ACADÉMIE
DES SCIENCES
INSTITUT DE FRANCE

Comptes Rendus
Mécanique

Elias Al Ghazal, Jad Mounayer, Beatriz Moya, Sebastian Rodriguez, Chady Ghnatios and Francisco Chinesta

Application of reduced-order models for temporal multiscale representations in the prediction of dynamical systems

Volume 354 (2026), p. 203-226

Online since: 24 March 2026

https://doi.org/10.5802/crmeca.355

The Comptes Rendus. Mécanique *are a member of the Mersenne Center for open scientific publishing*
www.centre-mersenne.org — e-ISSN : 1873-7234

ACADÉMIE DES SCIENCES
INSTITUT DE FRANCE

Comptes Rendus. Mécanique
2026, Vol. 354, p. 203-226
https://doi.org/10.5802/crmeca.355

Research article

Application of reduced-order models for temporal multiscale representations in the prediction of dynamical systems

Elias Al Ghazal [*,a], Jad Mounayer [a], Beatriz Moya [a], Sebastian Rodriguez [a], Chady Ghnatios [b] and Francisco Chinesta [a,c]

[a] PIMM Lab, ENSAM Institute of Technology, 75013 Paris, France

[b] University of North Florida, Department of Mechanical Engineering, 1 UNF Drive, Jacksonville, FL 32224, USA

[c] CNRS@CREATE LTD, Singapore

E-mail: elias.al_ghazal@ensam.eu

Abstract. Modeling and predicting the dynamics of complex multiscale systems remains a significant challenge due to their inherent nonlinearities and sensitivity to initial conditions, as well as limitations of traditional machine learning methods that fail to capture high frequency behaviors. To overcome these difficulties, we propose three approaches for multiscale learning. The first leverages the Partition of Unity (PU) method, integrated with neural networks, to decompose the dynamics into local components and directly predict both macro- and micro-scale behaviors. The second applies the Singular Value Decomposition (SVD) to extract dominant modes that explicitly separate macro- and micro-scale dynamics. Since full access to the data matrix is rarely available in practice, we further employ a sparse high-order SVD to reconstruct multiscale dynamics from limited measurements. Together, these approaches ensure that both coarse and fine dynamics are accurately captured, making the framework effective for real-world applications involving complex, multi-scale phenomena and adaptable to higher-dimensional systems with incomplete observations, by providing an approximation and interpretation in all time scales present in the phenomena under study.

Keywords. Multiscale dynamics, partition of unity, neural networks, singular value decomposition, sparse high order SVD, model reduction, data-driven modeling.

Funding. This work was supported by the IN-DEEP European Project. This Project has received funding from the EU's Horizon Europe research and innovation programme under the Marie Sklodowska-Curie GA No. 101119556. This research is also part of the DesCartes programme and is supported by the National Research Foundation, Prime Minister Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme. We acknowledge support from the French government, managed by the National Research Agency (ANR), under the CPJ ITTI.

Note. Submitted by invitation following the DTE-AICOMAS 2025 conference, held February 17-21, 2025.

Manuscript received 16 October 2025, revised and accepted 3 March 2026, online since 24 March 2026.

* Corresponding author

1. Introduction

Understanding the dynamics of systems governed by nonlinear behaviors is essential in many scientific and engineering fields, as it enables the prediction and control of complex processes [1]. Particularly, whether forecasting the weather, designing mechanical systems, or analyzing financial markets, accurately modeling and predicting systems characterized by multiple time scales remains a significant challenge. These systems exhibit a rich interplay between slow, long-term trends and fast, short-lived events, a phenomenon known as multi-scale dynamics [2]. Capturing both fast and slow dynamics simultaneously is further complicated by the inherent nonlinearity of many systems [3,4]. For example, in climate modeling, long-term trends such as global warming (coarse dynamics) interact with short-term fluctuations like weather changes (fine dynamics) [5]. These time scales are deeply intertwined, with short-term events influencing long-term trends and vice versa, making it difficult to capture the full scope of the system's behavior in a single model or approximation. The complexity and interplay of fast and slow processes can lead to chaotic patterns that are difficult to predict or understand without sophisticated techniques. Similarly, in mechanical systems, the interaction between high-frequency oscillations and low-frequency movements requires careful consideration of both fine and coarse dynamics. Such systems can exhibit sudden, dramatic changes in behavior depending on initial conditions or minor perturbations [6]. These challenges are amplified when the governing function $\dot{x} = f(x)$ that describes the system's dynamics is unknown.

When $\dot{x} = f(x)$ is known, traditional techniques such as numerical integration can be employed to solve the resulting differential equations and model the time evolution of the system's state $x(t)$. However, in many practical applications, the exact form of $f(x)$ is unknown, and only discrete observations of the system's state are available at different time points [7]. In such cases, the challenge shifts to inferring, or learning, the unknown forcing function $f(x)$ that drives the system's evolution. In machine learning, this is particularly challenging because the system is often observed as time-series data, and the goal is to discover the underlying dynamics that govern its behavior.

Several data-driven black-box machine learning methods have been proposed for this purpose [8,9]. However, model discovery is unstable, specially in multiscale scenarios, and can lead to explosion or vanishing of gradients [10]. To regularize the process, some methods leverage sparsity-promoting techniques to identify parsimonious models that accurately capture system dynamics, even in the presence of noise or limited data. However, these models may struggle when system dynamics involve multiple time scales or when the Fourier spectrum of $f(x)$ contains both low and high frequencies [11]. As stated in [12], physics-informed machine learning increases stability effects by leading the learning of dynamical system approximations towards local minima closer to the absolute minimum solution due to the imposition of physical restrictions, such as PDEs [7], thermodynamics [13–15], and other types of biases [16]. Moreover, structures such as Koopman operators [17–20], Sparse Identification of Nonlinear Dynamics (SINDy) [21], and neural operators [22–24], with DeepONets [25–27] being a particularly appealing option, have demonstrated great efficiency in this task. Nonetheless, these methods are susceptible to overfitting to the dominant low-frequency scales, thereby neglecting the high-frequency phenomena.

This is due to the fact that, in spite of the use of learning and physics biases, neural networks learn inherently low frequencies, and learning the fast frequencies tends to be part of a overfitting issue due to the optimization process. Authors in [28] showcase how deep neural networks suffer from a strong bias to low frequencies, which are learned faster than the high frequencies. These results are also supported by Zhi-Qin John Xu et al. [29].

Most solutions to this challenge propose learning in the frequency domain [29], where Fourier

neural operators (FNO) is one of the main methods in this field [25,30]. However, it is not always possible to learn efficiently in the frequency domain. In dynamical systems governed by $\dot{x} = f(x)$ the evolution of the state may depend solely on its current value through the function $f(x)$, rather than explicitly on time. Consequently, learning this problem in the frequency domain is challenging since the behavior is defined in the state space and not in the temporal domain.

Following this idea, and as a solution for the oversmoothing issues of machine learning, recent advances in dynamical systems modeling have focused on separating the dynamics into distinct levels: coarse (long-term) and fine (short-term) scales [31–33], proposing a sort of hierarchical learning scheme[34]. By isolating these scales, each can be modeled independently, improving both accuracy and interpretability [33]. This separation allows for a more precise understanding of how different time scales interact, without the computational inefficiencies and inaccuracies that arise from modeling them together [2].

The Multi-scale Deep Neural Network (MscaleDNN), for instance, captures features at multiple spatial scales [35]. In the field of physics-informed machine learning, we also find works dedicated to the treatment of different scales [36–38]. Such is the case of Wu et al. [39], which solves time-dependent linear transport equations using the so-called Asymptotic-Preserving Convolutional Deep Operator Networks (APCONs). In this case, separation of scales is performed by using the Knudsen number. However, in many cases, the identification of scales is not apparent or may remain undetermined in the absence of effective separation strategies. Often, only a bifurcation between two scales is performed, despite the fact that each scale may comprise multiple principal modes.

One effective technique for this separation is the use of structure-preserving and partitioning strategies that decompose the system's dynamics into local approximations valid within specific regions of the domain [40], such as in the case of the Partition of Unity (PU) [2]. By leveraging these approaches, both coarse and fine dynamics can be represented separately, ensuring that the model focuses on the relevant scales at each level and improving modeling accuracy and efficiency.

Also, reduced-order models aim to address this issue from a multiscaling approach by emphasizing the emergence of the principal modes inherent in the problems by contributing to the refinement of the learning process by unveiling the dynamic patterns inherent within its modes [41,42].

In this paper, we showcase the use of three different approaches to model the macro- and micro-scale dynamics of complex systems, improving both the accuracy and efficiency of predictions. These approaches are motivated by two key challenges: (i) the inherently multiscale nature of the problems, and (ii) their highly nonlinear dynamics with strong sensitivity to initial conditions. The first method employed is the Partition of Unity (PU), which is traditionally used for function decomposition in order to study local effects. In this case, the decomposition itself is learned from data to learn and reconstruct the original $f(x)$ by capturing both local variability and global structure [31]. The second approach tested in this work is the SVD-based approach to extract both macro- and micro-scale dynamics through dominant modes. The third approach proposed allows us to utilize a sparse high-order SVD, which allows learning multiscale modes from sparse measurements and provides a foundation for extending these techniques to higher-dimensional systems. Collectively, these contributions improve the accuracy and efficiency in the learning process of $f(x)$ without a priori knowledge of the separation of scales for complex nonlinear systems with strong sensitivity to initial conditions.

Consequently, this work is structured as follows. Section 2 elaborates on the use of PU for the approximation of multiscale dynamical systems. Section 3 presents the use of Singular Value Decomposition (SVD) for multiscale function approximations. Finally, In Section 4 we introduce the use of a spare-high order SVD learned with neural networks hierarchically to approximate the

different modes and scales of which the function is composed of, leading to a method to tackle also multiscale learning in multidimensional problems from sparse measurements.

2. Partition of unity method

The Partition of Unity (PU) method [43], widely used in computational mechanics, offers an effective means of handling multiscale dynamics in dynamical systems. Traditional models often struggle to capture the complex interactions between scales, and this limitation is further exacerbated in many machine learning approaches due to the well-known spectral bias issue [44]. The problem is especially pronounced when the system exhibits nonlinear and high-frequency behavior. To address these challenges, we propose a framework that integrates the PU method with machine learning. Specifically, we use a neural network to model the macro-scale dynamics and a trainable vector to capture fine-scale enrichments (as shown in Figure 1).

2.1. *Coarse (macro) approximation*

We begin with a coarse approximation of the governing dynamics $\dot{x} = f(x)$, represented as:

$$f(x) = \sum_{i=1}^{N} F_i N_i(x),$$

where $N_i(x)$ are piecewise linear shape functions that are equal to 1 at the macro node x_i and 0 at the neighboring macro nodes. Here, F_i is the nodal macro value corresponding to the macro node x_i (see the top of Figure 1(a)). This formulation provides a coarse-scale, macro-level representation of the system dynamics. However, it has limited ability to capture fine-scale transients.

2.2. *Fine-scale (micro) enrichment*

To address this limitation, the coarse approximation is augmented with a fine-scale enrichment that enhances the resolution of the system's dynamics at selected locations [2]. The enriched approximation can be written as:

$$f(x) = \sum_{i=1}^{N} F_i\, N_i(x)\, E_i(x),$$

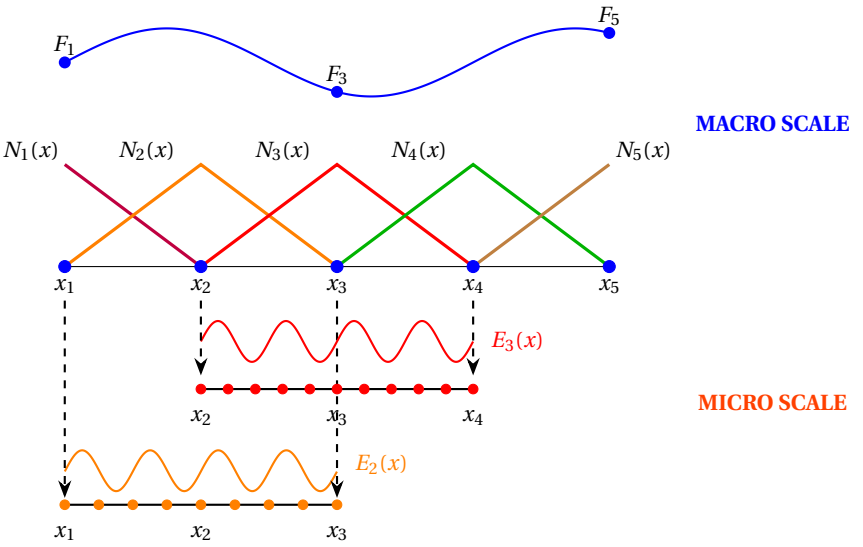
$$E_i(x) = \begin{cases} G(x - x_i), & \text{if } x \in \Omega_i, \\ 0, & \text{otherwise.} \end{cases}$$

and Ω_i is a local micro-domain centered at macro node x_i, defined to be twice the size of the macro element (see the bottom of Figure 1(a)).

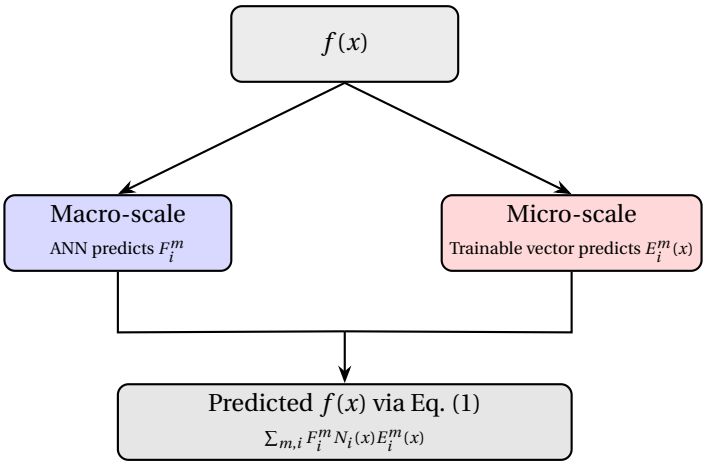
Although using a single enrichment function $G(x)$ is computationally efficient, it may not suffice to resolve the richness of the dynamics in complex systems. To overcome this limitation, a parsimonious multi-enrichment approach can be adopted. This approach is inspired by methods such as the Proper Generalized Decomposition (PGD) [45], which allow for the decomposition of the system's dynamics into multiple modes. The multi-enrichment formulation is given by:

$$f(x) = \sum_{m=1}^{M} \sum_{i=1}^{N} F_i^m\, N_i(x)\, G^m(x - x_i). \tag{1}$$

In this formulation, M represents the number of enrichment functions, and F_i^m and $G^m(x - x_i)$ are the unknown coefficients and enrichment functions, respectively, for each enrichment level m. The multi-enrichment approach captures the different frequencies present in the system's



(a) Shape functions of a multiscale approach. Top: macro shape functions provide coarse-scale approximation. Bottom: micro shape functions capture fine-scale details.



(b) Decomposition of $f(x)$ into macro- and micro-scale components. Macro-scale dynamics are predicted by a neural network (ANN), while micro-scale dynamics are captured by a trainable vector. The two components are combined to reconstruct the enriched approximation of $f(x)$.

Figure 1. Workflow for partition of unity.

dynamics by considering multiple levels of enrichment, with each level corresponding to a different frequency band. The first mode ($m = 1$) captures the low-frequency dynamics, while subsequent modes capture higher-frequency components. This hierarchical enrichment process allows the model to efficiently resolve multi-scale dynamics in a manner that progressively captures more detailed features of the system's behavior.

2.3. *Learning procedure*

To effectively learn the dynamics of the system from data, we parameterize the enrichment-based formulation using machine learning models. Specifically, we decompose the representation into two learnable components (see Figure 1(b)).

Macro-scale component: The macro coefficients F_i^m for each enrichment level m are modeled as outputs of a neural network with the corresponding x_i. The network captures the coarse-scale, low-frequency behavior of the system and learns a smooth, global representation across the domain.

Micro-scale component: In this framework, the micro-scale component E_i is decoupled from the global coordinate system by modeling it as a shared trainable vector rather than a functional mapping of the spatial input x. Because the micro-scale features are assumed to be spatially invariant relative to the macro-nodes, the enrichment does not require pointwise evaluation; instead, a consistent discrete profile is mapped onto every macro-element. By removing the dependence on a functional mapping $f(x)$, the micro-scale profile is treated directly as a set of optimization variables. These vector entries are updated during the training phase to learn the optimal local representation that minimizes the global residual, allowing the same micro-scale profile to be repeated across the entire domain independently of any absolute coordinates.

The key idea is that the macro component provides a smooth global approximation, while the micro component enhances local resolution using a compact, shared vector. This separation ensures that high-resolution details do not overwhelm the neural network's capacity, allowing it to generalize better across the domain. During training, both the neural network parameters and the enrichment vectors are jointly optimized by minimizing a loss function that compares the predicted output with the actual data. This learning approach leverages the structure-preserving properties of the partition of unity method while allowing for efficient, data-driven learning of multiscale system dynamics. The result is a compact and interpretable model that captures both global trends and localized transients with high fidelity.

2.4. *Numerical results*

In this section, we present some numerical results obtained by applying the Partition of Unity (PU) method to model and predict the dynamics of several types of dynamical systems. For each of these systems, data is generated by simulating their evolution under various initial conditions. The conditions span a broad range of states, ensuring that the model is exposed to diverse trajectories and system dynamics. Each data point consists of pairs $(x, \dot{x})$, where x represents the state of the system and $\dot{x}$ represents its time derivative.

The dataset is split into a training set (80% of the data) and a testing set (20% of the data). The training set is used to train the model, while the testing set is used to evaluate the model's predictive performance. Prior to splitting, the data is shuffled randomly to avoid any bias due to the ordering of the data.

We tested the partition of unity method using different numbers of modes, including setups with 1, 2, or more macro and micro modes. These configurations allow us to assess the impact of including different modes on the model's ability to capture both the global and fine-scale dynamics. By using a combination of both macro and micro modes, the model is able to capture large-scale global trends as well as fine-scale local fluctuations.

To determine the appropriate number of modes in the partition of unity framework, we adopt an adaptive enrichment strategy. Specifically, the model is trained incrementally by adding one mode at a time. After each enrichment step, the model's performance is evaluated

using the mean squared error (MSE) computed on the validation set. If the MSE remains above a predefined threshold of 10^{-2}, an additional mode is introduced to further enhance the model's representational capacity. This process continues until the MSE falls below the threshold, indicating that the model has achieved a satisfactory level of accuracy. In this way, the enrichment process is guided by a quantitative stopping criterion rather than a fixed number of modes, ensuring an optimal balance between model accuracy and computational efficiency.

In our implementation, the macro components of the PU model are represented using feed-forward neural networks. The macro neural network consists of three fully connected layers: the first maps the input to 64 hidden units, followed by a ReLU activation; the second maps to 32 hidden units with another ReLU; and the final layer outputs a scalar value. The micro components are modeled as trainable vectors that are directly optimized during training.

Training is performed over 300 epochs using the Adam optimizer with a learning rate of 10^{-3} and a weight decay of 10^{-4}. The loss function used to train the model is a normalized relative error defined as:

$$\text{Loss}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\|\boldsymbol{x} - \boldsymbol{y}\|_2}{\|\boldsymbol{y}\|_2} \times 100,$$

where $\boldsymbol{x}$ is the model output and $\boldsymbol{y}$ is the ground truth, and $\|\cdot\|_2$ denotes the L^2 norm.

After training, the model is first evaluated on the held-out testing set. To further assess its generalization ability, we then test the model using only the initial condition x_0. From this starting point, the trained model predicts the system's evolution over time via Runge–Kutta integration. The resulting trajectories are compared with the corresponding ground-truth dynamics.

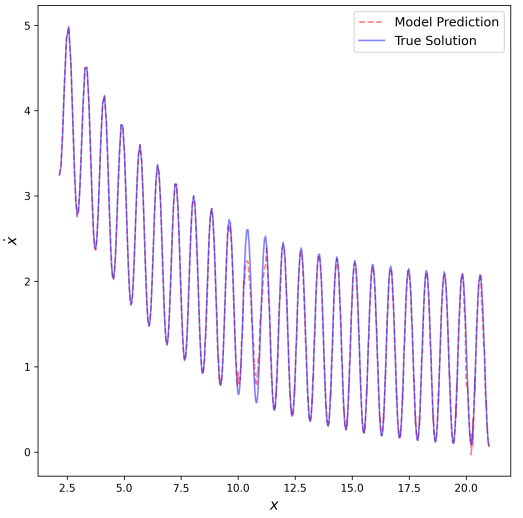
We begin with a simple example and gradually move to more complex systems. The aim is to evaluate the performance of the method in capturing both the macro and micro scales of the system's behavior while maintaining the computational efficiency. The macro and micro components depicted in the figures are generated by evaluating the distinct factors defined in Eq. (1). The macro-scale plots represent the function $F(x)$, which is defined over the global domain and evaluated as a function of the spatial input x. Conversely, the micro-scale plots visualize the shared trainable vector, which remains independent of the global coordinate system. To represent the micro-scale component graphically, the entries of this shared vector are mapped to a normalized local coordinate system within each macro-element. This formulation ensures that while $F(x)$ captures global variations, the micro-component provides a spatially invariant local profile that is repeated identically across every macro-element, regardless of its absolute position in the mesh.

2.4.1. *A first example*

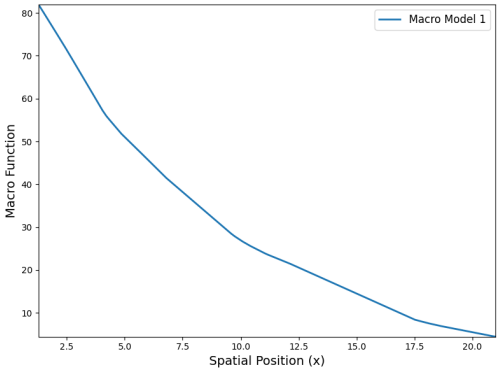
We start with a simple example, given by the following function:

$$f(x) = \sin(8x) + 5\exp(-0.2x) + 1. \tag{2}$$

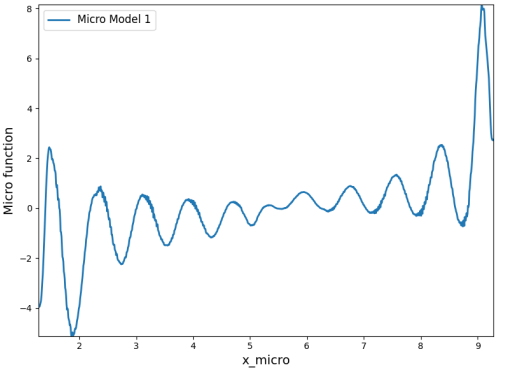
This system involves a combination of sinusoidal and exponential terms. For this simpler system, it is sufficient to use only one mode to achieve accurate predictions, as the resulting mean squared error (MSE) is 0.008, which lies below the predefined threshold of 10^{-2}. Specifically, the domain is discretized using 5 macro-elements, where each macro-element contains a corresponding micro-scale discretization defined by the shared trainable vector. By starting with such a simple example, we can build an understanding of the method and the foundation for more complex systems. The plots for the function, as well as the macro and micro components, are illustrated in Figure 2.



(a) Function plot for Eq. (2).



(b) Macro component of Eq. (2).



(c) Micro component of Eq. (2).

Figure 2. PU approximation for Eq. (2).

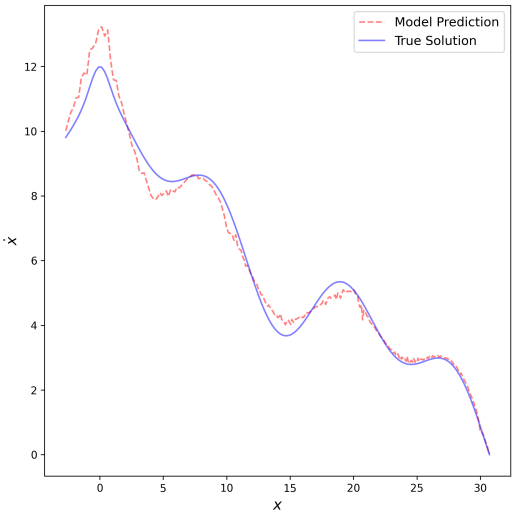
2.4.2. *A second example*

In this section, we consider a more complex example described by the following function:

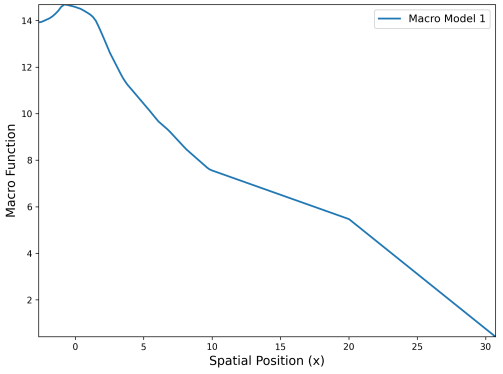
$$f(x) = A\left(\frac{\sin\left(\frac{1}{3}x\right) + \cos\left(\frac{2}{3}x\right) + \exp(-x^2) + \text{const}}{c}\right) - k \cdot x. \tag{3}$$

This example introduces more intricate dynamics, combining two sinusoidal functions at different frequencies with exponential terms and a linear component. The system exhibits more complex behavior, requiring the use of a multi-modes approximation to capture both the global and fine-scale dynamics. To resolve these features, we employ a 4 macro-elements, with the underlying local discretization remaining uniform across all elements to facilitate efficient learning.

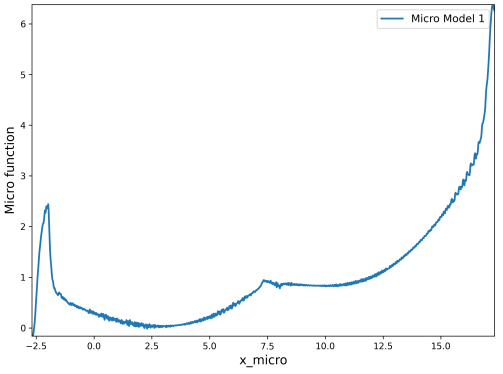
We first show the results obtained using a single mode, which clearly illustrates that the model does not exhibit a good fitting. The mean squared error (MSE) in this case is 0.276, which is above the predefined threshold value, confirming that a single mode is insufficient to capture the full behavior of the system. Figure 3 shows the overall function along with its corresponding macro



(a) Function plot using one mode for Eq. (3).



(b) Macro component using one mode for Eq. (3).



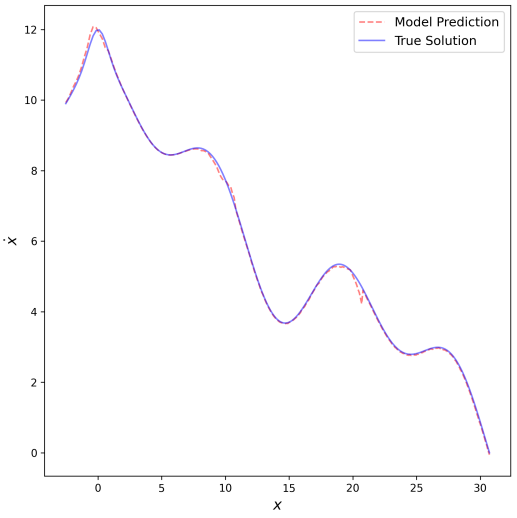
(c) Micro component using one mode for Eq. (3).

Figure 3. PU approximation for Eq. (3) using one mode.

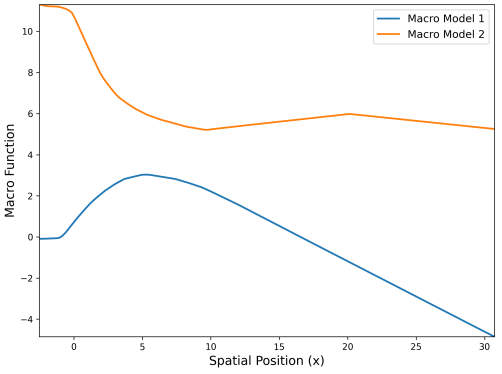
and micro components. These results highlight that the model does not fit well with only one mode, which is expected since the signal contains two distinct frequencies.

However, in this two-mode setup, the models are trained sequentially, in a parsimonious manner. The sequential training is handled using a wrapper module that allows multiple PU models to be combined and executed in sequence, ensuring that each mode is trained independently before composing the final model.

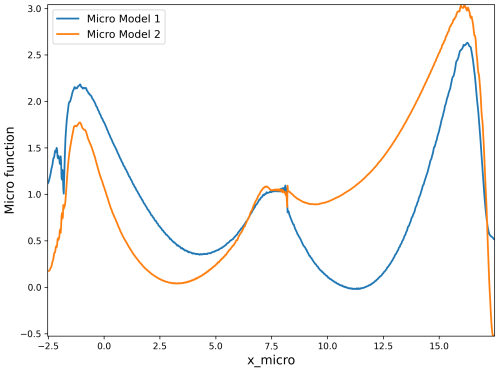
By using two modes, the results improve significantly, providing a much better fit to both the global and fine-scale dynamics. The model achieves an MSE of 0.04, confirming the accuracy and adequacy of the two-mode configuration. Figure 4 presents the improved function along with its corresponding macro and micro components. The addition of the second mode significantly enhances the fit, capturing both the large-scale and fine-scale dynamics more accurately.



(a) Function plot using two modes for Eq. (3).



(b) Macro component using two modes for Eq. (3).



(c) Micro component using two modes for Eq. (3).

Figure 4. PU approximation for Eq. (3) using two modes.

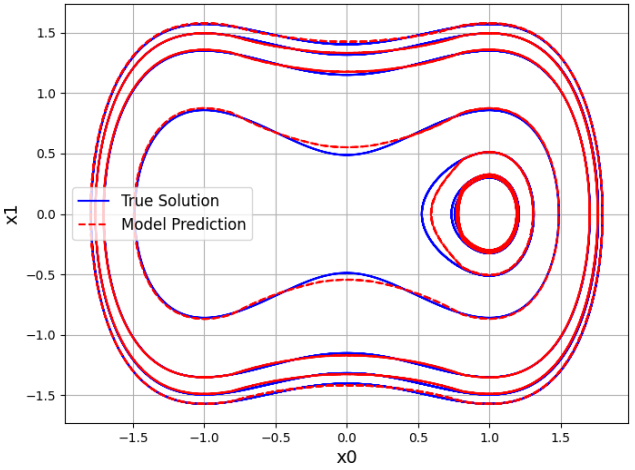
2.4.3. *Coupled systems*

Duffing system. The Duffing system is governed by the following differential equations:

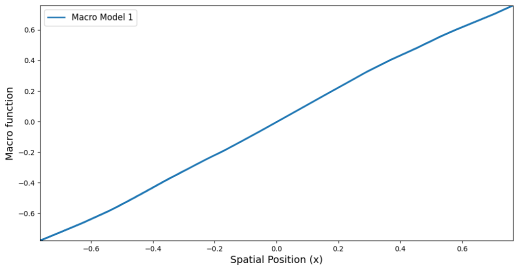
$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_0 - x_0^3 \end{bmatrix}. \tag{4}$$

Both variables x_0 and x_1 interact nonlinearly. To simplify learning, we assume $\dot{x}_1$ depends on x_0 and $\dot{x}_0$ depends on x_1, allowing the system to be split into two separate 1D problems. Two Partition of Unity (PU) models were trained: one predicts $\dot{x}_1$ from x_0, and the other predicts $\dot{x}_0$ from x_1. For this dynamical system, the input space is partitioned into 4 macro-elements, with their micro-elements.

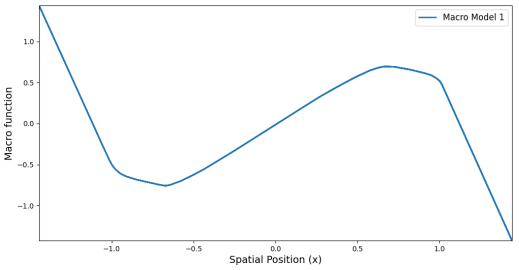
The results demonstrate that the PU method effectively captures the dynamics of the Duffing system, achieving a mean squared error of 0.001. Figure 5 shows the function and its corresponding macro and micro components, providing accurate predictions of the system behavior.



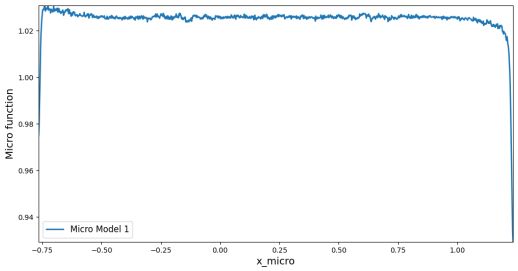
(a) Function plot for Eq. (4).



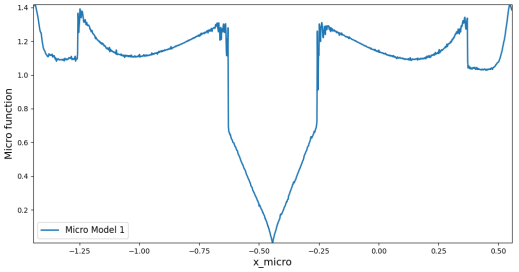
(b) Macro component 1 for Eq. (4).



(c) Macro component 2 for Eq. (4).



(d) Micro component 1 for Eq. (4).



(e) Micro component 2 for Eq. (4).

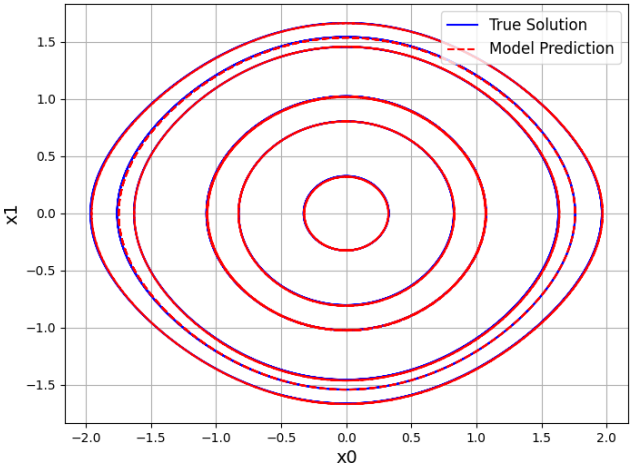
Figure 5. PU approximation for Eq. (4).

Harmonic oscillator system. The harmonic oscillator system is described by:

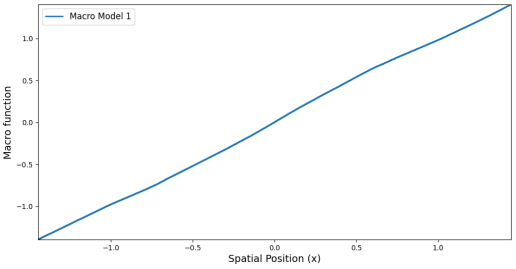
$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} x_1 \\ -\sin(x_0) \end{bmatrix}. \tag{5}$$

Similarly, we train two PU models for the oscillator: one predicting $\dot{x}_1$ from x_0 and the other predicting $\dot{x}_0$ from x_1, yielding two macro and micro components. The computational domain is discretized using 4 macro-elements, integrated with a refined micro-scale discretization to ensure high-fidelity capturing of the periodic oscillations.

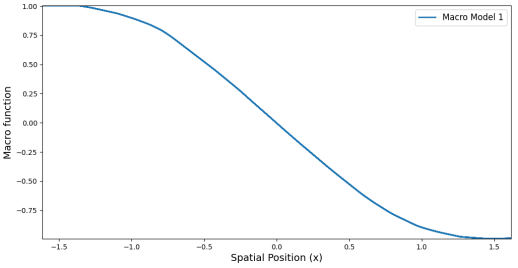
The results show that the PU method successfully models the harmonic oscillator, achieving a mean squared error (MSE) of 0.0006. Figure 6 shows the function and its corresponding macro



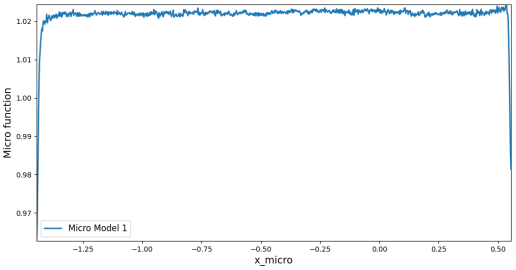
(a) Function plot for Eq. (5).



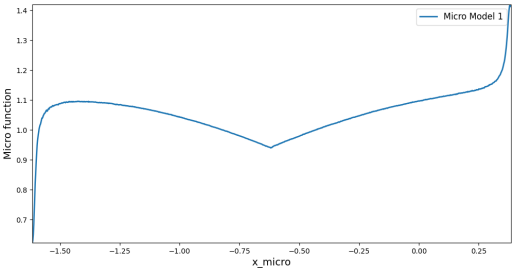
(b) Macro component 1 for Eq. (5).



(c) Macro component 2 for Eq. (5).



(d) Micro component 1 for Eq. (5).



(e) Micro component 2 for Eq. (5).

Figure 6. PU approximation for Eq. (5).

and micro components, capturing both coarse and fine-scale dynamics with high accuracy and efficiency.

2.4.4. *Summary of PU results*

In summary, the numerical results for all the considered systems demonstrate that the PU method is an effective tool for modeling and predicting complex, multi-scale dynamical systems. By separating each system's dynamics into coarse and fine scales, the method efficiently captures both global trends and detailed fluctuations. This provides a comprehensive view of the system's behavior with high accuracy and computational efficiency, as illustrated by the results for the Duffing and harmonic oscillator systems.

3. Singular Value Decomposition (SVD) for multi-scale function extraction

The Partition of Unity (PU) method constructs local approximations of the system's dynamics, capturing both macro- and micro-scale behaviors. Another method for analyzing the system is the singular value decomposition (SVD). By applying SVD to the data, the extracted modes can similarly be interpreted in terms of macro- and micro-scale components, providing a connection to the scale separation achieved by the PU method.

To perform the SVD, one could construct the data matrix by grouping overlapping macro-elements, i.e., combining each macro-element with its neighboring ones, similar to the overlapping approach used in the PU method. We instead use non-overlapping macro-elements. This choice avoids introducing strong linear dependencies between blocks, ensuring that the SVD extracts independent modes that represent the macro- and micro-scale structure of the system.

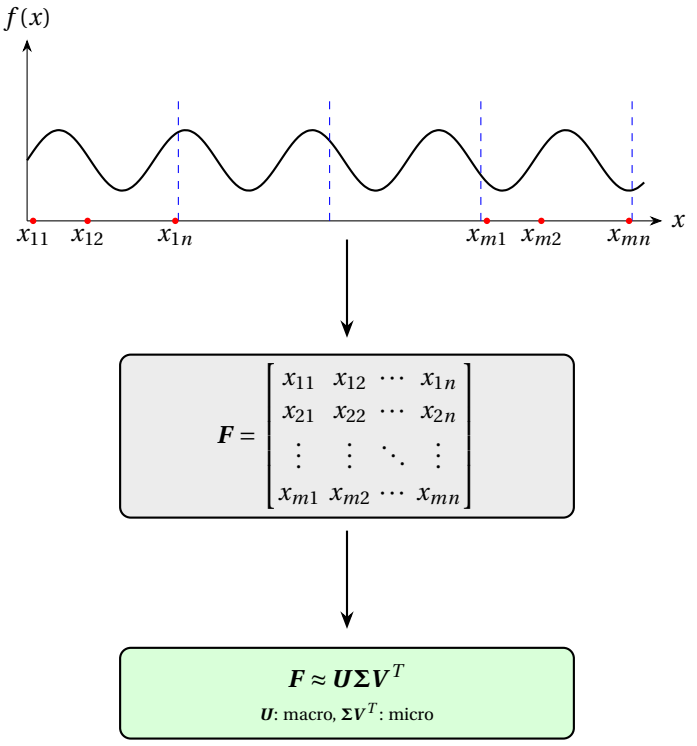


Figure 7. Illustration of the SVD workflow. The original function is divided into segments, transformed into a matrix $\boldsymbol{F}$, and then decomposed using SVD. The left singular vectors $\boldsymbol{U}$ capture the macro-scale structure, while $\boldsymbol{\Sigma}\boldsymbol{V}^T$ captures fine-scale variations (micro).

3.1. *Domain discretization*

In this section, we assume that the x-domain is equipped with a fine mesh that can capture all the scales involved in the dynamics, consisting of C nodes. A coarse mesh is then constructed, consisting of m macro-elements, each of which contains n micro-nodes, such that

$$C = m \times n.$$

Thus, each original node x_{ij} can be indexed by a pair (i, j), where $i = 1, \ldots, m$ denotes the macro-element index, and $j = 1, \ldots, n$ denotes the micro-node within the i-th macro-element.

The function nodal values $f(x_{ij})$, for all i, j, can now be expressed as a matrix $\boldsymbol{F}$, which has m rows and n columns. This representation does not introduce any complexity reduction at this stage, but it is well-known that matrices can be efficiently approximated in a hierarchical manner using the Singular Value Decomposition (SVD) (see Figure 7).

3.2. *SVD representation*

Instead of learning the matrix $\boldsymbol{F}$ directly, it is more efficient to learn its SVD representation. The SVD allows the matrix to be expressed as the sum of outer products of singular vectors:

$$\boldsymbol{F} \approx \sum_{i=1}^{T} \boldsymbol{U}_i \otimes \boldsymbol{V}_i,$$

where $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ are the left and right singular vectors, respectively, and T is the number of modes retained in the truncated approximation.

The truncated SVD representation enables a significant reduction in complexity, as it allows to approximate $\boldsymbol{F}$ using only the most significant singular values and vectors. By truncating the SVD to a limited number of modes T, we can achieve a compact representation of the matrix $\boldsymbol{F}$ that still captures the essential information needed to describe the system's dynamics. The matrix $\boldsymbol{F}$ used to compute the SVD has the micro and macro scales represented in the construction of each row and column. As a result, the left (truncated) singular vectors computed with the SVD will represent the micro functions equivalently to the PU, and the right (truncated) singular vectors computed with the SVD will represent the macro functions.

This approach allows for a more efficient representation of the system, as it significantly reduces the number of parameters required to model the system's evolution. The computational benefits of using the SVD approximation are particularly evident when dealing with large-scale systems or systems with many degrees of freedom.

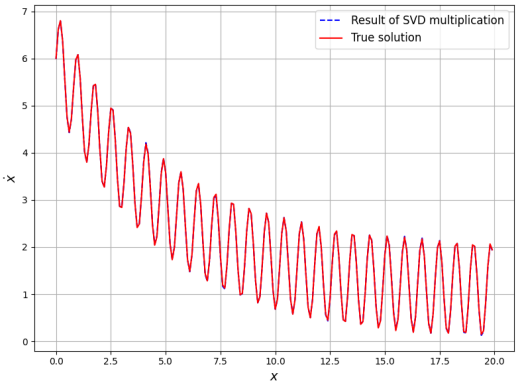
In summary, the SVD method offers a powerful way to approximate the nodal values of the system using a hierarchical decomposition that can be truncated to capture the essential features of the dynamics with reduced computational cost. This makes it an attractive technique for efficiently modeling complex systems with multi-scale behavior.

3.3. *Numerical results*

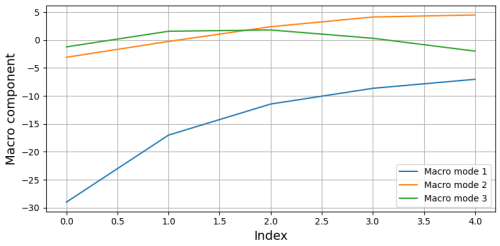
We now present the results obtained using the Singular Value Decomposition (SVD) method, applied to the same examples as the Partition of Unity (PU) method.

For each system, the SVD method was applied to approximate the nodal values $f(x_{ij})$ by decomposing the matrix $\boldsymbol{F}$. In the application of SVD, the selection of modes is typically determined by the energy captured by the modes or by the reconstruction error, as this reflects the decay behavior of the singular values in the analysis. In this work, we adopted an error-based criterion, selecting the number of modes such that the mean squared error (MSE) remained below 10^{-2}, consistent with the criterion established in the PU method.

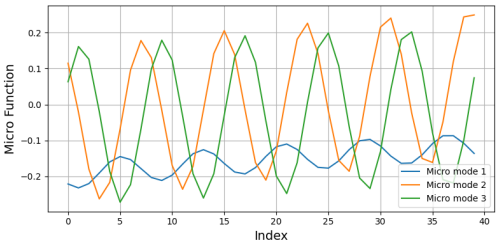
As illustrated in Section 2.4.1, we start with Eq. (2). As shown in Figure 8, the function and its corresponding macro and micro components are accurately represented when using three modes. The reconstruction achieved a mean squared error (MSE) of 1.56×10^{-4}, which is below the predefined threshold criterion of 10^{-2}, confirming the adequacy of the selected number of modes.



(a) Function plot using three modes for Eq. (2).



(b) Macro component using three modes for Eq. (2).



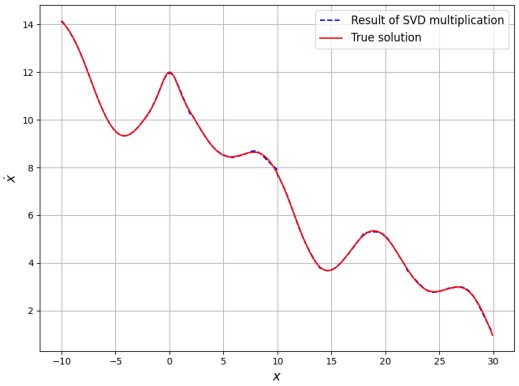
(c) Micro component using three modes for Eq. (2).

Figure 8. SVD approximation for Eq. (2) using three modes.

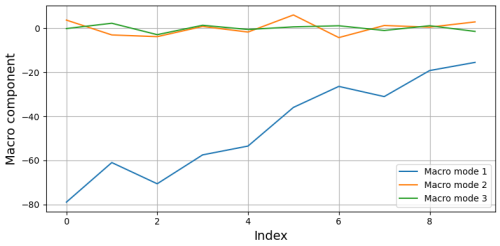
Next, we consider the dynamical system described by (3). As illustrated in Figure 9, the prediction obtained using three modes provides an accurate approximation of the system dynamics. The function plot, together with its corresponding macro and micro components, demonstrates that the SVD method effectively captures both the global and local features of the system. The reconstruction achieved a mean squared error (MSE) of 7.33×10^{-4}, which remains well below the predefined threshold, confirming the suitability of the selected mode configuration.

Finally, we examine the two dynamical systems given by (4) and (5). For the system described by (4), the SVD approximation is shown in Figure 10. The function plot, together with the corresponding macro and micro components, demonstrates that the chosen number of modes captures the system dynamics with good accuracy, yielding a mean squared error (MSE) of approximately 4.14×10^{-4}. Similarly, for the system described by (5), the results are presented in Figure 11, where the SVD-based reconstruction provides an accurate representation of the dynamics with a corresponding MSE of approximately 1.8×10^{-5}.

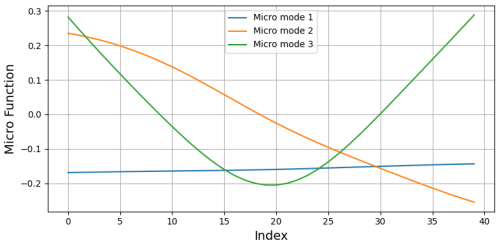
In summary, the numerical results demonstrate that the SVD method is a powerful approach for modeling and predicting complex, multi-scale dynamical systems. By decomposing the system into orthogonal modes and retaining only the most significant components, the method effectively reduces dimensionality while preserving essential dynamics. This enables accurate reconstruction of the system's behavior, capturing both large-scale patterns and fine-scale variations with reduced computational cost.



(a) Function plot using three modes for Eq. (3).



(b) Macro component using three modes for Eq. (3).



(c) Micro component using three modes for Eq. (3).

Figure 9. SVD approximation for Eq. (3) using three modes.

4. Sparse high-order SVD method

In practical situations, it is often not feasible to access the values of a target function f at all points in the computational domain. The measurement budget may only allow a sparse sampling of the domain, leading to an incomplete spatial dataset. This sparsity poses another challenge for data-driven modeling.

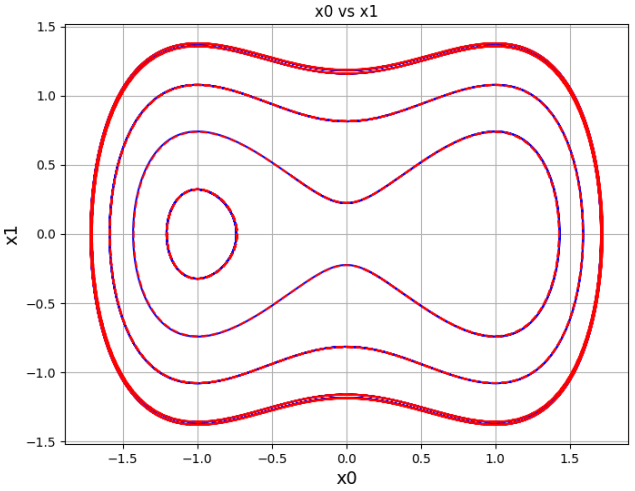
A naive approach would involve approximating the function directly via a neural network:

$$f(x) \approx NN(x),$$

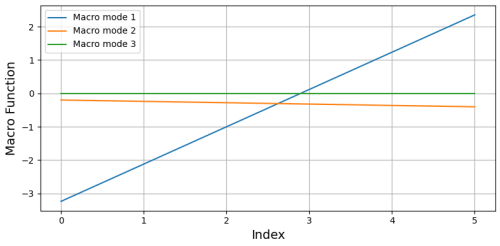
but such a general approximation involves two key limitations:

- high data requirements: training neural networks to approximate complex functions accurately typically requires a substantial amount of data to determine the network parameters;
- risk of overfitting: in multiscale problems, neural networks are prone to overfitting and spectral bias due to insufficient data, which impairs their ability to generalize beyond the training set, especially for high frequency data.

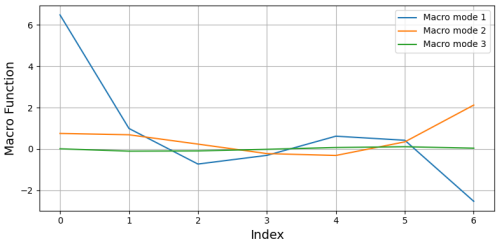
To address these challenges, we propose an approach that follows a sparse high-order SVD methodology, inspired by the PINN-PGD approach presented in [46]. In this manner, the algorithm is capable of learning from sparse data measurements. The scheme will follow an enrichment approach, where neural networks will learn the micro and macro functions profiting from their power to embed nonlinear correlations present in data. If the first micro and macro approximations do not accurately represent the dynamics, more modes (more neural network approximations of the functions) will progressively be added to the representation.



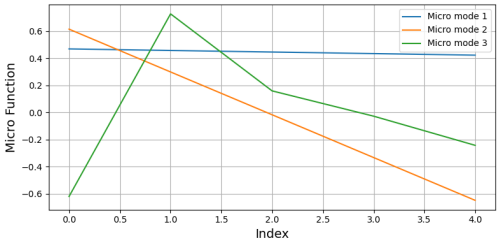
(a) Function plot using three modes for Eq. (4).



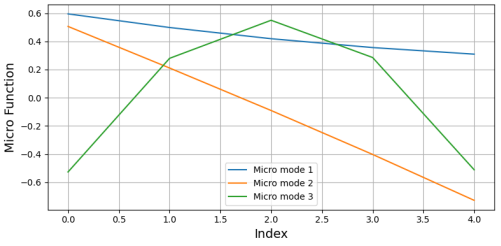
(b) Macro component 1 for Eq. (4).



(c) Macro component 2 for Eq. (4).



(d) Micro component 1 for Eq. (4).



(e) Micro component 2 for Eq. (4).

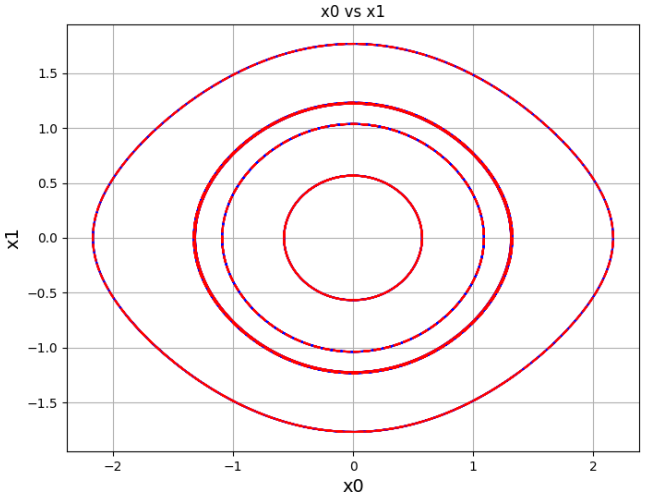
Figure 10. SVD approximation for Eq. (4) using three modes.

4.1. *Neural SVD representation of sparse observations*

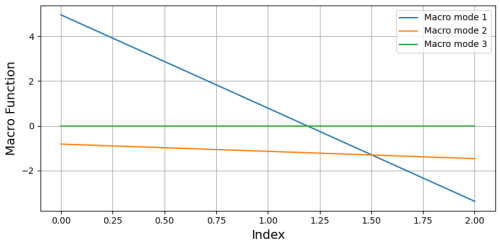
Following the spirit of SVD, we begin by organizing the observed values of f into a matrix $\boldsymbol{F}$, where each macro-cell corresponds to a single column of the matrix. However, due to the sparsity of measurements, this matrix remains incomplete, which prevents the direct application of classical SVD.

To approximate the low-rank structure of $\boldsymbol{F}$ despite its sparsity, we model its decomposition using time multiscale dimensional decomposition [46]:

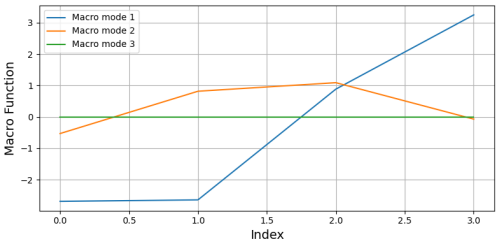
$$f(x_i, x_j) \approx NN_U(x_i) \cdot NN_V(x_j).$$



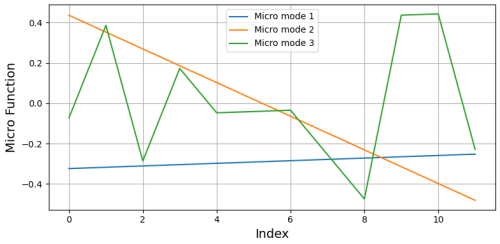
(a) Function plot using three modes for Eq. (5).



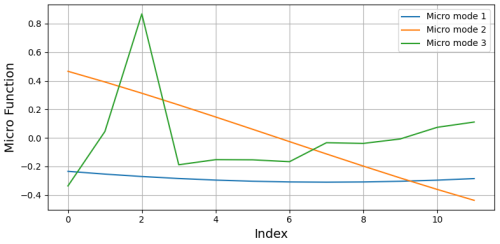
(b) Macro component 1 for Eq. (5).



(c) Macro component 2 for Eq. (5).



(d) Micro component 1 for Eq. (5).



(e) Micro component 2 for Eq. (5).

Figure 11. SVD approximation for Eq. (5) using three modes.

We further define:

- $U_i = NN_U(x_i)$, where x_i denotes the spatial coordinates within the macro-cell (i.e., the row index of the matrix);
- $V_j = NN_V(x_j)$, where x_j denotes the macro-cell identifier or coordinates (i.e., the column index).

The training loss is defined as [46]:

$$\mathscr{L}_{\text{data}} = \sum_{(i,j)\in\Omega} \left(f(x_i, x_j) - NN_U(x_i) \cdot NN_V(x_j)\right)^2,$$

where $\Omega \subset \{(i, j)\}$ denotes the set of sparse observed data points.

4.2. *Residual correction*

To enhance this data-driven model with physical consistency, we adopt a residual-based enrichment strategy. After the first approximation is learned, we define a residual:

$$r^{(1)}(x_i, x_j) = f(x_i, x_j) - NN_U^{(1)}(x_i) \cdot NN_V^{(1)}(x_j),$$

and fit a second neural network pair $(NN_U^{(2)}, NN_V^{(2)})$ to approximate this residual:

$$\mathscr{L}^{(2)} = \sum_{(i,j)\in\Omega} \left(r^{(1)}(x_i, x_j) - NN_U^{(2)}(x_i) \cdot NN_V^{(2)}(x_j)\right)^2.$$

This process can be repeated iteratively, resulting in a parsimonious enrichment:

$$\widehat{f}(x_i, x_j) = \sum_{k=1}^{K} \left\langle NN_U^{(k)}(x_i), NN_V^{(k)}(x_j) \right\rangle.$$

This approximation allows us to flexibly capture multiscale dynamics in a data-efficient and physics-consistent manner by decomposing the solution into a sum of learned low-rank components.

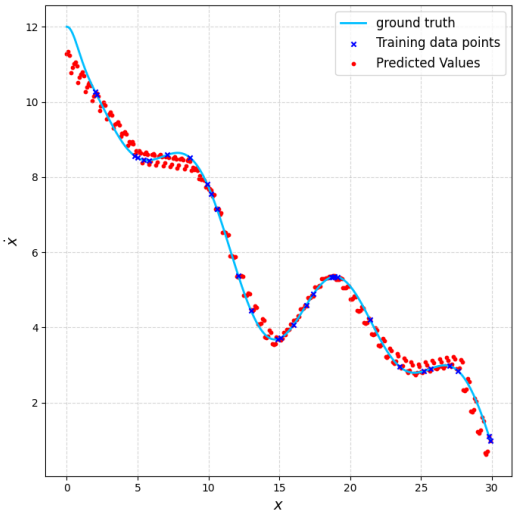
4.3. *Numerical results*

To evaluate the sparse high-order SVD framework, we apply it to the same representative one-dimensional function exhibiting multiscale characteristics defined in Eq. (3). We randomly mask 70% of the function values across the domain, producing a highly sparse observation matrix $\boldsymbol{F}$, with only 30% of its entries known. To ensure the variability of the sampled data is sufficient for capturing both scales, Figure 12 illustrates the positions of these observed entries within the $\boldsymbol{F}$ matrix. The goal is to reconstruct the full matrix, including the missing entries, using the sparse high-order SVD framework.

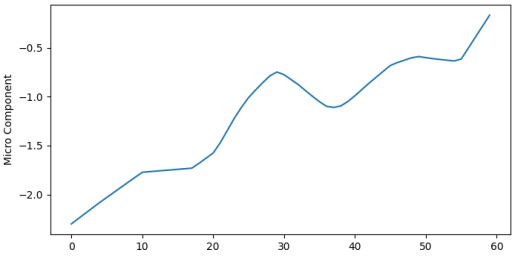
Figure 12. Sparsity pattern of the $\boldsymbol{F}$ matrix.

Following the formulation, we employ two separate neural networks to learn a low-rank approximation of the matrix:

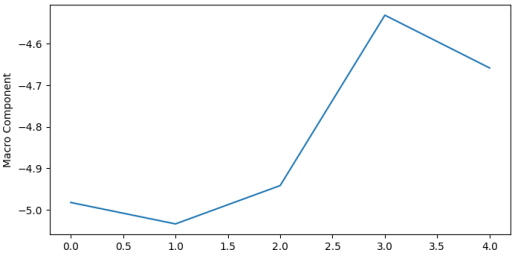
- $NN_U(x_i)$ predicts the latent representation of the micro-scale component (rows of $\boldsymbol{F}$);
- $NN_V(x_j)$ predicts the latent representation of the macro-scale component (columns of $\boldsymbol{F}$).



(a) Reconstructed function.



(b) Macro component.



(c) Micro component.

Figure 13. Sparse high-order SVD with a single macro-micro decomposition (Stage 1).

Each neural network consists of three hidden layers with nonlinear activation functions. The model is trained to minimize the reconstruction loss over the training observed entries.

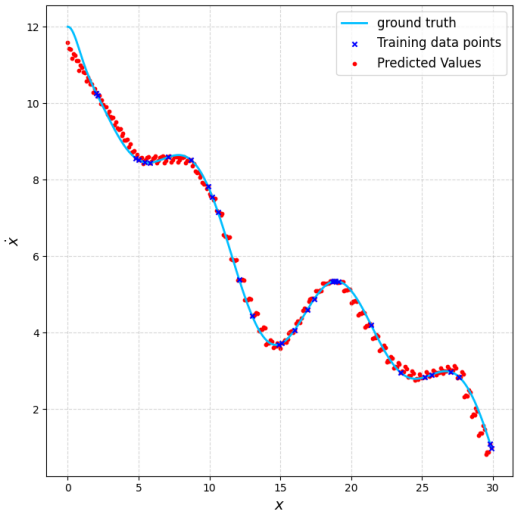
In this example, we tested the sparse high-order SVD model with two configurations: Stage 1 (single macro–micro decomposition) and Stage 2 (residual enrichment with an additional decomposition). In the Stage 1 setting, as shown in Figure 13, the model already provided a good-quality reconstruction, demonstrating that the macro and micro components effectively capture the coarse- and fine-scale dynamics of the signal, with a mean squared error of 0.08.

To refine the accuracy, Stage 2 introduces a residual enrichment process. To clearly analyze the nature of these corrections, Figures 14(b) and 14(c) represent the components learned during this enrichment. This enrichment process significantly improves the agreement with the ground truth, as shown in Figure 14, reducing the final MSE to 0.0007.

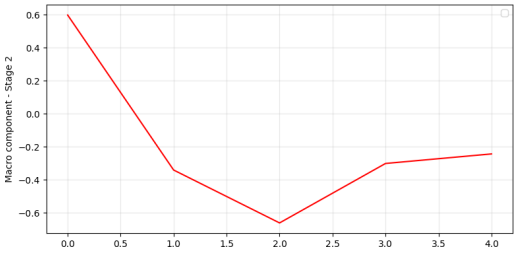
This numerical example confirms the ability of the sparse high-order SVD method to perform low-rank reconstruction of functions from sparse measurements in time.

5. Conclusion

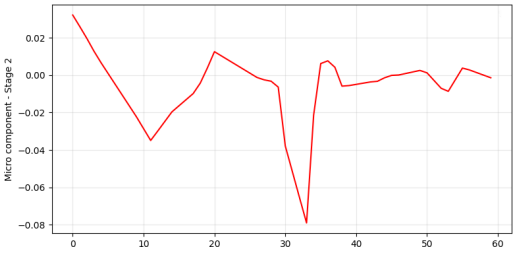
In this work, we proposed several approaches for analyzing and predicting the behavior of dynamical systems by decomposing their dynamics into micro (fine-scale) and macro (coarse-scale) components. One approach leverages the Partition of Unity (PU) method combined with



(a) Reconstructed function.



(b) Macro component.



(c) Micro component.

Figure 14. Sparse high-order SVD with residual enrichment (Stage 2).

neural networks to construct localized approximations, enabling the prediction of both macro- and micro-scale behaviors. Another approach uses Singular Value Decomposition (SVD) to extract dominant modes from the data, providing smooth global basis functions that capture the principal features of the system's dynamics across scales. To address scenarios with sparse or incomplete measurements, we further employ a sparse high-order SVD, which enables the reconstruction of multiscale dynamics directly from limited observations.

These approaches were applied to a variety of representative dynamical systems, including polynomial, sinusoidal, and exponential models, demonstrating their generality and robustness. The results confirmed that these methods effectively isolate and capture multi-scale features, leading to improvements in prediction accuracy and computational efficiency. Incorporating micro-macro decomposition with data-driven techniques provides a systematic way to model complex dynamics while maintaining interpretability and scalability, which is essential for high-dimensional and real-time applications.

Beyond predictive performance, these approaches contribute to a deeper understanding of how multi-scale phenomena can be disentangled and represented systematically. They also open avenues for integrating domain knowledge with data-driven techniques, enabling hybrid modeling approaches that balance physical interpretability and flexibility.

Future work will focus on extending these methods to more complex and realistic systems — such as turbulent flows, biological networks, or large-scale infrastructure systems — and improv-

ing performance by integrating with advanced machine learning architectures, including neural ODEs, physics-informed neural networks, or transformer-based sequence models. Additionally, adaptive schemes that dynamically adjust the level of resolution or partitioning based on data characteristics could further enhance model performance and generalizability. The integration of PU and SVD into broader modeling pipelines offers a promising direction for building intelligent, adaptive, and efficient models capable of addressing the challenges posed by real-world dynamical systems.

Acknowledgments

We would like to thank the research teams at PIMM Laboratory for their valuable insights and technical support throughout this project.

Declaration of interests

The authors do not work for, advise, own shares in, or receive funds from any organization that could benefit from this article, and have declared no affiliations other than their research organizations.

References

[1] W.-X. Wang, Y.-C. Lai and C. Grebogi, "Data Based Identification and Prediction of Nonlinear and Complex Dynamical Systems", 2017. Online at https://arxiv.org/abs/1704.08764v1.

[2] R. Ibáñez, A. Ammar, E. Cueto, A. Huerta, J.-L. Duval and F. Chinesta, "Multiscale Proper Generalized Decomposition Based on the Partition of Unity", *Int. J. Numer. Methods Eng.* **120** (2019), no. 6, pp. 727–747.

[3] R. Li, H. Wang and Y. Li, "Learning Slow and Fast System Dynamics via Automatic Separation of Time Scales", in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23)* (A. Singh and Y. Sun, eds.), ACM Press, 2023, pp. 4376–4386.

[4] G. Michel and G. P. Chini, "Multiple scales analysis of slow-fast quasilinear systems", *Proc. R. Soc. Lond., A, Math. Phys. Eng. Sci.* **475** (2019), no. 2223, article no. 20180630 (17 pages).

[5] S. Bordoni, S. M. Kang, T. A. Shaw, I. R. Simpson and L. Zanna, "The futures of climate modeling", *npj Clim. Atmos. Sci.* **8** (2025), article no. 99 (6 pages).

[6] V. N. Pilipchuk, A. F. Vakakis and M. A. F. Azeez, "Sensitive Dependence on Initial Conditions of Strongly Nonlinear Periodic Orbits of the Forced Pendulum", *Nonlinear Dyn.* **16** (1998), no. 3, pp. 223–237.

[7] M. Raissi, P. Perdikaris and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", *J. Comput. Phys.* **378** (2019), pp. 686–707.

[8] B. Chen, K. Huang, S. Raghupathi, I. Chandratreya, Q. Du and H. Lipson, "Automated discovery of fundamental variables hidden in experimental data", *Nat. Comp. Sci* **2** (2022), no. 7, pp. 433–442.

[9] N. L. Zhang, T. D. Nielsen and F. V. Jensen, "Latent variable discovery in classification models", *Artif. Intell. Med.* **30** (2004), no. 3, pp. 283–299.

[10] D. Chakraborty, S. W. Chung, T. Arcomano and R. Maulik, "Divide and conquer: Learning chaotic dynamical systems with multistep penalty neural ordinary differential equations", *Comput. Methods Appl. Mech. Eng.* **432(A)** (2024), article no. 117442.

[11] H. Zheng and G. Lin, "LES-SINDy: Laplace-Enhanced Sparse Identification of Nonlinear Dynamical Systems", 2024. Online at https://arxiv.org/abs/2411.01719.

[12] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang and L. Yang, "Physics-informed machine learning", *Nat. Rev. Phys.* **3** (2021), no. 6, pp. 422–440.

[13] P. Urdeitx, I. Alfaro, D. González, F. Chinesta and E. Cueto, "A comparison of single and double generator formalisms for thermodynamics-informed neural networks", *Comput. Mech.* **75** (2025), no. 6, pp. 1769–1785.

[14] A. Tierz, M. M. Iparraguirre, I. Alfaro, D. González, F. Chinesta and E. Cueto, "On the feasibility of foundational models for the simulation of physical phenomena", *Int. J. Numer. Methods Eng.* **126** (2025), no. 6, article no. e70027 (14 pages).

[15] Q. Hernández, A. Badías, D. González, F. Chinesta and E. Cueto, "Structure-preserving neural networks", *J. Comput. Phys.* **426** (2021), article no. 109950.

[16] R. Yu and R. Wang, "Learning dynamical systems from data: An introduction to physics-guided deep learning", *Proc. Natl. Acad. Sci. USA* **121** (2024), no. 27, article no. e2311808121 (10 pages).

[17] H. Bai and W. Ding, "KoNODE: Koopman-Driven Neural Ordinary Differential Equations with Evolving Parameters for Time Series Analysis", in *Proceedings of the 42nd International Conference on Machine Learning* (A. Singh, M. Fazel, D. Hsu, S. Lacoste-Julien, F. Berkenkamp, T. Maharaj, K. Wagstaff and J. Zhu, eds.), Proceedings of Machine Learning Research, vol. 267, PMLR, 2025, pp. 2484–2519.

[18] J. Nathaniel, C. Roesch, J. Buch, D. DeSantis, A. Rupe, K. Lamb and P. Gentine, "Deep Koopman operator framework for causal discovery in nonlinear dynamical systems", 2025. Online at https://arxiv.org/abs/2505.14828.

[19] S. E. Otto and C. W. Rowley, "Koopman operators for estimation and control of dynamical systems", *Ann. Rev. Control Rob. Auton. Syst.* **4** (2021), no. 1, pp. 59–87.

[20] C. Ghnatios, J. Mouterde, J. Tomezyk, J. Da Silva and F. Chinesta, "Learning Transformed Dynamics for Efficient Control Purposes", *Mathematics* **12** (2024), no. 14, article no. 2251 (22 pages).

[21] S. L. Brunton, J. L. Proctor and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems", *Proc. Natl. Acad. Sci. USA* **113** (2016), no. 15, pp. 3932–3937.

[22] J. Park, N. Yang and N. Chandramoorthy, "When are dynamical systems learned from time series data statistically accurate?", *Adv. Neural Inf. Process. Syst.* **37** (2024), pp. 43975–44008.

[23] K. Li and W. Ye, "D-FNO: A decomposed Fourier neural operator for large-scale parametric partial differential equations", *Comput. Methods Appl. Mech. Eng.* **436** (2025), article no. 117732.

[24] K. Michałowska, S. Goswami, G. E. Karniadakis and S. Riemer-Sørensen, "Neural operator learning for long-time integration in dynamical systems with recurrent neural networks", in *2024 International Joint Conference on Neural Networks (IJCNN)* (A. Hirose and H. Ishibuchi, eds.), IEEE Press, 2024, pp. 1–8.

[25] A. Cong, Y. Jin, Z. Lu, et al., "Transfer learning-based physics-informed DeepONets for the adaptive evolution of digital twin models for dynamic systems", *Nonlinear Dyn.* **113** (2025), pp. 19075–19102.

[26] C. Lin, Z. Li, L. Lu, S. Cai, M. Maxey and G. E. Karniadakis, "Operator learning for predicting multiscale bubble growth dynamics", *J. Chem. Phys.* **154** (2021), no. 10, article no. 104118.

[27] S. Goswami, M. Yin, Y. Yu and G. E. Karniadakis, "A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials", *Comput. Methods Appl. Mech. Eng.* **391** (2022), article no. 114587.

[28] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio and A. Courville, "On the spectral bias of neural networks", in *International conference on machine learning* (K. Chaudhuri and R. Salakhutdinov, eds.), Proceedings of Machine Learning Research, vol. 97, PMLR, 2019, pp. 5301–5310.

[29] Z.-Q. J. Xu, Y. Zhang, T. Luo, Y. Xiao and Z. Ma, "Frequency principle: Fourier analysis sheds light on deep neural networks", 2019. Online at https://arxiv.org/abs/1901.06523.

[30] G. Wen, Z. Li, K. Azizzadenesheli, A. Anandkumar and S. M. Benson, "U-FNO — An enhanced Fourier neural operator-based deep-learning model for multiphase flow", *Adv. Water Resources* **163** (2022), article no. 104180.

[31] S. Rodriguez, A. Pasquale, J. Mounayer, D. Canales, M. Beringhier, C. Ghnatios, A. Ammar and F. Chinesta, "A reduced simulation applied to viscoelastic fatigue of polymers using a time multi-scale approach based on partition of Unity method", *Adv. Model. Simul. Eng. Sci.* **12** (2025), no. 1, article no. 11 (16 pages).

[32] A. Pasquale, A. Ammar, A. Falcó, S. Perotto, E. Cueto, J.-L. Duval and F. Chinesta, "A separated representation involving multiple time scales within the Proper Generalized Decomposition framework", *Adv. Model. Simul. Eng. Sci.* **8** (2021), no. 1, article no. 26 (22 pages).

[33] S. Rodriguez, A. Pasquale, K. Nguyen, A. Ammar and F. Chinesta, "A time multiscale based data-driven approach in cyclic elasto-plasticity", *Comput. Struct.* **295** (2024), article no. 107277 (14 pages).

[34] M. Brenner, E. Weber, G. Koppe and D. Durstewitz, "Learning interpretable hierarchical dynamical systems models from time series data", 2024. Online at https://arxiv.org/abs/2410.04814.

[35] Z. Liu, W. Cai and Z.-Q. J. Xu, "Multi-scale deep neural network (MscaleDNN) for solving Poisson–Boltzmann equation in complex domains", 2020. Online at https://arxiv.org/abs/2007.11207.

[36] H. Wang, H. Yan, C. Rong, et al., "Multi-scale simulation of complex systems: a perspective of integrating knowledge and data", *ACM Comput. Surv.* **56** (2024), no. 12, pp. 1–38.

[37] L. Liu and W. Cai, "Multiscale DeepONet for nonlinear operators in oscillatory function spaces for building seismic wave responses", 2021. Online at https://arxiv.org/abs/2111.04860.

[38] S. E. Ahmed and P. Stinis, "A multifidelity deep operator network approach to closure for multiscale systems", *Comput. Methods Appl. Mech. Eng.* **414** (2023), article no. 116161.

[39] K. Wu, X.-B. Yan, S. Jin and Z. Ma, "Capturing the diffusive behavior of the multiscale linear transport equations by Asymptotic-Preserving Convolutional DeepONets", *Comput. Methods Appl. Mech. Eng.* **418** (2024), article no. 116531.

[40] R. S. Beddig, P. Benner, I. Dorschky, T. Reis, P. Schwerdtner, M. Voigt and S. W. R. Werner, "Structure-Preserving Model Reduction for Dissipative Mechanical Systems", in *Calm, Smooth and Smart* (P. Eberhard, ed.), Lecture Notes in Computational Science and Engineering, vol. 102, Springer, 2023, pp. 209–230.

[41] J. Baker, E. Cherkaev, A. Narayan and B. Wang, "Learning proper orthogonal decomposition of complex dynamics using heavy-ball neural ODEs", *J. Sci. Comput.* **95** (2023), no. 2, article no. 54.

[42] P. Sentz, K. Beckwith, E. C. Cyr, L. N. Olson and R. Patel, "Reduced basis approximations of parameterized dynamical partial differential equations via neural networks", *Found. Data Sci.* **7** (2025), no. 1, pp. 338–362. Sandia National Laboratories (SNL-NM), report number SAND–2025-04099J.

[43] J. M. Melenk and I. Babuška, "The partition of unity finite element method: Basic theory and applications", *Comput. Methods Appl. Mech. Eng.* **139** (1996), no. 1, pp. 289–314.

[44] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. A. Hamprecht, Y. Bengio and A. Courville, "On the Spectral Bias of Neural Networks", 2019. Online at https://arxiv.org/abs/1806.08734.

[45] C. Ghnatios, F. Masson, A. Huerta, A. Leygue, E. Cueto and F. Chinesta, "Proper Generalized Decomposition based dynamic data-driven control of thermal processes", *Comput. Methods Appl. Mech. Eng.* **213–216** (2012), pp. 29–41.

[46] C. Ghnatios and F. Chinesta, "A Parsimonious Separated Representation Empowering PINN-PGD-Based Solutions for Parametrized Partial Differential Equations", *Mathematics* **12** (2024), no. 15, article no. 2365 (13 pages).