical variable (PINN), enabling dynamic and context-aware prediction of the latent state at any given time instant. This fusion of latent representation and physics-informed modeling endows our approach with a unique ability to capture complex dynamics while preserving structural interpretability.

Thus, the encoder can be written

$$\begin{aligned} \varphi : \mathcal{X} \times \mathcal{T} &\longrightarrow \mathcal{Z} \\ (x_0, t) &\longmapsto z(t) := \varphi(x_0, t) \end{aligned} \tag{6}$$

where $\mathcal{Z} \subseteq \mathbb{R}^m$ denotes the set of measurement functions evaluated at time $t$ learned by the neural network.

Consequently, the decoder is defined as

$$\begin{aligned} \psi : \mathcal{Z} &\to \mathcal{X} \\ \varphi(x_0, t) &\longmapsto x(t) \end{aligned} \tag{7}$$

Thus, the decoder $\psi$ acts as the left inverse function of $\varphi$, in the sense that $\psi \circ \varphi(x_0, t) = x(t)$.

Finally, we approximate the action of the Lie operator on the finite-dimensional latent space by a matrix $\mathbf{L} \in \mathbb{R}^{m \times m}$, meaning that

$$\dot{z}(t) = \mathbf{L} z(t) \tag{8}$$

Compared to standard discrete-time autoencoders structures, we need to reconstruct the system dynamics in a continuous-time form in order to comply with the PINN framework. Since (PINNs) aim to learn dynamics that remain consistent

with the underlying physical principles, we enforce differential constraints on the latent dynamics to preserve the continuous-time structure of the system. If, intuitively, we assume that the latent-space dynamics are *equivalent* to the state-space ones, we can reconstruct the system dynamics exactly through the decoder's derivative. We formalize this intuition in the following result.

*Proposition 1:* Consider an autonomous dynamical system of the form (1) defined on a set $\mathcal{X}$ and so that its solutions exist for all positive times (i.e., it is *forward complete*). Let $\varphi : \mathcal{X} \times \mathcal{T} \to \mathcal{Z}$ and $\psi : \mathcal{Z} \to \mathcal{X}$ be such that $\psi \circ \varphi\big(x_0, t\big) = x(t)$ for all $x_0 \in \mathcal{X}$ and $t \geq t_0$, and assume that the restriction $\varphi_0(x) = \varphi(x, t_0)$ is surjective. Finally, let $\mathbf{F}_x$ and $\mathbf{F}_z$ represent the flows associated with the dynamics in the state-space $\mathcal{X}$ and in the latent space $\mathcal{Z}$ respectively.

If $\varphi$ encodes the latent dynamics, that is:

$$\forall x_0 \in \mathcal{X}, \forall t \geq t_0, \quad \varphi(x_0, t) = \mathbf{F}_z(\varphi(x_0, t_0), t) \tag{9}$$

then

$$\forall z_0 \in \mathcal{Z}, \forall t \geq t_0, \quad \mathbf{F}_x(\psi(z_0), t) = \psi(\mathbf{F}_z(z_0, t)) \tag{10}$$

*Proof:* First of all, we notice that the forward completeness of (1) and the dynamics encoding property of $\varphi$ guarantee that both flows are well defined for any $t \geq t_0$.

In order to prove (10), we apply $\psi$ to both sides of (9), obtaining

$$\forall x_0 \in \mathcal{X}, \forall t \geq t_0, \quad \psi \circ \varphi(x_0, t) = \psi \circ \mathbf{F}_z(\varphi(x_0, t_0), t)$$

Since $\psi \circ \varphi(x_0, t) = x(t)$, we deduce that

$$\forall x_0 \in \mathcal{X}, \forall t \geq t_0, \quad \mathbf{F}_x(x_0, t) = \psi \circ \mathbf{F}_z(\varphi(x_0, t_0), t) \tag{11}$$

Finally, the surjectivity of $\varphi_0$ means that for all $z_0 \in \mathcal{Z}$, there exists $x_0 \in \mathcal{X}$ so that $z_0 = \varphi(x_0, t_0)$; moreover, we immediately obtain that $\psi(z_0) = \psi \circ \varphi(x_0, t_0) = x_0$. By substituting in (11), we finally have

$$\forall z_0 \in \mathcal{Z}, \forall t \geq t_0, \quad \mathbf{F}_x(\psi(z_0), t) = \psi \circ \mathbf{F}_z(z_0, t) \tag{12}$$

which proves (10). ∎

*Corollary 1:* Let the hypotheses of Proposition 1 hold and, additionally, let $\psi \in \mathcal{C}^1(\mathcal{Z})$. Then, we can recover the state-space dynamics as

$$\dot{x}(t) = J_\psi(z(t))\dot{z}(t) \tag{13}$$

with $J_\psi$ denoting the Jacobian of $\psi$.

*Proof:* It suffices to take the time derivative of both sides of (10) and apply the chain rule on the right side. ∎

### A. Learning Architecture and Loss functions

To learn a physically consistent Lie operator, we introduce several complementary loss functions enforcing reconstruction accuracy, temporal consistency, and dynamical coherence in $\mathcal{X}$ and $\mathcal{Z}$.

For clarity in describing the proposed neural network architecture (Fig. 1), we introduce the following notations. Let

$$\mathbf{z}_T(x_0) := [\varphi(x_0, t_0), \varphi(x_0, t_2), \ldots, \varphi(x_0, t_N)] \tag{14}$$

denote the latent trajectory generated by the encoder. More concisely, we denote by $\Phi$ the encoder in (Fig. 1) which means

$$\mathbf{z}_T(x_0) := \Phi(x_0, T) \tag{15}$$

Similarly, let

$$\mathbf{x}_T(x_0) := [\psi(\varphi(x_0, t_0)), \ldots, \psi(\varphi(x_0, t_N))] \tag{16}$$

be the corresponding reconstructed trajectory in the state-space and more concisely, let $\Psi$ be the encoder in (Fig. 1) which means

$$\mathbf{x}_T(x_0) := \Psi(\mathbf{z}_T(x_0)). \tag{17}$$

Morevover, let

$$\dot{\mathbf{x}}_T(x_0) := [f(\psi(\varphi(x_0, t_0))), \ldots, f(\psi(\varphi(x_0, t_N)))] \tag{18}$$

be the associated time-derivative. In the same manner, let denote by $\dot{\tilde{\mathbf{z}}}_T(x_0)$ the time derivatives of the latent trajectory, and by $\dot{\tilde{\mathbf{z}}}_T(x_0)$ the derivatives predicted by the learned dynamics (cf. Eq. (5)).

Additionally, we denote by $\dot{\tilde{\mathbf{x}}}_T(x_0)$ the time derivatives of the state trajectory, and by $\dot{\tilde{\mathbf{x}}}_T(x_0)$ the derivatives induced by the latent dynamics through the decoder (cf. Eq. (8)).

With these notations introduced, we are now able to define different types of losses that will be used in the learning scheme.

- **Physical continuous losses:**

$$\mathcal{L}_1 = \left\| \dot{\tilde{\mathbf{x}}}_T(x_0) - \dot{\mathbf{x}}_T(x_0) \right\|_2^2 \tag{19}$$

$$\mathcal{L}_2 = \left\| \dot{\tilde{\mathbf{x}}}_T(x_0) - \dot{\mathbf{x}}_T(x_0) \right\|_2^2 \tag{20}$$

- **Physical boundary losses:**

$$\mathcal{L}_3 = \left\| [\mathbf{x}_T(x_0)]_1 - x_0 \right\|_2^2 \tag{21}$$

$$\mathcal{L}_4 = \left\| \left[\dot{\tilde{\mathbf{x}}}_T(x_0)\right]_1 - f(x_0) \right\|_2^2 \tag{22}$$

  with $[\mathbf{x}_T(x_0)]_1$, (resp. $\left[\dot{\tilde{\mathbf{x}}}_T(x_0)\right]_1$) is the first element of $\mathbf{x}_T(x_0)$ (resp. $\dot{\tilde{\mathbf{x}}}_T(x_0)$).

- **Latent continuous losse:**

$$\mathcal{L}_5 = \left\| \dot{\tilde{\mathbf{z}}}_T(x_0) - \dot{\tilde{\mathbf{z}}}_T(x_0) \right\|_2^2 \tag{23}$$

- **Latent boundary losses:**

$$\mathcal{L}_6 = \left\| \mathbf{L}\varphi(x_0, t_0) - \nabla\varphi(x_0, t_0)^T . f(x_0) \right\|_2^2 \tag{24}$$

$$\mathcal{L}_7 = \left\| \dot{\varphi}(x_0, t_0) - \nabla\varphi(x_0, t_0)^T . f(x_0) \right\|_2^2 \tag{25}$$

Here $\|.\|_2$ represents the $L^2$ norm.

Finally, the total loss is defined as a weighted sum of all individual losses:

$$\mathcal{L}_{global} := \sum_{i=1}^{7} \lambda_i \mathcal{L}_i \tag{26}$$

The weights $\lambda_i$ were selected according to the considered formulation, namely either the latent-dynamics approach that is to generate trajectories in the latent space with the analytical solution $z(t) = z_0 e^{\mathbf{L}t}$ or the PINN-based approach that is to generate trajectories from an initial condition $x_0$ and a time vector $T$.

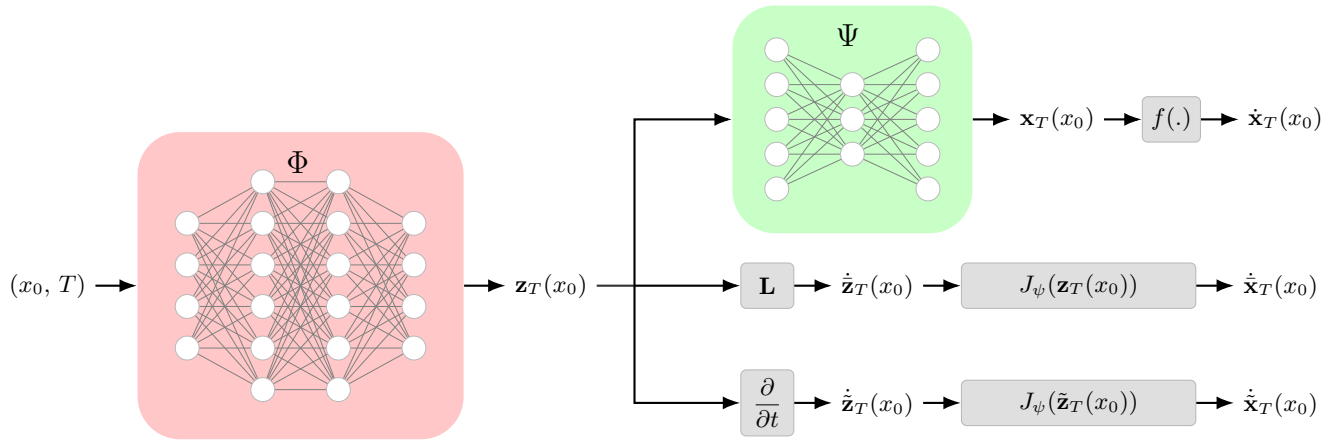


Fig. 1. Architecture of the neural network underlying the learning procedure

### B. Stability constraint on the Lie Operator

We impose a structural constraint on the Lie operator to ensure stability, as our study is limited to stable dynamical systems. In particular, following the parametrization of stable matrices proposed in [13], we define $\mathbf{L}$ as

$$\mathbf{L} := (J - R)Q \tag{27}$$

where $J$ is a skew-symmetric matrix, $R$ is a positive semi-definite matrix representing dissipation, and $P$ is a positive definite matrix (i.e. $J^\top = -J$, $R \succeq 0$, and $Q \succ 0$), ensuring that the learned operator remains Hurwitz stable [12], [13].

The proposed parametrization endows the learned operator with a dissipative structure. Indeed, to show this, consider the quadratic storage function $V(z) := \frac{1}{2} z^\top Q z, \quad \forall \quad z \in \mathcal{Z}$ which is positive definite since $Q \succ 0$. Using the skew-symmetry of $J$ and the semi-definite positivity of $R \succeq 0$, the time derivative along the trajectories governed by $\dot{z} = \mathbf{L}z$ is given by

$$\begin{aligned} \dot{V}(z) &= z^\top Q \dot{z} + \dot{z} Q z && (28)\\ &= z^\top Q(J - R)Qz + z^\top Q(-J - R)Qz && (29)\\ &= -z^\top QRQz \leq 0. && (30) \end{aligned}$$

This implies that the system is dissipative with respect to the storage function $V(z)$, and consequently Lyapunov stable. Moreover, if $R \succ 0$, the system is strictly dissipative, which guarantees asymptotic stability of the origin.

## IV. Results

We evaluate the proposed architecture on two systems: one presented in [11], which belongs to a class of polynomial nonlinear systems with a specific triangular structure, and another system that does not satisfy this structural assumption.

For all the examples the training, validation and test data were all generated by creating the graph of their vector field $f$ in a defined domain, henceforth denoted $\bar{\mathcal{X}} \subset \mathcal{X}$ such that for all vector $x \in \bar{\mathcal{X}}$, its elements $x_i$ are all in $[\bar{x}, \underline{x}]$ ($\bar{x}$ and $\underline{x}$ denote lower and upper bounds respectivel)y. The neural networks were trained using a time vector $T$ composed of $N = 100$ uniformly spaced values in the interval $[t_0, t_N] := [0, 1]$. We analyze the eigenvalues of the operator, and we generate trajectories with the analytical solution in the latent space using the matrix exponential of the Lie operator. Then, we decode to compare the trajectories to the trajectories generated with the well-known fourth-order Runge–Kutta method, henceforth denoted (RK4), from the original system. It is also worth noting that the stopping criterion was defined as a targeted final global loss lower than $10^{-3}$, and that all weighting coefficients $\lambda_i$ were set to 1 to ensure a fair comparison.

Moreover, in all considered examples, we focus on the Normalized Root Mean Squared Error (NRMSE) computed as:

$$\text{NRMSE} := \frac{\sqrt{\frac{1}{N_s}\sum_{j=1}^{N_s}\sum_{i=1}^{n}\left(x_i(t_j) - \hat{x}_i(t_j)\right)^2}}{\max_i \|x_i\|_2 - \min_i \|x_i\|_2}$$

where $N_s \gg N$ denotes the number of simulation samples, and $\max_i \|x_i\|_2$ (resp. $\min_i \|x_i\|_2$) denotes the maximum (resp. minimum) $L^2$ norm of the state vector $x_i$ across all samples.

Finally, all experiments were implemented in PyTorch and performed on a machine equipped with an NVIDIA RTX A4000 GPU.

### A. First Example : A Triangular polynomial system

We test the proposed architecture on the triangular polynomial nonlinear system presented in [11]. The structure of this system if of the form:

$$\begin{cases} \dot{x}_1 = f_1(x_1) \\ \dot{x}_2 = f_2(x_1, x_2) \\ \quad \vdots \\ \dot{x}_n = f_n(x_1, \ldots, x_n) \end{cases} \tag{31}$$

where each function $f_i$ is polynomial in its arguments. This particular structure allows the construction of a finite set of observables that is closed under the Koopman operator, leading to an explicit finite-dimensional linear representation of the dynamics. This analytical formulation provides

a benchmark that enables a direct comparison with our learned low-dimensional approximation. The specific system borrowed from [11] is given by:

$$\begin{cases} \dot{x}_1 &= a_1 x_1 \\ \dot{x}_2 &= a_2 x_2 + a_5 x_1^3 \\ \dot{x}_3 &= a_3 x_3 + a_6 x_1 x_2 + a_7 x_2^2 \\ \dot{x}_4 &= a_4 x_4 + a_8 x_1 x_2 x_3 \end{cases} \tag{32}$$

with $a_i = -0.5, \quad \forall i \in \{0, \ldots, 4\}$ and $a_i = -0.2, \quad \forall i \in \{5, \ldots, 8\}$. The training was performed according to the architecture shown in (Fig. 1) with the latent space dimension initially set to $m = 19$. This dimension corresponds to the exact solution formulated in [11].

After the training, we show in (Fig. 4) that the stability constraint of the obtained approximate operator $\mathbf{L}$ was respected. To evaluate the temporal behavior of the model, trajectories were simulated using a fixed integration step of $dt = 0.01$ over $N_s = 500$ time steps. This long horizon allows us to assess both transient and asymptotic regimes of the system. The trajectories were simulated using initial states conditions $x_i = 1, \ \forall \ i \in \{1, \ldots, 4\}$ and we observed a NRMSE of $10^{-3}$ (Fig. 2).

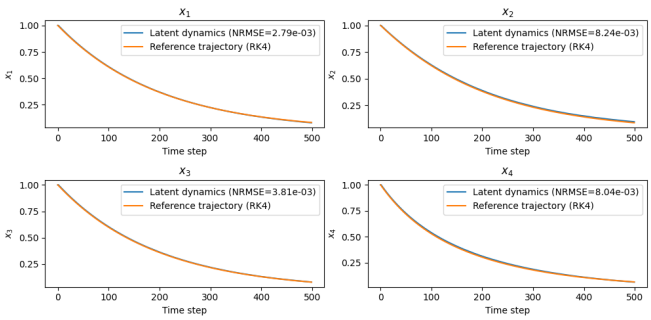


Fig. 2. For $m = 19$ : In orange the ground truth trajectories of the original system simulated with RK4. In blue the trajectories of the learned system reconstructed from the latent space via the decoder.

We then evaluated the proposed learning procedure using reduced latent dimensions (i.e. $m = 2, \ldots, m = 19$). We observed that an NRMSE of approximately $10^{-8}$ could be achieved with $m = 4$ (Fig. 3).

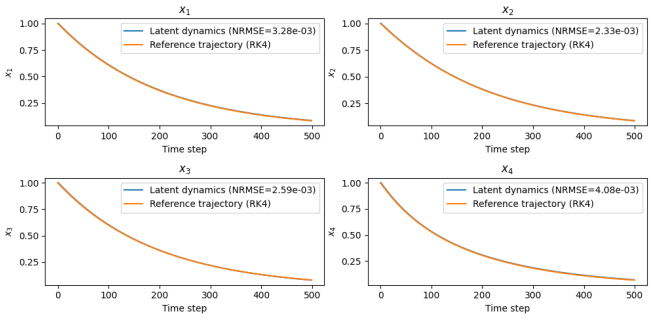


Fig. 3. For $m = 4$ : In orange the ground truth trajectories of the original system simulated with RK4. In blue the trajectories of the learned system reconstructed from the latent space via the decoder.

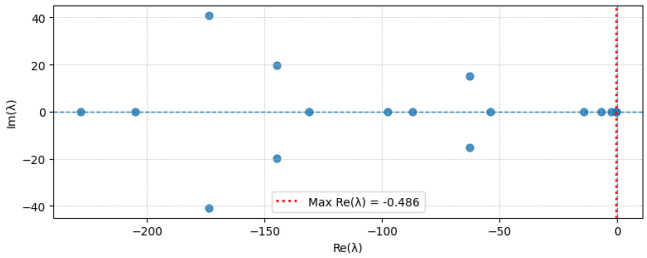


Fig. 4. First example: $\mathbf{L}$ spectrum with a latent dimension $m = 19$.

This illustrate the fact that our approched could lead to low order dimension Koopman continous-time operator with the same performance level while ensuring the stability constraint (Fig. 4 & 5).

We also consider the simulation time. Table I shows the comparaison with RK4 for the original system and with the latent space dynamics in two cases $m = 4$ and $m = 19$ over $N_s = 5.10^5$ time steps (this corresponds to the average value over 10 trials). We cleary obtained a speed up factor of $2-5$ for the two cases.

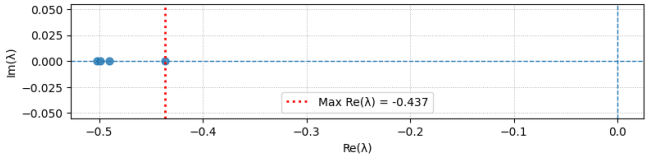


Fig. 5. First example: L spectrum with a latent dimension m = 4.

TABLE I

COMPARISON OF TRAJECTORY SIMULATION TIMES BETWEEN RK4 INTEGRATION AND LATENT-SPACE PROPAGATION $\psi(z_0 e^{\mathbf{L}t})$ OVER $N_s = 5.10^5$ TIME STEPS

| Simulation Method | Simulation time in (s) | Speed-up | Latent space dimension |
|---|---|---|---|
| RK4 | $1.1\times10^{-1}$ | $1\times$ | |
| Latent dynamics | $4.6\times10^{-2}$ | $2.4\times$ | 19 |
| Latent dynamics | $2.3\times10^{-2}$ | $4.8\times$ | 4 |

## B. Second Example

We propose to evaluate the learning approach on a polynomial nonlinear system that does not respect the triangular form (31) defined by:

$$\begin{cases} \dot{x}_1 &= -x_1 + x_2 x_1^2 \\ \dot{x}_2 &= -x_2 - x_1^3 \end{cases} \tag{33}$$

In this case, the training was performed in the same conditions as for the first example with the latent space dimension set to $m = 4$. For this dimension an NRMSE of approximately $10^{-3}$ was obtained. After the training, we show in (Fig. 6) that the stability constraint of the obtained approximate operator $\mathbf{L}$ was respected.

In the same manner as for the first example, we evaluate the temporal behavior of the model, with a fixed integration

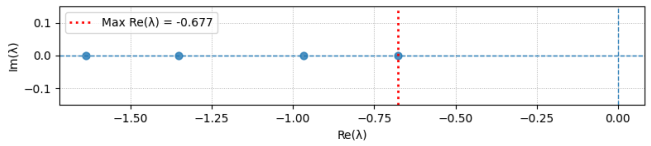


Fig. 6. Second example: $\mathbf{L}$ spectrum with a latent dimension $m = 4$.

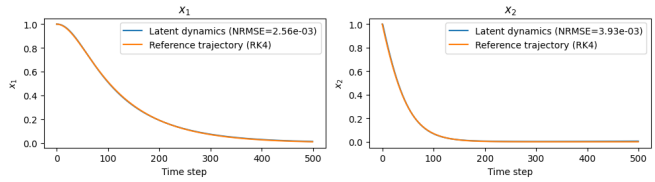


Fig. 7. For $m = 4$ : In orange the ground truth trajectories of the original system simulated with RK4. In blue the trajectories of the learned system reconstructed from the latent space via the decoder.

step of $dt = 0.01$ over $N_s = 5.10^2$ time steps. The trajectories were simulated using initial states conditions $x_i = 1,\ \forall\ i \in \{1, 2\}$ (Fig. 7).

Note that for this example as well, the simulation time is reduced by a factor 2.

The obtained results highlighted several important aspects of the proposed approach. First, the fact that all eigenvalues of the learned operator lie in the left half-plane confirms that the imposed structural constraints successfully enforce dissipativity and contribute to the stability of the learned dynamics. Second, the low reconstruction error observed for trajectories initialized from different initial conditions demonstrates the model's ability to generalize across the state-space, particularly near stable equilibria.

## V. Concluding Remarks and Perspectives

In this paper, we address the problem of approximating continuous-time nonlinear models using a structured approach that composes two static nonlinear transformations and a linear dynamic transformation. This is closely related to the Koopman formalism and the associated family of Koopman operators defined by the system's flow. Specifically, we assume the availability of a differential state-space model (as opposed to a recurrent state model) through its vector field $f$. Under this assumption, we propose an original learning scheme that leverages the discretization of the graph of $f$.

The proposed encoder is unconventional in that it does not output the current latent state but rather samples of the latent trajectory evaluated at user-specified time instants (temporal discretization). This design enables the integration of what we term Physical & Latent Continuous Losses (penalties enforcing consistency with physical and Koopman dynamics) and Physical & Latent Boundary Losses (penalties ensuring consistency of initial conditions) into the learning process. Concurrently, we introduce a structural stability constraint on the Koopman operator.

We numerically validate the methodology on two polynomial systems. The first, a 4th-order polynomial system, is known to admit a finite-dimensional Koopman representation of order 19, with a particular structure that allows for an analytical derivation. Our approximation methodology yields a continuous-time Koopman model whose simulations (evaluated using the NMRSE metric over a large set of trajectories) are arbitrarily close to those of the original model. Furthermore, the method demonstrates that a latent space of order 4 suffices to produce a highly accurate Koopman model—a result that is not evident from the existing literature. The second example confirms that the methodology is equally effective for more general polynomial systems.

In conclusion, the proposed approach effectively generates structured models that significantly reduce simulation time by leveraging linear simulation in the latent space, with the encoder and decoder serving as interfaces to the physical system. This methodology opens promising avenues for physics-informed approximation of a broad class of nonlinear differential models.

## References


[1] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control: Analysis and Design*, 2nd ed. Chichester, U.K.: Wiley, 2005.

[2] M. A. Hernández-Ortega, C. M. Rergis, A. Román-Messina, and E. R. Murillo-Aguirre, "Carleman linearization of differential-algebraic equations systems," *Results in Applied Mathematics*, vol. 28, p. 100660, 2025, doi: 10.1016/j.rinam.2025.100660.

[3] B. O. Koopman, "Hamiltonian systems and transformation in Hilbert space," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 17, no. 5, pp. 315–318, 1931.

[4] S. L. Brunton, M. Budisic, E. Kaiser, and J. N. Kutz, "Modern Koopman Theory for Dynamical Systems," *SIAM Review*, vol. 64, no. 2, pp. 229–340, 2022, doi: 10.1137/21M1401243.

[5] A. Frion, L. Drumetz, M. Dalla Mura, G. Tochon and A. A. E. Bey, "Neural Koopman Prior for Data Assimilation," in IEEE Transactions on Signal Processing, vol. 72, pp. 4191-4206, 2024, doi: 10.1109/TSP.2024.3416828.

[6] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *Journal of Fluid Mechanics*, vol. 656, pp. 5–28, 2010.

[7] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, "Spectral analysis of nonlinear flows," *Journal of Fluid Mechanics*, vol. 641, pp. 115–127, 2009.

[8] J. H. Tu, C. W. Rowley, D. M. Luchtenburg, S. L. Brunton, and J. N. Kutz, "On dynamic mode decomposition: Theory and applications," *Journal of Computational Dynamics*, vol. 1, no. 2, pp. 391–421, 2014.

[9] L. Lv, L. Liu, L. Bao, F. Sun, J. Dong, J. Zhang, X. Shan, K. Sun, H. Huang, and Y. Luo, "Multi-Segment Soft Robot Control Via Deep Koopman-Based Model Predictive Control," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, Atlanta, GA, USA, May 2025, pp. 9266–9272, doi: 10.1109/ICRA55743.2025.11127925.

[10] D. J. Alford-Lago, C. W. Curtis, A. T. Ihler, and O. Issan, "Deep learning enhanced dynamic mode decomposition," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 32, no. 3, p. 033116, 2022, doi: 10.1063/5.0073893.

[11] L. C. Iacob, M. Schoukens, and R. Tóth, "Finite dimensional Koopman form of polynomial nonlinear systems," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 6423–6428, 2023.

[12] G.-R. Duan and R. J. Patton, "A note on Hurwitz stability of matrices," *Automatica*, vol. 34, no. 11, pp. 1409–1412, 1998.

[13] N. Gillis and P. Sharma, "On computing the distance to stability for matrices using linear dissipative Hamiltonian systems,"*Automatica*, vol. 85, pp. 113–121, 2017. https://doi.org/10.1016/j.automatica.2017.07.047.

[14] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational Physics, Volume 378, 2019, Pages 686-707, ISSN 0021-9991, https://doi.org/10.1016/j.jcp.2018.10.045.