Received: 18 November 2023 | Revised: 8 January 2024 | Accepted: 18 January 2024

DOI: 10.1002/msd2.12102

RESEARCH ARTICLE

IJMSD International Journal of Mechanical System Dynamics Open Access WILEY

A multiscale differential-algebraic neural network-based method for learning dynamical systems

Yin Huang[1] | Jieyu Ding[2,3]

[1]College of Computer Science & Technology, Qingdao University, Qingdao, China

[2]School of Mathematics and Statistics, Qingdao University, Qingdao, China

[3]Center for Computational Mechanics and Engineering Simulation, Qingdao University, Qingdao, China

Correspondence

Prof. Jieyu Ding, School of Mathematics and Statistics, Qingdao University, Qingdao, China.
Email: djy@qdu.edu.cn

Funding information

National Natural Science Foundation of China, Grant/Award Numbers: 11772166, 12172186

Abstract

The objective of dynamical system learning tasks is to forecast the future behavior of a system by leveraging observed data. However, such systems can sometimes exhibit rigidity due to significant variations in component parameters or the presence of slow and fast variables, leading to challenges in learning. To overcome this limitation, we propose a multiscale differential-algebraic neural network (MDANN) method that utilizes Lagrangian mechanics and incorporates multiscale information for dynamical system learning. The MDANN method consists of two main components: the Lagrangian mechanics module and the multiscale module. The Lagrangian mechanics module embeds the system in Cartesian coordinates, adopts a differential-algebraic equation format, and uses Lagrange multipliers to impose constraints explicitly, simplifying the learning problem. The multiscale module converts high-frequency components into low-frequency components using radial scaling to learn subprocesses with large differences in velocity. Experimental results demonstrate that the proposed MDANN method effectively improves the learning of dynamical systems under rigid conditions.

KEYWORDS

dynamical systems learning, multibody system dynamics, differential-algebraic equation, neural networks, multiscale structures

1 | INTRODUCTION

The objective of dynamical system learning tasks is to forecast the future behavior of a system by leveraging observed data.[1] Dynamical systems provide a mathematical framework to describe the evolution of natural phenomena across time and space, commonly represented using differential equations. Solving these equations allows predictions about the future state of dynamical systems. Understanding complex physical dynamics across various spatial and temporal scales poses a significant research challenge, garnering attention from numerous scholars and having practical implications.[2–6]

Prediction in dynamical systems primarily relies on creating models derived from first principles. However, due to incomplete knowledge, these models based on physical laws often oversimplify or misrepresent the underlying structure of dynamical systems.[7] As a result, they tend to exhibit high biases and modeling errors that cannot be rectified by optimizing a few parameters. Consequently, it becomes essential to employ dynamical system learning methods that can accurately identify valid dynamic models based on observed trajectories. These methods play a crucial role in the analysis, simulation, and control of dynamic systems.[8–10]

Dynamical systems learning methods can be broadly classified into two categories: purely data-driven methods and physics-guided machine learning methods.[11] The availability of experimental data has led to the increasing popularity of purely data-driven methods, which utilize forward or backward models to fit the training data without requiring extensive involvement in model design or the derivation of learning and inference processes.[12–15] Among purely data-driven methods, neural networks have emerged as the dominant technology due to their exceptional ability to efficiently capture high-dimensional spatiotemporal dynamics from large data sets. One notable method in this category is the sparse identification of nonlinear dynamics (SINDy), which represents dynamic nonlinear differential equations as linear combinations of nonlinear candidate functions and approximates the model through sparse regression.[16] However, SINDy's reliance on numerical differentiation renders it sensitive to data noise, particularly when higher-order derivatives are involved. To mitigate this issue, researchers have developed alternative methods, such as integral identification[17] and weak form identification.[18] Deep learning architectures offer highly expressive models for function approximation and have demonstrated success in various scenarios.[19–21] Nonetheless, these models often exhibit a bias toward certain dynamic representations that do not strictly enforce the system's symmetries and conservation laws. Consequently, their generalization ability suffers, leading to physically implausible outcomes when applied to new, unseen states. This characteristic also contributes to their high variance and challenges in interpretation.[22] Moreover, training these models typically necessitates substantial data sets and lengthy computation times, rendering them impractical for many real-world applications. Thus, it is imperative to develop data-driven methods that yield physically plausible results.[23–26]

Physics-guided machine learning methods have emerged as a hybrid method that integrates the principles of physics with deep learning architectures. This novel method incorporates the physical constraints and geometric properties of the underlying system during the design and learning processes of neural networks. By leveraging the function approximation capabilities of neural networks, existing knowledge of the system's physics can be incorporated into the construction of physically constrained neural networks, leading to improved design, efficiency, and generalization capabilities.[27] In recent years, physically informed neural networks (PINNs) have been introduced as a means to incorporate the underlying physics during the training process and enhance desired system properties, addressing the limitations of classical neural networks.[28] For instance, Chen et al.[29] developed neural ordinary differential equations (Node) to uncover hidden ordinary differential equations (ODEs) from discrete data. However, these models often fail to preserve system energy and exhibit poor generalization capabilities.

To address these limitations, researchers have explored integrating physical principles, such as differential equations and symmetries, into deep neural networks (DNNs) as constraints.[30,31] Lutter et al.[12] proposed deep Lagrangian networks (DeLaNs), which employ two deep networks to model rigid body dynamics by parameterizing kinetic and potential energies. This formulation ensures the preservation of system energy, leading to superior long-term prediction and control performance. Greydanus et al.[32] introduced Hamiltonian neural networks (HNNs), followed by Cranmer et al.[33], who proposed Lagrangian neural networks (LNNs). Both approaches model the system holistically, regressing central quantities such as Lagrangian or Hamiltonian quantities to represent the system's dynamics and maintain physical properties like energy conservation and symmetry. To explicitly enforce constraints and simplify the learning problem, Finzi et al.[34] presented constrained Hamiltonian neural networks (CHNNs) and constrained Lagrangian neural networks (CLNNs). These frameworks embed the system in a Cartesian coordinate system and leverage Lagrange multipliers to explicitly enforce the constraints. As a result, they achieve simplification of the learning problem. Lu et al.[35] introduced modular Lagrangian networks (ModLaNets), which employ Euler Lagrangian equations to independently model system elements. This approach demonstrates improved performance in multibody tasks. Moreover, Gruver et al.[36] proposed mechanics neural networks (MechanicsNNs), utilizing an induction bias to deconstruct the HNN model and enhance computational efficiency. These models aim to strike a balance between accuracy and computational cost. However, these methods do not fully consider the impact of rigid conditions on accuracy, necessitating further research.

Through an in-depth exploration of the intrinsic dynamic properties inherent in multibody systems, a comprehensive understanding of their behavior in both scientific research and engineering applications can be achieved. This, in turn, equips us with indispensable tools for predicting and managing system behavior, thereby facilitating the resolution of critical problems. The dynamic modeling and prediction of multiphysics and multiscale systems present ongoing scientific challenges, as underscored by Karniadakis et al.[37] It is imperative to acknowledge that these systems often manifest rigid characteristics that exert a significant impact on the learning process. This effect becomes particularly pronounced in scenarios where parameters of system components undergo substantial changes or in the convergence of slow variables characterized by extensive ranges of motion and fast variables involving elastic deformations. As a result, neural network methods face limitations in addressing the inherent complexities of specific multibody dynamics equations. Given these constraints and drawing upon existing research findings, we posit a novel approach grounded in multiscale differential-algebraic neural networks (MDANN). Our method employs a system of differential-algebraic equations (DAE) with constraints explicitly enforced using Lagrange multipliers. This approach facilitates the learning of Lagrangian quantities, contributing to a more nuanced understanding of the system dynamics. Furthermore, the effectiveness of our proposed method is demonstrated through its application in a dynamically controlled environment. We introduce a multiscale module that integrates information from different frequencies, thereby enabling the learning of subprocesses with notable speed variations within the system. We conducted experimental validation of our proposed method, investigating coupled pendulum systems and double pendulum systems. The outcomes of these experiments provide compelling evidence substantiating the efficacy of our approach in acquiring a profound

understanding of dynamic systems. Additionally, we extend the application of our method to scalable structures, leveraging it to enhance precision in determining control forces. The results obtained from these applications distinctly showcase the robustness of our approach, affirming its capability to refine control strategies for dynamic systems.

The rest of the paper is organized as follows. Section 2 provides essential background knowledge to facilitate a better understanding of the subsequent discussions. Section 3 presents the MDANN method along with the technical implementation details. Section 4 showcases a series of numerical experiments. The paper concludes in Section 5.

2 | LEARNING DYNAMICS

The standard frameworks for Hamiltonian and Lagrangian dynamics are typically presented as sets of first- and second-order ODE, respectively. In 2018, Chen et al.[29] proposed Node method, a continuous DNN-based approach that utilizes neural networks to represent the time-continuous dynamics of hidden states.

The Node method models the time evolution of a hidden unit as a constant neural differential equation, as shown in Equation (1):

$$\frac{d\boldsymbol{x}}{dt} = f(\boldsymbol{x}(t), t, \theta), \tag{1}$$

where t represents time, $\boldsymbol{x}(t)$ is a continuous representation of the hidden state, and θ refers to the network weights and biases. The evolution function $f(\boldsymbol{x}(t), t, \theta)$, parameterized by the neural network, describes the dynamics of the hidden state over time. By solving the initial value problem (IVP) using the initial conditions $\boldsymbol{x}(t_0)$, the hidden state $\boldsymbol{x}(t_1)$ at the next moment is obtained. Therefore, the time evolution of the state $\boldsymbol{x}(t)$ can be represented as a time integral, as shown in Equation (2):

$$\begin{aligned}\boldsymbol{x}(t_1) &= \boldsymbol{x}(t_0) + \int_{t_0}^{t_1} f(\boldsymbol{x}(t), t, \theta)dt \\ &= \text{ODESolve}(\boldsymbol{x}(t_0), f, t_0, t_1, \theta),\end{aligned} \tag{2}$$

where ODESolve is an ODE numerical solver. To improve memory utilization efficiency, the concomitant sensitivity method is used during backpropagation.[29] Furthermore, the Node method allows for the incorporation of prior knowledge of physical principles into the network, enabling the parameterization of unknown physical quantities in the dynamics, such as mass, potential energy, Lagrangian, and Hamiltonian quantities.

3 | PROPOSED METHOD

This section introduces our proposed MDANN method. The method comprises two key modules: the Lagrangian mechanics module and the multiscale module. The Lagrangian mechanics module facilitates the learning of Lagrangian quantities by employing a system of DAE with explicit constraints. On the other hand, the multiscale module utilizes radial scaling to transform high-frequency components into low-frequency components, thereby enabling the learning of physical processes at various frequencies within Lagrangian quantities.

3.1 | Lagrangian mechanics module

The utilization of Lagrangian quantities to ensure energy conservation in a system has proven invaluable for comprehensively understanding the underlying physical processes.[7] By explicitly incorporating constraints, the learning of Lagrangian quantities is further enhanced, leading to improved data efficiency and prediction accuracy.

In this module, we employ the Lagrange multiplier method, which utilizes a system of DAE to explicitly handle constraints and obtain the Lagrangian quantities of the system. The module consists of two key components: a differential component and an algebraic component. The differential component focuses on the ODE of motion, which describes the temporal evolution of an object or system. It allows for the conservation of the Lagrangian quantity of the system. By considering variables such as position, velocity, and acceleration, we can infer the dynamic behavior over time. Differential equations provide the fundamental framework for describing the motion of an object or system. On the other hand, the algebraic component involves the algebraic constraint equations that capture the relationships within the system, including binding forces and other constraints. These equations effectively represent the interactions and constraints among the different components. Figure 1 depicts a schematic diagram of the Lagrangian mechanics module.

The system is characterized by a set of Cartesian coordinates $\boldsymbol{q}(t)$ that represent the configuration of a rigid body at time t. The Lagrangian function of the system, defined in Equation (3), is a fundamental component of our proposed methodology:

$$L(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) = T(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) - V(\boldsymbol{q}(t)), \tag{3}$$

where T represents the kinetic energy of the system, and V represents its potential energy. To efficiently learn Lagrangian quantities of physical systems, we parameterize the kinetic and potential energy using two separate neural networks. We leverage Lagrangian quantities as prior knowledge in this process.

In the context of mechanical systems, the kinetic energy T of the system is determined by Equation (4):

$$T(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) = \frac{1}{2}\dot{\boldsymbol{q}}(t)^{\mathrm{T}}\boldsymbol{M}(\boldsymbol{q}, t)\dot{\boldsymbol{q}}(t), \tag{4}$$

where the generalized mass matrix $\boldsymbol{M}(\boldsymbol{q}, t) = \nabla_{\dot{\boldsymbol{q}}}\nabla_{\dot{\boldsymbol{q}}}^{\mathrm{T}}L$, which is a constant matrix in Cartesian coordinates and does not depend on the state $\boldsymbol{q}(t)$. Therefore, learning the constant values of this matrix using a neural network can simplify the form of the Lagrangian function. The dynamics of a multibody system can be expressed using the DAE as shown in Equation (5):

$$\begin{cases}\boldsymbol{M}(\boldsymbol{q}, t)\ddot{\boldsymbol{q}} + \boldsymbol{\Phi}_{\boldsymbol{q}}^{\mathrm{T}}\boldsymbol{\lambda} = \boldsymbol{F}(\dot{\boldsymbol{q}}(t), \boldsymbol{q}(t), t) \\ \boldsymbol{\Phi}(\boldsymbol{q}, t) = \boldsymbol{0}\end{cases}. \tag{5}$$

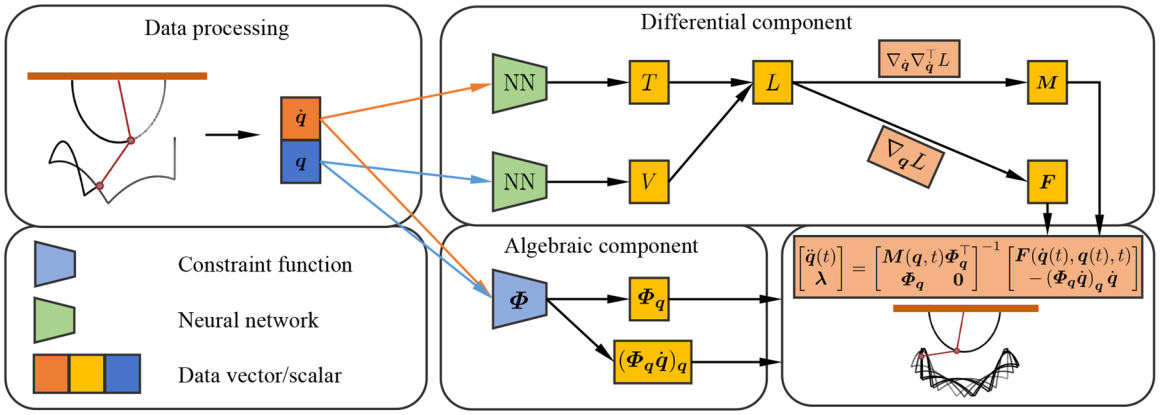

FIGURE 1 Schematic diagram of Lagrangian mechanics module.

The above equation describes the system using the generalized velocity $\dot{\boldsymbol{q}}$, the Lagrange multiplier $\boldsymbol{\lambda}$, the constraint equation $\boldsymbol{\Phi}$ for the position coordinate array $\boldsymbol{q}$, the generalized force matrix $\boldsymbol{F}$, and the Jacobi matrix of the constraint equation, $\boldsymbol{\Phi}_{\boldsymbol{q}}$. To obtain the velocity constraint equation $\dot{\boldsymbol{\Phi}}$ and the acceleration constraint equation $\ddot{\boldsymbol{\Phi}}$ for the system, we need to solve the constraint equation $\boldsymbol{\Phi}(\boldsymbol{q}, t)$ for the first and second-order derivatives with respect to time t, respectively. The velocity constraint equation, as shown in Equation (6), is given by

$$\dot{\boldsymbol{\Phi}}(\boldsymbol{q}, t) = \boldsymbol{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} + \boldsymbol{\Phi}_{t}(\boldsymbol{q}, t) = \boldsymbol{0}. \tag{6}$$

Similarly, the acceleration constraint equation, as shown in Equation (7), is given by

$$\begin{aligned}\ddot{\boldsymbol{\Phi}}(\boldsymbol{q}, t) = (\boldsymbol{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}} + \boldsymbol{\Phi}_{\boldsymbol{q}}(\boldsymbol{q}, t)\ddot{\boldsymbol{q}} + \boldsymbol{\Phi}_{\boldsymbol{qt}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} \\ + \boldsymbol{\Phi}_{t\boldsymbol{q}}(\boldsymbol{q}, t)\dot{\boldsymbol{q}} + \boldsymbol{\Phi}_{tt}(\boldsymbol{q}, t) = \boldsymbol{0}.\end{aligned} \tag{7}$$

These equations result in the system of index-1 DAE, given by Equation (8):

$$\begin{bmatrix}\boldsymbol{M}(\boldsymbol{q}, t) & \boldsymbol{\Phi}_{\boldsymbol{q}}^{\top} \\ \boldsymbol{\Phi}_{\boldsymbol{q}} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda}\end{bmatrix} = \begin{bmatrix}\boldsymbol{F}(\dot{\boldsymbol{q}}(t), \boldsymbol{q}(t), t) \\ -(\boldsymbol{\Phi}_{\boldsymbol{q}}\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}}\end{bmatrix}, \tag{8}$$

where the generalized force array $\boldsymbol{F} = \nabla_{\boldsymbol{q}}L = -\nabla_{\boldsymbol{q}}V$. Equation (8) can be reformulated as Equation (9), which explicitly expresses the acceleration of the generalized coordinates $\boldsymbol{q}(t)$ and the Lagrange multipliers $\boldsymbol{\lambda}$ as a function of the other variables. This reformulation can be particularly helpful in the numerical integration of the DAE.

$$\begin{bmatrix}\ddot{\boldsymbol{q}}(t) \\ \boldsymbol{\lambda}\end{bmatrix} = \begin{bmatrix}\boldsymbol{M}(\boldsymbol{q}, t) & \boldsymbol{\Phi}_{\boldsymbol{q}}^{\top} \\ \boldsymbol{\Phi}_{\boldsymbol{q}} & \boldsymbol{0}\end{bmatrix}^{-1}\begin{bmatrix}\boldsymbol{F}(\dot{\boldsymbol{q}}(t), \boldsymbol{q}(t), t) \\ -(\boldsymbol{\Phi}_{\boldsymbol{q}}\dot{\boldsymbol{q}})_{\boldsymbol{q}}\dot{\boldsymbol{q}}\end{bmatrix}. \tag{9}$$

3.2 | Multiscale module

Dynamical systems often exhibit multifrequency phenomena, particularly when their components vary significantly in parameters or involve a combination of slow variables with a wide range of motion and fast variables with elastic deformation. Consequently, the solutions of such systems comprise multiple frequency components that are superimposed on each other. DNNs excel at processing data with low-frequency content, as supported by the frequency principle (F-principle).[38] DNNs can rapidly learn the low-frequency content of data and achieve commendable generalization accuracy. However, neural networks often struggle when confronted with high-frequency data, leading to reduced convergence or even nonconvergence of the learning method. In the domain of multibody dynamics, the learning of dynamical systems poses challenges due to the frequency disparities between the motions of objects within the system.

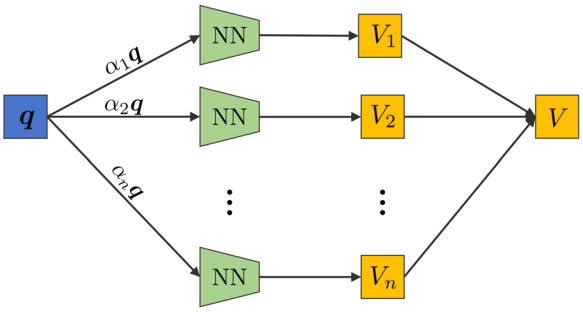

FIGURE 2 Schematic diagram of multiscale module.

To tackle this challenge, we employ a multiscale structure to preprocess the input data and extract frequency features that are better suited for learning. Specifically, we adopt the multiscale structure proposed by Liu[39] This approach has demonstrated its efficacy in facilitating the rapid learning of high-frequency components and expediting the solution of partial differential equations in comparison to traditional fully connected network structures. The multiscale module employs radial scaling to convert solution content from higher frequency ranges to lower ones, thereby rendering the solution content easier to learn. The module takes the state variable $\boldsymbol{q}$ as input and produces the potential energy V of the system as output. A schematic diagram representation of the module is depicted in Figure 2.

In our multiscale module, we utilize the linear combination property of energy to establish the potential energy of the system.[35] The potential energy comprises two main components: the potential energy, denoted as V_i, between the elements and the environment, and the potential energy, denoted as V_{ij}, between the elements themselves. The expression for the potential energy is presented in Equation (10).

$$V(\boldsymbol{q}) = \sum_i c_i V_i(\boldsymbol{q}_i) + \sum_{i,j,i\neq j} \frac{c_{ij}}{2} V_{ij}(\boldsymbol{q}_i, \boldsymbol{q}_j), \tag{10}$$

where the weight parameters of V_i and V_{ij} are denoted by c_i and c_{ij}, respectively. Additionally, V_{ij} is symmetric, that is, $V_{ij} = V_{ji}$. To obtain consistent frequency accuracy solutions, the position coordinates $\boldsymbol{q}$ are segmented into different frequency ranges using radial scaling. This segmentation allows for the utilization of m parallel subneural networks, denoted as V_θ, which are responsible for calculating the potential energy within each frequency range. Finally, we calculate the potential energy using a weighted summation, as shown in Equation (11).

$$\begin{aligned} V(\boldsymbol{q}) = &\sum_i c_i \sum_{k=1}^{m} V_i^{\theta_k}(\alpha_k \boldsymbol{q}_i) \\ &+ \sum_{i,j,i\neq j} \frac{c_{ij}}{2} \sum_{k=1}^{m} V_{ij}^{\theta_k}(\alpha_k \boldsymbol{q}_i, \alpha_k \boldsymbol{q}_j), \end{aligned} \tag{11}$$

where α_k and θ_k denote the scaling factor and network parameters of the k-th subnetwork, respectively. Additionally, we use residual connections to address the gradient vanishing issue.

During the training process, our model takes the system's states $(\boldsymbol{q}, \dot{\boldsymbol{q}})$ as inputs and predicts the corresponding states $\ddot{\boldsymbol{q}}$. To update the network parameters, we utilize the L_2 loss function, which is expressed in Equation (12):

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} (\ddot{\boldsymbol{q}}_i - \hat{\ddot{\boldsymbol{q}}}_i)^2, \tag{12}$$

where $\ddot{\boldsymbol{q}}$ and $\hat{\ddot{\boldsymbol{q}}}$ represent the true and predicted system states, respectively. N represents the number of samples in the data set. This loss function quantifies the discrepancy between the predicted and actual states, guiding the adjustment of network parameters during the training process.

4 | EXPERIMENTS

The experiments reported were conducted in the college laboratory of Qingdao University, utilizing existing infrastructure. The experimental setup comprised a server equipped with a Tesla P100 graphics card, an Intel Xeon CPU, and 13 GB of RAM, running on an Ubuntu 20.04 operating system. The algorithm was implemented using Python 3.9 and the PyTorch deep learning framework, with PyCharm as the integrated development environment.

To assess the effectiveness of the proposed method, we implemented it in three different systems: a double pendulum system, a coupled pendulum system, and a scissor-type deployable mast system. In investigating the double pendulum system, we utilized dynamic equations to build a comprehensive data set, comprising both a training set and a test set. The training set encompasses 100 samples with distinct initial conditions, and their evolution processes within the time range $t = [0, 5]$ are meticulously documented. For robust model training, we intentionally selected mutually independent initial conditions to capture the intricacies and diversity of the system's behavior. The test set consists of 100 samples with different initial conditions and no overlap with the training set samples. Their evolution processes within the time range $t = [0, 15]$ are recorded. The design of this data set aims to fully elucidate the generalization performance of the adopted methods in response to dynamic changes in the system. In the case of the coupled pendulum system, we applied the same methodology for generating both the training and test sets as employed in the double pendulum system, ensuring consistency throughout the study. While investigating the scissor-type deployable mast system, we calculated the expected trajectories of the system in accordance with mission design requirements. This process aimed to generate the essential data sets necessary for training neural networks. For model optimization, we applied the Adam algorithm, initializing weights from a normal distribution. The Tanh activation function was employed in this process. The configuration for multiscale frequency segments involved setting the number to 5, with scaling factors specified as {1, 2, 4, 8, 16}. To solve the dynamical system, we employed the dopri5 ODE solver with a time step of 0.01 s and used a relative tolerance error of 10^{-6} along with an absolute tolerance error of 10^{-9}. These choices were made for their accuracy and stability.

We employed the mean squared error (MSE) as an evaluation metric to assess the performance of the dynamical system learning method.[34] The MSE was utilized to assess the performance of the method using different error metrics, namely the position error and the energy violation error. The position error, denoted as $\mathrm{MSE}_{\boldsymbol{q}}$, is calculated according to Equation (13). This metric quantifies the deviation between the predicted and actual system states. Similarly, the energy violation error, denoted as MSE_E, is determined using Equation (14). This error metric measures the extent to which the method violates the energy constraints of the system.

$$\mathrm{MSE}_{\boldsymbol{q}} = \frac{1}{N} \sum_{i=1}^{N} (\boldsymbol{q}_i - \hat{\boldsymbol{q}}_i)^2 + \frac{1}{N} \sum_{i=1}^{N} (\dot{\boldsymbol{q}}_i - \hat{\dot{\boldsymbol{q}}}_i)^2, \tag{13}$$

$$\mathrm{MSE}_E = \frac{1}{N} \sum_{i=1}^{N} (E(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) - E(\hat{\boldsymbol{q}}_i, \hat{\dot{\boldsymbol{q}}}_i))^2, \tag{14}$$

where N stands for the number of samples in the data set. The vectors $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ correspond to the true state, while $\hat{\boldsymbol{q}}$ and $\hat{\dot{\boldsymbol{q}}}$ denote the predicted state. The function $E(\cdot)$ represents the total energy associated with the system.

4.1 | Coupled pendulum system

The coupled pendulum system is composed of three pendulums interconnected by two elastic springs. The system also experiences

stiffness problems when the springs are deformed elastically. The Lagrangian function is defined by Equation (15).

$$L = \sum_{i=1}^{3} \frac{1}{2} m_i \dot{\boldsymbol{x}}_i{}^{\top} \dot{\boldsymbol{x}}_i - \sum_{i=1}^{3} m_i g \boldsymbol{x}_i^{(2)} + \sum_{i=2}^{3} \frac{1}{2} k \left(\|\boldsymbol{x}_{i-1} - \boldsymbol{x}_i\| - r_{i-1}\right)^2, \tag{15}$$

where r and k represent the undeformed length and stiffness coefficient of the spring, respectively. For our experiments, we set $l_1 = l_2 = l_3 = 1$ m, $m_1 = 100$ kg, $m_2 = m_3 = 1$ kg, $r_1 = r_2 = 1$ m, and $k = 10$ N/m.

When assessing the efficacy of our method within the context of a coupled pendulum system, the evaluation of performance involves the quantification of the MSE pertaining to the predicted position and energy of the pendulum. The trajectory of a coupled pendulum system is visually depicted in Figure 3, providing valuable insights into the underlying dynamics. Notably, a comparative analysis with the reference solution elucidates discernible distinctions between the Node method and the MechanicsNN method, whereas the CLNN and MDANN methods exhibit a higher level of concordance.

Table 1 provides a thorough performance evaluation, demonstrating that the proposed method attains MSE_q of 3.214e−2 and MSE_E of 2.590e−3. To further elucidate these findings, Figure 4 illustrates the temporal evolution of errors, offering a dynamic perspective on the system's behavior. Notably, as the system evolves, the inherent complexity of its dynamics gradually intensifies, resulting in an escalating relative error. Nevertheless, the proposed method consistently outperforms alternative approaches, affirming its effectiveness in comprehensively capturing and simulating the behavior of rigid systems undergoing elastic deformation.

4.2 | Double pendulum system

The double pendulum system is composed of two small balls connected by massless rods. When the masses of the two pendulums differ significantly, the system experiences a stiffness problem. Its Lagrangian function is defined by Equation (16).

$$L = \sum_{i=1}^{2} \left(\frac{1}{2} m_i \dot{\boldsymbol{x}}_i{}^{\top} \dot{\boldsymbol{x}}_i - m_i g x_i^{(2)} \right), \tag{16}$$

where m_i represents the mass of the i-th object, g represents the acceleration of gravity in the $x^{(2)}$ direction, and $x^{(2)}$ represents the position of the object in that direction. Here, we used $l_1 = l_2 = 1$ m, $m_1 = 1$ kg, and $m_2 = 100$ kg.

The efficacy of the proposed method is assessed through its application to a double pendulum system. The obtained results are

TABLE 1 MSE comparison of results for coupled pendulum system.

Method	MSE_q	MSE_E
Node	2.353e−01 ± 1.063e−01	6.922e+02 ± 6.891e+02
MechanicsNN	4.727e−02 ± 7.575e−03	1.306e+01 ± 8.625e+00
CLNN	4.343e−02 ± 8.331e−03	2.490e+00 ± 1.426e+00
MDANN	3.214e−02 ± 1.132e−02	2.590e−02 ± 1.366e−02

Abbreviations: CLNN, constrained Lagrangian neural networks; MDANN, multiscale differential-algebraic neural network; MSE, mean squared error.

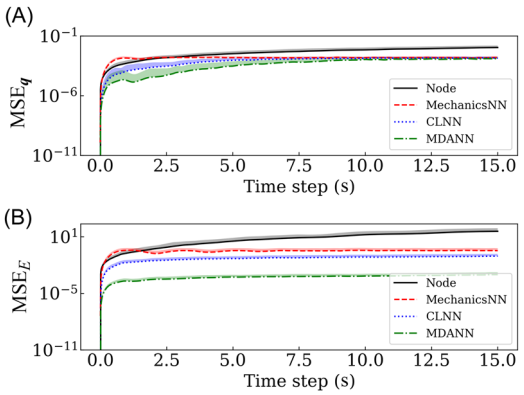

FIGURE 4 Accumulated mean squared error (MSE) over time for coupled pendulum system: (A) MSE of position, (B) MSE of energy.

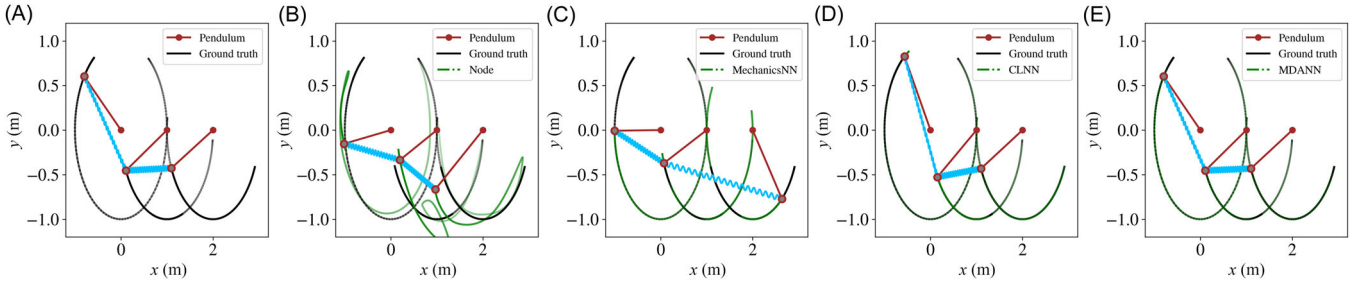

FIGURE 3 Comparison of results for coupled pendulum system: (A) ground truth, (B) Node, (C) MechanicsNN, (D) constrained Lagrangian neural networks, (E) multiscale differential-algebraic neural network.

presented in Table 2. The MSE for the proposed method's position is 9.638e−02, while the MSE for energy is 5.091e−01. In contrast, both Node and MechanicsNN exhibit signs of overfitting, underscoring their limitations in accurately capturing the evolution equations of the system, particularly in the face of substantial changes in mass. Remarkably, the proposed method outperforms CLNN in terms of both MSE metrics, highlighting its superior performance.

The trajectory of the double pendulum system is depicted in Figure 5, illustrating the precision of our method in predicting the system trajectory while adhering to fundamental physical principles. Figure 6 illustrates the temporal accumulation of errors, further emphasizing the superiority of the proposed method over alternative methods. Experimental findings substantiate the effectiveness of the proposed methodology in acquiring knowledge of rigid system dynamics.

We conducted an experiment to examine the impact of stiffness on the learning ability of a dynamical system using a double pendulum system. The stiffness of the system is determined by the disparity in speed between its subprocesses. To manipulate the stiffness, we adjusted the mass parameter m_2 with values of 100, 200, 300, and 400, thereby influencing the stiffness of the system. As the mass difference increases, so does the velocity difference between the subprocesses. The box plot in Figure 7 illustrates the variation in MSE_E associated with the mass parameter. The results indicate a discernible upward trend in the energy error as the system rigidity gradually intensifies. This escalation contributes to heightened complexity and poses challenges to the system's learning process. Nevertheless, the proposed method yields superior results in energy retention compared to other methods for learning dynamical systems. In conclusion, the proposed method effectively learns dynamical systems even under rigid conditions.

TABLE 2 MSE comparison of results for double pendulum system.

Method	MSE_q	MSE_E
Node	7.777e+01 ± 3.742e+01	7.077e+05 ± 3.868e+05
MechanicsNN	7.132e+00 ± 1.315e+01	1.970e+05 ± 4.908e+05
CLNN	1.345e−01 ± 2.838e−02	5.453e+01 ± 4.340e+01
MDANN	9.638e−02 ± 6.056e−02	5.091e−01 ± 5.553e−01

Abbreviations: CLNN, constrained Lagrangian neural networks; MDANN, multiscale differential-algebraic neural network; MSE, mean squared error.

4.3 | Scissor-type deployable mast system

The proposed method is subsequently applied in the context of a scissor-type deployable mast system.[40] The scissor-type deployable mast belongs to a category of structures with a negative Poisson's ratio that offers significant utility in the aerospace domain. These units possess a distinctive capability, enabling seamless transitions between two distinct states: a compact, folded configuration during ground handling and launch and the ability to dynamically expand upon reaching a designated orbital position. The dynamic expansion phase of this process carries the

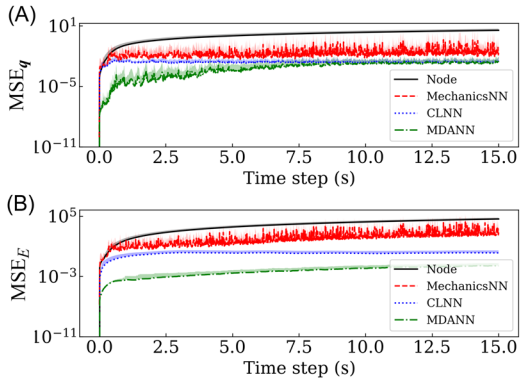

FIGURE 6 Accumulated mean squared error (MSE) over time for double pendulum system: (A) MSE of position, (B) MSE of energy.

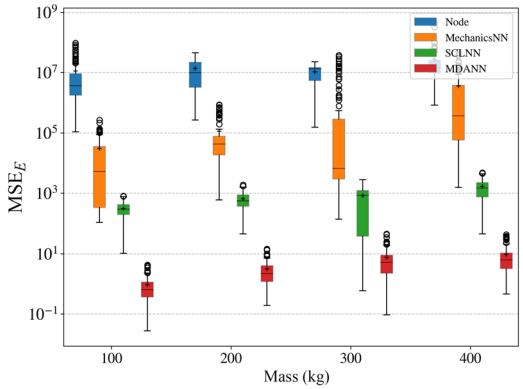

FIGURE 7 Comparison of energy results for double pendulum systems with different masses.

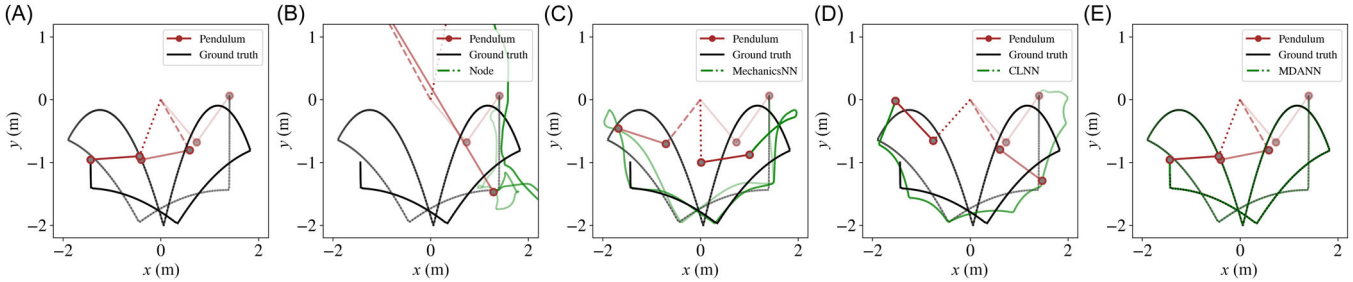

FIGURE 5 Comparison of results for double pendulum system: (A) ground truth, (B) Node, (C) MechanicsNN, (D) constrained Lagrangian neural networks, (E) multiscale differential-algebraic neural network.

potential for failure. Thus, it is necessary to conduct an analysis of the expanding process of scissor-type deployable masts through dynamic simulations and optimize the control forces required during the design phase. In this context, the proposed MDANN method is employed to optimize the control forces, ensuring a smooth and expeditious expansion of the scissor-type deployable mast.

The system is composed of multiple expandable units. Each expandable unit primarily consists of scissor hinges, orthogonal double-joint components, and sliding components configured in a rational manner to provide it with the capability to fold and expand, as show in Figure 8. Each side of the unit incorporates a scissor hinge, while the upper and lower ends are connected by orthogonal double-rotary joints and sliding joint components. The system bears a certain load at its top. Applying a driving force, the bottom orthogonal double-joint actuators move along the diagonals of the bottom quadrilateral, thereby achieving the folding and expanding function of the unit. Throughout this process, the length of the scissor rods remains constant, while the transverse rods adjust their length through the sliding components. In the fully extended state, the angle between the scissor rods of the scissor hinge is 90°.

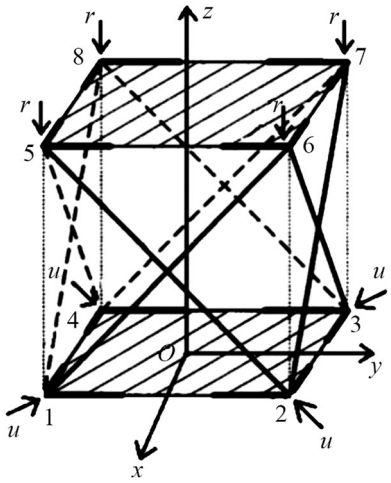

FIGURE 8 Schematic diagram of expandable unit.

When modeling the system using Cartesian coordinates, the state variables can be represented as Equation (17).

$$\boldsymbol{q} = \begin{bmatrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & x_3 & y_3 & z_3 & \cdots \\ x_{4n+1} & y_{4n+1} & z_{4n+1} & x_{4n+2} & y_{4n+2} & z_{4n+2} & & & & \\ x_{4n+3} & y_{4n+3} & z_{4n+3} & x_{4n+4} & y_{4n+4} & z_{4n+4} & & & & \end{bmatrix}^{\mathrm{T}}, \tag{17}$$

where n representing the number of deployable units. The generalized force vector, represented as $\boldsymbol{F}$, is defined in Equation (18).

$$\boldsymbol{F} = \begin{bmatrix} -\frac{\sqrt{2}}{2}u & \frac{\sqrt{2}}{2}u & 0 & -\frac{\sqrt{2}}{2}u & -\frac{\sqrt{2}}{2}u & 0 \\ \frac{\sqrt{2}}{2}u & -\frac{\sqrt{2}}{2}u & 0 & \frac{\sqrt{2}}{2}u & \frac{\sqrt{2}}{2}u & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & -r & 0 & 0 & -r & \\ 0 & 0 & -r & 0 & 0 & -r & \end{bmatrix}^{\mathrm{T}}, \tag{18}$$

where u denotes the control force applied to the system, while the load, represented as r, is maintained at a constant value of 5 N. The dimension of the generalized force vector $\boldsymbol{F}$ is $3(4n + 4) \times 1$. In the

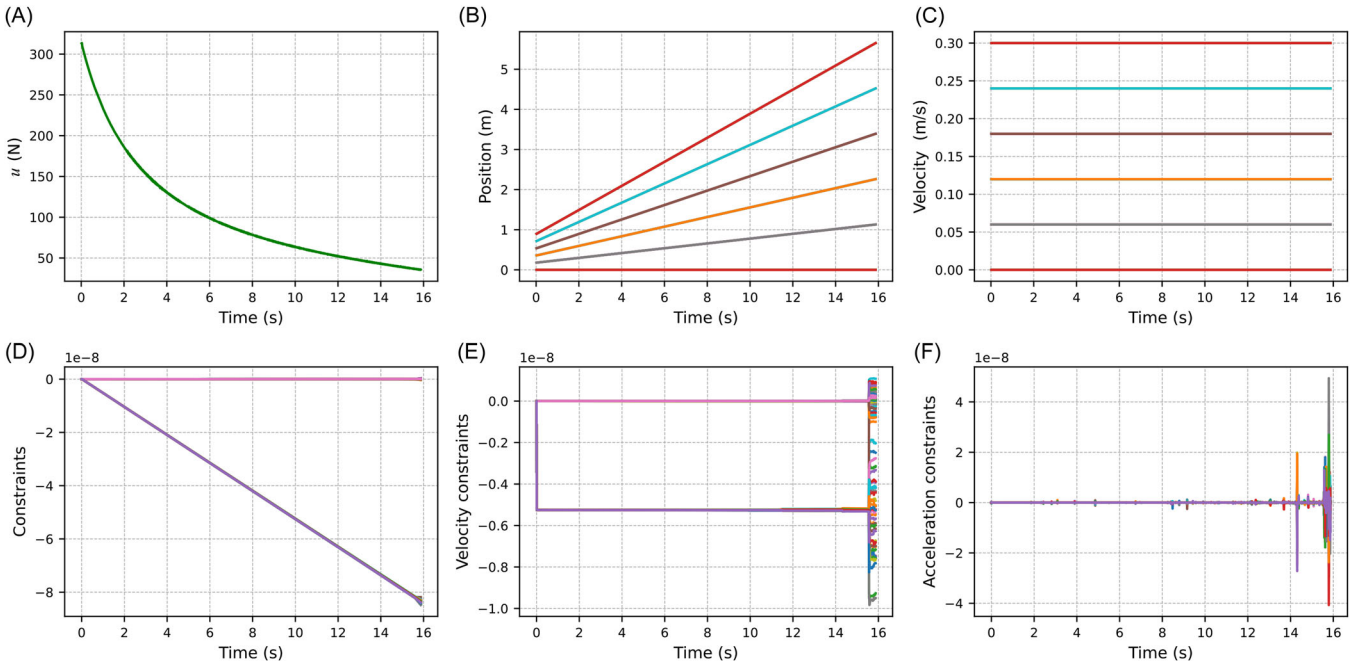

FIGURE 9 The results of the multiscale differential-algebraic neural network method for scissor-type deployable mast system: (A) control force, (B) deployment position, (C) deployment speed, (D) constraints, (E) velocity constraints, and (F) acceleration constraints.

context of a scissor-type deployable mast system, it is pertinent to note that their generalized coordinates are not mutually independent during the expanding process. Consequently, they must adhere to certain constraint conditions. In our analysis, we have taken into account constraints associated with both the rod length constraints $\boldsymbol{\Phi}_1$ and position constraints $\boldsymbol{\Phi}_2$. The constraint equations for the system can be expressed as Equation (19).

$$\boldsymbol{\Phi} = \begin{pmatrix} \boldsymbol{\Phi}_1 \\ \boldsymbol{\Phi}_2 \end{pmatrix} = \mathbf{0}, \tag{19}$$

where the dimensionality of the constraint equation $\boldsymbol{\Phi}$ is $(10 + 19n) \times 1$. In consideration of the quality parameters, we have defined the density of the mast as 3000 kg/m^3, the cross-sectional area as 0.03 m^2, and the length of the shear rod as 1.6 m. The dimension of the mass matrix $\boldsymbol{M}$ is given by $3(4n + 4) \times 3(4n + 4)$. In the system's initial state, the shear rod maintains a 10° inclination in a folded configuration. In this experiment, we have configured the number of expandable units in the system to be $n = 5$.

Our primary objective is to optimize the control force u to ensure a constant ascent rate of 0.3 m/s for the uppermost node of the scissor-type expandable mast. The outcomes of this optimization procedure are illustrated in Figure 9A, which demonstrates a gradual reduction in the control force over time.

To assess the effectiveness of our method in optimizing the control force, we directed our primary focus toward the nodes positioned identically within a specific section of the mast structure. Figure 9B,C illustrates the displacement variations and velocity fluctuations observed at the designated nodes. The experimental findings demonstrate the critical importance of optimizing control forces to ensure consistent and uniform motion across all units within the structure. Notably, we maintained a consistent velocity of approximately 0.3 m/s for the uppermost node, facilitating a seamless and expeditious deployment of the mast.

Moreover, as shown in Figure 9D–F, the system effectively maintains stringent constraints, velocity constraints, and acceleration constraints within precise and highly accurate ranges. In Figure 10, we observe the deployment of the scissor-type deployable mast, transitioning from its initial folded configuration at $t = 0$ s to achieving full extension by $t = 15.88$ s.

Furthermore, we conducted a comparative study to investigate the impact of varying numbers of units on the outcomes. The results are presented in Table 3, illustrating scenarios with 5, 10, and 15 units, respectively. It is evident that as the number of expandable units increases, the maximum height of the structure also rises. Additionally, the proposed method consistently maintains a relatively low maximum error in target speed, hovering around 1e−5. Moreover, the constraints demonstrate stable performance, remaining within a moderate range. To simulate scenarios involving a greater number of scalable units, we utilized Cinema 4D simulation software, as depicted in Figure 11. The experimental results unmistakably indicate that the proposed method effectively

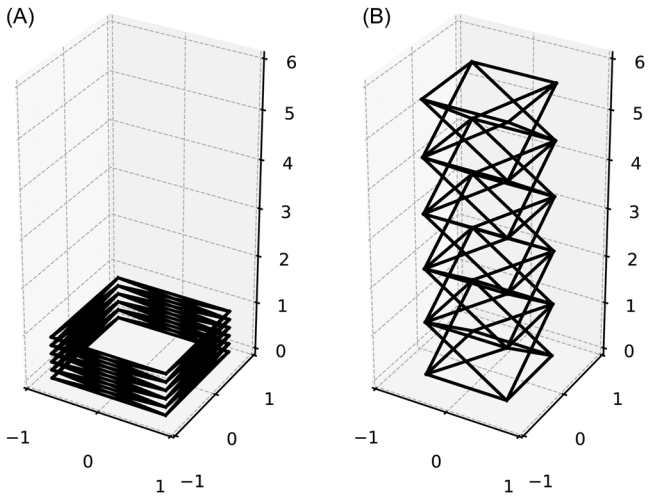

FIGURE 10 Schematic diagram of scissor-type deployable mast system: (A) folded state, (B) extended state.

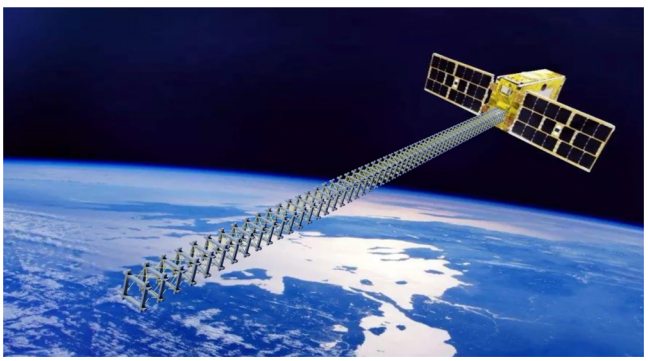

FIGURE 11 Schematic representation of scissor-type deployable mast system.

TABLE 3 Comparison of results for scissor-type deployable mast systems with different numbers of units.

The number of units	The maximum height of the expanded state	The maximum error of the target speed	The maximum error of the constraints	The maximum error of the velocity constraints	The maximum error of the acceleration constraints
5	5.65	9.9987e−06	8.4818e−08	9.8351e−09	4.9397e−08
10	11.31	9.9788e−06	9.2404e−07	2.0118e−07	1.4846e−06
15	16.97	9.9971e−06	4.2815e−07	1.3768e−07	2.7426e−05

optimizes the control force of the system, thereby achieving optimal control of the expansion process.

5 | CONCLUSION

Dynamical system learning is a challenging research area with significant practical applications. In this paper, we propose a novel MDANN method for addressing the challenges of learning dynamic systems, particularly those characterized by rigidity. Our method explicitly enforces constraints using a system of DAE, enabling better learning of Lagrangian quantities. Additionally, we introduce a multiscale structure to mitigate the problem of overlapping frequency components within physical processes. Subsequently, we apply the proposed method to address the challenges posed by the learning of a double pendulum system, a coupled pendulum system, and the optimization of control forces for a scissor-type deployable mast system. While our method exhibits notable enhancements in terms of evaluation metrics, there remains ample scope for performance enhancement, especially in rigid conditions. Future research endeavors should prioritize issues such as elevating the precision of dynamical system learning and unraveling the intricate relationship between sampling step size and learning accuracy.

ACKNOWLEDGMENTS

This study has been supported by the National Natural Science Foundations of China (Nos. 12172186 and 11772166).

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

DATA AVAILABILITY STATEMENT

Data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Jieyu Ding http://orcid.org/0000-0003-2085-5383

REFERENCES

1. Meidani K, Farimani AB. Identification of parametric dynamical systems using integer programming. *Expert Syst Appl*. 2023;219:119622.
2. Ding JY, Pan ZK, Chen LQ. Second order adjoint sensitivity analysis of multibody systems described by differential-algebraic equations. *Multibody Syst Dyn*. 2007;18:599-617.
3. Zheng M. Lie symmetries and conserved quantities of fractional nonconservative singular systems. *Int J Mech Syst Dyn*. 2023;3:274-279.
4. Huang W, Zou H, Pan Y, et al. Numerical simulation and behavior prediction of a space net system throughout the capture process: spread, contact, and close. *Int J Mech Syst Dyn*. 2023;3:265-273.
5. Eghbali M, Hosseini SA. A complex solution on the dynamic response of sandwich graphene-reinforced aluminum-based composite beams with copper face sheets under two moving constant loads on an elastic foundation. *Int J Mech Syst Dyn*. 2023;3:251-264.
6. Rui X, Zhang J, Wang X, Rong B, He B, Jin Z. Multibody system transfer matrix method: the past, the present, and the future. *Int J Mech Syst Dyn*. 2022;2(1):3-26.
7. Legaard C, Schranz T, Schweiger G, et al. Constructing neural network-based models for simulating dynamical systems. *Acm Comput Surv*. 2023;55(11):1-34.
8. Wang R, Walters R, Yu R. Approximately equivariant networks for imperfectly symmetric dynamics. Paper presented at: 39th International Conference on Machine Learning; July 17-23, 2022; Baltimore, MD. https://proceedings.mlr.press/v162/wang22aa.html
9. Wang K, Aanjaneya M, Bekris K. A first principles approach for data-efficient system identification of spring-rod systems via differentiable physics engines. Paper presented at: 2nd Conference on Learning for Dynamics and Control; June 10-11, 2020; The Cloud. https://proceedings.mlr.press/v120/wang20b.html
10. Yang TY, Rosca J, Narasimhan K, Ramadge PJ. Learning physics constrained dynamics using autoencoders. *Proc Adv Neural Inf Process Syst*. 2022;35:17157-17172.
11. Willard J, Jia X, Xu S, Steinbach M, Kumar V. Integrating scientific knowledge with machine learning for engineering and environmental systems. *Acm Comput Surv*. 2022;55(4):1-37.
12. Lutter M, Ritter C, Peters J. Deep Lagrangian networks: Using physics as model prior for deep learning. Paper presented at: 7th International Conference on Learning Representations; May 6-9, 2019; New Orleans, LA. https://openreview.net/forum?id=BklHpjCqKm
13. Yu H, Lu J, Zhang G. An online robust support vector regression for data streams. *Ieee T Knowl Data En*. 2020;34(1):150-163.
14. Shamshirband S, Hashemi S, Salimi H, et al. Predicting standardized streamflow index for hydrological drought using machine learning models. *Eng Appl Comp Fluid*. 2020;14(1):339-350.
15. Gomila R. Logistic or linear? Estimating causal effects of experimental treatments on binary outcomes using regression analysis. *J Exp Psychol Gen*. 2021;150(4):700-709.
16. Brunton SL, Proctor JL, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *P Natl Acad Sci Usa*. 2016;113(15):3932-3937.
17. Schaeffer H, McCalla SG. Sparse model selection via integral terms. *Phys Rev E*. 2017;96(2):023302.
18. Reinbold PA, Gurevich DR, Grigoriev RO. Using noisy or incomplete data to discover models of spatiotemporal dynamics. *Phys Rev E*. 2020;101(1):010203.
19. Berg J, Nyström K. A unified deep artificial neural network approach to partial differential equations in complex geometries. *Neurocomputing*. 2018;317:28-41.
20. Gong S, Meng Q, Wang Y, et al. Incorporating NODE with pre-trained neural differential operator for learning dynamics. *Neurocomputing*. 2023;528:48-58.
21. Lu L, Jin P, Pang G, Zhang Z, Karniadakis GE. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat Mach Intell*. 2021;3(3):218-229.
22. Finzi M, Stanton S, Izmailov P, Wilson AG. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. Paper presented at: 37th International Conference on Machine Learning; July 13-18, 2020; Virtual. https://proceedings.mlr.press/v119/finzi20a.html
23. Wang H, Liu Y, Wang S. Dense velocity reconstruction from particle image velocimetry/particle tracking velocimetry using a physics-informed neural network. *Phys Fluids*. 2022;34(1):017116.
24. Lin S, Chen Y. A two-stage physics-informed neural network method based on conserved quantities and applications in localized wave solutions. *J Comput Phys*. 2022;457:111053.
25. Haghighat E, Raissi M, Moure A, Gomez H, Juanes R. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Comput Method Appl Mech Eng*. 2021;379:113741.
26. Goswami S, Anitescu C, Chakraborty S, Rabczuk T. Transfer learning enhanced physics informed neural network for phase-field modeling of fracture. *Theor Appl Fract Mec*. 2020;106:102447.

27. Wang S, Wang H, Perdikaris P. Learning the solution operator of parametric partial differential equations with physics-informed deeponets. *Sci Adv*. 2021;7(40):eabi8605.
28. Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J Comput Phys*. 2019;378:686-707.
29. Chen RT, Rubanova Y, Bettencourt J, Duvenaud DK. Neural ordinary differential equations. *Proc Adv Neural Inf Process Syst*. 2018;31:6571-6583.
30. Cunha BZ, Droz C, Zine AM, Foulard S, Ichchou M. A review of machine learning methods applied to structural dynamics and vibroacoustic. *Mech Syst Signal Pr*. 2023;200:110535.
31. Baddoo PJ, Herrmann B, McKeon BJ, Nathan Kutz J, Brunton SL. Physics-informed dynamic mode decomposition. *P Roy Soc A-Math Phy*. 2023;479(2271):20220576.
32. Greydanus S, Dzamba M, Yosinski J. Hamiltonian neural networks. *Proc Adv Neural Inf Process Syst*. 2019;32:15379-15389.
33. Cranmer M, Greydanus S, Hoyer S, Battaglia P, Spergel D, Ho S. Lagrangian neural networks. Paper presented at: Advances in Neural Information Processing Systems 33; March 26-27, 2020; The Cloud. https://openreview.net/forum?id=iE8tFa4Nq
34. Finzi M, Wang KA, Wilson AG. Simplifying Hamiltonian and Lagrangian neural networks via explicit constraints. *Proc Adv Neural Inf Process Syst*. 2020;33:13880-13889.
35. Lu Y, Lin S, Chen G, Pan J. Modlanets: Learning generalisable dynamics via modularity and physical inductive bias. Paper presented at: 39th International Conference on Machine Learning; July 17-23, 2022; Baltimore, MD. https://proceedings.mlr.press/v162/lu22c.html
36. Gruver N, Finzi MA, Stanton SD, Wilson AG. Deconstructing the inductive biases of Hamiltonian neural networks. Paper presented at: 10th International Conference on Learning Representations; April 25-29, 2022; Virtual. https://openreview.net/forum?id=EDeVYpT42oS
37. Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. *Nat Rev Phys*. 2021;3(6):422-440.
38. Wang B. Multi-Scale Deep Neural Network (MscaleDNN) methods for oscillatory stokes flows in complex domains. *Commun Comput Phys*. 2020;28(5):2139-2157.
39. Liu Z. Multi-Scale Deep Neural Network (MscaleDNN) for solving Poisson-Boltzmann equation in complex domains. *Commun Comput Phys*. 2020;28(5):1970-2001.
40. Ding J, Liu J. Optimal control based on modified genetic algorithm for the deployment process of scissor-type deployable mast. Paper presented at: 8th Asian Conference on Multibody Dynamics; August 7-10, 2016; Kyoto, KP. https://www.jstage.jst.go.jp/article/jsmeacmd/2016.8/0/2016.8_29_1289270/_article/-char/ja/

AUTHOR BIOGRAPHIES

Yin Huang was born in 1995. He is currently a PhD candidate at the College of Computer Science & Technology, Qingdao University, China. He received his Master's degree from Liaoning University of Technology, China, in 2021. His research interest is multibody dynamics and computer vision.

Jieyu Ding is currently a professor at the School of Mathematics and Statistics, and the Center for Computational Mechanics and Engineering Simulation of Qingdao University, China. She received her PhD degree in Shanghai University, and Shanghai Institute of Applied Mathematics and Mechanics, China, in 2008. Her research interest includes multibody system dynamics, design optimization, and optimal control.

How to cite this article: Huang Y, Ding J. A multiscale differential-algebraic neural network-based method for learning dynamical systems. *Int J Mech Syst Dyn*. 2024;4:77-87. doi:10.1002/msd2.12102