Chaos, Solitons and Fractals 212 (2026) 119017

Contents lists available at ScienceDirect

Chaos, Solitons and Fractals

journal homepage: www.elsevier.com/locate/chaos

Sparse semi-interpretable networks for discovering nonlinear dynamics with time-varying parameters and switching structures

Binghang Xiao [a], Jianzhe Huang [a,b,*], Siyuan Xing [c,**]

[a] School of Aeronautics and Astronautics, Shanghai Jiao Tong University, Shanghai, 200240, China
[b] School of Information Science and Engineering, East China University of Science and Technology, Shanghai, 200237, China
[c] Mechanical Engineering Department, California Polytechnic State University, San Luis Obispo, CA, 93407-0403, USA

ARTICLE INFO

Keywords:
Data-driven equation discovery
Sparse regression
Semi-interpretable networks
Time-varying nonlinear dynamics
Switching dynamics

ABSTRACT

Data-driven equation discovery has emerged as a new focal point in nonlinear dynamics. While significant progress has been made in identifying governing equations of time-invariant dynamical systems, the existing approaches still encounter considerable obstacles when applied to scenarios with time-varying parameters or switching structures. To bridge this gap, this paper introduces a new sparse semi-interpretable network architecture. The framework integrates an interpretable neural network, called SREINet, to identify the backbone of governing equations, with a hypernetwork designed to capture time-varying dynamics and structural transitions. In addition, a periodic pruning mechanism is applied to the hypernetwork's output to enhance structural sparsity and mitigate the influence of measurement noise. By performing symbolic regression on the hypernetwork's output, our approach can extract explicit, closed-form governing equations for time-varying and switching nonlinear dynamics. We validate the method across three representative systems with varying levels of nonlinearity and dimensionality, including a Kuramoto oscillator with logarithmic time-varying coupling strength. Finally, we demonstrate that the framework can also be used for early prediction of tipping points.

1. Introduction

In many real-world applications, the underlying physical processes are nonstationary, driven by governing laws that evolve through continuous parameter modulations or discrete structural switches. These variations often arise in response to external forcings, environmental cycles, or operational switches. In climate science, the continuous accumulation of greenhouse gases acts as an external radiative forcing that, for example, leads to time-varying thermal gradients that may cause irreversible transitions in atmospheric and oceanic circulation [1]. In epidemiology, effective transmission rates fluctuate due to seasonal cycles [2], often accompanied by structural shifts in dynamics caused by policy interventions. In aerospace, morphing unmanned aerial vehicles (UAVs) undergo significant shifts in their physical parameters and aerodynamic characteristics during structural transitions [3], while control laws may switch abruptly between different flight phases [4]. Although automatic discovery of these time-varying physical laws from data has critical implications for scenarios such as tipping-point prediction [5,6] and structural health monitoring [7], the current model-discovery tools are still limited in handling such cases. This paper aims to bridge this gap by developing a new machine-learning framework that can simultaneously discover the symbolic skeleton of governing equations

* Corresponding author at: School of Information Science and Engineering, East China University of Science and Technology, Shanghai, 200237, China.
** Corresponding author.
E-mail addresses: sixing@calpoly.edu (S. Xing), hjianzhe@ecust.edu.cn (J. Huang).

https://doi.org/10.1016/j.chaos.2026.119017
Received 13 April 2026; Received in revised form 17 August 2026; Accepted 18 August 2026
Available online 29 August 2026

and the corresponding temporal evolution of their parameters. Prior to elaborating on this approach, we briefly review the evolution of data-driven modeling in nonlinear dynamics.

Data-driven modeling dates back to the 17th century when Johannes Kepler distilled the fundamental laws of planetary motion directly from meticulous observational data [8]. In recent years, this field has drawn extensive attention mainly due to the explosion of data volume and increasing complexity of systems (e.g., climate and neural systems) that make first-principles modeling intractable [9]. To tackle these challenges, researchers have developed diverse modeling strategies that generally fall into two primary paradigms: implicit and explicit methods. The implicit methods leverage the universal approximation of neural networks [10] to learn the underlying system dynamics. Specialized architectures like neural ordinary differential equations [11], Recurrent Neural Network (RNN) and its variants (e.g., LSTM [12], GRU [13], and reservoir computing [14]) have been developed to learn dynamics via latent-space representations [15–17]. As an alternative to neural-network-based approaches, Koopman operator [18,19] provides an alternative implicit modeling framework by lifting nonlinear dynamics into a higher-dimensional functional space where the dynamics become linear. While these implicit models excel in predictive performance, they often lack interpretability and provide few insights on the underlying physical laws.

On the other hand, explicit data-driven modeling aims to extract parsimonious governing equations for nonlinear dynamics, with three dominant paradigms: symbolic regression (SR), sparse regression, and neural-symbolic approach. Symbolic regression [20] searches for the best-fit symbolic expressions using Genetic Algorithms [21], while sparse regression [22,23] formulates the problem as a sparse optimization problem that selects the best-fit equations from a collection of predefined candidate functions. Neural-symbolic approach embeds symbolic operations into interpretable network structures, incorporating sparse optimization and deep learning techniques to distill the most parsimonious governing equations. For instance, Equation Learner (EQL) [24] builds feedforward layers from elementary functions to discover symbolic expressions directly from data. In addition, Kolmogorov–Arnold Networks (KANs) [25] leverage the Kolmogorov–Arnold representation theorem [26,27] to express multivariate functions as a sum of learnable univariate functions. While KAN is inherently interpretable, extracting explicit governing equations usually requires converting edge splines into learnable symbolic functions or employing post-training symbolic regression. Another recently proposed interpretable network [28,29], Sparse Regression Embedded Interpretable Network (SREINet), utilizes the separable property of candidate functions to design a multilayer network that inherently embeds the matrix-formulation of sparse regression. By incorporating a masking-based pruning algorithm, the method demonstrates the capability to identify high-dimensional nonlinear dynamical systems.

Despite these advancements, most current research in equation discovery has focused on dynamical systems with stationary parameters. The data-driven identification of time-varying nonlinear dynamics remains a nascent challenge that has only recently garnered research interest. A common strategy to address time-varying coefficients is to employ sliding windows or segmented regression [30,31], where the entire time series is divided into several continuous time windows. The governing equations are individually identified in these segments where the system parameters are assumed to be locally stationary. Afterward, time-varying parameters are reconstructed via numerical interpolation [32]. However, the performance of this approach is sensitive to time-window size: narrow windows can resolve fast parameter variations yet exhibit high noise susceptibility [33], wider windows suppress noise effectively but introduce estimation lag and spurious averaging effects [34]. For this reason, the window size has to be tuned via trial and error [35], which lacks generalizability and imposes a significant manual tuning burden. Another recent approach to addressing this issue is to introduce a hypernetwork [36] that takes time as input and generates the coefficients, either directly or indirectly. This way, the time-varying coefficients can be learned from the input–output mapping of the hypernetwork. Depending on how temporal information is represented, such generators can be implemented through simple feedforward mappings, recurrent or attention-based models, or variational encoders. For instance, Dynamic Sparse Identification of Nonlinear Dynamics (DSINDy) employs a variational autoencoder (VAE) to generate time series for time-varying coefficients of sparse dynamical equations [37]. Hyper-EQL (HEQL) [38] integrates meta-weight units into the EQL framework, enabling the symbolic network to identify time-varying coefficients with invariant structures of underlying governing equations. However, existing hypernetwork-assisted equation discovery methods mainly use hypernetworks to generate time-varying coefficients for a prescribed sparse model or to modulate symbolic-network weights under an invariant equation structure. In these settings, the hypernetwork primarily addresses parametric time variation, while the simultaneous identification of the equation skeleton and its temporal/structural evolution remains less explored.

This paper proposes a new semi-interpretable machine-learning framework named H-SREINet. It couples interpretable SREINet, which extracts the structural skeleton of governing equations, with a black-box hypernetwork (multilayer perceptron) that learns the mapping relationship between time and parameters. The two networks are trained within a unified, end-to-end framework, further enhanced by a periodic, S-shaped masking-based pruning scheme to enforce structural parsimony. By applying symbolic regression to the pruned hypernetwork output, this architecture can extract explicit governing equations for nonlinear dynamical systems with time-varying coefficients and switching behaviors. The closed-form symbolic representations extracted by H-SREINet provide a basis for downstream stability analysis, including equation-based tipping-point inference. We demonstrate the performance of this framework through three classical examples with distinct types of nonlinearities and dimensions, followed by an idealized AMOC example that illustrates how recovered equations can be used to infer a tipping time outside the training interval.

The paper is organized as follows. Section 2 presents the framework of the proposed H-SREINet. Section 3 demonstrates the performance of the system identification on various dynamical systems. Section 4 concludes the paper and outlines directions for future research.

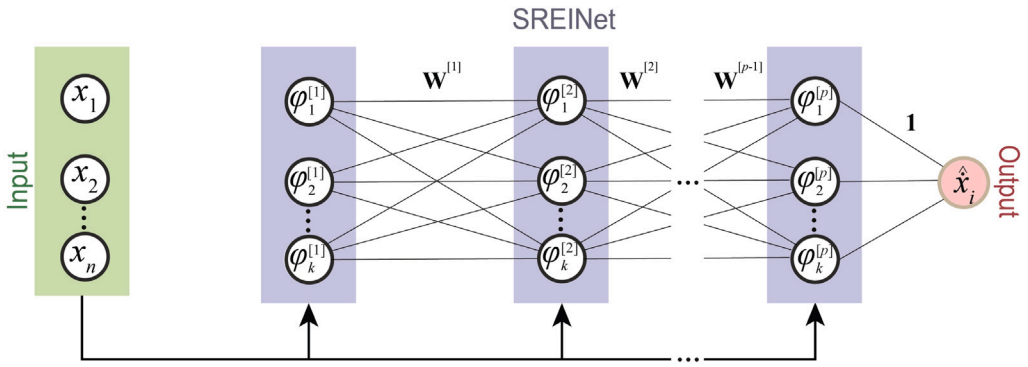

Fig. 1. Architecture of SREINet adapted from Ref. [29]. The neurons of each layer are replaced by univariate functions (with $\varphi_1^{[j]} = 1$).

2. Methodology

2.1. SREINet structure

The structure of the SREINet [29] is illustrated in Fig. 1, where the inputs correspond to system states $\boldsymbol{x} = [x_1, x_2, \dots, x_n]^T$, and the output is a scalar representing the time derivative of a designated dimension. The model is trained separately for each dimension in a sequential manner, which will be elaborated in the following training section. Structurally, the states $\boldsymbol{x}$ are shared across the network and fed in parallel into each hidden layer. The SREINet is different from the classical artificial neural networks in several aspects. First, its layers are bias-free. Second, the model's neurons are defined by a library of atomic candidate functions (e.g., constants, polynomials, and exponents) to define its neurons in the following form $\varphi(\zeta) \in \{1, \zeta, \zeta^2, \dots, \sin(\zeta), \sin(2\zeta), \dots, \exp(\zeta), \exp(2\zeta), \dots\}$, where ζ is a feature of the state $\boldsymbol{x}$, and repeated atomic functions are allowed. This way, we define the atomic functions of the jth layer $\boldsymbol{\varphi}^{[j]}$:

$$\boldsymbol{\varphi}^{[j]} = \left[\varphi_1^{[j]}, \varphi_2^{[j]}, \varphi_3^{[j]}, \dots, \varphi_k^{[j]}\right]^T, \tag{1}$$

where the constant 1 is always assigned as $\varphi_1^{[j]}$. The third distinction arises in forward propagation. To illustrate this, let us assume the atomic functions are univariate monomials. The output of the second layer is expressed as

$$\boldsymbol{y}^{[2]} = \boldsymbol{\varphi}^{[2]} \odot \mathbf{W}^{[1]} \cdot \boldsymbol{y}^{[1]}, \tag{2}$$

where $\odot$ and the centered dot denote the Hadamard product and matrix multiplication, respectively. The term $\mathbf{W}^{[1]} \cdot \boldsymbol{y}^{[1]}$ generates linear combinations of the first-layer outputs, which are then multiplied element-wise by the atomic functions of the second layer. Consequently, these operations construct all computational paths for candidate functions up to second-order monomials (e.g., $1, x_1, \dots, x_1x_2, \dots, x_n^2$). By increasing the network depth, SREINet can systematically generate all monomials up to an order equivalent to the number of layers. In practice, one or two copies of the base set of atomic functions are typically sufficient, as governing equations are generally sparse within the candidate space and require only a few forward paths to accurately capture the underlying dynamics. This architecture implies that any governing equation decomposable into a sum of product-form candidate functions can be effectively captured by SREINet. Consequently, the network maintains inherent interpretability, as its layers directly map to the structural decomposition of the underlying dynamics.

2.2. Time-varying coefficients represented by hypernetworks

Building upon the SREINet architecture, we now integrate it with a hypernetwork structure to enable the learning of nonlinear dynamics with time-varying coefficients, as shown in Fig. 2. For each weight matrix, we assign a hypernetwork that takes time as input and outputs the corresponding weights: i.e., $\mathbf{W}^{[j]}(t) = \mathrm{NN}^{[j]}(t)$. This way, the output $\boldsymbol{y}^{[j]}$ of the jth layer reads

$$\boldsymbol{y}^{[j]} = \boldsymbol{\varphi}^{[j]} \odot \mathbf{W}^{[j-1]}(t) \cdot \boldsymbol{y}^{[j-1]}. \tag{3}$$

In this paper, the hypernetwork is implemented as a multilayer perceptron (MLP), which is simple and enables formulating the task as a pointwise optimization problem, thereby enhancing computational efficiency. Although alternative sequence-modeling architectures, such as RNNs and their variants, could be employed, they typically require the computationally expensive process of Backpropagation Through Time (BPTT). With the hypernetwork represented by MLP, the output (state derivatives) $\hat{\dot{x}}_i$ is in the form

$$\hat{\dot{x}}_i(\boldsymbol{x}, t) = \boldsymbol{\varphi}^{[p+1]} \cdot \boldsymbol{\varphi}^{[p]}(\boldsymbol{x}) \odot \mathbf{W}^{[p-1]}(t) \cdots \mathbf{W}^{[2]}(t) \cdot \boldsymbol{\varphi}^{[2]}(\boldsymbol{x}) \odot \mathbf{W}^{[1]}(t) \cdot \boldsymbol{\varphi}^{[1]}(\boldsymbol{x}), \tag{4}$$

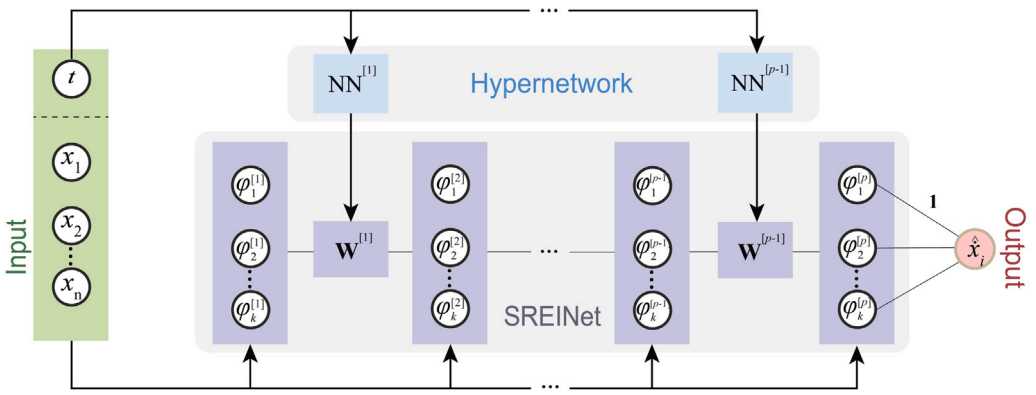

Fig. 2. Architecture of hypernetwork-enhanced SREINet. Each layer is paired with a hypernetwork that takes time t as input and outputs the layer weights.

where $\boldsymbol{\varphi}^{[p+1]}$ is assigned as $\mathbf{1}_{k\times 1}$ for simplicity. We call this new network structure H-SREINet, where the letter "H" represents the hypernetwork. The architecture of H-SREINet forms a semi-interpretable structure where the bottom interpretable SREINet captures the dominant terms of the vector field and the top black-box hypernetworks predict time-varying weights. Therefore, the equation skeleton can be recovered from the active paths of SREINet, while the hypernetwork supplies the corresponding time-dependent coefficients. Unlike standard black-box networks, the structure of H-SREINet serves as a parametric factorization of the vector field. Specifically, any specific candidate function equivalent to $\prod \varphi_i^{\alpha_i}$ is formed by the product of neurons across layers, meaning its time-varying coefficient $\zeta_{\boldsymbol{\alpha}}(t)$ ($\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_k]$, $|\boldsymbol{\alpha}| \le p$) is the sum of all path weights that contribute to that candidate function. More specifically, with a single copy of neurons, the time-varying coefficient $\zeta_{\boldsymbol{\alpha}}(t)$ can be calculated through the nonlinear map:

$$\xi_{\boldsymbol{\alpha}}(t) = \sum_{J\in\mathcal{P}_{\boldsymbol{\alpha}}} \mathbf{W}^{[1]}_{j_1 j_2}(t)\mathbf{W}^{[2]}_{j_2 j_3}(t)\cdots\mathbf{W}^{[p-1]}_{j_{p-1} j_p}(t), \tag{5}$$

where $\mathcal{P}_{\boldsymbol{\alpha}}$ denotes the set of all index sequences $J = \{j_1, j_2, \dots, j_p\}$ such that the occurrence of each index i in J satisfies

$$\sum_{m=1}^{p} \mathbb{1}(j_m = i) = \alpha_i, \tag{6}$$

with $\mathbb{1}(\cdot)$ being the indicator function. In practice, these coefficients can be calculated by a recursive depth-first search algorithm to explore all active (nonzero) paths. As each identified coefficient is a function of time, it can be easily integrated with symbolic regression to obtain its symbolic expression, which will be further discussed in Section 2.4.

2.3. Learning algorithm

With the H-SREINet structure, we apply a dimensionwise scheme to sequentially learn the vector field

$$\hat{x}_j = \hat{\boldsymbol{f}}_j(\boldsymbol{x}, t) \approx \mathcal{N}(\boldsymbol{x}, t, \boldsymbol{\theta}_j),\ \ j = 1, 2, \dots, n, \tag{7}$$

which can be formulated through the following optimization problem:

$$\arg\min_{\boldsymbol{\theta}_j} \sum_{i=1}^{m} \frac{1}{m} \left\| \dot{x}_j^{(i)} - \mathcal{N}(\boldsymbol{x}^{(i)}, t, \boldsymbol{\theta}_j) \right\|_2 + \sum_{k=1}^{p-1} R(\boldsymbol{W}_j^{[k]}(t)), \tag{8}$$

where $\boldsymbol{\theta}_j$ denotes the corresponding hypernetwork weights, and R represents a sparsity promoting operator applied to the weights or the outputs of hypernetworks. Theoretically, the sparsity promoting operator should be applied to the coefficients of the original equations. However, the coefficient reconstruction will lead to the calculation of $(n+p)!/n!p!$ number of weights during each training step, which is computationally expensive, especially for high-dimensional situations. Alternatively, we apply the sparsity promoting operator to the weights of SREINet, as the sparsity of coefficients implies the sparsity of network weights.

We note that the concept of applying hypernetworks such as MLP and VAE to learn time-varying weights has been explored earlier. However, it is in fact a challenging task to distill governing equations in these scenarios as these weights are no longer constant but functions of time, represented by the outputs of hypernetworks. This means that traditional sparsity promoters such as Lasso [39] and Ridge regression [40] have to penalize across the whole sampling time domain. To address this challenge, we develop a mask-based structural pruning scheme specifically designed for hypernetwork architectures. Unlike traditional static pruning, our method evaluates the significance of each weight by considering its dynamic evolution across the entire temporal domain. Specifically, for each weight matrix, we construct a corresponding binary mask $\mathcal{M}$. A weight $w_{ij}^{[k]}(t)$ (an output of a hypernetwork)

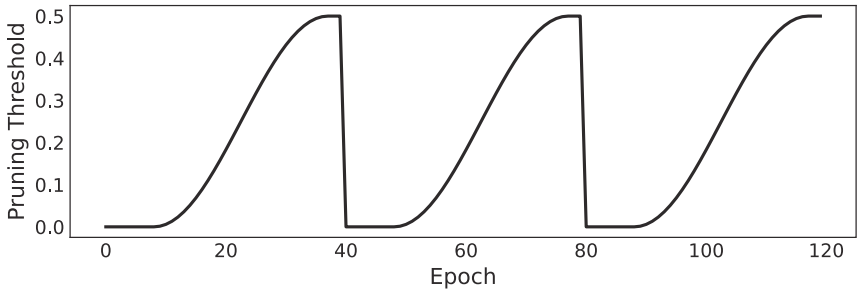

Fig. 3. An example of periodic piecewise pruning strategy with 3 pruning periods and each period has 40 pruning epochs.

is identified as redundant only if its magnitude remains consistently below a predefined threshold τ across a set of s uniformly sampled time points $T = \{t_1, t_2, \dots, t_s\}$. Once a weight is flagged, we mask the corresponding element in the mask to zero: i.e.,

$$\mathcal{M}_{ij}^{[k]} = \begin{cases} 0, & \text{for } \max\limits_{z\in[1,2,\dots,s]} \left|w_{ij}^{[k]}(t_z)\right| \le \tau(E), \\ 1, & \text{for } \max\limits_{z\in[1,2,\dots,s]} \left|w_{ij}^{[k]}(t_z)\right| > \tau(E), \end{cases} \tag{9}$$

where the threshold τ varies with the epoch number E, which will be explained subsequently. Notably, we adopt a more conservative criterion that compares the max absolute value of the weight at these sampled temporal points with the threshold. Alternatively, one can also use the mean and variance of weights for the same purpose. This max-based criterion is intended to preserve weights that become important only over a short time interval, such as near an abrupt switching event. If the sampled temporal grid contains points within the transition region, a transiently active weight can still exceed the threshold and survive pruning. In addition, the MLP-based hypernetwork can approximate sharp step-like coefficient changes when sufficient samples are available around the switching time. However, if the sampling is too sparse and no data points capture the transition, the abrupt change is not represented in the training data; in that case, the hypernetwork may smooth over the transition and the pruning procedure may incorrectly remove or underestimate the associated term.

Once the binary masks are determined, we prune the specific parameters in the kth hypernetwork's output layer that generate the redundant weights. Specifically, for each zeroed entry $\mathcal{M}_{ij}^{[k]} = 0$, we identify the corresponding output neuron in the hypernetwork and set all incoming weights associated with that specific neuron to zero. Since the output layer is constructed without a bias term, this structural pruning ensures that the hypernetwork's mapping for $w_{ij}^{[k]}(t)$ is strictly nullified across the entire temporal domain.

In addition to the masking mechanism, we employ an S-shaped pruning schedule to dynamically and periodically adjust the pruning threshold τ, as shown in Fig. 3. This strategy is characterized by a "gradual-rapid-gradual" progression and periodic reset. The former facilitates a smooth transition from dense to sparse representations while the latter provides a recovery mechanism, allowing key terms that may have been erroneously pruned during early training stages to be recovered. Such a pruning profile has been shown to significantly enhance training robustness and mitigate the complexity of hyperparameter tuning [29].

The S-shaped schedule is designed to allow the hypernetwork weights to evolve freely in the early stages before progressively imposing sparsity. While various smooth or sigmoid-like functions can achieve this, we implement a periodic, piecewise-defined schedule as a representative example. Within each period E_{period}, the profile is structured as:

$$\tau(E) = \begin{cases} 0, & 0 \le E < E_{\text{zero}}, \\ g(E), & E_{\text{zero}} \le E < E_{\text{zero}} + E_{\text{ramp}}, \\ A, & E_{\text{zero}} + E_{\text{ramp}} \le E \le E_{\text{period}}, \end{cases} \tag{10}$$

where A is the maximum amplitude of the pruning schedule, and $g(E)$ can be constructed with sinusoidal function as:

$$g(E) = \frac{A}{2}\left[1 - \cos\left(\pi \frac{E - E_{\text{zero}}}{E_{\text{ramp}}}\right)\right]. \tag{11}$$

2.4. Training procedure

Having introduced the sparsity-promoting pruning method, we are ready to present the details of training. The flowchart of training procedure is shown in Fig. 4. The process begins by collecting a sample set $\{\boldsymbol{x}_i\}_{i=1}^m$ (w/o noise) followed by estimating time derivatives using finite difference. The estimates then undergo digital filtering to mitigate noise. The processed dataset $\{\boldsymbol{x}_i, \dot{\boldsymbol{x}}_i\}_{i=1}^m$ is fed into the H-SREINet, which is trained in conjunction with the presented pruning strategy. The pseudo-code for the training procedure is given in Algorithm 1.

Again, the dynamics for each dimension are trained sequentially. This decoupled approach enhances memory efficiency and improves scalability for high-dimensional systems, though a parallel implementation remains feasible in principle. At the start of each epoch, a pruning threshold is determined based on the current epoch index. This threshold generates binary masks that

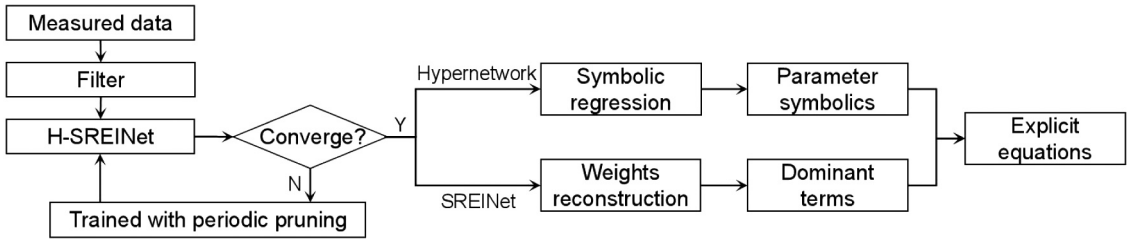

Fig. 4. Training flowchart.

prune the time-varying weights by zeroing the corresponding connections in the hypernetwork's output layer, or the last-layer links responsible for generating the weight values. Since the weights are time-dependent, we evaluate the hypernetwork's output across a uniformly sampled temporal grid. A weight is subject to pruning if its maximum absolute value (L_∞ norm) over the sampled interval falls below the predefined threshold, as defined in Eq. (9).

The hypernetwork configuration is selected based on a sensitivity analysis of the MLP architecture, activation function, batch size, and learning rate. The complete results are reported in Appendix A.

Algorithm 1 Training procedure of H-SREINet

Require: Data dimension: n; nonlinearity order: p; pruning coefficients: $N_{\text{period}}, N_{\text{zero}}, N_{\text{ramp}}$; Pruning epochs per period: E_{period}; Invariant test input: I_{test}; Function library length: k.

pruning_schedule_ramp($N_{\text{period}}, N_{\text{zero}}, N_{\text{ramp}}$)
for $i = 1, 2, \ldots, n$ **do**
  for epoch $= 1, 2, \ldots, E_{\text{period}} * N_{\text{period}}$ **do**
    τ = pruning_schedule[epoch]
    train_dataset.shuffle(dataset)
    Mask = prune_model_and_generate_masks(τ, I_{test})
    for (x_l, y_l) in dataset **do**
      train_loss[l] = train_step(dataset.batch_size, Mask)
    end for
    if early_stopping_check is met **then**
      Break
    end if
  end for
end for

After convergence, the next step is to extract the explicit form of governing equations from H-SREINet. This is accomplished through two parallel steps: (1) identifying and reconstructing dominant terms from SREINet and (2) recovering explicit expressions of their weights through SR [41]. First, a Depth-First Search (DFS) [42] algorithm is employed to traverse the network and identify all active paths whose links survived the pruning process; paths sharing identical symbolic structures are merged by summing their respective weights. This step recovers the dominant equation skeleton independently of the subsequent symbolic-regression step. Since these weights are inherently functional rather than constant, we feed a uniform non-temporal distribution into the embedded Hypernetwork to generate the corresponding coefficient outputs across the sampled domain. Subsequently, symbolic regression is applied to these input–output pairs to distill the precise mathematical forms of the functional coefficients. Finally, by assembling these recovered symbolic coefficients with their respective dominant terms, we reconstruct the closed-form governing equations that are expected to provide more physical insights than black-box models.

3. Results

To understand how H-SREINet performs in practice, we examine three nonlinear systems that vary in their dimensionality, the complexity of their nonlinearities, and their time-varying parameter patterns. For each system, we generate a training dataset $\{\boldsymbol{x}_i\}_{i=1}^m$ (or $\{\boldsymbol{x}_i, \dot{\boldsymbol{x}}_i\}_{i=1}^m$) by simulating a set of trajectories from different initial conditions, ensuring the dataset thoroughly covers the system's dynamic range. In this context, recovering the governing equations is generally more demanding than in static cases. Because the hypernetwork introduces a larger number of trainable parameters to handle the time-dependent parameters of the system, a more substantial volume of data is required to ensure the model learns these complex relationships faithfully.

In the following examples, time-derivatives of states are either directly sampled or estimated via the finite difference method. Where measurement noise is present, a moving average filter with a window length of 7 is employed to denoise the signals, which has proven more effective than the Butterworth [43] and Savitzky–Golay [44] filters for this specific task.

As stated in Section 2.4, the hypernetwork is implemented as an MLP with two hidden layers, each containing 16 neurons activated by the Gaussian Error Linear Unit (GELU) function. H-SREINet is initialized with a Gaussian distribution $\mathcal{N}(\mu = 0, \sigma =$

Table 1
Identified time-varying parameters of the Duffing system via H-SREINet with symbolic regression.

	$\alpha(t)$	$\delta(t)$	$\beta(t)$
Ground truth	$2+\cos(3t)$	$0.5+0.5\sin(t)$	$-0.5t+2$
Noise free	$2.0133+\cos(2.9943t)$	$0.5009+0.4979\sin(t)$	$-0.5203t+2.0140$
30 dB SNR	$1.9513+\cos(3.0712t)$	$0.4846+0.4697\sin(t)$	$-0.4684t+1.9349$
20 dB SNR	$1.9237+\cos(2.9033t)$	$0.5823+0.3810\sin(t)$	$-0.5988t+2.0724$

Table 2
Training and validation losses under different noise levels of the Duffing system.

Noise level	Train loss	Val loss	Eqs. Discovered?
Noise free	7.33e−5	7.88e−5	✓
30 dB SNR	2.87e−1	2.81e−1	✓
20 dB SNR	2.65	2.49	✓

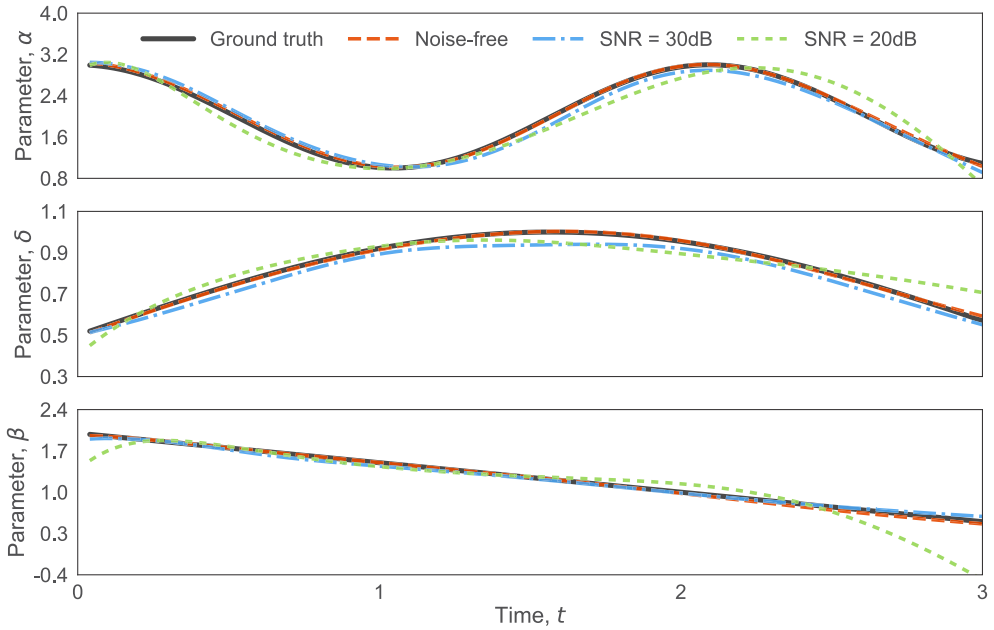

Fig. 5. Time-varying parameters predicted by the hypernetwork of H-SREINet under different noise levels.

0.001), trained using the Adam optimizer with a learning rate of 0.01, a mini-batch size of 256, and an early stopping criterion at a loss of 10^{-7}. The total number of training epochs is governed by the pruning periods and pruning epochs. The pruning schedule consists of 30 periods, with 30 pruning epochs in each period for all cases, except for the Rössler system, which requires more training epochs to adapt to the switching behavior. SR is implemented via PySR [41], with 50 evolutionary iterations and a maximum expression complexity of 10. We adopt basic binary operators (+, −, ∗) and task-adaptive unary operators including trigonometric, logarithmic, and polynomial functions. All cases are run on an 11th Gen Intel(R) Core(TM) i7-11390H CPU @ 3.40 GHz (2.92 GHz).

3.1. Example 1: Time-variant Duffing oscillator

We first apply H-SREINet to learn the dynamics of a time-varying Duffing oscillator in the form

$$\begin{aligned}\dot{x}_1 &= x_2,\\ \dot{x}_2 &= -\delta(t)x_2 - \alpha(t)x_1 - \beta(t)x_1^3,\end{aligned} \tag{12}$$

where $\delta(t)=0.5+0.5\sin(t)$, $\alpha(t)=2+\cos(3t)$, $\beta(t)=-0.5t+2$. While these parameter profiles are specifically designed to benchmark the performance of H-SREINet, they mirror realistic engineering conditions. Specifically, the periodically fluctuating damping δmimics the behavior of semi-active control devices such as MR/ER fluid dampers [45]. The periodic linear stiffness α represents systems under parametric excitation [46], such as structures subjected to periodic axial loads. Lastly, the linear evolution of the nonlinear stiffness β accounts for cumulative thermal effects [47].

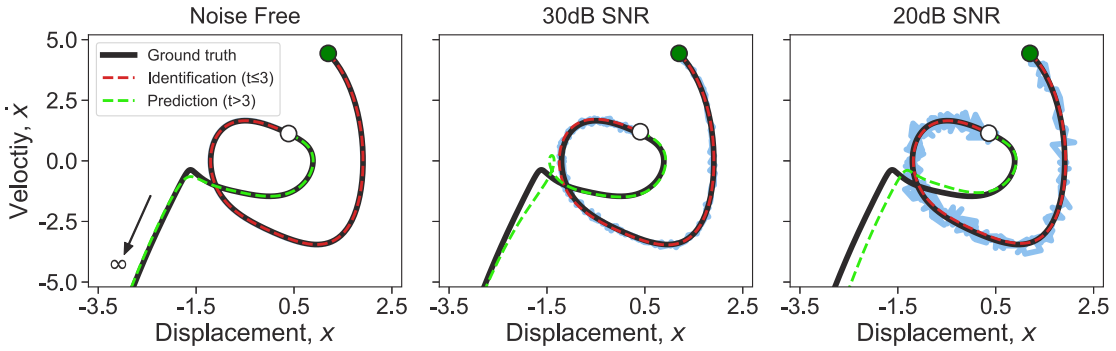

Fig. 6. Comparison of simulated trajectories of ground truth and identified model under noisy data. From left to right: noise free, 30 dB SNR, and 20 dB SNR. The H-SREINet framework utilizes symbolic regression to discover the explicit functional forms of time-varying coefficients. These coefficients are integrated into the structural skeleton of SREINet to construct closed-form governing equations. Under the tested noise levels, the discovered governing equations can still predict beyond the training horizon (green), with visible degradation as noise increases.

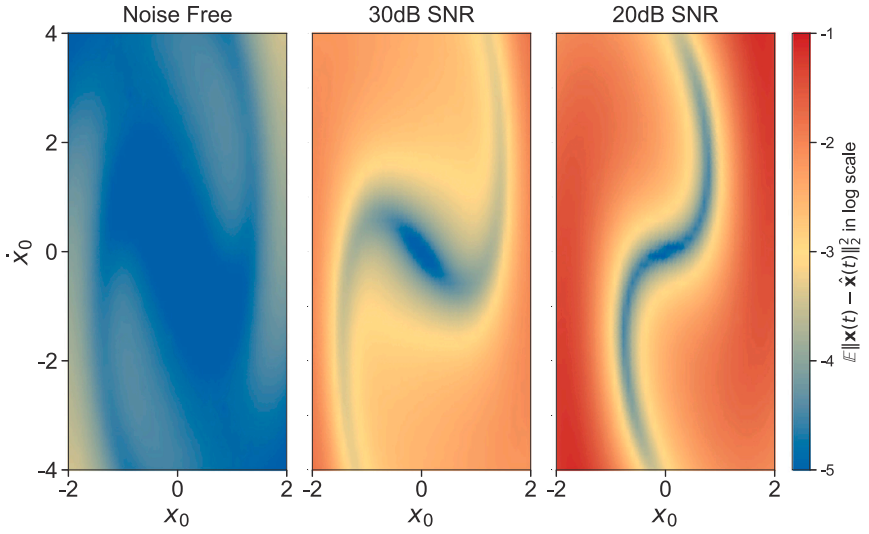

Fig. 7. The prediction error distribution over the initial-condition plane of the identified time-varying Duffing oscillator. From left to right: noise-free, 30 dB SNR and 20 dB SNR, respectively.

We generate a dataset with 6000 points from 20 distinct trajectories simulated for a duration of $T = 3$, with a time step $dt = 0.01$. Additional datasets with signal-to-noise ratios (SNRs) of 20 dB and 30 dB are generated for comparison. To construct the H-SREINet, we use a three-layer SREINet, assigning each layer a single copy of atomic functions $\{1, x, \dot{x}, \exp(\dot{x})\}$, while the pruning threshold is set to 0.5.

We present the predicted time-varying parameters across various noise levels within the training horizon $[0,\ 3]$ in Fig. 5. Without noise, the hypernetworks match the ground truth closely. As noise level increases, the predictions gradually deviate from the ground truth, which are more apparent near initial and final time, especially for 20 dB SNR. These boundary discrepancies arise from the edge effects inherent in the moving-average process, where the truncation of the sliding window at the two tails prevents effective noise reduction and biases the parameter identification. For this reason, we have removed the first and last three points when performing symbolic regression. The discovered symbolic forms, as shown in Table 1, capture the symbolic structures of the parameters and recover their functional coefficients, including both constant offsets and time-dependent components. Besides, the average training loss and validation loss of two dimensions are listed in Table 2. Although the training and validation losses increase significantly with the rise of noise level, H-SREINet recovers governing equations in all tested cases up to 20 dB SNR. We note that 20 dB SNR can represent moderate noise in some practical settings, but it should not be interpreted as a severe-noise regime. Stronger-noise cases are therefore examined separately in Appendix B. The identification error is at the level of 10^{-3} without noise and increases to the level of 10^{-2} (δ) and 10^{-1} (α and β) with 20 dB noise.

How does the agreement in parameter symbolics from Table 1 translate to prediction accuracy? To answer this question, we simulate the identified symbolic models from a trained initial condition and compare the trajectories with the ground truth, as illustrated in Fig. 6. The dynamics of the identified models align closely with the actual trajectory within the training time span,

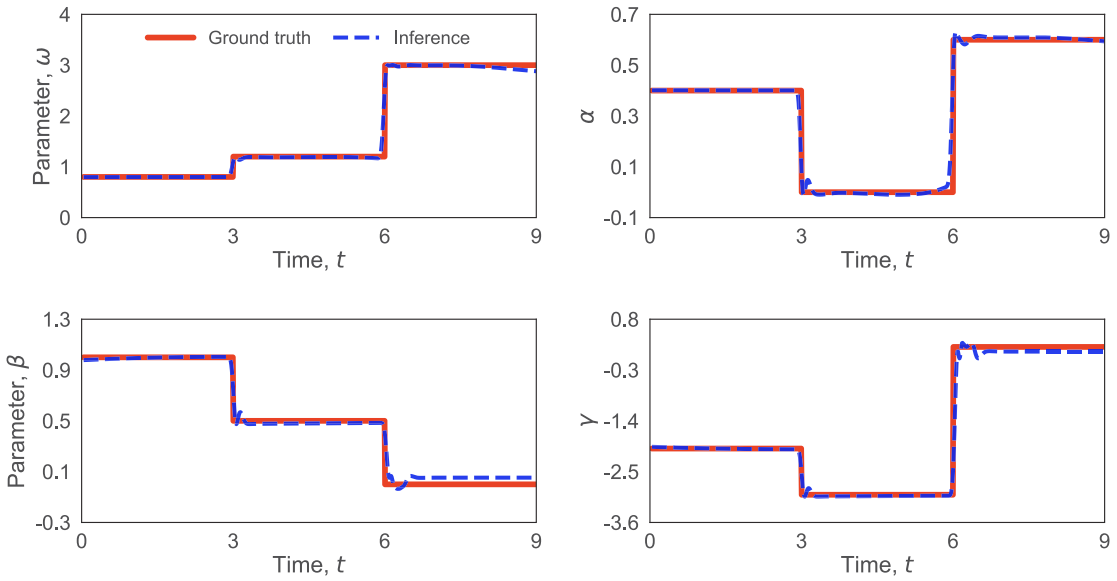

Fig. 8. Time-varying parameters inferred by the H-SREINet of the Rössler system.

regardless of noise. All models reproduce the dynamics in the untrained region up to $t = 8$, although small deviations are observed under noise. Notably, due to the linear time-varying nature of the nonlinear stiffness, the actual system eventually enters an unstable regime where the trajectory diverges. H-SREINet characterizes this transition from stable oscillation to instability in this example, suggesting its potential use in equation-based tipping-point analysis. We will explore this topic further in the last example. To further examine the generalization ability of these models, we perform numerical simulations with initial conditions sampled from the phase space $\{(x_1, x_2)|x_1 \in [-2, 2] \text{ and } x_2 \in [-4, 4]\}$. For each simulation, we record a trajectory with $t \in [0, 6]$, and the prediction error was quantified using Mean Squared Error (MSE), calculated as $\mathbb{E}\|\boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)\|_2^2$. The error distribution over the phase plane of initial conditions is visualized in Fig. 7, where the color mapping represents the log_{10}(MSE) across the sampled phase plane. In the noise-free scenario, the model achieves high precision with the MSE consistently maintained below 10^{-4} throughout the entire domain (indicated by the deep blue regions). As the signal-to-noise ratio decreases to 30 dB and 20 dB, the prediction error increases, particularly for initial conditions farther from the origin. Notably, while the error magnitude rises in the high-energy regions of the phase space at 20 dB, the model still maintains relatively high predictive accuracy near the equilibrium points. Thus, the results support stable prediction under the tested moderate-noise conditions, while the stronger-noise study in Appendix B shows that the method degrades and may fail when the SNR becomes much lower.

3.2. Example 2: Switching Rössler system

In our second example, we show that H-SREINet can also be applied to nonlinear dynamical systems with piecewise continuous coefficients. A special case of such systems is switching dynamical systems, where the governing laws undergo abrupt changes at pre-determined time instants, a common phenomenon in scheduled industrial batch processes and PWM (pulse-width-modulation) controlled systems. Consider a modified Rössler system with time-varying structures:

$$\begin{aligned} \dot{x} &= -\omega(t)y - z, \\ \dot{y} &= \omega(t)x + \alpha(t)y, \\ \dot{z} &= xz + \gamma(t)z + \beta(t), \end{aligned} \tag{13}$$

where coefficients $\omega(t)$, $\alpha(t)$, $\beta(t)$ and $\gamma(t)$ are piecewise constant over the intervals $I_1 = [0, 3)$, $I_2 = [3, 6)$, and $I_3 = [6, 9)$. In this case, the system dynamics with $t > 9$ is not considered.

We collect 40 trajectories with random initial conditions under 40 dB SNR for a duration of $T = 9$, resulting in a total of $40 \times 900 = 36{,}000$ points. Notably, due to the characteristics of switching components, more trajectories and pruning periods are required to better capture the system dynamics. We conduct the same identification process as the aforementioned Duffing example. For the configuration of H-SREINet, the function library takes $\{1, x, y, z\}$, the depth of hidden layers takes 2, the pruning threshold takes 0.2, the pruning period is 80 and each period has 80 pruning epochs. Upon convergence of H-SREINet, the mean of training losses across the three dimensions is 3.46e−1 while the mean of validation losses is 3.19e−1, respectively.

The parameters estimated by the hypernetwork across the training time period are presented in Fig. 8, where the dashed line and solid line represent the estimation and the ground truth, respectively. The estimation aligns with the reference within each piecewise continuous segment, while small deviations are observed during transition. We attribute this to discrete temporal sampling, where

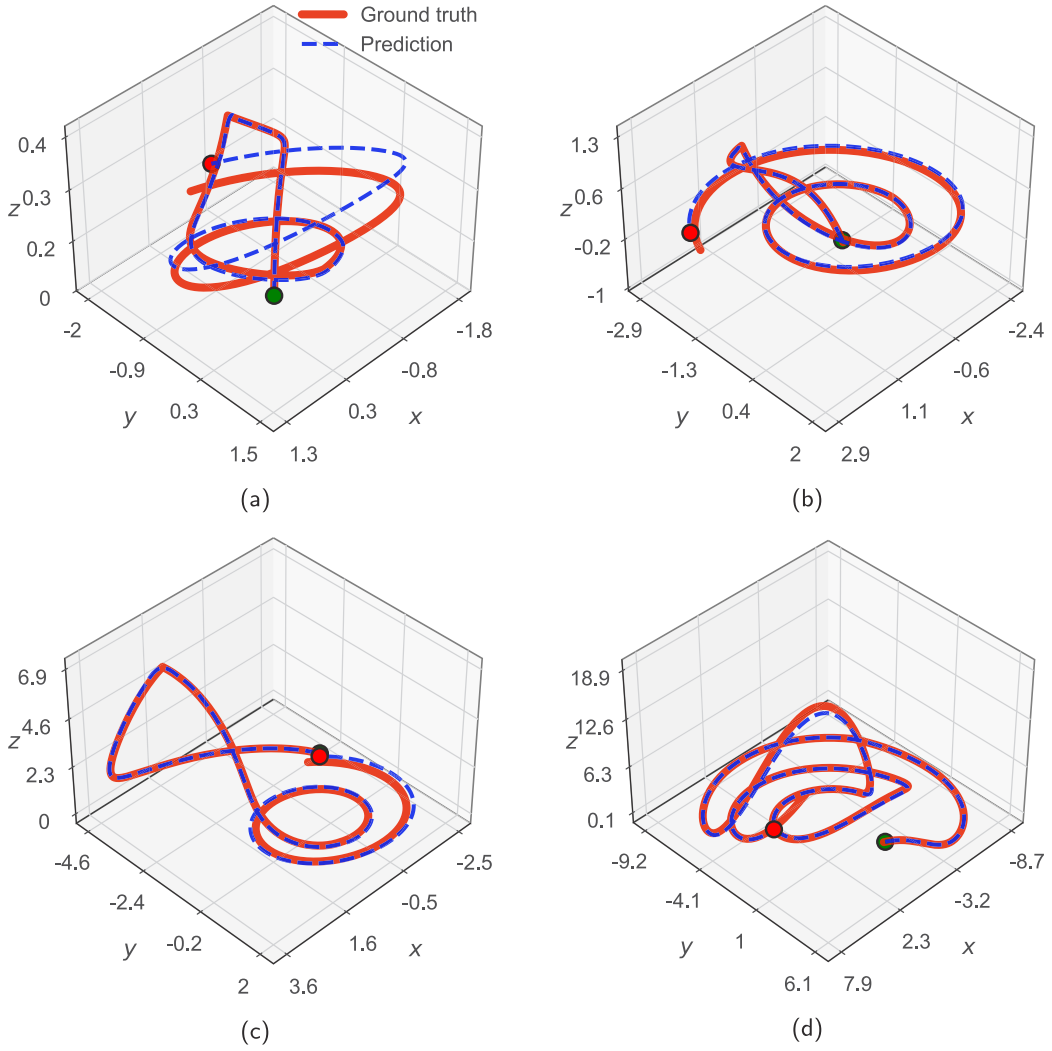

Fig. 9. The phase trajectories of the discovered Rössler system equations under different initial states as (a) $[0, 0, 0]$, (b) $[-1, -1, -1]$, (c) $[-2, -2, 0]$, (d) $[2, 5, 5]$.

Table 3
Identified time-varying parameters of the Rössler system via H-SREINet with symbolic regression.

Metric	Time	$\omega(t)$	$\widehat{\omega}(t)$	$\alpha(t)$	$\widehat{\alpha}(t)$	$\beta(t)$	$\widehat{\beta}(t)$	$\gamma(t)$	$\widehat{\gamma}(t)$
Coeff. value	$[0, 3)$	0.8	0.8009	0.4	0.3964	1	0.9930	−2	−2.0047
	$[3, 6)$	1.2	1.2083	0	0.0064	0.5	0.4831	−3	−3.0155
	$[6, 9]$	3.0	2.9669	0.6	0.6055	0	0.0497	0.2	0.0817
Coeff. MSE	$[0, 9]$	4.93e−3		8.20e−4		1.76e−3		3.05e−2	

the sampling set fails to precisely capture the transition. Consequently, the hypernetwork performs smooth interpolation across the piecewise continuous segments. While increasing the sampling density could theoretically improve transition capture, we do not pursue this route as a reduced sampling interval Δt inevitably amplifies the noise sensitivity of finite-difference approximations, thereby compromising the overall identification accuracy. Despite this small deviation, the hypernetwork achieves good predictive accuracy. The estimated parameters via symbolic regression at each time interval, and the resulting MSE of hypernetwork predictions for time-varying parameters are listed in Table 3.

To understand the extrapolation capability of the identified model, we simulate its dynamics using four additional initial conditions that are not contained in the training set. The initial conditions are separated from each other so that the trajectories sample distinct nonlinear features of the system. The comparisons between ground truth and prediction are shown in Fig. 9, where

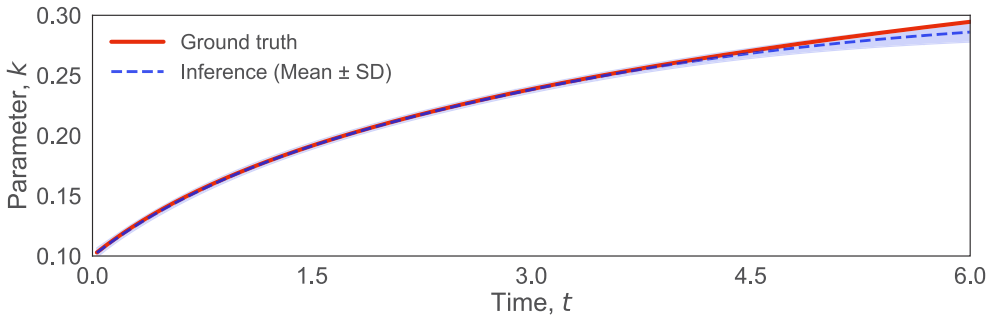

Fig. 10. Time evolution of the mean of the parameters k_i predicted by the hypernetwork ($i = 1, 2, \ldots, 5$.). The shaded area indicates the standard deviation around the mean. The increased deviation after $t > 4.5$ is attributed to the strong coupling law that induces oscillator synchronization. This restricts the dynamics to a low-dimensional manifold, thereby making it difficult to fully capture the system's nonlinearity.

Table 4
Discovered $\widehat{k}_N$ and $\widehat{\omega}_i$ of the Kuramoto oscillators and the corresponding analytical forms. k_i are identical across all dimensions.

Dimension i	k_i	$\widehat{k}_i$	ω_i	$\widehat{\omega}_i$
1	$0.1 + 0.1\ln(t+1)$	$0.1334 + 0.0818\ln(t+0.6419)$	0.7	0.7089
2	–	$0.1196 + 0.0871\ln(t+0.7296)$	1.3	1.2991
3	–	$0.1120 + 0.0936\ln(t+0.8594)$	0.9	0.8990
4	–	$0.1088 + 0.0943\ln(t+0.8756)$	1.4	1.3978
5	–	$0.1115 + 0.0937\ln(t+0.8778)$	0.6	0.6025

Table 5
Effects of threshold on identification performance of H-SREINet. The semi-interpretable network discovers the correct equations across a range of threshold, indicating the robustness of the method.

Threshold	$\lvert e_{\text{esti\_para}}\rvert_{\text{ave}}$ (10^{-5})	Train loss (10^{-5})	Val loss (10^{-5})	Eqs. Discovered?
0.02	3.15	4.65	4.97	×
0.04	1.23	4.67	4.51	✓
0.06	1.01	4.56	4.37	✓
0.08	1.72	4.37	4.61	✓
0.10	1.91	4.67	4.72	✓
0.12	1.21	5.05	5.14	✓
0.14	795	203	204	×
0.16	3123	123	125	×
0.18	3761	1349	1350	×
0.20	5028	1213	1163	×

the green circle represents the initial point and the red one depicts the end point at $t = 9$. In all tested cases, the prediction (blue dashed line) closely matches the reference (red solid line).

3.3. Example 3: Kuramoto oscillators with logarithmic time-varying coefficients

In order to test the ability of the proposed method in a higher-dimensional setting, we consider an extended Kuramoto model with a time-varying coupling law. The original Kuramoto model has been used extensively to study the synchronization of coupled phase oscillators and phase transition from a disordered state to a collective synchronized state [48]. The differential equations of the modified Kuramoto oscillator read

$$\dot{\theta}_i = \frac{K_i(t)}{N} \sum_{j=1, j\neq i}^{N} \sin(\theta_j - \theta_i) + \omega_i, \tag{14}$$

where θ_i is the phase angle in radian, N denotes the number of oscillators, ω_i represents the inherent frequency, and $K(t)$ stands for the time-varying coupling strength.

We use 5 nodes with random inherent frequencies $[\omega_1, \omega_2, \omega_3, \omega_4, \omega_5] = [0.7, 1.3, 0.9, 1.4, 0.6]$. $K_i(t)$ takes the form: $K_i = 0.5 + 0.5\ln(t+1), i = 1, 2, \ldots, 5$. For convenience, define $k_i \triangleq K_i/N$. We generate a dataset of 12,000 samples from 20 trajectories over $t \in [0, 6]$ with a 40 dB SNR, incorporating direct measurements of both states and derivatives. To learn the dynamics of this Kuramoto

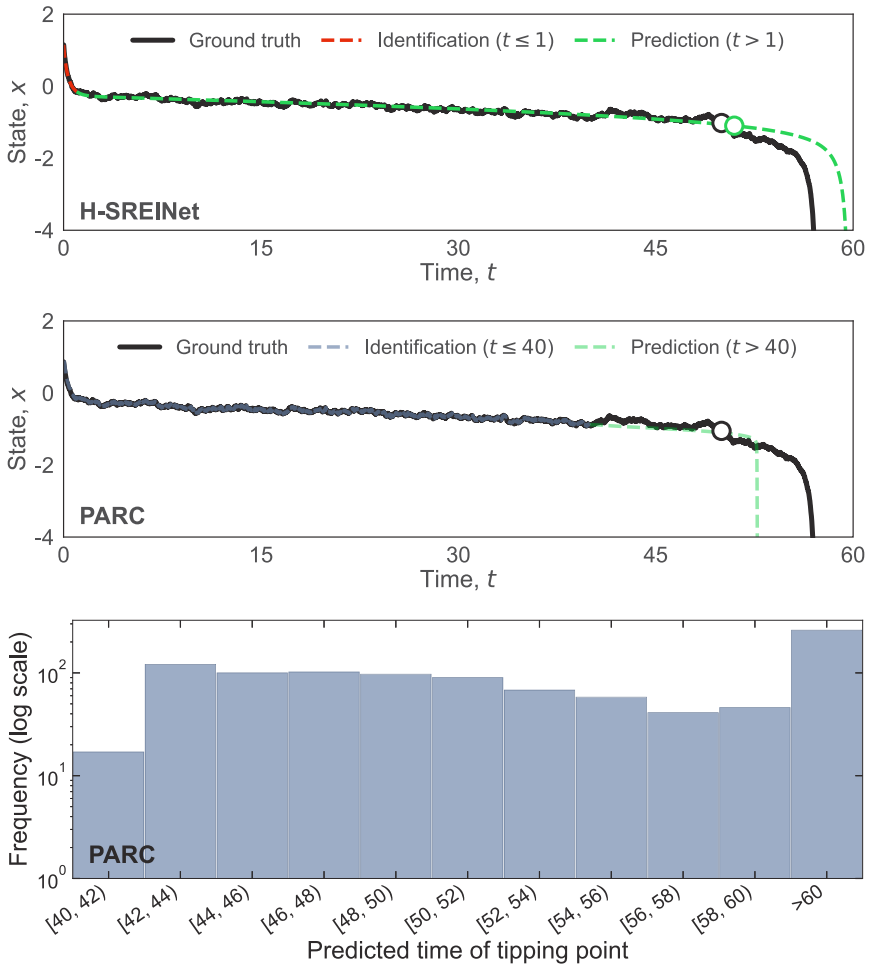

Fig. 11. Comparison of H-SREINet and PARC approaches for tipping-point prediction. (upper) The identified H-SREINet model extrapolates the pre- and post-tipping dynamics from the recovered governing equation. The analytical tipping point (black circle) emerges at $t = 50$, and the predicted one appears at $t = 50.97$. (middle) The comparative PARC approach predicts the collapse of the state trajectory at approximately $t = 52.50$ when the time-varying λ is provided. (bottom) Histogram of predicted AMOC tipping points from 1000 PARC realizations. Initial condition: $x_0 = 1.2$.

model, we apply a H-SREINet that is composed by a 2-layer SREINet with atomic functions $\{1, \sin\theta_1, \ldots, \sin\theta_5, \cos\theta_1, \ldots, \cos\theta_5\}$ and a pruning threshold of 0.05.

The skeleton of the discovered model is identical to that of Eq. (14), although the cross-coupling terms $\sin(\theta_j - \theta_i)$ are expressed as $\sin\theta_j \cos\theta_i - \cos\theta_i \sin\theta_j$. The time evolution of the mean of the parameter k_i ($i = 1, 2, \ldots, 5$) is illustrated in Fig. 10. The hypernetwork of H-SREINet can capture the evolution of the coupling strength until $t > 4.5$. Beyond this point, the strong coupling law induces network synchronization, causing the dynamics to evolve on a low-dimensional manifold. In this case, the nonlinearity is not fully reflected by the data; consequently, identifying the exact parameters k_i becomes challenging. Nevertheless, the learned parameters exhibit only minor fluctuations around the mean. With symbolic regression, the expressions of estimated parameters $\widehat{k}_N$ and $\widehat{\omega}_i$ are well aligned with analytical forms, as shown in Table 4.

When the method is scaled to high-dimensional problems, the number of time-varying weights increases quadratically. In our experience, the rapid expansion of the candidate function library increases the risk of overfitting. This makes it challenging to select an appropriate pruning threshold, especially when using the classical Sequential Thresholding Least Square (STLSQ) method [23]. A robust pruning algorithm should ideally remain stable across a broad range of thresholds. To test this, we evaluate the convergence of our algorithm by iterating through an array of threshold values. As shown in Table 5, a small threshold (e.g., 0.02) leads to overfitting, while an excessively large threshold overprunes the system by eliminating dominant terms. However, there is a stable regime between 0.04 and 0.12 where the loss amplitude remains consistent and the correct governing equations are identified. This suggests that the threshold sensitivity is unlikely to pose a significant challenge in practical applications.

Table 6
AMOC system parameters identified by the H-SREINet.

Parameter	A	$\widehat{A}$	m	$\widehat{m}$	$\lambda(t)$	$\widehat{\lambda}(t)$
Value	0.95	0.9488	−1.3	−1.3015	$-1+0.02t$	$-1.001-0.01964t$

3.4. Application: Tipping point prediction

Having seen the H-SREINet's capability to discover nonlinear dynamics with time-varying parameters and switching structures, we now show how the identified governing equations can be further utilized for tipping-point prediction. We consider predicting the collapse of the Atlantic Meridional Overturning Circulation (AMOC) [49], which has been analyzed through a 1D stochastic proxy model for sea surface temperature (SST) [50]:

$$\dot{X}(t) = -A(X(t)-m)^2 - \lambda(t) + \sigma dB_t, \tag{15}$$

where $X(t)$ is a stochastic variable exhibiting a tipping transition, while A and m denote system constants, σ represents the noise amplitude, and dB_t stands for a Brownian motion. At $\lambda = 0$, a saddle–node bifurcation occurs, marking the tipping point and indicating the potential collapse of the AMOC. We will show that under weak noise, H-SREINet is still applicable to discover the drift term (deterministic part), which is sufficient for tipping prediction.

To generate the dataset, we numerically solve this stochastic equation using the Euler–Maruyama method with a noise strength of $\sigma = 0.1$. We collect 20 trajectories of the state x and its derivative $\dot{x}$ for $t \in [0, 1]$, encompassing 2000 data points. Admittedly, obtaining multiple trajectories from a single climate system is rarely feasible. However, this setup serves as an idealized scenario to validate the effectiveness of the H-SREINet.

We employ a 3-layer (higher than the actual nonlinear order) SREINet with atomic functions $\{1, x\}$ and set the pruning threshold to 0.5. Due to the limited number of data points, the batch size is decreased to 64. Training stops early at epoch 562 when the early-stopping loss criterion is met. H-SREINet accurately identifies the drift term, where the estimated symbolic form of the parameters and their corresponding ground-truth are presented in Table 6.

To predict a tipping point, we select one trajectory from the training set and continue simulating both the synthetic model and identified model until the dynamics after tipping points are clearly observed. The comparison between the ground truth and prediction from the discovered model is illustrated in the upper panel of Fig. 11. The system collapses at $t = 50$, whereas the discovered model forecasts a tipping point at $t = 50.97$. With short transient data over $t \leq 1$, the recovered equation can extrapolate both pre- and post-tipping dynamics in this idealized example.

We further compare our H-SREINet against Parameter-Adaptable Reservoir Computing (PARC) [5]. After training solely on time series over $t \in [0, 40]$, one predicted state trajectory with known time-varying λ is demonstrated in the middle panel of Fig. 11, where the system trajectory collapses at approximately $t = 52.50$. This PARC approach yields predicted tipping points that follow a concentrated statistical distribution, as visualized in the bottom panel of Fig. 11. Among 1000 independent realizations, 260 trials predict the tipping event to occur after $t = 60$, and 51.00% of predictions fall within the interval $[42, 52)$.

Remarkably, there is an essential difference between the two approaches. The prediction reliability of H-SREINet hinges on the accuracy of the recovered equations. In addition, it requires multiple short-duration trajectories in this example. In contrast, the PARC approach follows an end-to-end paradigm. It generally requires pre-specified time-varying parameters as inputs and longer training trajectories to achieve satisfactory prediction accuracy, without explicitly reconstructing governing dynamical equations.

3.5. Performance comparison with existing methods

We compare H-SREINet with three existing methods (HEQL, TV-SINDy, and DSINDy) through the Duffing oscillator in Example 1. For HEQL [38], we adopt a two-layer structure with 2 constant nodes, 4 identity nodes, and 2 product nodes per layer. Time-Varying SINDy (TV-SINDy) [30] uses a candidate library including polynomials up to third order, trained with L_1 regularization over sliding windows of size 20. DSINDy [37] shares the identical function library with TV-SINDy, and is optimized via joint Kullback–Leibler (KL) divergence loss and L_1 sparsity penalty. A unified maximum pruning threshold is applied across all approaches: time-varying parameters are retained if the maximum absolute values of their trajectories surpass this threshold. A discussion on the time cost and memory requirements of these various methods is given in Appendix C.

Corresponding inferred time-varying parameters under different noise levels for the training horizon $t \in [0, 3]$ are illustrated in Fig. 12. Under the tested noise-free, 30 dB, and 20 dB SNR conditions, H-SREINet maintains better fitting consistency than HEQL, TVSINDy and DSINDy. As noise intensity increases, the baseline methods suffer from severe trajectory oscillations and prominent drift away from the true time-varying parameter evolution.

The MSE of identified parameters under three noise scenarios is summarized in Table 7, alongside the judgment of successful governing equation discovery for each competing method. It can be observed that the proposed H-SREINet achieves the lowest MSE values for time-varying parameters across noise-free, 30 dB and 20 dB SNR and successfully identifies the correct dynamical equations under the test settings. By contrast, HEQL fails to recover the true equation structure at 20 dB SNR, while TVSINDy and DSINDy produce redundant terms and miss the correct governing equations when noise is introduced.

Furthermore, the values listed under the Complexity column in Table 7 denote discovered number of terms. H-SREINet identifies the three physically meaningful dominant terms of the Duffing system without introducing spurious redundant terms under three noise scenarios. In contrast, the baseline methods generate redundant terms under 20 dB SNR. The comparison of discovered governing equations at selected time instants is presented in Appendix D.

Table 7
MSE of identified time-varying parameters of the Duffing system by different methods, where the values marked in the Complexity denote discovered number of terms. H-SREINet discovers the correct equation structures with smaller coefficient errors under the tested noise-free, 30 dB, and 20 dB SNR conditions.

Methods	Noise free			30 dB SNR			20 dB SNR			Complexity			
	e_α	e_δ	e_β	e_α	e_δ	e_β	e_α	e_δ	e_β	Ref.	Clean	30 dB	20 dB
H-SREINet	1.0e−3	3.4e−5	7.1e−4	9.3e−3	1.5e−3	1.5e−3	3.6e−2	2.4e−3	5.9e−2	3	3	3	3
HEQL	7.2e−3	5.5e−4	1.4e−2	2.4e−2	5.0e−4	1.6e−1	8.6e−2	1.6e−2	3.9e−1	3	3	3	7
TVSINDy	4.4e−3	1.1e−3	9.9e−3	2.0e−2	8.7e−3	7.9e−2	1.6e−1	3.6e−2	4.7e−1	3	3	4	8
DSINDy	5.6e−2	6.1e−3	4.3e−2	4.4e−1	1.4e−1	2.8	3.5	1.1	21.8	3	3	9	9

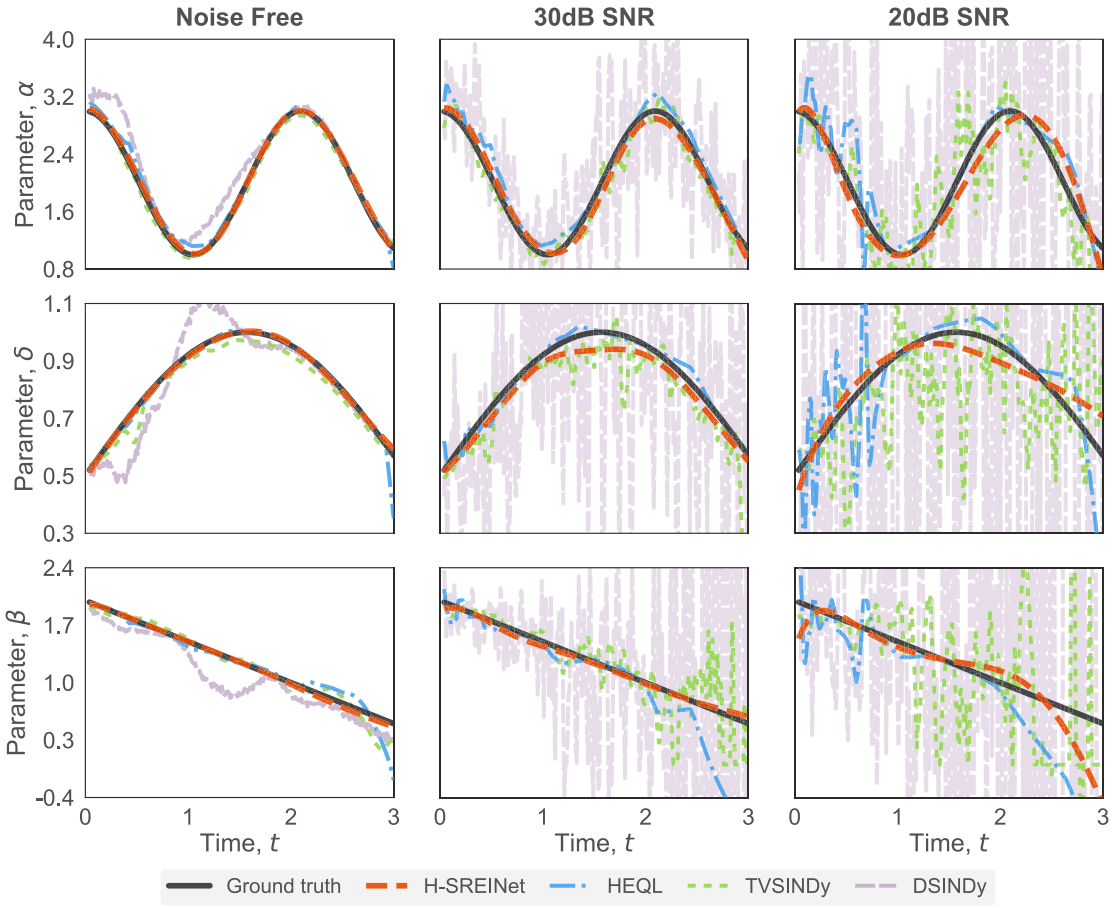

Fig. 12. Comparison of inferred time-varying parameters under different noise levels. Lower SNR represents higher noise level.

Table 8
H-SREINet performance under varying dimensionalities of the Kuramoto system.

Dimension	Weights/Dimension	Train loss	Val loss	Coeff error	Eqs. Discovered?
5	2240	5.37e−5	4.98e−5	3.14e−4	✓
10	7360	6.44e−5	5.85e−5	3.17e−4	✓
15	15,680	7.19e−5	5.19e−5	3.14e−4	✓
20	27,200	6.40e−5	6.33e−5	3.92e−4	✓

3.6. Scaling H-SREINet to high-dimensional systems

To demonstrate the potential of H-SREINet for high-dimensional systems, we revisit the Kuramoto oscillators and increase their nodes to 10, 15 and 20. We keep K_i unchanged, set a pruning threshold of 0.05 for the 10-node case, and reduce the threshold to 0.025 for the 15-node and 20-node scenarios. For these systems, we generate a larger dataset of 30,000 samples from 50 trajectories

and increase the pruning periods to 40, with 40 pruning epochs in each period. As summarized in Table 8, H-SREINet maintains comparable training and validation losses, as well as coefficient errors for the time-varying coupling strength k_i and frequency ω_i. Throughout these test cases, H-SREINet recovers the underlying governing equations, suggesting its potential applicability to higher-dimensional systems. This analysis focuses on scaling with system dimension; comprehensive profiling of wall-clock training cost, memory requirements, and scaling with library size is beyond the scope of the present work and will be investigated in future studies.

4. Conclusions

In this study, we propose a semi-interpretable and end-to-end machine learning architecture for discovering nonlinear dynamics with time-varying and switching characteristics. This work is motivated by the broader need to extend equation discovery from stationary benchmark systems to nonstationary scientific and engineering systems whose governing laws evolve in time. Building upon the structure of SREINet, this framework introduces hypernetworks to capture time-varying/switching coefficients. In this formulation, the SREINet backbone identifies the structural skeleton of the vector field, while the hypernetwork represents the temporal evolution of the associated coefficients. The governing equations are discovered using supervised learning that incorporates a masking-based pruning scheme with a periodically varying pruning threshold. Using symbolic regression, H-SREINet recovers explicit governing equations that can support downstream analysis such as data-driven equation discovery, digital twin modeling, and tipping-point inference. The validation results show that the proposed method can identify interpretable governing equations for several representative systems, including a time-varying Duffing oscillator, a switching Rössler system, and a Kuramoto oscillator network with logarithmic time-varying coefficients. Additional comparisons with existing methods on the Duffing oscillator show that H-SREINet achieves lower coefficient errors and cleaner equation structures under various noise conditions. We also demonstrate, through an idealized AMOC example, that the discovered governing equation can be used to infer tipping behavior beyond the training interval when the recovered dynamics are sufficiently accurate.

Despite these advancements, the method has several limitations that should be addressed in future research. First, the success of discovering underlying governing equations relies on the selection of candidate functions. When critical candidate functions are missing, the algorithm can produce overfitted equations that capture only local dynamics and diverge quickly from the ground truth outside of this local region. This has been a known issue for the explicit methods. Thus, developing a mechanism for the intelligent selection of candidate functions is essential. This dependence also affects the SR step used to convert hypernetwork outputs into closed-form coefficient expressions. SR is most reliable for smooth, low-noise coefficient trajectories with sparse expressions and an adequate function library; noisy fluctuations, missing functions, or overly complex coefficient laws may lead to spurious or structurally incorrect expressions [51,52]. Therefore, denoising strategies [53], noise-resilient model components [54], problem-specific operator selection, and independent validation of recovered coefficient laws remain important. Second, the training data must cover a sufficient region of phase space that represents the nonlinear characteristics of the system under investigation. In practice, this can be achieved either by introducing persistent external excitation (included in the identification process) or sampling data from diverse trajectories. The present framework also assumes regularly sampled and fully observed state data, with derivatives either measured or estimated from the available trajectories. Since H-SREINet constructs candidate functions directly from the observed state variables, it cannot be directly applied to partial-state data without an additional state-reconstruction or latent-variable modeling step. Irregular sampling and strongly data-limited settings may require trajectory-based training or data-assimilation strategies. Finally, the use of hypernetworks increases the amount of training data required, which may not be feasible for data-limited scientific problems (e.g., AMOC). To mitigate this, the point-wise optimization framework could be transitioned to a trajectory-based approach using backpropagation through time (BPTT).

CRediT authorship contribution statement

Binghang Xiao: Writing – original draft, Visualization, Validation, Methodology, Investigation. **Jianzhe Huang:** Writing – review & editing, Supervision, Methodology. **Siyuan Xing:** Writing – review & editing, Supervision, Software, Methodology, Conceptualization.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work, the authors used Google Gemini in order to refine the language and improve the readability of the manuscript. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Table 9
Sensitivity analysis of hyperparameters on parameter estimation performance, where ※ denotes the optimal value within each corresponding category.

Category	Architecture	Batch size	Activation	Learning rate	Parameter error	Train loss	Val loss
Architecture	[16]	256	GELU	0.01	0.0139	0.4815	0.4707
	[16 16] ※	256	GELU	0.01	0.0061	0.4747	0.4712
	[16 16 16]	256	GELU	0.01	0.0100	0.4800	0.4801
	[32]	256	GELU	0.01	0.2523	0.5221	0.5280
	[32 32]	256	GELU	0.01	0.0554	0.4901	0.4903
	[32 32 32]	256	GELU	0.01	0.3434	0.5571	0.5517
Batch size	[16 16]	64	GELU	0.01	0.0892	0.5205	0.5038
	[16 16]	128	GELU	0.01	0.0145	0.4874	0.4559
	[16 16]	**256** ※	GELU	0.01	0.0061	0.4747	0.4712
	[16 16]	512	GELU	0.01	0.0297	0.4741	0.4717
	[16 16]	1024	GELU	0.01	0.0146	0.4779	0.4625
Activation	[16 16]	256	Tanh	0.01	0.1134	0.5219	0.5258
	[16 16]	256	Sigmoid	0.01	1.9633	8.7810	8.5277
	[16 16]	256	**GELU** ※	0.01	0.0061	0.4747	0.4712
	[16 16]	256	ELU	0.01	0.3660	0.5603	0.5493
	[16 16]	256	ReLU	0.01	0.9343	5.0380	4.9067
Learning rate	[16 16]	256	GELU	0.1	2.3549	12.6733	12.7633
	[16 16]	256	GELU	0.05	2.3527	12.9733	13.2567
	[16 16]	256	GELU	**0.01** ※	0.0061	0.4747	0.4712
	[16 16]	256	GELU	0.005	0.0128	0.4760	0.4660
	[16 16]	256	GELU	0.001	0.2357	0.9476	0.9390

Table 10
Identified time-varying parameters under low SNR conditions via H-SREINet with symbolic regression.

	$\alpha(t)$	$\delta(t)$	$\beta(t)$	Train loss	Val loss	Eqs. Discovered?
Ground truth	$2+\cos(3t)$	$0.5+0.5\sin(t)$	$-0.5t+2$	–	–	–
15 dB SNR	$2.08+\cos(3.03t)$	$0.40+0.65\sin(t)$	$-0.39t+1.69$	14.4	14.3	✓
10 dB SNR	$2.45+\cos(3.24t)$	$\sin(0.78t+0.37)$	$-0.31t+1.29$	46.7	48.0	×
5 dB SNR	$2.42+\cos(t)$	$\sin\{\sin[0.53+\sin(t)]\}$	$-t^2+2.18$	145	140	×

Appendix A. Hypernetwork hyperparameter sensitivity analysis

The hyperparameter sensitivity analysis of the H-SREINet framework is conducted on the Duffing oscillator with 30 dB SNR, as shown in Table 9. The parameter errors and training and validation losses are calculated by averaging over 3 random seeds. After performance comparison, a mini-batch size of 256, the GELU activation function, a learning rate of 0.01, and a network architecture with two hidden layers of 16 and 16 hidden units are found to be appropriate for the target system.

Appendix B. Performance of H-SREINet under strong noise

Table 10 summarizes the parameter estimation results and corresponding losses of H-SREINet under 15 dB, 10 dB and 5 dB SNR. Both training and validation losses rise evidently with increasing noise. At 15 dB SNR, the approach correctly recovers the full equation structure with relatively minor parameter deviations. At 10 dB SNR, the approach still identifies three dominant terms, but the recovered form of $\delta(t)$ has a different symbolic form. When the SNR drops to 5 dB, H-SREINet fails to reconstruct the true governing equation, and the inferred parameters deviate severely from ground truth. These trends are further intuitively illustrated in Fig. 13, where the trajectory mismatch between estimated parameters and ground truths gradually enlarges under stronger noise.

Appendix C. Discussion on the time cost and memory requirements

To provide a quantitative estimate of the computational cost, we evaluate the identification methods on the Duffing system with 20 dB SNR and report their wall-clock training time, equation-extraction time, and estimated memory usage in Table 11. Similar to previous experiments, all tests are performed on an 11th Gen Intel(R) Core(TM) i7-11390H CPU.

To make a consistent comparison among methods implemented using different computational frameworks, including neural-network-based and matrix-based approaches, we define the training time as the algorithmic runtime from the beginning of optimization to its completion. Accordingly, the reported values exclude model initialization for H-SREINet and HEQL, as well as construction of the data matrices used by TVSINDy and DSINDy. Although TVSINDy achieves the shortest training time for this relatively low-dimensional example, H-SREINet requires less training time than the other two neural-network-based methods, HEQL and DSINDy.

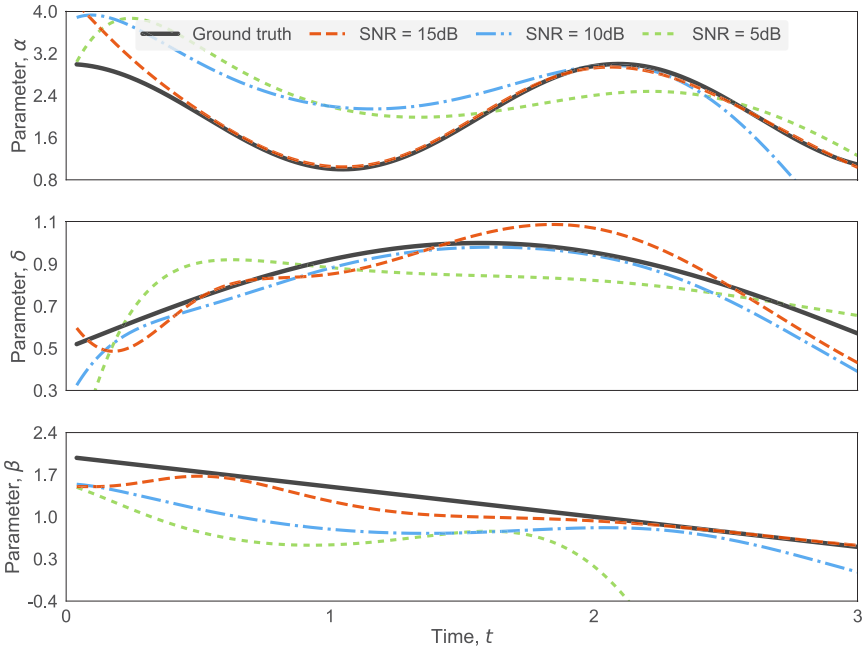

Fig. 13. Time-varying parameters inferred by the hypernetwork of H-SREINet under low SNR. Lower SNR corresponds to larger noise intensity.

Table 11
Computational cost of different identification methods for the Duffing system at 20 dB SNR.

Method	Training time (s)	Eq. extraction time (s)	Estimated memory
H-SREINet	107.6	0.13	113 kB
HEQL	115.1	173.5	1.89 MB
TVSINDy	2.3	N/A	1.13 kB
DSINDy	250.8	0.12	109 MB

Unlike TVSINDy, the neural-network-based methods require an additional post-processing step to extract the governing equations from the trained models. H-SREINet and DSINDy require comparable equation-extraction times, whereas HEQL is substantially slower because its extraction procedure repeatedly converts numerical representations into symbolic expressions using SymPy at individual time instances.

Direct comparison of profiler-reported memory consumption across the different implementations can be misleading because framework-level profiling, particularly for TensorFlow-based implementations, includes runtime and memory-management overhead that is not directly attributable to the identification algorithm itself. We therefore estimate the algorithm-level memory requirement. For the neural-network-based methods, the estimate is in the form

$$M = M_{\text{param}} + M_{\text{batch}} + M_{\text{forward/backward}} + M_{\text{gradient}} + M_{\text{optimizer}},$$

where the individual terms account for model parameters, mini-batch data, forward/backward-pass with intermediate atomic functions, gradients, and optimizer states, respectively.

All memory estimates assume FP32 precision. As shown in Table 11, H-SREINet has a substantially smaller estimated memory requirement than HEQL and DSINDy. This is expected as H-SREINet yields a more compact structure than HEQL. In addition, the large memory requirement of DSINDy is mainly associated with its CNN-based autoencoder architecture which incorporates an encoder with neurons [2, 16, 32, 2400, 4, 4] and a decoder with [4, 2400, 32, 16, 9, 2700, 2673]. This layout introduces significant additional network parameters when mapping measurement trajectories to coefficient representations.

Appendix D. Discovered governing equations at selected time instants

The comparison of discovered governing equations at selected time instants is presented in Table 12. Rows of the table are grouped by three noise scenarios, noise-free, 30 dB SNR, and 20 dB SNR, with two representative time instants listed under each noise level.

Table 12
Discovered governing equations of the Duffing system at selected time instants.

Noise level	Time	Ground truth	H-SREINet	HEQL	TVSINDy	DSINDy
Noise free	0.2	$\dot{x}_2 = -0.60x_2 - 2.83x_1 - 1.90x_1^3$	$\dot{x}_2 = -0.59x_2 - 2.86x_1 - 1.91x_1^3$	$\dot{x}_2 = -0.60x_2 - 2.90x_1 - 1.91x_1^3$	$\dot{x}_2 = -0.57x_2 - 2.83x_1 - 1.85x_1^3$	$\dot{x}_2 = -0.54x_2 - 3.24x_1 - 1.77x_1^3$
	2.8	$\dot{x}_2 = -0.67x_2 - 1.48x_1 - 0.60x_1^3$	$\dot{x}_2 = -0.67x_2 - 1.53x_1 - 0.55x_1^3$	$\dot{x}_2 = -0.67x_2 - 1.52x_1 - 0.51x_1^3$	$\dot{x}_2 = -0.64x_2 - 1.51x_1 - 0.31x_1^3$	$\dot{x}_2 = -0.65x_2 - 1.53x_1 - 0.44x_1^3$
30 dB SNR	0.2	$\dot{x}_2 = -0.60x_2 - 2.83x_1 - 1.90x_1^3$	$\dot{x}_2 = -0.57x_2 - 2.91x_1 - 1.90x_1^3$	$\dot{x}_2 = -0.57x_2 - 2.99x_1 - 2.00x_1^3$	$\dot{x}_2 = -0.57x_2 - 2.88x_1 - 1.86x_1^3 - 0.01x_1x_2^2$	$\dot{x}_2 = -0.32x_2 - 3.33x_1 - 1.74x_1^3 + 0.01x_1^2 - 0.12x_1x_2 - 0.04x_2^2 - 0.13x_1^2x_2 - 0.02x_1x_2^2 - 0.01x_2^3$
	2.8	$\dot{x}_2 = -0.67x_2 - 1.48x_1 - 0.60x_1^3$	$\dot{x}_2 = -0.63x_2 - 1.47x_1 - 0.65x_1^3$	$\dot{x}_2 = -0.67x_2 - 1.59x_1 + 0.45x_1^3$	$\dot{x}_2 = -0.56x_2 - 1.32x_1 - 0.00x_1^3 - 0.79x_1x_2^2$	$\dot{x}_2 = -0.03x_2 - 1.84x_1 + 1.55x_1^3 + 1.11x_1^2 - 0.84x_1x_2 + 0.09x_2^2 - 2.37x_1^2x_2 - 0.68x_1x_2^2 - 0.23x_2^3$
20 dB SNR	0.2	$\dot{x}_2 = -0.60x_2 - 2.83x_1 - 1.90x_1^3$	$\dot{x}_2 = -0.60x_2 - 2.90x_1 - 1.85x_1^3$	$\dot{x}_2 = -0.38x_2 - 3.12x_1 - 1.97x_1^3 - 0.29x_1^2 - 0.02x_2^2 - 0.37x_1x_2 - 0.09x_1^2x_2$	$\dot{x}_2 = -0.68x_2 - 2.70x_1 - 1.96x_1^3 - 0.13x_1^2 + 0.02x_1x_2 - 0.01x_2^2 + 0.11x_1^2x_2 + 0.00x_1x_2^2$	$\dot{x}_2 = -0.30x_2 - 5.77x_1 - 2.29x_1^3 - 1.05x_1^2 + 0.14x_1x_2 + 0.07x_2^2 + 0.19x_1^2x_2 + 0.55x_1x_2^2 - 0.03x_2^3$
	2.8	$\dot{x}_2 = -0.67x_2 - 1.48x_1 - 0.60x_1^3$	$\dot{x}_2 = -0.75x_2 - 1.66x_1 - 0.06x_1^3$	$\dot{x}_2 = -0.68x_2 - 1.71x_1 + 0.75x_1^3 - 0.09x_1^2 - 0.13x_2^2 - 0.22x_1x_2 - 0.16x_1^2x_2$	$\dot{x}_2 = -0.58x_2 - 1.49x_1 + 1.34x_1^3 + 2.20x_1^2 - 0.06x_1x_2 + 0.01x_2^2 + 0.33x_1^2x_2 - 0.57x_1x_2^2$	$\dot{x}_2 = -1.80x_2 - 1.67x_1 + 2.22x_1^3 + 1.36x_1^2 + 3.51x_1x_2 - 1.84x_2^2 + 8.08x_1^2x_2 - 0.61x_1x_2^2 + 0.06x_2^3$

Data availability

Data will be made available on request.

References

[1] Lenton TM, Held H, Kriegler E, Hall JW, Lucht W, Rahmstorf S, Schellnhuber HJ. Tipping elements in the Earth's climate system. Proc Natl Acad Sci 2008;105(6):1786–93.
[2] Jagan M, DeJonge MS, Krylova O, Earn DJ. Fast estimation of time-varying infectious disease transmission rates. PLoS Comput Biol 2020;16(9):e1008124.
[3] Lee S, Chung W, Son H. Online parameter identification framework for a multirotor UAV: Application to an arm stretchable morphing multirotor. Mech Syst Signal Process 2022;166:108468.
[4] Aboudorra M, Saini A, Franchi A. Gain scheduling position control for Fully-Actuated morphing Multi-Rotor UAVs. In: International conference on unmanned aircraft systems. ICUAS, 2024, p. 15–22.
[5] Panahi S, Kong LW, Moradi M, Zhai ZM, Glaz B, Haile M, Lai YC. Machine learning prediction of tipping in complex dynamical systems. Phys Rev Res 2024;6(4).
[6] Li X, Zhu Q, Zhao C, Zhao B, Zhang X, Duan X, Lin W. Ultra-early prediction of tipping points: Integrating dynamical measures with reservoir computing. 2026.
[7] Farrar CR, Worden K. Structural health monitoring: A machine learning perspective. Chichester, UK: John Wiley & Sons; 2012.
[8] Brunton SL, Kutz JN. Data-driven science and engineering: Machine learning, dynamical systems, and control. 2nd ed.. Cambridge University Press; 2022.
[9] Yu R, Wang R. Learning dynamical systems from data: An introduction to physics-guided deep learning. Proc Natl Acad Sci 2024;121(27).
[10] Pillonetto G, Aravkin A, Gedon D, Ljung L, Ribeiro AH, Schön TB. Deep networks for system identification: a survey. Automatica 2025;171.
[11] Tang Z, Wang H, Li J. Different layers models based on neural ordinary differential equations for computing time-varying problems: A survey. IEEE Access 2026.
[12] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput 1997;9(8):1735–80.
[13] Cho K, van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN Encoder–Decoder for statistical machine translation. In: Proceedings of the 2014 conference on empirical methods in natural language processing. EMNLP, 2014, p. 1724–34.
[14] Jaeger H. The "echo state" approach to analysing and training recurrent neural networks-with an erratum note. Technical report 148, Bonn, Germany: German National Research Center for Information Technology GMD; 2001.
[15] Su Y, Fu J, Lin C, Lai X, Zheng Z, Lin Y, He Q. A novel deep learning multi-step prediction model for dam displacement using chrono-initialized LSTM and sequence-to-sequence framework. Expert Syst Appl 2025;271.
[16] Vlachas PR, Byeon W, Wan ZY, Sapsis TP, Koumoutsakos P. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. Proc R Soc A 2018;474(2213).
[17] Pathak J, Hunt B, Girvan M, Lu Z, Ott E. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. Phys Rev Lett 2018;120(2).
[18] Mezić I. Spectral properties of dynamical systems, model reduction and decompositions. Nonlinear Dynam 2005;41(1).
[19] Williams MO, Rowley C, Kevrekidis I. A Kernel-Based approach to data-driven Koopman spectral analysis. J Comput Dyn 2015;2(2).
[20] Schmidt M, Lipson H. Distilling free-form natural laws from experimental data. Science 2009;324(5923):81–5.
[21] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Addison-Wesley; 1989.
[22] Wang W-X, Yang R, Lai Y-C, Vassilios K, Grebogi C. Predicting catastrophes in nonlinear dynamical systems by compressive sensing. Phys Rev Lett 2011;106(15):154101, (4 pages).
[23] Brunton SL, Proctor JL, Kutz JN. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. Proc Natl Acad Sci 2016;113(15).
[24] Sahoo S, Lampert C, Martius G. Learning equations for extrapolation and control. In: International conference on machine learning. 2018.
[25] Liu Z, Wang Y, Vaidya S, Ruehle F, Halverson J, Soljačić M, Hou TY, Tegmark M. KAN: Kolmogorov–Arnold networks. In: International Conference on Learning Representations. ICLR, 2025.
[26] Kolmogorov AN. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. Transl Am Math Soc 1963;2(28).
[27] Braun J, Griebel M. On a constructive proof of Kolmogorov's superposition theorem. Constr Approx 2009;30:653–75.

[28] Xing S. CombOpNet: a neural-network accelerator for SINDy. J Vib Test Syst Dyn 2025;9(01):1–20.
[29] Xing S, Han Q, Charalampidis E, Lai YC. Data-driven discovery of spatiotemporal dynamical systems with sparse interpretable neural networks. 2025.
[30] Li S, Kaiser E, Laima S, Li H, Brunton SL, Kutz JN. Discovering time-varying aerodynamics of a prototype bridge by sparse identification of nonlinear dynamical systems. Phys Rev E 2019;100(2):022220.
[31] Xiao B, Huang J, Zhongliang J. A fast system estimation algorithm for a discontinuous dynamical model with coefficients coupling. Mech Syst Signal Process 2025;229.
[32] Ma H, Lu X, Zhang L. Extracting parametric dynamics from time-series data. Nonlinear Dynam 2023;111(16):15177–99.
[33] Luo W, Billings SA. Adaptive model selection and estimation for nonlinear systems using a sliding data window. Signal Process 1995;46(2):179–202.
[34] Li Y, Wei HL, Billings SA, Sarrigiannis PG. Identification of nonlinear time-varying systems using an online sliding-window and common model structure selection (CMSS) approach with applications to EEG. Int J Syst Sci 2016;47(11):2671–81.
[35] Liu L, Li F, Liu W, Xia H. Sliding window iterative identification for nonlinear closed-loop systems based on the maximum likelihood principle. Internat J Robust Nonlinear Control 2025;35(3):1100–16.
[36] Ha D, Dai AM, Le QV. HyperNetworks. In: International conference on learning representations. 2017.
[37] Voina D, Brunton SL, Kutz JN. Deep generative modeling for identification of noisy, Non-Stationary dynamical systems. 2024.
[38] Zhang M, Kim S, Lu PY, Soljačić M. Deep learning and symbolic regression for discovering parametric equations. IEEE Trans Neural Netw Learn Syst 2023.
[39] Ranstam J, Cook JA. LASSO regression. J Br Surg 2018;105(10):1348.
[40] McDonald GC. Ridge regression. Wiley Interdiscip Rev: Comput Stat 2009;1(1):93–100.
[41] Cranmer M. Interpretable machine learning for science with PySR and SymbolicRegression. jl. 2023.
[42] Even S. Graph algorithms. 2nd ed.. Cambridge University Press; 2011.
[43] Selesnick IW, Burrus CS. Generalized digital butterworth filter design. IEEE Trans Signal Process 1998;46(6).
[44] Candan Ç, Inan H. A unified framework for derivation and implementation of Savitzky–Golay filters. Signal Process 2014;104.
[45] Zhu WQ, Luo M, Dong L. Semi-active control of wind excited building structures using MR/ER dampers. Probab Eng Mech 2004;19(3).
[46] Butikov EI. Parametric excitation of a linear oscillator. Eur J Phys 2004;25(4).
[47] Xiao B, Huang J, Tang D, Xu Z, Jing Z. Dynamic modeling of the electrical actuation system of the hypersonic aircraft considering the temperature effects. Aerosp Syst 2024;7(2).
[48] Kuramoto Y. International symposium on mathematical problems in theoretical physics. Lecture Notes in Phys 1975;30(420).
[49] Lohmann J, Ditlevsen PD. Risk of tipping the overturning circulation due to increasing rates of ice melt. Proc Natl Acad Sci 2021;118(9).
[50] Ditlevsen P, Ditlevsen S. Warning of a forthcoming collapse of the atlantic meridional overturning circulation. Nat Commun 2023;14(4254).
[51] Grindle R. Perils and pitfalls of symbolic regression (Master's thesis), The University of Vermont and State Agricultural College; 2021.
[52] Raghav SS, Kumar ST, Balaji R, Sanjay M, Shunmuga C. Interactive symbolic regression-a study on noise sensitivity and extrapolation accuracy. In: Proceedings of the genetic and evolutionary computation conference companion. 2024, p. 2076–82.
[53] Tonda A, Zhang H, Chen Q, Xue B, Zhang M, Lutton E. Comparing Data Transformation Techniques for system identification with standard symbolic regression. In: Proceedings of the genetic and evolutionary computation conference companion. 2025, p. 655–8.
[54] Sun C, Shen S, Tao W, Xue D, Zhou Z. Noise-resilient symbolic regression with dynamic gating reinforcement learning. In: Proceedings of the AAAI conference on artificial intelligence. Vol. 39, 2025.