SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

September 2024, Vol. 67, Iss. 9, 192201:1–192201:18
https://doi.org/10.1007/s11432-022-4050-4

Physics-informed deep Koopman operator for Lagrangian dynamic systems

Xuefeng WANG[1], Yang CAO[1], Shaofeng CHEN[1] & Yu KANG[1,2,3*]

[1]*Department of Automation, University of Science and Technology of China, Hefei* 230027*, China;*
[2]*State Key Laboratory of Fire Science, University of Science and Technology of China, Hefei* 230027*, China;*
[3]*Institute of Advanced Technology, University of Science and Technology of China, Hefei* 230027*, China*

Received 14 January 2022/Revised 13 June 2022/Accepted 4 December 2023/Published online 20 August 2024

Abstract Accurate mechanical system models are crucial for safe and stable control. Unlike linear systems, Lagrangian systems are highly nonlinear and difficult to optimize because of their unknown system model. Recent research thus used deep neural networks to generate linear models of original systems by mapping nonlinear dynamic systems into a linear space with a Koopman observable function encoder. The controller then relies on the Koopman linear model. However, without physical information constraints, ensuring control consistency between the original nonlinear system and the Koopman system is tough, as the learning process of the Koopman observation function is unsupervised. This paper thus proposes a two-stage learning algorithm that uses structural subnetworks to build a physics-informed network topology to simultaneously learn the Koopman observable functions and the system energy representation. In the Koopman matrix learning session, a quadratic-constrained optimization problem is solved to ensure that the Koopman representation satisfies the energy difference matching hard constraint. The proposed energy-preserving deep Lagrangian Koopman (EPDLK) framework effectively represents the dynamics of the Lagrangian system while ensuring control consistency. The effectiveness of EPDLK is compared with those of various Koopman observable function construction methods in multistep prediction and trajectory tracking tasks. EPDLK achieves better control consistency by guaranteeing energy difference matching, which facilitates the application of the control law generated on the Koopman system directly to the original nonlinear Lagrangian system.

Keywords Koopman operator, Lagrangian, physics-informed, nonlinear, model predictive control (MPC)

1 Introduction

The Lagrangian system can characterize various critical dynamic systems, such as multirotor drones, humanoid robots, and robotic arms [1–3]. However, traditional control approaches require precise dynamic models, which are frequently difficult to provide in real-world scenarios. Moreover, even when accurate dynamic models are available, their strong nonlinearity results in considerable difficulty in designing corresponding controllers.

With the development of data acquisition techniques, obtaining state data for dynamic systems has become relatively easy compared with the high cost and specialized knowledge required for modeling based on the first-principles mechanism. Therefore, data-driven modeling (DDM) and control are becoming important research directions when only the type of system model is known, and the parameters of the system model are unknown [4, 5].

In recent years, the Koopman operator [6] approach for dynamic systems has received attention from researchers, providing a theory for constructing a global infinite-dimensional linearized representation of nonlinear systems. It thus makes mature linear system control techniques promising for nonlinear system controller design [7, 8]. However, the infinite-dimensional nature of the Koopman operator makes it impractical for real-world applications. To address this limitation, researchers have proposed DDM techniques, such as dynamic mode decomposition (DMD) [9] and its variant, extended DMD (EDMD) [10], to compute a finite-dimensional Koopman representation from the time-series data of nonlinear system

* Corresponding author (email: kangduyu@ustc.edu.cn)

states. Although DDM techniques are adequate, traditional EDMD approaches require a manual observation function design and involve significant subjectivity and potential model bias, making the explicit modeling of latent variable relationships in dynamic systems challenging. Given the excellent performance of deep neural networks (DNNs) in capturing relationships among latent variables in complex systems [11], research works have used deep autoencoder (DAE) frameworks to learn appropriate Koopman embeddings automatically [11,12]. Moreover, building upon such learned Koopman representations using DAE, the model predictive control (MPC) method has successfully been applied to various nonlinear control tasks [13–15].

In current Koopman-based MPC studies, the typical pipeline involves solving an optimization problem within the Koopman linear system and directly applying the generated control law to the original nonlinear system. This pipeline potentially requires the Koopman linear system to satisfy "constraints on control consistency", ensuring that the optimal control laws generated in both the Koopman linear system and the original nonlinear system are consistent [7, 8, 13, 14]. However, learning Koopman representations using DAEs is an unsupervised process susceptible to various factors, such as random initializations and data noise, among others [16–18]. As we have demonstrated in our research, the Koopman linear system representation is subject to multivaluedness without additional constraints, can compromise the constraints on control consistency, and can render the control objectives unattainable. Furthermore, it may even lead to dangerous behaviors in practice [15, 19].

In the context of Lagrangian systems where dissipation is negligible, drawing inspiration from energy-shaping control theory [20–22], we propose a Koopman representation approach that adheres to energy difference matching (EDM) constraints. The issue of the multivaluedness of the Koopman representation is addressed by ensuring a consistent energy difference between states in the original Lagrangian system and corresponding states in the Koopman representation. Consequently, the energy-optimal control laws generated within the Koopman framework are equally optimal for the original nonlinear system. This approach guarantees the consistency of control laws between the original system and its Koopman representation.

In scenarios where only the time series data of original Lagrangian system states are known, with unknown system model parameters, we propose a two-stage learning algorithm to ensure that the corresponding Koopman representation adheres to EDM constraints: energy-preserving deep Lagrangian Koopman (EPDLK). This algorithm separates the learning of Koopman representation into distinct phases: learning Koopman observable functions and Koopman matrices independently.

In the Koopman observable function learning stage, this paper proposes a physics-informed network topology called deep Lagrangian Koopman (DLK) based on the physical structure of the Lagrangian system. By introducing Lagrangian structural subnetworks and physical constraint loss functions, DLK achieves a structure-preserving Koopman representation while identifying the parameters of the Lagrangian system and learning the Koopman observation function.

During the Koopman matrix learning stage, we introduce EDM constraints as hard constraints, transforming the Koopman matrix solution problem into a quadratic optimization problem with quadratic equality constraints. This ensures that the EPDLK representation satisfies the EDM constraints without solving complex neural network optimization problems with hard constraints.

Because of the Euler-Lagrangian representation, the EPDLK operator is suitable for representing the dynamic processes of any rigid robot structure without additional object domain knowledge. For different robot systems, only the input and output dimensions of the EPDLK network must be changed without changing the network structure and training process.

Then, in the state prediction and control tasks of a typical Lagrangian system (cart-pole), the proposed EPDLK framework is compared with other advanced methods for building Koopman observation functions, such as radial basis functions (RBFs) [7], polynomial basis [23], Hermite polynomials used in ACD-EDMD [24], deep MLP (multilayer perceptron) [12], ELU-NN [25], and deep convolutional neural network (CNN) [26], in terms of state prediction accuracy, computational cost, and control performance. The proposed method can construct energy-preserving Koopman observation functions so that the control laws designed in Koopman linear systems can be directly applied to the corresponding nonlinear system control tasks.

In summary, this paper makes the following contributions:

(1) **A Lagrangian Koopman representation framework with EDM constraints.** The proposed framework addresses the inherent multivaluedness problem in Koopman representations created through DAEs by ensuring a consistent energy difference between states in the original system and their

counterparts in the Koopman representation. This approach guarantees that the control law remains a consistent between the original system and the Koopman representation.

(2) **EPDLK algorithm.** The proposed EPDLK algorithm efficiently constructs Koopman representations adhering to EDM constraints by dividing learning into two stages: first using the DLK network for learning Koopman observables and then optimizing Koopman matrices under EDM constraints. This approach streamlines learning into a quadratic optimization task, ensuring compliance with essential constraints while simplifying the process and avoiding the complexities of hard-constrained neural network optimization.

(3) **DLK network and Koopman matrix optimization.** In addition, the proposed DLK network achieves the joint learning of the Koopman observation function and the system energy function of a Lagrangian system in a data-driven manner by introducing Lagrangian structural subnetworks and a physical constraint loss function.

The remainder of this paper is organized as follows. Section 2 introduces the preliminaries on the Koopman theory and Lagrangian dynamics and then formulates the problems of multistep prediction and controller design. The main results of the equivalence of control consistency and EDM, the energy-preserving Koopman operator for the Lagrangian system, and the deep Koopman controller are given in Sections 3–5, respectively. Numerical simulation results are given in Section 6. Finally, the conclusion is given in Section 7.

2 Preliminaries

In model-based robot control, dynamic systems must be modeled first. This article's main problem is the use of data-driven methods to construct linear embedded models for nonlinear robot systems with unknown structures. In this section, we first introduce the Koopman operator of the controlled system. Koopman observation functions can map the nonlinear state space into a linear embedding space for a nonlinear dynamic system. At the same time, the existing methods of constructing the Koopman operator and observation function are introduced. Then, we discuss the Koopman-based control process. Finally, we describe the Lagrangian equation of the dynamic robot system.

2.1 Koopman operator for dynamic systems

Consider a discrete-time nonlinear controlled dynamic system $\boldsymbol{x}$-system:

$$\boldsymbol{x}_{k+1} = F(\boldsymbol{x}_k, \boldsymbol{u}_k). \tag{1}$$

The continuous dynamic system $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$ can be rewritten as a discrete-time system by defining the sampling time interval T:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}(t+T) = \boldsymbol{x}(t) + \int_t^{t+T} f(\boldsymbol{x}(\tau), \boldsymbol{u}(\tau))\mathrm{d}\tau = F(\boldsymbol{x}_k, \boldsymbol{u}_k), \tag{2}$$

where the state of the dynamic system $\boldsymbol{x}(t) \in \mathcal{S} \subset \mathbb{R}^s$ is denoted as $\boldsymbol{x}_k$, T is the sampling time interval, and $\boldsymbol{u}(t) \in \mathcal{M} \subset \mathbb{R}^m$ is the control input at time t.

We are concerned with constructing a linear model for the nonlinear system (1) suitable for control design methods such as MPC.

The Koopman operator theory [6] states that for any nonlinear system $\boldsymbol{x}$-system, there exists a set of observation functions $\boldsymbol{\phi}(\boldsymbol{x}_k)$ that map the $\boldsymbol{x}$-system to an infinite-dimensional linear system in Hilbert space.

The Koopman operator $\mathcal{K}$ is an infinite-dimensional linear operator:

$$[\mathcal{K}\boldsymbol{\phi}](\boldsymbol{x}_k) = \boldsymbol{\phi}\left(F(\boldsymbol{x}_k, \boldsymbol{u}_k)\right) = \boldsymbol{\phi}(\boldsymbol{x}_{k+1}), \tag{3}$$

where $\boldsymbol{\phi}(\boldsymbol{x}_k) = \{\phi_1(\boldsymbol{x}_k), \ldots, \phi_n(\boldsymbol{x}_k), \ldots\} : \mathcal{S} \to \mathbb{R}$.

Let

$$\boldsymbol{z}_k = \phi(\boldsymbol{x}_k), \quad \boldsymbol{x}_k = \psi(\boldsymbol{z}_k), \tag{4}$$

where ψ is the inverse function of ϕ.

Eq. (3) can be rewritten as

$$\boldsymbol{z}_{k+1} = \mathcal{K}_z \boldsymbol{z}_k + \mathcal{K}_u \boldsymbol{u}_k, \tag{5}$$

where $\mathcal{K}_z, \mathcal{K}_u$ are the submatrices of the Koopman operator $\mathcal{K}$, the state and control matrices for the state variable $\boldsymbol{z}$.

In other words, the Koopman operator defines a linear dynamical system $\boldsymbol{z}$-system, which determines the mapping relation from $\boldsymbol{z}_k$ to $\boldsymbol{z}_{k+1}$. The Koopman operator enables obtaining the properties of the original nonlinear $\boldsymbol{x}$-system (1) by studying the properties of the $\boldsymbol{z}$-system (5) using well-established linear system analysis techniques.

2.2 Finite-dimensional approximating the Koopman operator

The infinite-dimensional Koopman operator is not feasible in practice, and finite-dimensional subspace approximations can be constructed using the following data-driven methods: DMD [9] algorithm and its variant EDMD [27].

Provided the dataset of dynamic systems' trajectories $\mathcal{D} = \{\mathcal{T}_n\}_{n=0}^N$, where $\mathcal{T} = \{\boldsymbol{x}_k, \boldsymbol{u}_k\}_{k=0}^T$ is a trajectory, DMD transforms the Koopman operator approximation problem into the following optimization problem, which is a problem of a typical least square:

$$\min_{\mathcal{K}_x, \mathcal{K}_u} \sum_{k=0}^{N} \|\boldsymbol{x}_{k+1} - \mathcal{K}_x \boldsymbol{x}_k - \mathcal{K}_u \boldsymbol{u}_k\|^2. \tag{6}$$

For EDMD, by constructing basis functions (4) as observation functions, the corresponding optimization problem becomes

$$\min_{\mathcal{K}_z, \mathcal{K}_u} \mathcal{L}_{\text{lin}}, \tag{7}$$

where

$$\text{Linear loss} : \mathcal{L}_{\text{lin}} = \sum_{k=0}^{N} \|\boldsymbol{z}_{k+1} - \mathcal{K}_z \boldsymbol{z}_k - \mathcal{K}_u \boldsymbol{u}_k\|^2. \tag{8}$$

The original infinite-dimensional system state vector becomes a finite-dimensional vector $\boldsymbol{z} \in \mathbb{R}^n$, and the corresponding finite-dimensional Koopman operator matrices are $\mathcal{K}_z \in \mathbb{R}^{n \times n}$, $\mathcal{K}_u \in \mathbb{R}^{n \times m}$.

In the traditional EDMD algorithm, Koopman observation functions are hand-crafted or randomly selected, such as polynomial basis functions or RBFs [7]. In recent years, researchers have explored learning-based methods that use DNN autoencoders [11,13,28,29] to learn both the observation function $\phi(\boldsymbol{x})$ and the Koopman operator matrix $\mathcal{K}_z, \mathcal{K}_u$ from data. When using DNN autoencoders to learn observation functions, the above optimization problem (7) must be solved with the optimization problem of state reconstruction:

$$\min_{\alpha} \mathcal{L}_{\text{recon}}, \tag{9}$$

where

$$\text{Reconstruction loss} : \mathcal{L}_{\text{recon}} = \sum_{k=0}^{N} \|\boldsymbol{x}_k - \psi_\alpha(\phi_\alpha(\boldsymbol{x}_k))\|^2. \tag{10}$$

Moreover, $\phi_\alpha(\boldsymbol{x})$, $\psi_\alpha(\boldsymbol{z})$ are respectively the DNN representations of the observation function and its inverse function, which can be called an encoder and a decoder.

The learning process of Koopman observation functions (7) and (9) adopts an autoencoder learning framework, which is an unsupervised learning process with only the state of the original $\boldsymbol{x}$-system as the supervisory signal and linear constraints (7) on the $\boldsymbol{z}$-system. There is no ground truth of the $\boldsymbol{z}$-system. Thus, the multivaluedness arises.

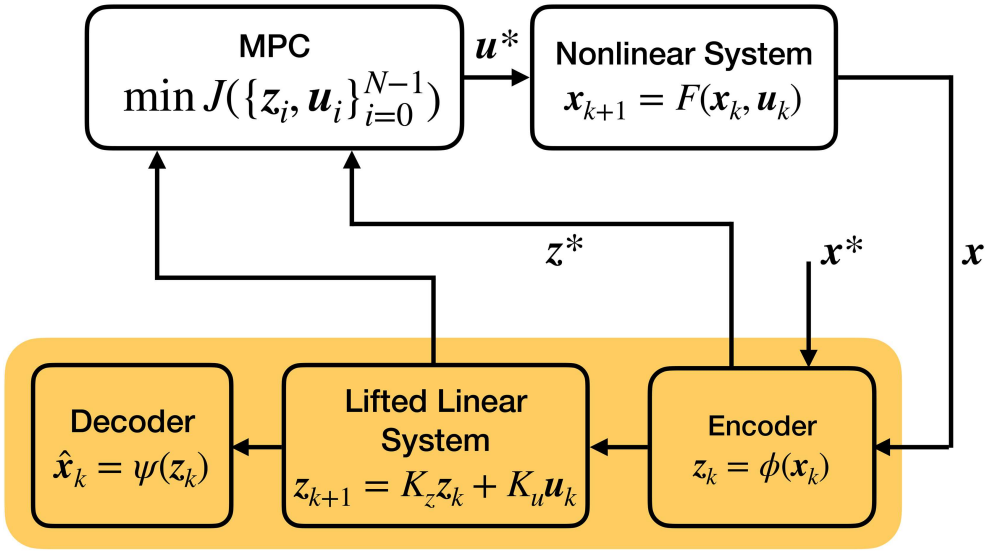

Figure 1 (Color online) Koopman-based control process.

2.3 Koopman-based control process

For the trajectory tracking problem, the nonlinear MPC design for the original nonlinear $\boldsymbol{x}$-system (1) can be reduced to solving the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{u}} \quad & J_x = \sum_{i=0}^{T} (\boldsymbol{x}_i - \boldsymbol{x}_i^*)^{\mathrm{T}} Q_x (\boldsymbol{x}_i - \boldsymbol{x}_i^*) + \sum_{i=0}^{T-1} \boldsymbol{u}_i^{\mathrm{T}} R \boldsymbol{u}_i \\ \text{s.t.} \quad & \boldsymbol{x}_{i+1} = F(\boldsymbol{x}_i, \boldsymbol{u}_i), \\ & \boldsymbol{x}(0) = \boldsymbol{x}_0, \end{aligned} \tag{11}$$

where $\boldsymbol{x}^*$ is the target trajectory, $Q_x > \boldsymbol{0}, R \geqslant \boldsymbol{0}$ are the weight matrices of the system state and control inputs.

The original nonlinear $\boldsymbol{x}$-system is mapped to a linear $\boldsymbol{z}$-system in the embedding space using the Koopman observation functions, thus transforming the nonlinear MPC problem Eq. (11) into the corresponding linear MPC problem:

$$\begin{aligned} \min_{\boldsymbol{z},\boldsymbol{u}} \quad & J_z = \sum_{i=0}^{T} (\boldsymbol{z}_i - \boldsymbol{z}_i^*)^{\mathrm{T}} Q_z (\boldsymbol{z}_i - \boldsymbol{z}_i^*) + \sum_{i=0}^{T-1} \boldsymbol{u}_i^{\mathrm{T}} R \boldsymbol{u}_i \\ \text{s.t.} \quad & \boldsymbol{z}_{i+1} = \mathcal{K}_z \boldsymbol{z}_k + \mathcal{K}_u \boldsymbol{u}_k, \\ & \boldsymbol{z}(0) = \phi(\boldsymbol{x}_0), \end{aligned} \tag{12}$$

where $\boldsymbol{z}^* = \phi(\boldsymbol{x}^*)$, and $Q_z > \boldsymbol{0}, R \geqslant \boldsymbol{0}$ are the weight matrices.

The conventional procedure for Koopman control [7,8,13,14] is to solve the linear MPC (12) and apply the generated control law directly to the original nonlinear $\boldsymbol{x}$-system, as shown in Figure 1. This requires the control law $\boldsymbol{u}_z^*$ generated by (12) to be consistent with the control law $\boldsymbol{u}_x^*$ generated by (11).

2.4 Lagrangian dynamic systems

The Euler-Lagrange equation is a common form of the equation used to describe the motion of a mechanical system. It uses generalized coordinates q to represent system configurations. In this paper, we focus on a class of Lagrangian systems with nonconservative forces $\boldsymbol{u}$:

$$\frac{d}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial L}{\partial \boldsymbol{q}} = \boldsymbol{u}, \tag{13}$$

where the Lagrangian and energy are usually expressed as

$$L = T - V, \quad E = T + V, \tag{14}$$

where $T(\dot{\boldsymbol{q}}, \boldsymbol{q}) = \frac{1}{2}\dot{\boldsymbol{q}}^{\mathrm{T}} H(\boldsymbol{q})\dot{\boldsymbol{q}}$ is the kinetic energy and $H(\boldsymbol{q}) : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is the symmetric and positive definite generalized inertia matrix. Moreover, $V(\boldsymbol{q})$ is the potential energy.

By subsituting (14) into (15), we can rewrite (15) in the following equivalent form:

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{u}(t), \tag{15}$$

where $C(\boldsymbol{q}, \dot{\boldsymbol{q}}) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is the Coriolis matrix and $\boldsymbol{g}(\boldsymbol{q}) : \mathbb{R}^n \to \mathbb{R}^n$ is the derivative of the potential energy V with respect to generalized coordinates q: $\boldsymbol{g}(\boldsymbol{q}) = \partial V(\boldsymbol{q})/\partial \boldsymbol{q}$. The ordinary differential equation can represent any robot system with holonomic constraints, such as coupled pendulums [30] and legged robots [31].

2.5 Problem statement

Problem 1 (Multistep trajectory prediction)**.** Given a trajectory $\mathcal{T} = \{\boldsymbol{x}_k, \boldsymbol{u}_k\}_{k=0}^T$ and a model $\mathcal{F}_{\boldsymbol{\alpha}}$, with the initial state $\boldsymbol{x}_0$ and control sequence $\{\boldsymbol{u}_k\}_{k=0}^T$, find the parameters $\boldsymbol{\alpha}$ that minimize

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \sum_{k=1}^{T} ||\boldsymbol{x}_k - \hat{\boldsymbol{x}}_k||_2 \\ &\text{s.t. } \hat{\boldsymbol{x}}_{k+1} = \mathcal{F}_{\boldsymbol{\alpha}}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k), \ \ \hat{\boldsymbol{x}}_0 = \boldsymbol{x}_0, \ \ \forall k = 1, \ldots, T. \end{aligned} \tag{16}$$

Problem 2 (Model-based controller design)**.** Given the initial state $\boldsymbol{x}_0$ and reference state trajectory $\mathcal{X}^* = \{\boldsymbol{x}_k^*\}_{k=0}^T$, the control law $\boldsymbol{u} = \boldsymbol{\pi}(\mathcal{F}_{\boldsymbol{\alpha}}, \boldsymbol{x}_0, \mathcal{X}^*)$ is designed on the basis of the learned model $\mathcal{F}_{\boldsymbol{\alpha}}$ such that $\limsup_{k\to\infty} ||\boldsymbol{x}_k - \boldsymbol{x}_k^*||_2$ is bounded.

3 EDM and control consistency

In this section, we first demonstrate the multivalued uncertainty of unconstrained Koopman representation and then provide the relationship between EDM constraints and control consistency between the Lagrangian system and the corresponding Koopman representation.

3.1 Multivalued uncertainty of the unconstrained Koopman representation

For a given set of state sequences $\{(\boldsymbol{x}_0, \boldsymbol{u}_0), (\boldsymbol{x}_1, \boldsymbol{u}_1), \ldots, (\boldsymbol{x}_N, \boldsymbol{u}_N)\}$, let us assume that $\{\phi_1, \psi_1, A_1, B_1\}$ are the optimal Koopman observables and Koopman matrix obtained on the basis of (7) and (9) that is, the optimal solution of the following optimization problem:

$$\begin{aligned} \phi^*, \psi^*, \mathcal{K}_x^*, \mathcal{K}_u^* =& \mathrm{argmin}_{\mathcal{K}_z, \mathcal{K}_u, \phi, \psi} \mathcal{L}_{\mathrm{Lin}} + \mathcal{L}_{\mathrm{Recon}} \\ =& \mathrm{argmin}_{\mathcal{K}_z, \mathcal{K}_u, \phi, \psi} \sum_{k=0}^{N-1} \left[\|\phi(\boldsymbol{x}_{k+1}) - \mathcal{K}_z\phi(\boldsymbol{x}_k) - \mathcal{K}_u\boldsymbol{u}_k\|^2 + \|\boldsymbol{x}_k - \psi(\phi(\boldsymbol{x}_k))\|^2 \right], \end{aligned} \tag{17}$$

where $\phi^* = \phi_1, \psi^* = \psi_1, \mathcal{K}_x^* = A_1, \mathcal{K}_u^* = B_1$. Let

$$J_1 = \mathcal{L}_{\mathrm{Lin},1} + \mathcal{L}_{\mathrm{Recon},1} = \sum_{k=0}^{N-1} \left[\|\phi_1(\boldsymbol{x}_{k+1}) - A_1\phi_1(\boldsymbol{x}_k) - B_1\boldsymbol{u}_k\|^2 + \|\boldsymbol{x}_k - \psi_1(\phi_1(\boldsymbol{x}_k))\|^2 \right]. \tag{18}$$

Theorem 1. There exists another set of Koopman representations $\{\phi_2, \psi_2, A_2, B_2\} \neq \{\phi_1, \psi_1, A_1, B_1\}$ such that $J_2 = J_1$. This means that the optimal solution for the optimization problem (17) is not unique.

Proof. We have

$$J_2 = \sum_{k=0}^{N-1} \left[\|\phi_2(\boldsymbol{x}_{k+1}) - A_2\phi_2(\boldsymbol{x}_k) - B_2\boldsymbol{u}_k\|^2 + \|\boldsymbol{x}_k - \psi_2(\phi_2(\boldsymbol{x}_k))\|^2 \right]. \tag{19}$$

For any $\psi_1(\phi_1(\boldsymbol{x}))$, we can construct $\psi_2(\phi_2(\boldsymbol{x})) = \psi_1(\phi_1(\boldsymbol{x}))$ such that

$$\|\boldsymbol{x}_k - \psi_2(\phi_2(\boldsymbol{x}_k))\|^2 = \|\boldsymbol{x}_k - \psi_1(\phi_1(\boldsymbol{x}_k))\|^2, \ \forall k = 0, 1, \ldots, N. \tag{20}$$

Thus, we only need to prove the existence of ϕ_2, A_2, B_2 so that the following equation about $\mathcal{L}_{\text{Lin}}$ holds to prove the theorem:

$$\mathcal{L}_{\text{Lin},2} = \sum_{k=0}^{N-1} \|\phi_2(\boldsymbol{x}_{k+1}) - A_2\phi_2(\boldsymbol{x}_k) - B_2\boldsymbol{u}_k\|^2 = \mathcal{L}_{\text{Lin},1} = \sum_{k=0}^{N-1} \|\phi_1(\boldsymbol{x}_{k+1}) - A_1\phi(\boldsymbol{x}_k) - B_1\boldsymbol{u}_k\|^2. \quad (21)$$

Let

$$\begin{aligned} P_k^2 &= \|\phi_1(\boldsymbol{x}_{k+1}) - A_1\phi(\boldsymbol{x}_k) - B_1\boldsymbol{u}_k\|^2, \\ Q_k^2 &= \|\phi_2(\boldsymbol{x}_{k+1}) - A_2\phi(\boldsymbol{x}_k) - B_2\boldsymbol{u}_k\|^2. \end{aligned} \quad (22)$$

Without loss of generality, let us assume A_1 as an invertible matrix and $P_0 \neq P_1$. If we let the following conditions hold:

$$\begin{cases} Q_0 = P_1, \\ Q_1 = P_0, \\ Q_k = P_k, \quad k = 2, 3, \ldots, N-1, \end{cases} \quad (23)$$

then we have

$$\begin{aligned} \mathcal{L}_{\text{Lin},2} &= \sum_{k=0}^{N-1} Q_k^2 = Q_0^2 + Q_1^2 + \sum_{k=2}^{N-1} Q_k^2 \\ &= P_1^2 + P_0^2 + \sum_{k=2}^{N-1} P_k^2 = \mathcal{L}_{\text{Lin},1}. \end{aligned} \quad (24)$$

That is, as long as we find $\{\phi_2, \psi_2, A_2, B_2\}$ that satisfies the condition (23), we can prove Theorem 1.

Let $\phi_2(\boldsymbol{x}) = \phi_1(\boldsymbol{x}) + g(\boldsymbol{x}), A_2 = A_1, B_2 = B_1$, where

$$g(\boldsymbol{x}_k) \begin{cases} \neq 0, & \text{if } k \in 0, 1, \\ = 0, & \text{if } k \in 2, 3, \ldots, N-1. \end{cases} \quad (25)$$

Apparently, the condition (23) corresponds to the following equations:

$$\begin{cases} \phi_1(\boldsymbol{x}_1) + g(\boldsymbol{x}_1) - A_1[\phi_1(\boldsymbol{x}_0) + g(\boldsymbol{x}_0)] - B_1\boldsymbol{u}_0 = \phi_1(\boldsymbol{x}_2) - A_1[\phi_1(\boldsymbol{x}_1)] - B_1\boldsymbol{u}_1, \\ \phi_1(\boldsymbol{x}_2) - A_1[\phi_1(\boldsymbol{x}_1) + g(\boldsymbol{x}_1)] - B_1\boldsymbol{u}_1 = \phi_1(\boldsymbol{x}_1) - A_1[\phi_1(\boldsymbol{x}_0)] - B_1\boldsymbol{u}_0. \end{cases} \quad (26)$$

Solving the above equations, we get

$$\begin{cases} g(\boldsymbol{x}_0) = A_1^{-1}(A_1^{-1} - I)H, \\ g(\boldsymbol{x}_1) = A_1^{-1}H, \end{cases} \quad (27)$$

where

$$H = (\phi_1(\boldsymbol{x}_2) - \phi_1(\boldsymbol{x}_1)) - A_1(\phi_1(\boldsymbol{x}_1) - \phi_1(\boldsymbol{x}_0)) - B_1(\boldsymbol{u}_1 - \boldsymbol{u}_0). \quad (28)$$

Herein, I is the identity matrix. Eq. (27) implies the existence of a function $g(\boldsymbol{x})$ that satisfies the Koopman representation condition (23).

Finally, there exists a Koopman representation $\{\phi_2, \psi_2, A_2, B_2\} \neq \{\phi_1, \psi_1, A_1, B_1\}$ such that $J_2 = J_1$, thus proving that the optimal solution of (17) is not unique.

Theorem 1 shows that in the absence of constraints on the Koopman representation, multiple different Koopman observation functions and Koopman matrices $\{\phi_1, \psi_1, A_1, B_1\}, \{\phi_2, \psi_2, A_2, B_2\}$ can be generated for the same state $\boldsymbol{x}_k$, all of which are optimal solutions to the optimization problem (17). Solving MPC (12) for $\phi_1(\boldsymbol{x})$ and $\phi_2(\boldsymbol{x})$, respectively, the obtained $\boldsymbol{u}_{\phi 1}^*$ is not necessarily equal to $\boldsymbol{u}_{\phi 2}^*$, indicating that the unconstrained Koopman embedding representation is multivalued. This makes it challenging to guarantee that the control law generated by (11) is the same as that generated by (12).

3.2 EDM constraints

In Subsection 2.1, we introduced a form of Koopman representation. As previously mentioned, to apply control laws generated by the Koopman system directly to the original system, they must satisfy the control consistency constraints. However, because of the multivalued nature of the unconstrained DAE learning of Koopman representations, as proven in Subsection 3.1, it cannot guarantee control consistency between the two systems.

Consider an $\boldsymbol{x}$-system with an initial state $\boldsymbol{x}_0$, a target state $\boldsymbol{x}_T$, an initial system energy $E(\boldsymbol{x}_0)$, and a target system energy $E(\boldsymbol{x}_T)$. Energy shaping control theory suggests that the system and controller can be viewed as energy converters, where the control process of transforming the $\boldsymbol{x}$-system from its initial state $\boldsymbol{x}_0$ to the target state $\boldsymbol{x}_T$ is equivalent to transferring the system energy from the initial value $E(\boldsymbol{x}_0)$ to the target value $E(\boldsymbol{x}_T)$ through control input energy $E(\boldsymbol{u})$.

For a passive system with negligible dissipation, such as a Lagrangian system, the energy transfer process satisfies the following relationship:

$$E(\boldsymbol{x}_T) = E(\boldsymbol{x}_0) + E(\boldsymbol{u}_x), \tag{29}$$

where $\boldsymbol{u}_x$ is the control input that drives the system state from $\boldsymbol{x}_0$ to $\boldsymbol{x}_T$.

We notice that we can ensure the following EDM constraint between the $\boldsymbol{x}$-system and its corresponding Koopman representation $\boldsymbol{z}$-system:

$$E(\boldsymbol{z}_{k+1}) - E(\boldsymbol{z}_k) = E(\boldsymbol{x}_{k+1}) - E(\boldsymbol{x}_k) \quad \forall \boldsymbol{z}_k = \phi(\boldsymbol{x}_k),\ k \in \{0, \ldots, T-1\}. \tag{30}$$

Therefore,

$$E(\boldsymbol{u}_x) = E(\boldsymbol{u}_z). \tag{31}$$

This means that the control sequence that is energy optimal for the $\boldsymbol{z}$-system must also be energy optimal for the $\boldsymbol{x}$-system. Thus, a new control paradigm for nonlinear systems can be constructed. First, design the Koopman linear $\boldsymbol{z}$-system satisfying the EDM constraints, and then design the control law in the $\boldsymbol{z}$-system and impose it directly on the original nonlinear system. This paradigm avoids the complex problem of designing control laws for nonlinear systems.

4 EPDLK operator

In this section, we discuss how to construct a Koopman representation that satisfies EDM constraints in a data-driven manner with only the Lagrangian system state time-series data known. Two major steps are required to achieve the above task: (1) identification of the original system energy function $E(x)$ and (2) solving DAE neural network optimization problems with hard constraints.

The typical learning process for deep Koopman representation involves constructing a Koopman observation function encoder ϕ and decoder ψ using DNNs such as MLP and CNN, along with a loss function $\mathscr{L}$. This process entails batch-inputting state data samples $\boldsymbol{x}$ and $\boldsymbol{u}$, optimizing parameters to minimize $\mathscr{L}$. To construct a deep Koopman operator that satisfies the EDM constraint, according to (10), (8), and (30), the following optimization problem must be solved:

$$\begin{aligned} \min_{\phi,\psi,\mathcal{K}_z,\mathcal{K}_u} \quad & \mathcal{L}_{\text{Recon}} + \mathcal{L}_{\text{Lin}} \\ \text{s.t.} \quad & \Delta E(\boldsymbol{z}_k) = \Delta E(\boldsymbol{x}_k) \quad \forall \boldsymbol{z}_k = \phi(\boldsymbol{x}_k),\ k \in \{0, \ldots, T-1\}. \end{aligned} \tag{32}$$

However, conventional DNNs have difficulty learning Lagrangian systems' energy function $E(\boldsymbol{x})$. Additionally, to satisfy the EDM constraint for Koopman representation, we must optimize DNNs with hard constraints.

Because the energy function $E(\boldsymbol{z})$ of the $\boldsymbol{z}$-system can be expressed as a quadratic form of $\boldsymbol{z}$, that is, $E(\boldsymbol{z}) = \boldsymbol{z}^{\mathrm{T}}\boldsymbol{z}$, and $\Delta E(\boldsymbol{x})$ can be calculated using (14), the EDM constraint can be expressed as

$$(\mathcal{K}_z\boldsymbol{z} + \mathcal{K}_u\boldsymbol{u})^{\mathrm{T}}(\mathcal{K}_z\boldsymbol{z} + \mathcal{K}_u\boldsymbol{u}) - \boldsymbol{z}^{\mathrm{T}}\boldsymbol{z} - \Delta E(\boldsymbol{x}) = 0. \tag{33}$$

$\mathcal{L}_{\text{recon}}$ is the reconstruction loss, used to guide the learning of Koopman observable functions ϕ_α and ψ_α, which are represented using DNNs with highly nonlinear features. Once ϕ_α, ψ_α have been learned, the optimization of $\mathcal{L}_{\text{Lin}}$ only needs to learn $\mathcal{K}_{\text{z}}$ and $\mathcal{K}_{\text{u}}$, which is a linear quadratic optimization problem.

Therefore, we can decouple the learning of the Koopman representation into two stages:

Stage 1: Koopman observable function learning

$$\min_{\phi,\psi} \quad \mathcal{L}_{\text{Recon}}; \tag{34}$$

Stage 2: Koopman matrix learning

$$\begin{aligned} \min_{\mathcal{K}_z,\mathcal{K}_u} \quad & \mathcal{L}_{\text{Lin}} \\ \text{s.t.} \quad & \text{Eq. (33)}. \end{aligned} \tag{35}$$

We propose the DLK network based on the system's physical structure during the first stage of learning the Koopman observable functions. This network achieves structure-preserving Koopman representation by introducing Lagrangian structural subnetworks and physical constraint loss functions while identifying Lagrangian system parameters during Koopman observable function learning. In the second stage of learning the Koopman matrices, we introduce an EDM constraint, transforming the Koopman matrix solution problem into a quadratic optimization problem with quadratic equality constraints. This is to ensure that the Koopman operator satisfies the EDM constraint without solving complex neural network optimization problems with hard constraints.

Next, we introduce the DLK network and present the Koopman matrix learning algorithm with EDM constraints.

4.1 DLK network

In Koopman observation function construction, DNNs [11, 13, 29] can discover more objective observable functions adaptively compared with handcrafted polynomial basis functions or RBFs with randomly selected weights. However, conventional neural network structures are insufficient for learning physical structures [32], such as the positive-definiteness of the inertia matrix in Lagrangian systems. This section thus proposes a physics-informed DLK network that introduces Lagrangian subnetworks and physics-informed loss functions. The DLK network achieves structure-preserving Koopman observable functions by constraining the learned Koopman embedding space state to the physically feasible solution space. The DLK network is accomplished while simultaneously training the Koopman observation function and identifying the Lagrangian system parameters.

As depicted in Figure 2, the DLK network consists of two subnetworks: an encoder and a decoder. The input of the encoder subnetwork is the original system state $\boldsymbol{x} = [\boldsymbol{q}, \dot{\boldsymbol{q}}]^{\mathrm{T}}$, whereas the output is the Koopman state $\boldsymbol{z} = \phi(\boldsymbol{x})$. Meanwhile, the input of the decoder subnetwork is the Koopman observation function $\phi(\boldsymbol{x})$, whereas the output is the reconstructed value of the original system state $\boldsymbol{x}$, i.e., $\hat{\boldsymbol{x}} = \psi(\boldsymbol{z})$. Besides the conventional reconstruction loss function, we introduce a physics-informed loss function that ensures that the Koopman encoder learns the correct Lagrangian physical structure.

4.1.1 *Structural parameter subnetworks*

By embedding the physical structure into the DNNs and modularizing the learning of $V(\boldsymbol{q}), H(\boldsymbol{q}), C(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the nonlinear Lagrangian system energy representation is reconstructed from the data.

The proposed encoder is shown in Figure 2. In the Koopman Encoder, we first use a feedforward network to learn the input state data features and represent $V(\boldsymbol{q}), H(\boldsymbol{q}), C(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in the feature space as feedforward networks. Finally, these middle-layer features and the original state $\boldsymbol{x} = \{\boldsymbol{q}, \dot{\boldsymbol{q}}\}$ are input into a fusion network to obtain the Koopman embedding space states $\boldsymbol{z}$. The feature representation network, the network for learning $C(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and the fusion network are similar full-connection network structures, called rectified linear unit (ReLU) networks, with a linear output layer and other hidden layers with ReLU as the activation function.

The Cholesky decomposition, which represents $H(\boldsymbol{q})$ as the product of a lower-triangular matrix, is used to embed the symmetrical positive-definite inertia matrix in the original nonlinear system Lagrangian equations into the network structure:

$$\begin{aligned} H(\boldsymbol{q}) &= L_\alpha(\boldsymbol{q}) L_\alpha^{\mathrm{T}}(\boldsymbol{q}), \\ L_\alpha &= L_d + L_o. \end{aligned} \tag{36}$$

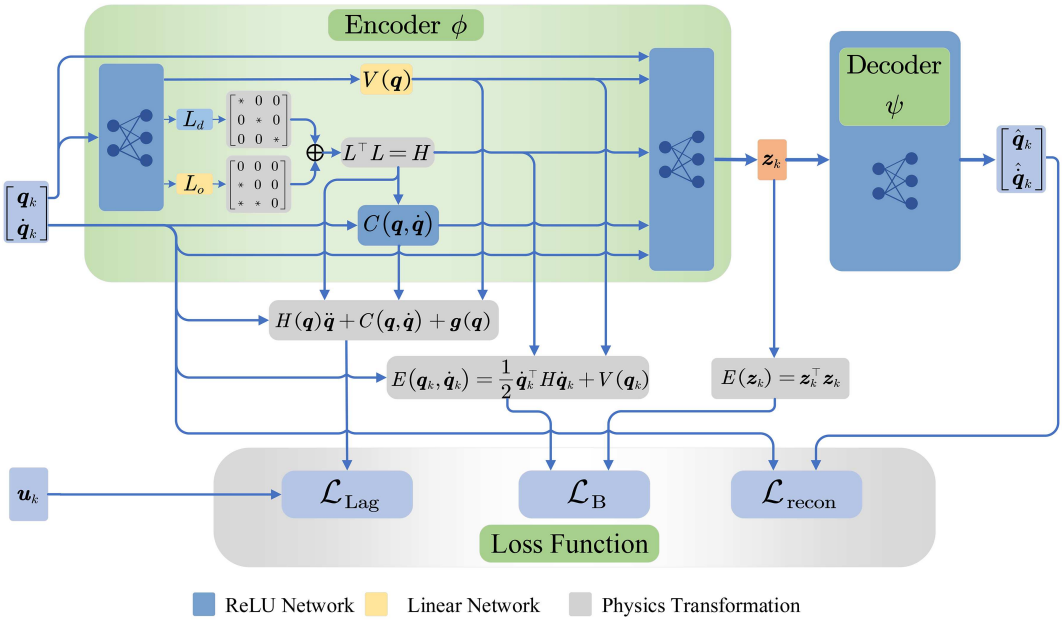

Figure 2 (Color online) DLK network.

A linear network represents the off-diagonal elements L_o in the L_α matrix. In contrast, the ReLU network represents the diagonal elements L_d without the linear output layer to ensure that all diagonal elements of L_α are positive and $H_\alpha(\boldsymbol{q}) > 0$.

In contrast to the Koopman encoder ϕ, the Koopman decoder ψ learns the mapping from the Koopman embedding space to the original nonlinear system space. This process is supervised learning with the original nonlinear system state as the label, which can be well represented using a regular neural network. Here, we use a five-layer ReLU network as a decoder.

4.1.2 *Physics-informed loss function*

Lagrangian structure parameter loss function. This loss function ensures that the DLK framework learns the correct structural parameters. The time series of state $\dot{\boldsymbol{q}}_k$ in the training trajectory samples are numerically differentiated to obtain $\ddot{\boldsymbol{q}}_k$. At the same time, the derivative of the potential energy $V(\boldsymbol{q}_k)$ concerning the input state $\boldsymbol{q}_k$, that is, the gravity $\boldsymbol{g}(\boldsymbol{q}_k) = \partial V(\boldsymbol{q}_k)/\partial \boldsymbol{q}_k$, can be easily calculated with the help of modern automatic differentiation software frameworks, such as Pytorch [33]. According to (15), the Lagrangian structural parameter loss function is constructed as

$$\text{Lagrangian loss}: \quad \mathcal{L}_{\text{Lag}} = H_\alpha(\boldsymbol{q}_k)\ddot{\boldsymbol{q}}_k + C_\alpha(\boldsymbol{q}_k,\dot{\boldsymbol{q}}_k)\dot{\boldsymbol{q}}_k + \boldsymbol{g}(\boldsymbol{q}_k) - \boldsymbol{u}_k. \tag{37}$$

Differentiable barrier loss function. The feasible set of Koopman matrix learning is nonempty if and only if $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z} + \Delta E(\boldsymbol{x}) > 0$ (see Subsection 4.2 for the proof). In neural network optimization problems featuring inequality constraints, the barrier function is commonly leveraged to convert such constraints into an objective function. In so doing, the original optimization problem can be rendered unconstrained. Notably, the barrier function exhibits rapid growth when the associated constraint is breached while remaining at zero in cases where the said constraint is upheld. To guarantee that the inequality constraint is obtained, a differentiable barrier loss function is crafted:

$$\text{Barrier loss}: \quad \mathcal{L}_{\text{B}} = \min\{0, \boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{z}_k + \Delta E(\boldsymbol{x}_k) - \delta\}, \tag{38}$$

where $\Delta E(\boldsymbol{x}_k)$ is the energy difference between the states $\boldsymbol{x}_{k+1}$ and $\boldsymbol{x}_k$ and $\delta > 0$ is a small positive value used to avoid the case where the value of $\boldsymbol{z}_k^{\mathrm{T}}\boldsymbol{z}_k + \Delta E(\boldsymbol{x}_k)$ is too close to 0.

Combining the loss function in (10) and the physics-informed loss function proposed in (37) and (38), then, by solving the following optimization problem in the Stage 1 and training the DLK neural network, physics-informed Koopman observable functions are learned:

$$\min_{\phi,\psi} \quad \mathcal{L}_{\text{ob}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{Lag}} + \mathcal{L}_{\text{B}}. \tag{39}$$

Algorithm 1 Learning algorithm of the Koopman observable functions

Input: Trajectory dataset $\mathcal{D} = \{\{\boldsymbol{x}_k^n, \boldsymbol{u}_k^n\}_{k=0}^T\}_{n=0}^N$, the encoder network ϕ, and the decoder network ψ constructed in Subsection 4.1.1, with the maximum number of epochs $\mathrm{Ep}_{\max}$;
Initialize network weights of ϕ, ψ, Epoch = 0, batch size bs, and training termination error $\epsilon > 0$;
while Epoch $< \mathrm{Ep}_{\max}$ and $|\mathcal{L}_{\mathrm{ob}}| \geqslant \epsilon$ **do**
  Reset the training episodes;
  while the current epoch is not terminated **do**
    Sample a batch of state and control sequences $X_k = \{\boldsymbol{x}_k, \ldots, \boldsymbol{x}_{k+bs}\}, U_k = \{\boldsymbol{u}_k, \ldots, \boldsymbol{u}_{k+bs}\}$ from the dataset $\mathcal{D}$;
    Calculate the Koopman space state Z_k corresponding to X_k: $Z_k = \phi(X_k)$;
    Calculate the Koopman space state Z_{k+1} corresponding to X_{k+1}: $Z_{k+1} = \phi(X_{k+1})$;
    Obtain the reconstruction states with decoder ψ: $\hat{X}_k = \psi(Z_k)$, $\hat{X}_{k+1} = \psi(Z_{k+1})$;
    Obtian $\mathcal{L}_{\mathrm{ob}}$ according to (10), (37), (38), and (39);
    Solve the optimization problem (39) with an Adam optimizer to update the parameters of ϕ, ψ;
  end while
end while
Output: ϕ, ψ.

The Adam optimization algorithm [34] is used to solve the above optimization problem. The Koopman operator learning algorithm is shown in Algorithm 1.

4.2 Koopman matrix learning with EDM hard constraints

When the Koopman observable functions ϕ, ψ are learned, we fix their parameters for the Koopman matrix learning.

In the case of fixed parameters of ϕ, ψ, $\boldsymbol{z}$, $\boldsymbol{z}_+$ in (35) is known. Therefore, the original optimization problem with $\boldsymbol{z}$, $\boldsymbol{z}_+$, $\mathcal{K}_x$, $\mathcal{K}_u$ as variables is transformed into a quadratic optimization problem related to the Koopman matrices $\mathcal{K}_x$, $\mathcal{K}_u$:

$$\begin{aligned} \min \quad & \mathcal{L}_{\mathrm{Lin}}(\mathcal{K}_z, \mathcal{K}_u) = ||\boldsymbol{z}_+ - \mathcal{K}_z\boldsymbol{z} - \mathcal{K}_u\boldsymbol{u}||_2^2 \\ \text{s.t.} \quad & (\mathcal{K}_z\boldsymbol{z} + \mathcal{K}_u\boldsymbol{u})^{\mathrm{T}}(\mathcal{K}_z\boldsymbol{z} + \mathcal{K}_u\boldsymbol{u}) - \boldsymbol{z}^{\mathrm{T}}\boldsymbol{z} - \Delta E(\boldsymbol{z}) = 0. \end{aligned} \tag{40}$$

Let

$$S = \boldsymbol{z}_+ - \mathcal{K}_z\boldsymbol{z} - \mathcal{K}_u\boldsymbol{u}. \tag{41}$$

Eq. (40) can be rewritten as

$$\begin{aligned} \min \quad & \mathcal{L}_{\mathrm{Lin}} = S^{\mathrm{T}}S \\ \text{s.t.} \quad & S^{\mathrm{T}}S - 2\boldsymbol{z}_+^{\mathrm{T}}S + C = 0, \end{aligned} \tag{42}$$

where

$$C = \boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ - \boldsymbol{z}^{\mathrm{T}}\boldsymbol{z} - \Delta E(\boldsymbol{x}). \tag{43}$$

This optimization problem is a quadratic optimization problem with quadratic constraints. Because of $\boldsymbol{z}_+ \neq \boldsymbol{0}$, the problem has nontrivial solutions.

4.2.1 *Global minimum*

We define the Lagrangian function of (42) as

$$L(S, \lambda) = S^{\mathrm{T}}S - \lambda(S^{\mathrm{T}}S - 2\boldsymbol{z}_+^{\mathrm{T}}S + C). \tag{44}$$

An optimal solution of (42) must satisfy the following KKT conditions:

$$\begin{aligned} &\frac{\partial L}{\partial S} = S - \lambda S + \lambda\boldsymbol{z}_+ = 0, \\ &\frac{\partial L}{\partial \lambda} = S^{\mathrm{T}}S - 2\boldsymbol{z}_+^{\mathrm{T}}S + C = 0. \end{aligned} \tag{45}$$

According to [35], the global minimum of (42) must also satisfy the second-order necessary condition:

$$\lambda \leqslant 1. \tag{46}$$

Algorithm 2 Koopman matrix learning

Input: Dataset of the original $\boldsymbol{x}$-system state trajectories $\mathscr{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t\}$, control input trajectories $\mathscr{U} = \{\boldsymbol{u}_1, , \boldsymbol{u}_2, \ldots, \boldsymbol{u}_{t-1}\}$, and the encoder network ϕ and decoder network ψ learned in Algorithm 1;

For each sample state $\boldsymbol{x}$ in $\mathscr{X}$, compute its corresponding Koopman states and let $\mathscr{Z} = \{\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{t-1}\}$ and $\mathscr{Z}_+ = \{\boldsymbol{z}_2, \boldsymbol{z}_3, \ldots, \boldsymbol{z}_t\}$;

Compute the energy difference sequence $\Delta E_x = \{\Delta E(\boldsymbol{x}_1), \ldots, \Delta E(\boldsymbol{x}_{t-1})\}$ of the $\boldsymbol{x}$-system based on (14);

Substitute $\mathscr{Z}, \mathscr{Z}_+, \Delta E_x$ into (43), (49), and (51); obtain $\mathscr{S}^* = \{S_1^*, S_2^*, \ldots, S_{t-1}^*\}$;

Let $\boldsymbol{\eta}^{(n+m)\times(t-1)} = \begin{bmatrix}\mathscr{Z} \\ \mathscr{U}\end{bmatrix}$;

Calculate the generalized inverse solution for the Koopman matrices:

$$[\mathcal{K}_x, \mathcal{K}_u] = (\mathscr{Z}_+ - \mathscr{S}^*)\boldsymbol{\eta}^{\mathrm{T}}(\boldsymbol{\eta}\boldsymbol{\eta}^{\mathrm{T}})^{-1};$$

Output: $\mathcal{K}_x, \mathcal{K}_u$.

Algorithm 3 Deep Koopman MPC

Input: Reference trajectory $(X_{\mathrm{ref}}, U_{\mathrm{ref}})$, time window T.

while the termination status is not reached **do**
  Obtain the current state $\boldsymbol{x}_k$;
  Obtain the Koopman state of $\boldsymbol{x}_k$: $\boldsymbol{z}_0 = \phi(\boldsymbol{x}_k)$;
  Solve (52) to get the control sequence $\{\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{T-1}\}$;
  Deploy first control input $\boldsymbol{u}_0$ to the plant;
end while

From (45), we have

$$S = \frac{-\lambda}{1-\lambda}\boldsymbol{z}_+, \tag{47}$$

and

$$\frac{1}{(1-\lambda)^2}\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ - (\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ - C) = 0. \tag{48}$$

When $\lambda \neq 1$, we have

$$\begin{aligned}\lambda_1 &= 1 + \sqrt{(\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+)/(\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ - C)}, \\ \lambda_2 &= 1 - \sqrt{(\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+)/(\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ - C)}.\end{aligned} \tag{49}$$

According to (43), we can get

$$\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ - C = \boldsymbol{z}^{\mathrm{T}}\boldsymbol{z} + \Delta E(\boldsymbol{x}). \tag{50}$$

Because of $\boldsymbol{z}_+^{\mathrm{T}}\boldsymbol{z}_+ \geqslant 0$, the feasible set of (49) is nonempty if and only if $\boldsymbol{z}^{\mathrm{T}}\boldsymbol{z} + \Delta E(\boldsymbol{x}) > 0$. Through the loss function $\mathcal{L}_B$ constructed in Subsection 4.1.2, we can ensure that the inequality holds. According to (46), the feasible solution is $\lambda = \lambda_2$, which means that the global minimum of (42) is

$$S^* = \frac{-\lambda_2}{1-\lambda_2}\boldsymbol{z}_+. \tag{51}$$

4.2.2 *Learning Koopman matrices*

In the previous section, we obtained the optimal solution of S for a single sample pair $\{\boldsymbol{x}, \boldsymbol{x}_+, \boldsymbol{z}, \boldsymbol{z}_+\}$. The Koopman matrices $\mathcal{K}_z, \mathcal{K}_u$ can be learned by using many sample pairs from the dataset of dynamic system states. The specific process is described in Algorithm 2.

In summary, this two-stage learning algorithm constructs an energy-preserving deep Koopman representation of the Lagrangian system. The learned Koopman observable functions and Koopman matrices can be used for tasks such as nonlinear system state prediction and controller design.

5 Deep Koopman control

This section explores how to combine EPDLK with MPC methods to transform the original nonlinear system control problem into a linear MPC problem in the Koopman linear space state (see Algorithm 3). After obtaining the energy-preserving Koopman linear $\boldsymbol{z}$-system, we can design the MPC in the $\boldsymbol{z}$-system and impose the generated control law directly on the original nonlinear $\boldsymbol{x}$-system. MPC aims to synthesize the control sequence to minimize the objective function in a time window T. The EPDLK framework

maps the original nonlinear system into a linear dynamical system, and the controller design of the original nonlinear system can be transformed into the following convex quadratic programming problem, thus reducing the difficulty of the solution:

$$\begin{aligned} \min_{\boldsymbol{z},\boldsymbol{u}} \quad & \sum_{i=0}^{T} \left(\boldsymbol{z}_i - \boldsymbol{z}_i^*\right)^{\mathrm{T}} Q \left(\boldsymbol{z}_i - \boldsymbol{z}_i^*\right) + \sum_{i=0}^{T-1} \boldsymbol{u}_i^{\mathrm{T}} R \boldsymbol{u}_i \\ \text{s.t.} \quad & \boldsymbol{z}_{i+1} = \mathcal{K}_z \boldsymbol{z}_i + \mathcal{K}_u \boldsymbol{u}_i, \\ & \boldsymbol{z}_i \in [\boldsymbol{z}_{\min}, \boldsymbol{z}_{\max}], \quad \boldsymbol{u}_i \in [\boldsymbol{u}_{\min}, \boldsymbol{u}_{\max}], \\ & \boldsymbol{z}_0 = \phi(\boldsymbol{x}_k), \quad \boldsymbol{z}_i^* = \phi(\boldsymbol{x}_{k+i}^*), \\ & \boldsymbol{z}_{\min} = \phi(\boldsymbol{x}_{\min}), \quad \boldsymbol{z}_{\max} = \phi(\boldsymbol{x}_{\max}), \end{aligned} \tag{52}$$

where $Q \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{n\times n}$ are positive semidefinite cost matrices. The encoder ϕ maps the desired trajectory $\boldsymbol{x}^*$ and state bounds $\boldsymbol{x}_{\min}, \boldsymbol{x}_{\max}$ in the original nonlinear space to the corresponding values $\boldsymbol{z}^*, \boldsymbol{z}_{\min}, \boldsymbol{z}_{\max}$ in the Koopman embedding space.

For the reference trajectory tracking task, at the kth step, with $\boldsymbol{z}_0 = \phi(\boldsymbol{x}_k)$ as the initial state, solve (52) within a time window T to obtain the control sequence $\{\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots, \boldsymbol{u}_{T-1}\}$. Then, the first control command $\boldsymbol{u}_0$ is used directly as input to the original plant.

6 Experimental results

6.1 Experimental setup

6.1.1 *Simulation plant*

To validate the performance of the proposed framework, we studied the cart-pole system, a typical system described by Lagrangian dynamics:

$$\begin{bmatrix} m_c + m_p & m_p l\cos\theta \\ m_p l\cos\theta & m_p l^2 \end{bmatrix} \begin{bmatrix} \ddot{\xi} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} 0 & -m_p l\sin\theta \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\xi} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ -m_p l\sin\theta \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tau, \tag{53}$$

where ξ is the horizontal position of the cart, θ is the angle between the pole and the vertical direction, and m_c, m_p are the cart's and pole's masses, respectively. l is the length of the pole, g is the acceleration due to gravity, and τ is the horizontal force input to the cart.

Unlike the conventional system identification method for estimating the model parameters m_c, m_p, l, our method does not learn the structural parameters of a specific system but learns the Lagrangian dynamical system representation. For the cart-pole system with unknown model parameters, the control state data set $\mathcal{D} = \{\xi_k, \theta_k, \dot{\xi}_k, \dot{\theta}_k, \tau_k\}_{k=0}^{N}$ is obtained by inputting the random control sequence, and it is used for Koopman embedding representation learning. The linear model obtained by learning is combined with the MPC method to design the controller. Its control task is to move the cart-pole from the initial state at a certain distance from the origin to the origin.

6.1.2 *Data generation*

To construct the dataset $\mathcal{D}$, we first sample initial states in the interval $(\xi, \theta, \dot{\xi}, \dot{\theta}) \in [-3, 3] \times [-0.2, 0.2] \times [-0.1, 0.1] \times [-0.1, 0.1]$. By inputting a random control sequence, 300 state trajectory datasets with a duration of 2 s are generated, where 240 trajectories are used as training sets and the remaining 60 trajectories are used as multistep prediction test sets. The sampling frequency of trajectory data is 100 Hz. With the generated dataset, the EPDLK framework is constructed using the algorithm described in Section 4, and the Koopman linear model is learned. The neural structure parameters of the DLK encoder and decoder used in this experiment are shown in Table 1.

6.1.3 *Baselines*

We compare the proposed method with the (1) local linearization model (Linear), (2) handcrafted Edmd-poly [23] model with polynomial basis function as observation functions, (3) handcrafted Edmd-rbf [7] model with RBF as observation functions and the SOTA of Koopman observation function learning

Table 1 Neural structure parameters of DLK

Network structures		Node number	Activation function
Encoder	Feature network	$\{2, 16, 16\}$	ReLU
	V	$\{16, 2\}$	–
	L_d	$\{16, 8, 2\}$	ReLU
	L_o	$\{16, 1\}$	–
	$C(q, \dot{q})$	$\{6, 8, 4\}$	ReLU
	Fusion network	$\{13, 16, 16\}$	ReLU
Decoder		$\{16, 16, 16, 4\}$	ReLU

Table 2 Comparison on time complexity and parameter numbers

	Linear	Edmd-poly	Edmd-rbf	Hermite	MLP	ELU-NN	DCNN	EPDLK
Time complexity	$O(n^2)$	$O(nv)$	$O(nv)$	$O(nv)$	$O(Lv^2)$	$O(Lv^2)$	$O(LvkC^2)$	$O(v^2)$
Number of parameters	16	64	64	128	1664	1664	1709	1024

methods, (4) Hermite polynomials used in ACD-EDMD [24], (5) deep MLP [12], (6) ELU-NN [25], and (7) deep CNN [26].

The time complexity of the above methods is shown in Table 2. Therein, n is the dimension of $\boldsymbol{x}$-system states, v is the $\boldsymbol{z}$-system state dimension, L is the number of neural network layers, k is the convolutional kernel width, and C is the number of channels of deep CNN. In each time step, the time complexity of the MPC optimization problem (52) is $O(Nv^3)$, which shows that the most time-consuming part is the MPC optimization in the Koopman-based controller design process.

For comparison convenience and considering the Markovian nature of the Lagrangian system, the input state-control sequence length for the Koopman encoder in this study is uniformly set to 1. Additionally, through experimental validation, we uniformly set the output state dimension of the Koopman encoder to 16. The number of neurons in each layer of the deep MLP neural network encoder is $\{4, 32, 32, 16\}$ with the ReLU activation function. The number of neurons in each layer of ELU-NN is the same as that of deep MLP, but the dropout mechanism is introduced in the training session, and the ELU-type activation function is used. In the DCNN method, a three-layer CNN is constructed using a padding convolution kernel with shape $3 * 1$. The number of parameters for each method is shown in Table 2. In the local linearization model, the structural parameters of the cart-pole system are known. The Edmd-poly, Edmd-rbf, and Hermite methods are implemented using sci-kit learn, whereas the neural networks in deep MLP, ELU-NN, DCNN, and EPDLK are implemented using PyTorch [33].

6.1.4 *Ablation setup*

To evaluate the model representation capability of the proposed EPDLK framework, we constructed the degraded EPDLK framework EPDLK-NPI (EPDLK without physics information). By replacing the Lagrangian structural subnetworks in DLK with neural network layers with the same number of parameters, removing the loss function terms $\mathcal{L}_{\text{Lag}}$ and $\mathcal{L}_{\text{B}}$ from the loss function $\mathcal{L}_{\text{ob}}$, EPDLK is degraded to a conventional deep MLP. By comparing EPDLK with EPDLK-NPI, we demonstrate the necessity of introducing the physical information structure.

Meanwhile, to evaluate the data utilization efficiency of the EPDLK framework, we randomly selected 1/6 of the original trajectory training dataset (i.e., 40 trajectory data points) to train the EPDLK model and called the trained model EPDLK-FD (fewer data).

6.2 Multistep trajectory prediction

First, we compare the multistep prediction performance of various methods on the test set. Under the condition that only the initial state $\boldsymbol{x}_0 = [\xi_0, \theta_0, \dot{\xi}_0, \dot{\theta}_0]$ and the control sequence $\boldsymbol{u} = \{\tau_0, \ldots, \tau_{T-1}\}$ are given, we predict forward the system state value within 200 time steps (corresponding to the real-time value of 2 s).

Figure 3 shows the multistep prediction error distribution of the EPDLK method and other benchmark methods on the 60 trajectories in the test set. Figures 3(a) and (c) show semilogarithmic graphs. Apparently, the local linearization, Edmd-poly, and Hermite methods can achieve relatively accurate predictions in the initial period. However, as the number of prediction steps increases, the error increases

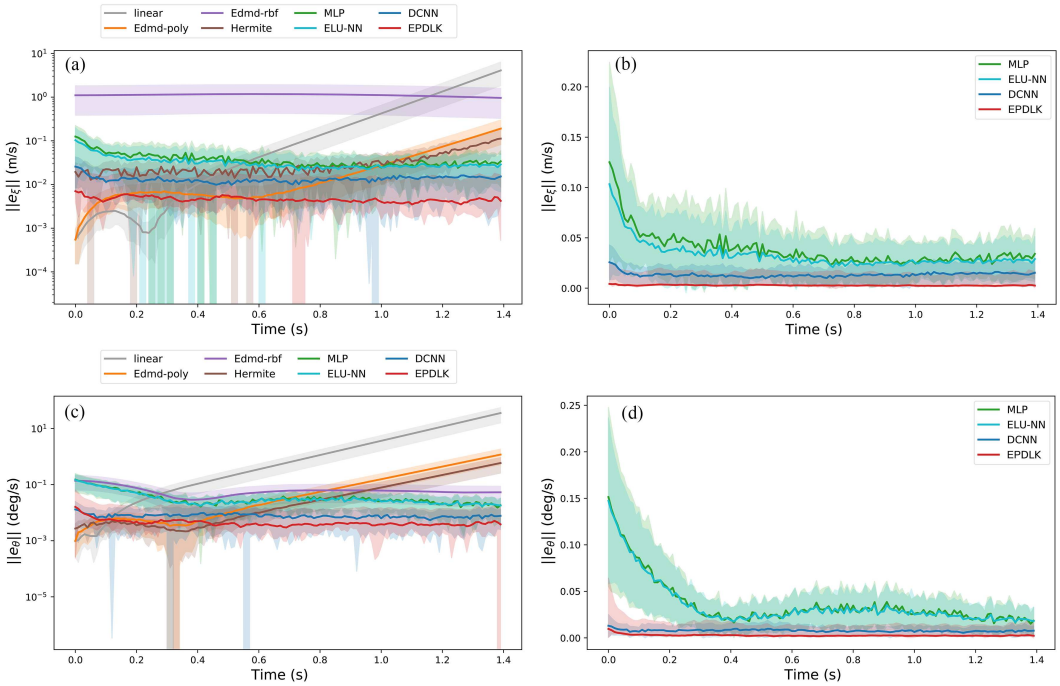

Figure 3 (Color online) Multistep prediction error distribution of the cart-pole system state horizontal position ξ (top) and angle θ (bottom). The solid line is the mean value of the error, whereas the color interval represents the error distribution. (a) Semilogarithmic graphs of the distribution of $\|e_\xi\|$; (b) distribution of $\|e_\xi\|$ based on DNN; (c) semilogarithmic graphs of the distribution of $\|e_\theta\|$; (d) distribution of $\|e_\theta\|$ based on DNN.

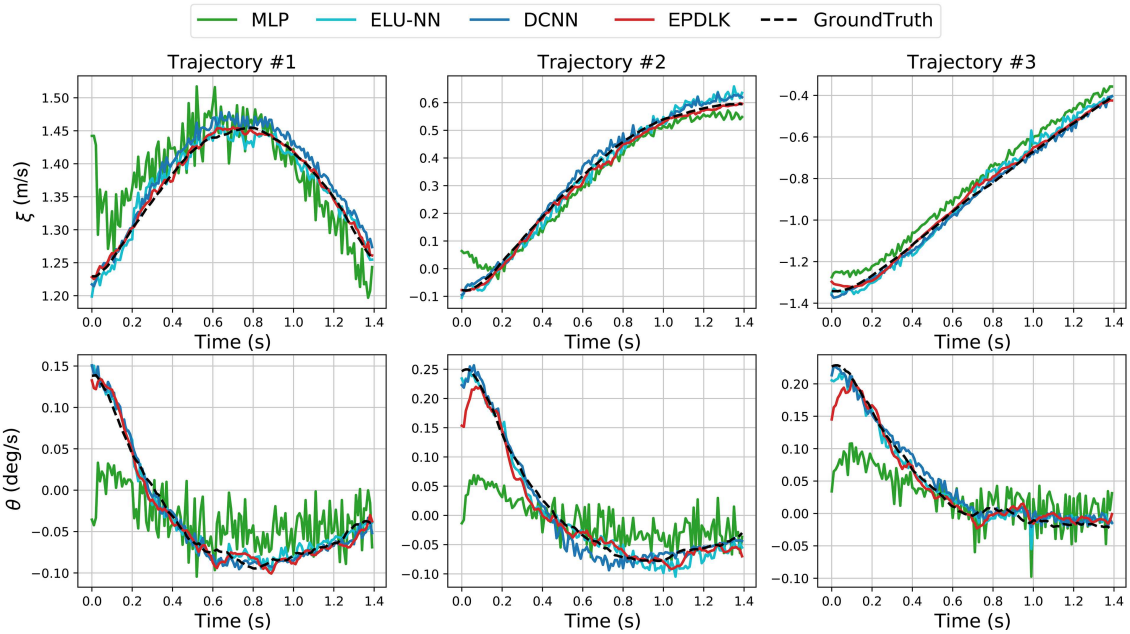

Figure 4 (Color online) Comparison of the multistep predictions of cart-pole system states: horizontal position ξ (top) and angle θ (bottom).

exponentially. The prediction accuracy of the Edmd-rbf method for angle θ is better than that for the horizontal position ξ. Figures 3(b) and (d) show the prediction error distributions of the four methods using DNNs.

Figure 4 compares the prediction results of three different trajectories. According to Figures 3 and 4, compared with MLP, ELU-NN, and DCNN, the EPDLK method can achieve accurate long horizontal multistep prediction, and the variance of the prediction results is slight.

Table 3 Performance comparison on MSE and computational efficiency for the 200-Step trajectory prediction task

	Linear	Edmd-poly	Edmd-rbf	Hermite	MLP	ELU-NN	DCNN	EPDLK-NPI	EPDLK-FD	EPDLK
MSE of ξ (m/s)	33.321	1.088	1.238	0.479	0.054	0.046	0.027	0.062	0.024	0.013
MSE of θ (deg/s)	291.542	6.866	0.076	2.992	0.047	0.045	0.032	0.056	0.029	0.012
Average time cost of one step (ms)	0.1525	0.1467	0.1583	0.1810	0.4667	0.4478	0.4832	0.3010	0.2842	0.2875

Table 3 shows the mean square error (MSE) of the multistep prediction of each method on the 60 trajectories in the test set and the corresponding average time consumed in single-step prediction. In the ablation experiment, the EPDLK model degenerates into a conventional deep MLP because of the removal of the physical information representation components. Under the same model parameters as the EPDLK model, the EPDLK-NPI model, although its time complexity did not change significantly, the prediction error increased by 371.79%, showing that the structural subnetwork and the energy-preserving loss function component played a significant role in building a physically feasible Koopman representation.

The EPDLK-LD model performed better when only 1/6 of the dataset was used for training. The prediction error only increased to 0.024 and 0.029, respectively, which was still better than those of other comparison methods, showing that, by introducing physical information, the solution space is limited to a subspace that satisfies physical feasibility, which can reduce the need for data samples.

6.3 Model-based control

We compared the performance of models generated by various methods in trajectory tracking control tasks. The goal of the control task is to return to the origin within 2 s from the initial position.

We randomly selected 25 different groups of initial positions and target positions and used nonlinear MPC to generate state reference trajectories and corresponding control inputs when the model parameters of the cart-pole system were known. We designed the MPC controller for trajectory tracking on the Koopman linear system constructed using four methods (MLP, ELU-NN, DCNN, and EPDLK). Figure 5 shows the trajectory tracking error results. As shown in the figure, because of the energy-preserving property of the EPDLK framework, the tracking results of the controller designed in the Koopman $\boldsymbol{z}$-system are almost identical to the control results generated by NMPC in the original nonlinear $\boldsymbol{x}$-system. However, the trajectory tracking results of other algorithms have a certain degree of deviation. In particular, Subsection 6.2 shows that the prediction results of the linear, Edmd-poly, Edmd-rbf, and Hermiter methods diverged with time, which was challenging to use in controller design. Thus, the control results are not shown here.

In Table 4, we give the MSE of trajectory tracking and the average time cost to solve the MPC optimization problem of one step on the Koopman linear system generated by each method. The tracking results on multiple trajectories show that the energy-preserving Koopman observation function can achieve model control consistency by ensuring energy consistency between the Koopman linear $\boldsymbol{z}$-system and the original nonlinear $\boldsymbol{x}$-system. In particular, after removing the Lagrangian structural subnetwork and the energy-preserving loss function, the performance of the control law corresponding to the EPDLK-NPI method does not differ significantly from that corresponding to the conventional deep MLP, showing the effectiveness of the proposed EPDLK framework for the control of Lagrangian dynamic systems. The results of the EPDLK method further demonstrate the high data utilization of the EPDLK framework with the introduction of physical information. Because the MPC solving process occupies the primary time consumption in the Koopman-based MPC design process, different methods of constructing Koopman observation functions have little impact on the time cost of MPC controller generation under the condition that the dimensionality of the Koopman linear system is consistent.

7 Conclusion

We propose a physics-informed DLK network topology EPDLK for learning the Koopman linear representation of nonlinear systems by encoding the physical structure of the Lagrangian system into the Koopman linear representation. Meanwhile, by constructing a two-stage Koopman representation learning algorithm, the learned Koopman linear embedding representation satisfies the EDM constraint. It thus achieves control consistency between the Koopman linear system and the original nonlinear system. Then, the learning Koopman linear embedding model is combined with MPC to realize the trajectory

Table 4 Performance comparison on MSE and computational efficiency for the trajectory tracking task

	MLP	ELU-NN	DCNN	EPDLK-NPI	EPDLK-LD	EPDLK
MSE of ξ (m/s)	1.115	1.757	0.228	1.431	0.047	0.022
MSE of θ (deg/s)	0.136	0.207	0.077	0.152	0.0112	0.0067
Average time cost of one step (ms)	7.92	7.81	8.10	7.33	7.40	7.37

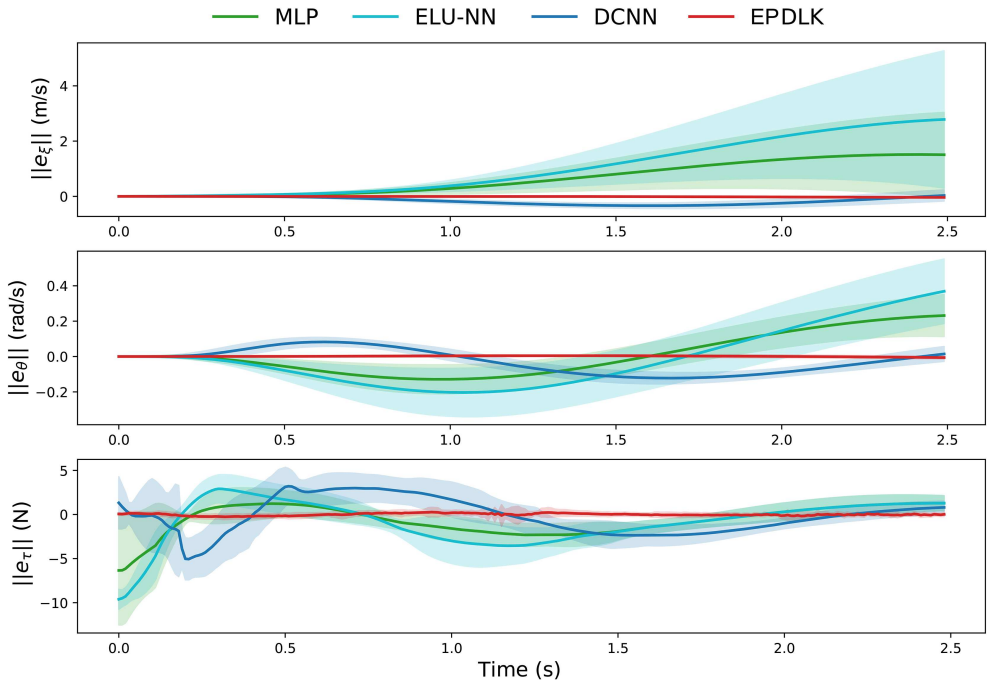

Figure 5 (Color online) Comparison of trajectory tracking control: error of states ξ (top) and θ (center) and error of control inputs (bottom).

tracking control task. Several experiments show that the proposed method is significantly better than the local linearization method, the EDMD method of manually designing the observable functions, and some current methods for building Koopman observation functions using DNNs. These results show that physics-informed energy-preserving deep Koopman embedded representation can obtain the linear representation of a dynamic system model with high physical fidelity, achieve high-precision, long-time-horizon prediction, and achieve control consistency between the Koopman linear system and the original nonlinear system. In future research, we plan to apply the proposed EPDLK algorithm to actual robots, addressing complex issues such as observation noise, model uncertainty [36], and time delay [37], where factors like the length of Koopman encoding could significantly impact the precision of Koopman representation.

Acknowledgements This work was supported in part by National Natural Science Foundation of China (Grant No. 62033012), China Postdoctoral Science Foundation (Grant No. 2023M733384), and Fundamental Research Funds for the Central Universities (Grant No. WK2100000044).

References

1 Chung S J, Slotine J J E. Cooperative robot control and concurrent synchronization of Lagrangian systems. IEEE Trans Robot, 2009, 25: 686–700
2 Yue M, Ning Y G, Yu S Z, et al. Composite following control for wheeled inverted pendulum vehicles based on human-robot interaction. Sci China Inf Sci, 2019, 62: 50206
3 Lv W J, Kang Y, Qin J H. FVO: floor vision aided odometry. Sci China Inf Sci, 2019, 62: 012202
4 Xie K D, Jiang Y, Yu X, et al. Data-driven cooperative optimal output regulation for linear discrete-time multi-agent systems by online distributed adaptive internal model approach. Sci China Inf Sci, 2023, 66: 170202
5 Li J N, Nie H, Chai T Y, et al. Reinforcement learning for optimal tracking of large-scale systems with multitime scales. Sci China Inf Sci, 2023, 66: 170201
6 Koopman B O. Hamiltonian systems and transformation in Hilbert space. Proc Natl Acad Sci USA, 1931, 17: 315–318
7 Korda M, Mezić I. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. Automatica, 2018, 93: 149–160
8 Brunton S L, Brunton B W, Proctor J L, et al. Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. PLoS One, 2016, 11: e0150171
9 Schmid P J. Dynamic mode decomposition of numerical and experimental data. J Fluid Mech, 2010, 656: 5–28

10 Williams M O, Kevrekidis I G, Rowley C W. A data-driven approximation of the Koopman operator: extending dynamic mode decomposition. J Nonlinear Sci, 2015, 25: 1307–1346
11 Lusch B, Kutz J N, Brunton S L. Deep learning for universal linear embeddings of nonlinear dynamics. Nat Commun, 2018, 9: 1
12 Folkestad C, Pastor D, Mezic I, et al. Extended dynamic mode decomposition with learned Koopman eigenfunctions for prediction and control. In: Proceedings of American Control Conference (ACC), 2020. 3906–3913
13 Li Y, He H, Wu J, et al. Learning compositional Koopman operators for model-based control. In: Proceedings of International Conference on Learning Representations, New Orleans, 2019
14 Morton J, Witherden F D, Kochenderfer M J. Deep variational Koopman models: inferring Koopman observations for uncertainty-aware dynamics modeling and control. In: Proceedings of IJCAI International Joint Conference on Artificial Intelligence, Macao, 2019. 3173–3179
15 Folkestad C, Pastor D, Burdick J W. Episodic Koopman learning of nonlinear robot dynamics with application to fast multirotor landing. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Paris, 2020. 9216–9222
16 Lutter M, Ritter C, Peters J. Deep Lagrangian networks: Using physics as model prior for deep learning. In: Proceedings of International Conference on Learning Representations, Vancouver, 2018
17 Cranmer M, Greydanus S, Hoyer S, et al. Lagrangian neural networks. In: Proceedings of ICLR Workshop on Integration of Deep Neural Models and Differential Equations, Addis Ababa, 2020. 1–9
18 Greydanus S, Dzamba M, Yosinski J. Hamiltonian neural networks. 2019. ArXiv:1906.01563
19 Bahari M, Nejjar I, Alahi A. Injecting knowledge in data-driven vehicle trajectory predictors. Transp Res Part C-Emerg Technol, 2021, 128: 103010
20 Ortega R, van der Schaft A J, Mareels I, et al. Putting energy back in control. IEEE Control Syst Mag, 2001, 21: 18–33
21 Wang Y Z, Feng G. On finite-time stability and stabilization of nonlinear port-controlled Hamiltonian systems. Sci China Inf Sci, 2013, 56: 1–14
22 Franco E, Garriga-Casanovas A. Energy-shaping control of soft continuum manipulators with in-plane disturbances. Int J Robot Res, 2021, 40: 236–255
23 Kaiser E, Kutz J N, Brunton S L. Data-driven discovery of Koopman eigenfunctions for control. Mach Learn-Sci Technol, 2021, 2: 035023
24 Shi L, Karydis K. ACD-EDMD: analytical construction for dictionaries of lifting functions in Koopman operator-based nonlinear robotic systems. IEEE Robot Autom Lett, 2022, 7: 906–913
25 Yeung E, Kundu S, Hodas N. Learning deep neural network representations for Koopman operators of nonlinear dynamical systems. In: Proceedings of American Control Conference (ACC), 2019. 4832–4839
26 van der Heijden B, Ferranti L, Kober J, et al. DeepKoCo: efficient latent planning with a task-relevant Koopman representation. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021. 183–189
27 Williams M O, Hemati M S, Dawson S T M, et al. Extending data-driven Koopman analysis to actuated systems. IFAC-PapersOnLine, 2016, 49: 704–709
28 Morton J, Witherden F D, Kochenderfer M J, et al. Deep dynamical modeling and control of unsteady fluid flows. In: Proceedings of Advances in Neural Information Processing Systems, Montéral, 2018. 9258–9268
29 Azencot O, Erichson N B, Lin V, et al. Forecasting sequential data using consistent Koopman autoen-coders. In: Proceedings of International Conference on Machine Learning, 2020. 475–485
30 Greenwood D T. Advanced Dynamics. Cambridge: Cambridge University Press, 2006
31 Miller K, Clavel R. The lagrange-based model of delta-4 robot dynamics. Robotersysteme, 1992, 8: 49–54
32 Cao W M, Zheng C T, Yan Z Y, et al. Geometric deep learning: progress, applications and challenges. Sci China Inf Sci, 2022, 65: 126101
33 Paszke A, Gross S, Massa F, et al. Pytorch: an imperative style, high-performance deep learning library. In: Proceedings of Advances in Neural Information Processing Systems, 2019
34 Kingma D P, Ba J. Adam: a method for stochastic optimization. In: Proceedings of International Conference on Learning Representations, San Diego, 2015
35 Jeyakumar V, Rubinov A M, Wu Z Y. Non-convex quadratic minimization problems with quadratic constraints: global optimality conditions. Math Program, 2007, 110: 521–541
36 Wang T, Kang Y, Li P F. Adaptive event-triggered distributed model predictive control for tracking consensus of multiagent systems. Sci Sin-Tech, 2023, 53: 1885–1894
37 Kang Y, Zhao Y B. Dynamic data packing towards the optimization of QoC and QoS in networked control systems. Sci China Tech Sci, 2016, 59: 72–80