PHYSICS-INFORMED STOCHASTIC CONFIGURATION MACHINE: A BACKPROPAGATION-FREE NEURAL NETWORK WITH FAST TRAINING INVOLVING NONLINEAR DIFFERENTIAL EQUATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

While Physics-Informed Neural Networks (PINNs) have emerged as a transformative paradigm for solving complex differential equations, their reliance on backpropagation-based gradient descent and automatic differentiation (AD) imposes significant computational bottlenecks and severe non-convex optimization challenges. To overcome these fundamental limitations, we propose the *Physics-Informed Stochastic Configuration Machine (PI-SCM)*, a novel backpropagation-free framework for both forward and inverse problems in differential equations. The core mathematical contribution lies in the analytical evaluation of local Jacobians for nonlinear differential operators, which facilitates a linearized representation of the physical loss and projects it into a unified, linearized algebraic subspace. This reformulation allows for the explicit determination of network weights via a sequence of linearized least-squares problems, effectively bypassing the iterative traps of traditional nonlinear optimizers. We develop a progressive algorithmic suite comprising localized construction (PI-SC-I), sliding-window correction (PI-SC-II), and global correction (PI-SC-III), and establish their universal approximation properties. Extensive experiments demonstrate that PI-SCM achieves robust parameter identification and substantially higher predictive accuracy than the PIELM baselines while accelerating training by orders of magnitude compared with standard PINNs. Our work provides a highly efficient and scalable foundation for next-generation, real-time Scientific Machine Learning applications.

1 INTRODUCTION

Nonlinear ordinary and partial differential equations (ODEs/PDEs) describe physical systems across science and engineering (Farlow, 1993). Alongside classical numerical solvers, scientific machine learning integrates physical knowledge into learned approximations (Baker et al., 2019). Physics-Informed Neural Networks (PINNs) (Raissi et al., 2019) embed governing equations and initial/boundary conditions into the training loss, providing a mesh-free approach to solution approximation and parameter identification, including settings with sparse observations.

However, training PINNs using gradient-based optimizers, such as Adam and L-BFGS (Wang et al., 2023), presents two main challenges. First, nonlinear physical residuals produce nonconvex objectives, and imbalanced gradients across loss terms can hinder convergence (Rathore et al., 2024). Second, evaluating derivatives through automatic differentiation (AD) (e.g., TensorFlow (Abadi et al., 2015), Pytorch (Paszke et al., 2017)) and repeatedly backpropagating parameter gradients can incur substantial computational and memory costs (Cuomo et al., 2022). Together, these challenges motivate methods that incorporate physical constraints without backpropagation.

Randomized networks offer an alternative by fixing randomly configured hidden parameters and solving for output weights through linear least squares. Stochastic Configuration Networks (SCNs) (Wang & Li, 2017) combine this analytical training mechanism with incremental node construction under a supervisory inequality, providing a universal approximation property under the stated conditions. This combination makes SCNs a natural starting point for efficient physics-informed learning.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

The key difficulty is that fixing hidden parameters does not make a nonlinear differential operator linear in the output weights. Physics-Informed Extreme Learning Machine (PIELM) (Dwivedi & Srinivasan, 2020) obtains output weights through linear least squares for linear PDEs but cannot retain this formulation for nonlinear PDEs. Local Extreme Learning Machine (LocELM) (Dong & Li, 2021) and the neural network basis method (NNBM) (Huang & Wu, 2025) extend randomized networks to nonlinear PDEs without backpropagation but still require iterative nonlinear optimization, such as the Gauss–Newton iterations. Frozen-PINNs (Datar et al., 2026) combine randomized network features with least-squares initialization and adaptive ODE solvers for time-dependent PDEs, but this formulation cannot apply to nonlinear static PDEs. Physics-Informed Stochastic Configuration Network (PISCN) (Xu et al., 2025) relies on gradient descent for its optimization. Despite their different training strategies, these randomized network methods share a common limitation: physics-informed constraints do not guide their hidden parameter selection or node construction. This gap motivates a unified framework that incorporates physical constraints into both hidden node construction and output weight estimation while retaining analytical training for nonlinear differential equations. Further background and related work are provided in Appendix A.

We address this challenge with the **Physics-Informed Stochastic Configuration Machine (PI-SCM)**. The physical residual is locally linearized, allowing the physics-informed and observational constraints to be assembled into a unified linear least-squares system. This formulation supports both unsupervised and semi-supervised forward learning and joint state and parameter estimation for inverse problems. Our contributions are threefold:

1. **Physics-informed node construction and analytical training.** We integrate physical, boundary, and data constraints into a unified residual that guides both node construction and output weight estimation. For each candidate node, the output weights are obtained by solving a single linear least-squares problem without AD, backpropagation, or an inner sequence of Gauss–Newton iterations.

2. **Progressive construction and correction.** PI-SC-I performs localized construction, PI-SC-II applies sliding-window correction, and PI-SC-III applies global correction, offering different trade-offs between accuracy and computational cost. Under the stated assumptions, the framework ensures monotonic residual reduction and satisfies the universal approximation property.

3. **Unified forward and inverse learning.** Jointly estimating physical parameters and network coefficients enables state reconstruction and parameter identification within the same formulation. Experiments on five benchmark systems show that PI-SCM can achieve competitive or better predictive accuracy than deep PINN baselines while reducing training time by approximately 36–107$\times$ across the evaluated forward and inverse tasks.

2 Preliminaries of PI-SCM and Problem Statement

2.1 Network Architecture of PI-SCM

PI-SCM incrementally constructs a single hidden layer network. Given an input coordinate $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^\top$, the k-th hidden node output is $h_k(\boldsymbol{x}) = g(\boldsymbol{w}_k^\top \boldsymbol{x} + b_k)$, where g is the activation function, $\boldsymbol{w}_k \in \mathbb{R}^n$ is the hidden weight vector, and $b_k \in \mathbb{R}$ is the hidden bias. With L hidden nodes and output weights $\boldsymbol{\beta}_k \in \mathbb{R}^m$, the state approximation is $\boldsymbol{u}_L(\boldsymbol{x}) = \sum_{k=1}^{L} \boldsymbol{\beta}_k h_k(\boldsymbol{x})$. The hidden parameters $\boldsymbol{w}_k$ and b_k are randomly assigned within a problem-dependent adaptive range $[-\tau, \tau]$ and remain fixed, while the output weights are determined analytically using pseudoinverse solvers. This architecture serves as the foundation for solving the problems defined below.

2.2 Problem Statement

Consider a differential system of order $q \geq 1$ on an open bounded domain $\Omega \subset \mathbb{R}^n$ with boundary $\partial\Omega$ and state $\boldsymbol{u} : \Omega \to \mathbb{R}^m$. For a multi-index $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_n) \in \mathbb{N}_0^n$, let $|\boldsymbol{\gamma}| = \sum_{j=1}^{n} \gamma_j$ and $D^{\boldsymbol{\gamma}}\boldsymbol{u}(\boldsymbol{x}) = \frac{\partial^{|\boldsymbol{\gamma}|}\boldsymbol{u}(\boldsymbol{x})}{\partial x_1^{\gamma_1} \partial x_2^{\gamma_2} \cdots \partial x_n^{\gamma_n}}$, with $D^{\mathbf{0}}\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{u}(\boldsymbol{x})$. For any nonnegative integer s, define $\mathbb{I}_s = \{\boldsymbol{\gamma} \in \mathbb{N}_0^n : |\boldsymbol{\gamma}| \leq s\}$ and the generalized differential state $\mathcal{D}^{(s)}\boldsymbol{u}(\boldsymbol{x}) = \{D^{\boldsymbol{\gamma}}\boldsymbol{u}(\boldsymbol{x})\}_{\boldsymbol{\gamma} \in \mathbb{I}_s}$. The differential system is

$$\mathbf{F}\Big(\boldsymbol{x}, \mathcal{D}^{(q)}\boldsymbol{u}(\boldsymbol{x}); \boldsymbol{\lambda}\Big) = \mathbf{0}, \quad \forall \boldsymbol{x} \in \Omega, \qquad \mathcal{B}\Big(\boldsymbol{x}, \mathcal{D}^{(q_b)}\boldsymbol{u}(\boldsymbol{x})\Big) = \mathbf{0}, \quad \forall \boldsymbol{x} \in \partial\Omega, \tag{1}$$

where $0 \leq q_b \leq q$, $\mathbf{F}$ is the differential operator, $\mathcal{B}$ represents the boundary or initial constraint operator, and $\boldsymbol{\lambda} \in \mathbb{R}^{n_p}$ is the physical parameter vector. Let $\Omega_p \subset \Omega$, $\Omega_b \subset \partial\Omega$, and $\Omega_d \subset \Omega$ denote the finite sets of interior collocation points, boundary/initial points, and observational data points.

Forward problem. Given $\boldsymbol{\lambda}$, the objective is to determine the optimal set of network parameters $\{\boldsymbol{w}_k, b_k, \boldsymbol{\beta}_k\}_{k=1}^{L}$ such that $\boldsymbol{u}_L$ satisfies the governing equation and boundary/initial constraints in Eq. (1) over Ω_p and Ω_b, respectively, while matching available observations over Ω_d.

Inverse problem. When $\boldsymbol{\lambda}$ is unknown, the objective is to jointly determine $\{\boldsymbol{w}_k, b_k, \boldsymbol{\beta}_k\}_{k=1}^{L}$ and identify $\boldsymbol{\lambda}$ such that $\boldsymbol{u}_L$ satisfies the governing equation and boundary/initial constraints while fitting the observations over Ω_d.

3 Physics-Informed Stochastic Configuration Machine

3.1 Forward Problem

We begin with **PI-SC-I**, the localized construction algorithm. Suppose the network has generated $L-1$ hidden nodes, and the current state approximation is given by $\boldsymbol{u}_{L-1}(\boldsymbol{x}) = \sum_{k=1}^{L-1} \boldsymbol{\beta}_k h_k(\boldsymbol{x})$, where $\boldsymbol{\beta}_k \in \mathbb{R}^m$ denotes the output weight vector, $h_k(\boldsymbol{x}) = g(\boldsymbol{w}_k^\top \boldsymbol{x} + b_k)$ represents the output of the k-th hidden node, $g(\cdot)$ is the activation function, and $\boldsymbol{w}_k, b_k$ are the randomly configured input weights and bias, respectively.

At step L, we define the physical residual as $\boldsymbol{e}_L^p := \mathbf{F}\big(\boldsymbol{x}, \mathcal{D}^{(q)}\boldsymbol{u}_L\big)$ and the corresponding initial/boundary residual as $\boldsymbol{e}_L^b := \mathcal{B}\big(\boldsymbol{x}, \mathcal{D}^{(q_b)}\boldsymbol{u}_L\big)$. When evaluating the addition of the L-th node $h_L(\boldsymbol{x})$, we have $\boldsymbol{u}_L(\boldsymbol{x}) = \boldsymbol{u}_{L-1}(\boldsymbol{x}) + \boldsymbol{\beta}_L h_L(\boldsymbol{x})$. By examining the residual difference between step L and step $L-1$, when the differential operator $\mathbf{F}$ is nonlinear, the physical residual increment is nonlinear with respect to the newly added output weight $\boldsymbol{\beta}_L$. This nonlinearity directly precludes the use of standard linear least squares methods for explicitly determining $\boldsymbol{\beta}_L$.

To decouple this nonlinearity and project it into a linear subspace, we apply the first-order Taylor expansion around the previously established state $\mathcal{D}^{(q)}\boldsymbol{u}_{L-1}$. By analytically evaluating the local Jacobians of the differential operator with respect to every derivative component up to order q, the physical residual is linearized as

$$\boldsymbol{e}_L^p = \mathbf{F}\Big(\boldsymbol{x}, \mathcal{D}^{(q)}\boldsymbol{u}_L\Big) \approx \mathbf{F}\Big(\boldsymbol{x}, \mathcal{D}^{(q)}\boldsymbol{u}_{L-1}\Big) + \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{L-1})\left(D^{\boldsymbol{\gamma}}\boldsymbol{u}_L - D^{\boldsymbol{\gamma}}\boldsymbol{u}_{L-1}\right), \tag{2}$$

For each $\boldsymbol{\gamma} \in \mathbb{I}_q$, the local Jacobian at the previous state is $\boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{L-1}) = \frac{\partial \mathbf{F}}{\partial (D^{\boldsymbol{\gamma}}\boldsymbol{u})}\big|_{\mathcal{D}^{(q)}\boldsymbol{u}=\mathcal{D}^{(q)}\boldsymbol{u}_{L-1}}$. In particular, the zero-order case $\boldsymbol{\gamma} = \mathbf{0}$ gives $\boldsymbol{J}_{D^0\boldsymbol{u}} = \boldsymbol{J}_{\boldsymbol{u}}$. Because $\boldsymbol{\beta}_L$ is independent of $\boldsymbol{x}$, the differential increment satisfies $D^{\boldsymbol{\gamma}}\boldsymbol{u}_L - D^{\boldsymbol{\gamma}}\boldsymbol{u}_{L-1} = \boldsymbol{\beta}_L D^{\boldsymbol{\gamma}} h_L(\boldsymbol{x})$ for $\boldsymbol{\gamma} \in \mathbb{I}_q$. Substituting this relation into (2) yields

$$\boldsymbol{e}_L^p \approx \boldsymbol{e}_{L-1}^p + \boldsymbol{A}_L(\boldsymbol{x})\boldsymbol{\beta}_L, \tag{3}$$

where

$$\boldsymbol{A}_L(\boldsymbol{x}) = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{L-1}) D^{\boldsymbol{\gamma}} h_L(\boldsymbol{x}). \tag{4}$$

The nonlinear local residual evaluation is thus reduced to a linear mapping with respect to $\boldsymbol{\beta}_L$. For the initial or boundary conditions, the operator can be formulated as $\mathcal{B}\big(\boldsymbol{x}, \mathcal{D}^{(q_b)}\boldsymbol{u}\big) = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}}\boldsymbol{u} - \boldsymbol{g}_{bc}(\boldsymbol{x})$, where $\boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x})$ denotes the coefficient matrix and $\boldsymbol{g}_{bc}(\boldsymbol{x})$ represents the prescribed boundary data. The boundary residual difference between steps L and $L-1$ is $\boldsymbol{e}_L^b - \boldsymbol{e}_{L-1}^b = \left(\sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}} h_L(\boldsymbol{x})\right)\boldsymbol{\beta}_L$. This expression is linear with respect to $\boldsymbol{\beta}_L$, which yields

$$\boldsymbol{e}_L^b = \boldsymbol{e}_{L-1}^b + \boldsymbol{C}_L(\boldsymbol{x})\boldsymbol{\beta}_L, \tag{5}$$

where $\boldsymbol{C}_L(\boldsymbol{x}) = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}} h_L(\boldsymbol{x})$.

For semi-supervised learning, let $\boldsymbol{e}_L^d := \boldsymbol{u}_{data}(\boldsymbol{x}) - \boldsymbol{u}_L(\boldsymbol{x})$ denote the observational residual, where $\boldsymbol{u}_{data}(\boldsymbol{x})$ denotes the observations. The residual satisfies exactly $\boldsymbol{e}_L^d = \boldsymbol{e}_{L-1}^d + \boldsymbol{D}_L(\boldsymbol{x})\boldsymbol{\beta}_L$, where $\boldsymbol{D}_L(\boldsymbol{x}) = -h_L(\boldsymbol{x})\boldsymbol{I}_m$.

Assuming that at the L-th step, we aim to simultaneously minimize $\boldsymbol{e}_L^p$, $\boldsymbol{e}_L^b$, and $\boldsymbol{e}_L^d$, the comprehensive optimization objective for the PI-SC-I algorithm is formulated as the following linear least squares loss function:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}_L) = &\frac{1}{N_p}\left\|\boldsymbol{A}_L(\boldsymbol{x})\boldsymbol{\beta}_L + \boldsymbol{e}_{L-1}^p\right\|_{\Omega_p}^2 + \frac{\omega_{bc}}{N_b}\left\|\boldsymbol{C}_L(\boldsymbol{x})\boldsymbol{\beta}_L + \boldsymbol{e}_{L-1}^b\right\|_{\Omega_b}^2 \\ &+ \frac{\omega_{data}}{N_d}\left\|\boldsymbol{D}_L(\boldsymbol{x})\boldsymbol{\beta}_L + \boldsymbol{e}_{L-1}^d\right\|_{\Omega_d}^2,\end{aligned} \tag{6}$$

where ω_{bc} and ω_{data} are the penalty parameters balancing the initial/boundary conditions and the observational data against the physical domain. The notations $\|\cdot\|_{\Omega_p}^2$, $\|\cdot\|_{\Omega_b}^2$, and $\|\cdot\|_{\Omega_d}^2$ represent the standard discrete squared L^2 norms evaluated over $\Omega_p = \{\boldsymbol{x}_i^p\}_{i=1}^{N_p}$, $\Omega_b = \{\boldsymbol{x}_i^b\}_{i=1}^{N_b}$, and $\Omega_d = \{\boldsymbol{x}_i^d\}_{i=1}^{N_d}$, respectively, where N_p, N_b, and N_d are the corresponding numbers of physical, boundary, and data points. At the L-th step, we define

$$\tilde{\boldsymbol{e}}_L = \begin{bmatrix} \frac{1}{\sqrt{N_p}} \underset{i=1,\dots,N_p}{\mathrm{col}} \left(\boldsymbol{e}_L^p(\boldsymbol{x}_i^p)\right) \\ \sqrt{\frac{\omega_{bc}}{N_b}} \underset{i=1,\dots,N_b}{\mathrm{col}} \left(\boldsymbol{e}_L^b(\boldsymbol{x}_i^b)\right) \\ \sqrt{\frac{\omega_{data}}{N_d}} \underset{i=1,\dots,N_d}{\mathrm{col}} \left(\boldsymbol{e}_L^d(\boldsymbol{x}_i^d)\right) \end{bmatrix}, \boldsymbol{M}_L = \begin{bmatrix} \frac{1}{\sqrt{N_p}} \underset{i=1,\dots,N_p}{\mathrm{col}} \left(\boldsymbol{A}_L(\boldsymbol{x}_i^p)\right) \\ \sqrt{\frac{\omega_{bc}}{N_b}} \underset{i=1,\dots,N_b}{\mathrm{col}} \left(\boldsymbol{C}_L(\boldsymbol{x}_i^b)\right) \\ \sqrt{\frac{\omega_{data}}{N_d}} \underset{i=1,\dots,N_d}{\mathrm{col}} \left(\boldsymbol{D}_L(\boldsymbol{x}_i^d)\right) \end{bmatrix}, \boldsymbol{v}_L = -\tilde{\boldsymbol{e}}_{L-1}, \tag{7}$$

where $\mathrm{col}(\cdot)$ denotes the vertical concatenation operator. With these definitions, the objective in (6) can be written as $\mathcal{L}(\boldsymbol{\beta}_L) = \|\boldsymbol{M}_L\boldsymbol{\beta}_L - \boldsymbol{v}_L\|^2$. Consequently, the minimum-norm solution of the linearized least squares problem is explicitly computed as $\widehat{\boldsymbol{\beta}}_L = \boldsymbol{M}_L^\dagger \boldsymbol{v}_L$. To control the Taylor truncation error of the true nonlinear residual, the adopted output-weight increment is defined as

$$\boldsymbol{\beta}_L^* = \alpha \boldsymbol{M}_L^\dagger \boldsymbol{v}_L, \tag{8}$$

where $\dagger$ denotes the Moore-Penrose pseudo-inverse operator, and $\alpha \in (0, 1]$ acts as a damping factor. Introducing this parameter guarantees the existence of a feasible parameter configuration that enforces strict monotonic descent of the true residual.

To rigorously guarantee the convergence of the algorithm, we introduce a supervisory node selection mechanism. Using $\tilde{\boldsymbol{e}}_L$ defined in (7), the comprehensive residual norm at the L-th step is given by $\mathcal{E}_L^2 = \|\tilde{\boldsymbol{e}}_L\|^2$. In the PI-SC-I algorithm, we establish a contraction evaluation metric $\xi_L = (r+\mu_L)\mathcal{E}_{L-1}^2 - \mathcal{E}_L^2$, where $r \in (0, 1)$ and $\mu_L = \frac{1-r}{L+1}$. During the generation of the L-th hidden node, we perform T_{max} independent trials by randomly sampling the hidden parameters $(\boldsymbol{w}_L, b_L)$ and calculating their corresponding optimal output weights $\boldsymbol{\beta}_L^*$. Full candidate configurations satisfying $\xi_L > 0$ are deposited into a feasible pool. The algorithm ultimately selects the optimal parameter set that maximizes ξ_L, thereby completing the PI-SC-I localized update and enforcing the steepest residual descent.

Remark 1. *Unlike Gauss–Newton methods, which repeatedly linearize a fixed nonlinear least-squares problem, PI-SCM integrates each linearization into node construction and requires only one linear least-squares solve for each candidate node. α and the supervisory node selection control the Taylor truncation error and enforce the residual descent required for the universal approximation property stated in Section 3.4.*

In the original SCN framework (Wang & Li, 2017), a global evaluation of the output weights was designed after the local node selection to overcome the slow convergence caused by localized construction. Here, we adopt an analogous global correction, defining the **PI-SC-III** algorithm. Let $\boldsymbol{H}_L(\boldsymbol{x}) = [h_1(\boldsymbol{x})\boldsymbol{I}_m, h_2(\boldsymbol{x})\boldsymbol{I}_m, \dots, h_L(\boldsymbol{x})\boldsymbol{I}_m]$ denote the global hidden layer output matrix, and the global output weight vector is denoted as $\boldsymbol{\beta}_{glob} = [\boldsymbol{\beta}_1^\top, \boldsymbol{\beta}_2^\top, \dots, \boldsymbol{\beta}_L^\top]^\top$. The globally updated state approximation is denoted as $\boldsymbol{u}_L(\boldsymbol{x}) = \boldsymbol{H}_L(\boldsymbol{x})\boldsymbol{\beta}_{glob}$.

Similarly, considering the nonlinearity of $\mathbf{F}$, we apply the first-order Taylor expansion around the locally constructed state $\mathcal{D}^{(q)}\boldsymbol{u}_{loc}$. Specifically, let $\boldsymbol{\beta}_{loc}^* = [\boldsymbol{\beta}_1^\top, \boldsymbol{\beta}_2^\top, \dots, (\boldsymbol{\beta}_L^*)^\top]^\top$ denote the configuration achieved by the PI-SC-I algorithm at the current step, yielding the local state approximation $\boldsymbol{u}_{loc}(\boldsymbol{x}) = \boldsymbol{H}_L(\boldsymbol{x})\boldsymbol{\beta}_{loc}^*$ and its corresponding physical residual $\boldsymbol{e}_{loc}^p$.

The local Jacobian matrices are then evaluated at the current localized configuration $\boldsymbol{u}_{loc}(\boldsymbol{x})$. The physical residual is thereby linearized via these Jacobian operators as

$$\boldsymbol{e}_L^p \approx \boldsymbol{e}_{loc}^p + \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{loc})\left(D^{\boldsymbol{\gamma}}\boldsymbol{H}_L(\boldsymbol{x})\boldsymbol{\beta}_{glob} - D^{\boldsymbol{\gamma}}\boldsymbol{u}_{loc}(\boldsymbol{x})\right) = \boldsymbol{A}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob} - \boldsymbol{b}(\boldsymbol{x}), \quad (9)$$

where $\boldsymbol{A}_{glob}(\boldsymbol{x}) = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{loc}) D^{\boldsymbol{\gamma}}\boldsymbol{H}_L(\boldsymbol{x})$, and $\boldsymbol{b}(\boldsymbol{x}) = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{loc}) D^{\boldsymbol{\gamma}}\boldsymbol{u}_{loc}(\boldsymbol{x}) - \boldsymbol{e}_{loc}^p$. Following the same rationale, the initial/boundary condition residual is formulated as

$$\boldsymbol{e}_L^b = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}}\boldsymbol{H}_L(\boldsymbol{x})\boldsymbol{\beta}_{glob} - \boldsymbol{g}_{bc}(\boldsymbol{x}) = \boldsymbol{C}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob} - \boldsymbol{g}_{bc}(\boldsymbol{x}), \quad (10)$$

where $\boldsymbol{C}_{glob}(\boldsymbol{x}) = \sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}}\boldsymbol{H}_L(\boldsymbol{x})$. The data residual is formulated as:

$$\boldsymbol{e}_L^d = \boldsymbol{u}_{data}(\boldsymbol{x}) - \boldsymbol{H}_L(\boldsymbol{x})\boldsymbol{\beta}_{glob} = \boldsymbol{u}_{data}(\boldsymbol{x}) - \boldsymbol{D}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob}, \quad (11)$$

where $\boldsymbol{D}_{glob}(\boldsymbol{x}) = \boldsymbol{H}_L(\boldsymbol{x})$.

Building upon these global algebraic formulations, the comprehensive objective function for the PI-SC-III algorithm is constructed as a linear least squares problem:

$$\begin{aligned}\mathcal{L}_{glob}(\boldsymbol{\beta}_{glob}) = \frac{1}{N_p}\left\|\boldsymbol{A}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob} - \boldsymbol{b}(\boldsymbol{x})\right\|_{\Omega_p}^2 + \frac{\omega_{bc}}{N_b}\left\|\boldsymbol{C}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob} - \boldsymbol{g}_{bc}(\boldsymbol{x})\right\|_{\Omega_b}^2 \\ + \frac{\omega_{data}}{N_d}\left\|\boldsymbol{D}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob} - \boldsymbol{u}_{data}(\boldsymbol{x})\right\|_{\Omega_d}^2.\end{aligned} \quad (12)$$

The three terms in (12) can be combined into a unified linear least squares system. The corresponding weighted global design matrix $\boldsymbol{M}_{glob}$ and target vector $\boldsymbol{v}_{glob}$ are given below:

$$\boldsymbol{M}_{glob} = \begin{bmatrix} \frac{1}{\sqrt{N_p}} \underset{i=1,\ldots,N_p}{\operatorname{col}} \left(\boldsymbol{A}_{glob}(\boldsymbol{x}_i^p)\right) \\ \sqrt{\frac{\omega_{bc}}{N_b}} \underset{i=1,\ldots,N_b}{\operatorname{col}} \left(\boldsymbol{C}_{glob}(\boldsymbol{x}_i^b)\right) \\ \sqrt{\frac{\omega_{data}}{N_d}} \underset{i=1,\ldots,N_d}{\operatorname{col}} \left(\boldsymbol{D}_{glob}(\boldsymbol{x}_i^d)\right) \end{bmatrix}, \boldsymbol{v}_{glob} = \begin{bmatrix} \frac{1}{\sqrt{N_p}} \underset{i=1,\ldots,N_p}{\operatorname{col}} \left(\boldsymbol{b}(\boldsymbol{x}_i^p)\right) \\ \sqrt{\frac{\omega_{bc}}{N_b}} \underset{i=1,\ldots,N_b}{\operatorname{col}} \left(\boldsymbol{g}_{bc}(\boldsymbol{x}_i^b)\right) \\ \sqrt{\frac{\omega_{data}}{N_d}} \underset{i=1,\ldots,N_d}{\operatorname{col}} \left(\boldsymbol{u}_{data}(\boldsymbol{x}_i^d)\right) \end{bmatrix}. \quad (13)$$

Accordingly, the objective in (12) can be written as $\mathcal{L}_{glob}(\boldsymbol{\beta}_{glob}) = \left\|\boldsymbol{M}_{glob}\boldsymbol{\beta}_{glob} - \boldsymbol{v}_{glob}\right\|^2$. The output weights in PI-SC-III are obtained from the global pseudo-inverse:

$$\tilde{\boldsymbol{\beta}}_{glob}^* = \boldsymbol{M}_{glob}^{\dagger}\boldsymbol{v}_{glob}. \quad (14)$$

To control the Taylor truncation error of the true nonlinear residual and guarantee monotonic convergence under PI-SC-III, we introduce a discrete line search mechanism. We define the global search direction as $\Delta\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}_{glob}^* - \boldsymbol{\beta}_{loc}^*$. The output weights are given by $\boldsymbol{\beta}(\alpha) = \boldsymbol{\beta}_{loc}^* + \alpha\Delta\boldsymbol{\beta}$, with $\alpha \in [0, 1]$.

Let $\mathcal{E}_L^2(\alpha)$ denote the comprehensive true residual norm evaluated with the candidate weights $\boldsymbol{\beta}(\alpha)$. The algorithm computes in parallel a discrete set of candidate step sizes, denoted as $\mathbb{A} \subset [0, 1]$, which includes the boundary limits 1.0 and 0.0. The best scaling factor α^* is selected as $\alpha^* = \arg\min_{\alpha\in\mathbb{A}} \mathcal{E}_L^2(\alpha)$. The configuration is then adopted as the final updated weights $\boldsymbol{\beta}^* = \boldsymbol{\beta}(\alpha^*)$.

Although PI-SC-III improves accuracy, its global correction becomes increasingly expensive as the network width L grows. PI-SC-II follows the same correction principle but restricts its scope through a sliding window (Wang & Li, 2017). When $L \leq W$, PI-SC-II coincides with the global correction. When $L > W$, it freezes the first $L - W$ nodes and corrects only the most recent W nodes. The active window fits the physical, boundary, and observational discrepancies left by the frozen block using the same linearization as PI-SC-III. PI-SC-II is then evaluated using the same discrete line search as PI-SC-III. Because the corresponding design matrix contains at most mW columns, the correction cost is controlled by W rather than the total network width L. The detailed derivation of PI-SC-II for forward problems is provided in Appendix B.1.

It is worth noting that utilizing either PI-SC-II or PI-SC-III after each node selection (PI-SC-I) is not necessary. To reduce computational overhead, a correction interval L_g is introduced such that either update is performed every L_g nodes.

3.2 Inverse Problem

For inverse problems, the localized construction in PI-SC-I jointly estimates the output weight of the newly added node and the physical parameter increment $\Delta\boldsymbol{\lambda}_L = \boldsymbol{\lambda}_L - \boldsymbol{\lambda}_{L-1}$. Let $\boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{L-1}, \boldsymbol{\lambda}_{L-1}) = \left.\frac{\partial \mathbf{F}}{\partial \boldsymbol{\lambda}}\right|_{\mathcal{D}^{(q)}\boldsymbol{u}=\mathcal{D}^{(q)}\boldsymbol{u}_{L-1}, \boldsymbol{\lambda}=\boldsymbol{\lambda}_{L-1}}$. Together with the state linearization represented by $\boldsymbol{A}_L$ in Eq. (4), the joint linearized physical equation is

$$[\boldsymbol{A}_L(\boldsymbol{x}) \quad \boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{L-1}, \boldsymbol{\lambda}_{L-1})] \begin{bmatrix} \boldsymbol{\beta}_L \\ \Delta\boldsymbol{\lambda}_L \end{bmatrix} \approx -\boldsymbol{e}^p_{L-1}. \tag{15}$$

Here, $\boldsymbol{\theta}_L = [\boldsymbol{\beta}_L^\top, \Delta\boldsymbol{\lambda}_L^\top]^\top$. Because the boundary and observational constraints do not depend directly on $\boldsymbol{\lambda}$, they remain linear in $\boldsymbol{\theta}_L$. Combining these constraints with the linearized physical equation yields a linear least-squares objective for estimating the state and physical parameters.

The state and parameter candidate is damped and filtered by the established supervisory node selection mechanism used in the forward problem. PI-SC-II and PI-SC-III use the same method to jointly correct the output weights and physical parameters. The complete derivations for inverse problems are provided in Appendix B.2. The detailed training procedures are presented in Appendix B.3.

3.3 Hidden Parameter Scale Selection

The candidate range $\Upsilon = \{\tau_1, \tau_2, \ldots, \tau_{end}\}$ controls expressivity and numerical stability through $\mathbf{w}_L \sim U(-\tau, \tau)$ and $\mathrm{b}_L \sim U(-\tau, \tau)$. For $h_L(\boldsymbol{x}) = g(\boldsymbol{w}_L^\top \boldsymbol{x} + b_L)$ and $\boldsymbol{\gamma} \in \mathbb{I}_q$, the chain rule gives $D^{\boldsymbol{\gamma}} h_L(\boldsymbol{x}) = g^{(|\boldsymbol{\gamma}|)}(\boldsymbol{w}_L^\top \boldsymbol{x} + b_L) \prod_{j=1}^n w_{L,j}^{\gamma_j}$, where $w_{L,j}$ is the j-th hidden weight and $g^{(|\boldsymbol{\gamma}|)}$ is the corresponding activation derivative. Substitution into (4) yields

$$\boldsymbol{A}_L(\boldsymbol{x}) = \sum_{\boldsymbol{\gamma} \in \mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{L-1}) g^{(|\boldsymbol{\gamma}|)}(\boldsymbol{w}_L^\top \boldsymbol{x} + b_L) \prod_{j=1}^n w_{L,j}^{\gamma_j}. \tag{16}$$

The weight factor $\prod_{j=1}^n w_{L,j}^{\gamma_j}$ scales as $\mathcal{O}(\tau^{|\boldsymbol{\gamma}|})$. Large τ can therefore amplify higher-order derivative contributions and worsening the conditioning of $\boldsymbol{M}_L$ in (7). Conversely, small τ limits expressivity and can slow convergence. Therefore, we use $\Upsilon = a \times \{1, 2, 4, 8, 16, 32, 64\}$, with $a > 0$ chosen to balance state and derivative contributions. PI-SCM searches these ranges in ascending order, expanding the range only when the current one fails to yield an accepted residual decrease.

3.4 Universal Approximation Property

We now state the assumptions and universal approximation property for PI-SCM in forward problems. Detailed arguments including proofs are provided in Appendix C.

Assumption 1. *Let $\boldsymbol{z} = \operatorname{col}_{\boldsymbol{\gamma} \in \mathbb{I}_q} (D^{\boldsymbol{\gamma}}\boldsymbol{u}) = \operatorname{col}\big(\mathcal{D}^{(q)}\boldsymbol{u}\big)$. The activation function satisfies $g \in C^q(\mathbb{R})$. The generalized state $\boldsymbol{z}(\boldsymbol{x})$ is uniformly bounded and all $\boldsymbol{x} \in \Omega_p \cup \Omega_b \cup \Omega_d$. Define $\boldsymbol{z}_{\mathrm{col}}(\boldsymbol{u}) = \operatorname{col}_{\boldsymbol{x} \in \Omega_p \cup \Omega_b \cup \Omega_d} (\boldsymbol{z}(\boldsymbol{x}))$. Let $\mathbb{K}$ be the closure of the reachable values of $\boldsymbol{z}_{\mathrm{col}}(\boldsymbol{u})$. The differential operator $\mathbf{F}$ is assumed to be twice continuously Fréchet differentiable (C^2) with respect to $\boldsymbol{z}$ on an open neighborhood of the local states represented in $\mathbb{K}$.*

Assumption 2. *Let Γ be the admissible hidden-node family, and let $\boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h)$ denote the weighted design matrix defined by Eq. (7) at state $\boldsymbol{z}_{\mathrm{col}}$ for candidate node h, so that $\boldsymbol{M}_L(h) = \boldsymbol{M}\big(\boldsymbol{z}_{\mathrm{col}}(\boldsymbol{u}_{L-1}), h\big)$. There exist a finite integer J and fixed reference nodes $h_1, \ldots, h_J \in \Gamma$ such that, for every $\boldsymbol{z}_{\mathrm{col}} \in \mathbb{K}$, the block matrix*

$$\boldsymbol{M}_{\mathrm{ref}}(\boldsymbol{z}_{\mathrm{col}}) = [\boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h_1), \ldots, \boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h_J)] \tag{17}$$

has full row rank. Moreover, for each fixed j, the rank of $\boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h_j)$ is constant over $\mathbb{K}$.

Theorem 1. *Suppose that Assumptions 1 and 2 hold. There exist constants $\kappa \in (0, 1]$ and $\bar{\alpha} \in (0, 1]$ such that, for any $r \in (1 - \kappa\bar{\alpha}, 1)$ and any nonnegative sequence $\{\mu_L\}$ satisfying $\lim_{L \to +\infty} \mu_L = 0$ and $\mu_L \leq 1 - r$, there exists a candidate hidden node $h_L \in \Gamma$ at each step L such that, for any*

$\alpha \in ((1-r)/\kappa, \bar{\alpha}]$, *the output weight configuration* $\boldsymbol{\beta}_L(\alpha) = \alpha \boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L$ *guarantees that the following inequality holds:*

$$\mathcal{E}_{L-1}^2 - \mathcal{E}_L^2 \geq \delta_L. \tag{18}$$

where $\delta_L = (1 - r - \mu_L)\mathcal{E}_{L-1}^2$. *By accepting the node and the output weights that satisfy (18), the sequence of* $\{\mathcal{E}_L^2\}$ *monotonically decreases, satisfying* $\lim_{L\to+\infty} \mathcal{E}_L^2 = 0$.

Remark 2. *Explicitly computing* κ *and* $\bar{\alpha}$ *is unnecessary. Instead,* α *and* r *are adaptively selected using the supervisory node selection mechanism. Both* α *and* r *are initialized at the first values of their respective ordered lists,* $\mathbb{A}_1$ *and* $\mathbb{S}_r$*, and each advanced to its next value only when candidate nodes fail to satisfy the node selection.*

Let $\mathcal{E}_L^2(\boldsymbol{\beta}) = \|\tilde{\boldsymbol{e}}_L(\boldsymbol{\beta})\|^2$ and $\mathcal{E}_{L,loc}^2 = \mathcal{E}_L^2(\boldsymbol{\beta}_{loc}^*)$. $\tilde{\boldsymbol{\beta}}_{win}^*$ is defined in (24).

Theorem 2. *Suppose the conditions of Theorem 1 hold. We define the global search direction as* $\Delta\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}_{glob}^* - \boldsymbol{\beta}_{loc}^*$. *Then, there exists a constant* $\bar{\alpha} \in [0, 1]$ *such that for any* $\alpha \in [0, \bar{\alpha}]$*, the output weight configuration* $\boldsymbol{\beta} = \boldsymbol{\beta}_{loc}^* + \alpha\Delta\boldsymbol{\beta}$ *guarantees that the following inequality holds:*

$$\mathcal{E}_L^2(\boldsymbol{\beta}) \leq \mathcal{E}_{L,loc}^2. \tag{19}$$

By accepting the configuration that satisfies this inequality, the sequence of $\{\mathcal{E}_L^2\}$ *monotonically decreases, satisfying* $\lim_{L\to+\infty} \mathcal{E}_L^2 = 0$.

Theorem 3. *Suppose the conditions of Theorem 1 hold. We define the sliding-window search direction as* $\Delta\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}_{win}^* - \boldsymbol{\beta}_{loc}^*$. *Then, there exists constant* $\bar{\alpha} \in [0, 1]$ *such that for any* $\alpha \in [0, \bar{\alpha}]$*, the output weight configuration* $\boldsymbol{\beta} = \boldsymbol{\beta}_{loc}^* + \alpha\Delta\boldsymbol{\beta}$ *guarantees that the following inequality holds:*

$$\mathcal{E}_L^2(\boldsymbol{\beta}) \leq \mathcal{E}_{L,loc}^2. \tag{20}$$

By accepting the configuration that satisfies this inequality, the sequence of $\{\mathcal{E}_L^2\}$ *monotonically decreases, satisfying* $\lim_{L\to+\infty} \mathcal{E}_L^2 = 0$.

4 Experimental Results

This section evaluates PI-SCM on forward problems involving the Van der Pol oscillator, the two-dimensional Helmholtz equation, the Allen–Cahn equation, the five-dimensional reaction–diffusion equation, and the two-dimensional Bratu equation, together with inverse problems involving the Van der Pol, Helmholtz, and Bratu equations.

For the forward and inverse experiments below, all tabulated values are reported as mean $\pm$ standard deviation over 100 independent trials, and state RMSE is evaluated on 10,000 randomly sampled test coordinates. The governing equations and experimental setup of the benchmark systems are provided in Appendix D.1–D.5. The activation-function study and additional ablation studies are reported in Appendix D.6 and Appendix D.7, respectively.

4.1 Forward Problems: Results and Analysis

Table 1 reports training time and RMSE under the unsupervised and semi-supervised paradigms.

Table 1 shows that PI-SCM provides a strong combination of predictive accuracy and computational efficiency. PI-SC-I has the lowest construction cost, PI-SC-II balances accuracy and efficiency, and PI-SC-III generally achieves the highest accuracy. With $L_g = 1$, PI-SC-III trains 19 to 51 times faster than the corresponding deep PINNs and reduces RMSE by factors of 2.2 to 2.8 on Van der Pol, over four orders of magnitude on Helmholtz, 49 to 85 on Allen–Cahn, and 16 to 17 on Bratu. On reaction–diffusion, it achieves comparable RMSE while training 22 to 23 times faster.

The correction intervals provide further acceleration. We use $L_g = 50$ for Van der Pol, $L_g = 10$ for Helmholtz, Allen–Cahn, and reaction–diffusion, and $L_g = 30$ for Bratu. Relative to $L_g = 1$, these settings reduce the PI-SC-III training time by approximately 46% to 66% and increase the speedups over the deep PINNs to approximately 36 to 107 times, while broadly preserving predictive accuracy.

Compared with the PIELM baselines, PI-SC-III at these intervals lowers RMSE on Van der Pol, Helmholtz, Allen–Cahn, and reaction–diffusion by factors ranging from approximately 1.7 to 6.6 $\times$

Table 1: Forward results across the five benchmark systems.

System	Algorithm	Paradigm	Time (s)	RMSE
Van der Pol oscillator	PINN (Raissi et al., 2019) (1×300)	Unsupervised	21.3 ± 0.57	$3.69e{-1} \pm 9.18e{-2}$
		Semi-supervised	27.5 ± 0.89	$8.05e{-2} \pm 4.02e{-3}$
	PINN (4×64)	Unsupervised	40.0 ± 1.02	$5.49e{-3} \pm 3.03e{-3}$
		Semi-supervised	50.8 ± 1.08	$1.34e{-3} \pm 5.22e{-4}$
	PISCN (Xu et al., 2025)	Unsupervised	–	–
		Semi-supervised	13.8 ± 0.37	$7.58e{-2} \pm 2.59e{-2}$
	PIELM(NNBM) (Huang & Wu, 2025)	Unsupervised	0.50 ± 0.10	$1.49e{-1} \pm 4.63e{-1}$
		Semi-supervised	0.43 ± 0.03	$1.84e{-3} \pm 4.24e{-3}$
	PI-SC-I	Unsupervised	0.53 ± 0.02	$3.88e{-1} \pm 1.41e{-1}$
		Semi-supervised	0.60 ± 0.01	$1.23e{-1} \pm 3.82e{-2}$
	PI-SC-II ($L_g = 1$)	Unsupervised	1.37 ± 0.04	$1.30e{-2} \pm 5.73e{-3}$
		Semi-supervised	1.57 ± 0.05	$5.27e{-3} \pm 2.63e{-3}$
	PI-SC-II ($L_g = 50$)	Unsupervised	0.73 ± 0.03	$8.45e{-2} \pm 8.28e{-2}$
		Semi-supervised	0.86 ± 0.03	$7.72e{-3} \pm 2.10e{-3}$
	PI-SC-III ($L_g = 1$)	Unsupervised	1.44 ± 0.04	$1.93e{-3} \pm 9.86e{-4}$
		Semi-supervised	1.64 ± 0.04	$6.04e{-4} \pm 1.76e{-4}$
	PI-SC-III ($L_g = 50$)	Unsupervised	0.75 ± 0.04	$8.94e{-2} \pm 9.73e{-2}$
		Semi-supervised	0.83 ± 0.02	$4.62e{-4} \pm 1.60e{-4}$
Helmholtz equation	PINN (1×400)	Unsupervised	40.5 ± 1.23	$2.62e{-2} \pm 1.97e{-2}$
		Semi-supervised	53.2 ± 1.51	$1.13e{-2} \pm 4.50e{-3}$
	PINN (4×64)	Unsupervised	75.6 ± 1.11	$3.28e{-2} \pm 1.43e{-2}$
		Semi-supervised	105 ± 1.38	$1.44e{-2} \pm 5.06e{-3}$
	PIELM (Dwivedi & Srinivasan, 2020)	Unsupervised	0.02 ± 0.00	$9.09e{-3} \pm 3.83e{-3}$
		Semi-supervised	0.02 ± 0.00	$6.61e{-3} \pm 3.84e{-3}$
	PI-SC-I	Unsupervised	0.57 ± 0.01	$1.91e{-1} \pm 9.82e{-2}$
		Semi-supervised	0.66 ± 0.01	$1.57e{-1} \pm 7.25e{-2}$
	PI-SC-II ($L_g = 1$)	Unsupervised	1.99 ± 0.05	$5.19e{-3} \pm 2.05e{-3}$
		Semi-supervised	2.20 ± 0.06	$4.71e{-3} \pm 1.78e{-3}$
	PI-SC-II ($L_g = 10$)	Unsupervised	1.17 ± 0.06	$6.40e{-3} \pm 2.21e{-3}$
		Semi-supervised	1.32 ± 0.07	$6.65e{-3} \pm 2.65e{-3}$
	PI-SC-III ($L_g = 1$)	Unsupervised	2.31 ± 0.04	$1.02e{-6} \pm 4.79e{-7}$
		Semi-supervised	2.54 ± 0.05	$1.04e{-6} \pm 8.75e{-7}$
	PI-SC-III ($L_g = 10$)	Unsupervised	1.21 ± 0.07	$1.38e{-6} \pm 1.93e{-6}$
		Semi-supervised	1.35 ± 0.06	$1.07e{-6} \pm 4.80e{-7}$
Allen–Cahn equation	VS-PINN (Ko & Park, 2025) (1×500)	Unsupervised	31.8 ± 0.79	$5.40e{-1} \pm 2.63e{-2}$
		Semi-supervised	36.2 ± 0.94	$4.65e{-1} \pm 4.67e{-2}$
	VS-PINN (4×64)	Unsupervised	65.0 ± 1.06	$5.23e{-1} \pm 4.14e{-2}$
		Semi-supervised	71.6 ± 1.35	$7.75e{-2} \pm 1.06e{-1}$
	PIELM(NNBM)	Unsupervised	1.18 ± 0.18	$3.48e{-2} \pm 1.15e{-1}$
		Semi-supervised	1.06 ± 0.30	$3.12e{-2} \pm 7.43e{-2}$
	PI-SC-I	Unsupervised	1.59 ± 0.04	$5.28e{-1} \pm 3.50e{-3}$
		Semi-supervised	1.73 ± 0.03	$3.16e{-1} \pm 2.47e{-2}$
	PI-SC-II ($L_g = 1$)	Unsupervised	3.13 ± 0.07	$1.69e{-1} \pm 4.75e{-2}$
		Semi-supervised	3.41 ± 0.06	$2.19e{-2} \pm 9.77e{-3}$
	PI-SC-II ($L_g = 10$)	Unsupervised	1.83 ± 0.08	$6.45e{-2} \pm 5.95e{-2}$
		Semi-supervised	1.96 ± 0.04	$9.94e{-3} \pm 4.39e{-3}$
	PI-SC-III ($L_g = 1$)	Unsupervised	3.49 ± 0.14	$6.16e{-3} \pm 2.07e{-3}$
		Semi-supervised	3.70 ± 0.07	$1.57e{-3} \pm 2.88e{-4}$
	PI-SC-III ($L_g = 10$)	Unsupervised	1.82 ± 0.08	$1.63e{-2} \pm 5.47e{-2}$
		Semi-supervised	2.01 ± 0.08	$1.57e{-3} \pm 2.66e{-4}$
Reaction–diffusion equation	PINN (1×1000)	Unsupervised	382 ± 4.30	$1.62e{-2} \pm 2.41e{-3}$
		Semi-supervised	395 ± 4.05	$1.31e{-2} \pm 1.62e{-3}$
	PINN (4×64)	Unsupervised	379 ± 2.71	$5.12e{-3} \pm 3.38e{-3}$
		Semi-supervised	398 ± 6.83	$5.19e{-3} \pm 3.32e{-3}$
	PIELM(NNBM)	Unsupervised	1.61 ± 0.03	$1.29e{-2} \pm 1.35e{-3}$
		Semi-supervised	1.62 ± 0.02	$1.29e{-2} \pm 1.24e{-3}$
	PI-SC-I	Unsupervised	5.17 ± 0.16	$1.01e{-1} \pm 6.46e{-3}$
		Semi-supervised	5.43 ± 0.06	$9.35e{-2} \pm 5.80e{-3}$
	PI-SC-II ($L_g = 1$)	Unsupervised	9.31 ± 0.09	$2.05e{-2} \pm 1.39e{-3}$
		Semi-supervised	9.79 ± 0.13	$1.97e{-2} \pm 1.63e{-3}$
	PI-SC-II ($L_g = 10$)	Unsupervised	5.47 ± 0.09	$2.30e{-2} \pm 1.74e{-3}$
		Semi-supervised	5.76 ± 0.09	$2.23e{-2} \pm 1.53e{-3}$
	PI-SC-III ($L_g = 1$)	Unsupervised	17.1 ± 1.38	$6.27e{-3} \pm 4.47e{-4}$
		Semi-supervised	17.4 ± 0.10	$6.15e{-3} \pm 4.42e{-4}$
	PI-SC-III ($L_g = 10$)	Unsupervised	5.84 ± 0.10	$6.48e{-3} \pm 4.80e{-4}$
		Semi-supervised	6.28 ± 0.13	$6.39e{-3} \pm 4.63e{-4}$
Bratu equation	PINN (1×300)	Unsupervised	86.5 ± 1.24	$3.64e{-4} \pm 1.42e{-4}$
		Semi-supervised	95.8 ± 1.31	$4.10e{-4} \pm 1.23e{-4}$
	PINN (4×64)	Unsupervised	133 ± 1.11	$1.42e{-4} \pm 1.11e{-4}$
		Semi-supervised	148 ± 1.32	$1.15e{-4} \pm 8.73e{-5}$
	PIELM(NNBM)	Unsupervised	0.13 ± 0.00	$9.84e{-6} \pm 8.88e{-8}$
		Semi-supervised	0.20 ± 0.00	$7.17e{-6} \pm 8.76e{-8}$
	PI-SC-I	Unsupervised	1.25 ± 0.06	$1.83e{-2} \pm 4.13e{-3}$
		Semi-supervised	1.31 ± 0.04	$1.54e{-2} \pm 3.47e{-3}$
	PI-SC-II ($L_g = 1$)	Unsupervised	2.56 ± 0.09	$8.61e{-6} \pm 2.11e{-6}$
		Semi-supervised	2.73 ± 0.07	$7.11e{-6} \pm 1.65e{-6}$
	PI-SC-II ($L_g = 30$)	Unsupervised	1.27 ± 0.07	$1.03e{-5} \pm 4.33e{-7}$
		Semi-supervised	1.39 ± 0.07	$8.03e{-6} \pm 5.06e{-7}$
	PI-SC-III ($L_g = 1$)	Unsupervised	2.72 ± 0.09	$8.25e{-6} \pm 2.36e{-6}$
		Semi-supervised	2.93 ± 0.05	$7.01e{-6} \pm 1.85e{-6}$
	PI-SC-III ($L_g = 30$)	Unsupervised	1.27 ± 0.05	$9.16e{-6} \pm 1.56e{-7}$
		Semi-supervised	1.39 ± 0.06	$6.59e{-6} \pm 1.79e{-7}$

10^3, and reduces the unsupervised and semi-supervised RMSEs on Bratu from 9.84×10^{-6} to 9.16×10^{-6} and from 7.17×10^{-6} to 6.59×10^{-6}, respectively. The particularly high Helmholtz accuracy follows from its linear operator, for which the PI-SCM linearization is exact. PIELM

Table 2: Inverse results across the Van der Pol, two-dimensional Helmholtz, and Bratu systems, where the identified parameters are μ, k^2, and λ, respectively.

System	Algorithm	$\lvert\lambda_{pred} - \lambda_{true}\rvert$	Time (s)	State RMSE
Van der Pol oscillator	PINN (1×300)	$2.88e{-1} \pm 1.09e{-1}$	22.0 ± 0.51	$5.85e{-2} \pm 1.35e{-2}$
	PINN (4×64)	$2.35e{-4} \pm 2.01e{-4}$	51.8 ± 0.94	$9.51e{-4} \pm 3.42e{-4}$
	PI-SC-I	$4.01e{-1} \pm 4.69e{-2}$	0.71 ± 0.02	$1.08e{-1} \pm 2.05e{-2}$
	PI-SC-II ($L_g = 1$)	$1.91e{-3} \pm 1.59e{-3}$	1.64 ± 0.03	$4.80e{-3} \pm 2.10e{-3}$
	PI-SC-II ($L_g = 50$)	$1.52e{-3} \pm 7.45e{-4}$	1.10 ± 0.02	$1.18e{-3} \pm 6.06e{-4}$
	PI-SC-III ($L_g = 1$)	$5.08e{-4} \pm 2.75e{-4}$	1.95 ± 0.04	$5.96e{-4} \pm 2.00e{-4}$
	PI-SC-III ($L_g = 50$)	$8.20e{-4} \pm 4.67e{-4}$	1.14 ± 0.07	$5.69e{-4} \pm 1.64e{-4}$
Helmholtz equation	PINN (1×400)	$1.09e{+1} \pm 2.87e{+0}$	50.6 ± 1.19	$2.82e{-1} \pm 1.27e{-1}$
	PINN (4×64)	$6.72e{-1} \pm 6.27e{-1}$	101 ± 2.13	$2.36e{-2} \pm 1.22e{-2}$
	PI-SC-I	$3.72e{+1} \pm 1.93e{+1}$	1.49 ± 0.03	$2.17e{-1} \pm 9.90e{-2}$
	PI-SC-II ($L_g = 1$)	$5.76e{-1} \pm 5.62e{-1}$	3.14 ± 0.07	$4.56e{-3} \pm 1.93e{-3}$
	PI-SC-II ($L_g = 10$)	$1.98e{+0} \pm 2.44e{+0}$	1.92 ± 0.06	$7.85e{-3} \pm 6.54e{-3}$
	PI-SC-III ($L_g = 1$)	$8.42e{-4} \pm 1.60e{-3}$	3.51 ± 0.10	$6.70e{-5} \pm 1.15e{-4}$
	PI-SC-III ($L_g = 10$)	$5.07e{-4} \pm 7.64e{-4}$	2.02 ± 0.08	$3.84e{-5} \pm 4.88e{-5}$
Bratu equation	PINN (1×300)	$7.30e{-2} \pm 3.50e{-2}$	97.1 ± 1.08	$2.64e{-3} \pm 1.29e{-3}$
	PINN (4×64)	$2.05e{-3} \pm 7.78e{-3}$	145 ± 3.73	$1.42e{-4} \pm 2.35e{-4}$
	PI-SC-I	$7.01e{-1} \pm 1.60e{-2}$	1.58 ± 0.05	$1.90e{-2} \pm 5.27e{-4}$
	PI-SC-II ($L_g = 1$)	$1.29e{-4} \pm 2.87e{-6}$	3.13 ± 0.04	$6.99e{-6} \pm 1.24e{-7}$
	PI-SC-II ($L_g = 30$)	$1.27e{-4} \pm 4.10e{-6}$	1.69 ± 0.04	$7.86e{-6} \pm 6.09e{-7}$
	PI-SC-III ($L_g = 1$)	$1.31e{-4} \pm 1.65e{-6}$	3.57 ± 0.07	$6.17e{-6} \pm 1.87e{-5}$
	PI-SC-III ($L_g = 30$)	$1.31e{-4} \pm 1.85e{-6}$	1.66 ± 0.04	$6.50e{-6} \pm 1.58e{-7}$

provides the lowest training cost, whereas PI-SCM achieves substantially higher accuracy across all systems while retaining practical training efficiency. These results demonstrate the benefit of using physics-informed constraints to guide both node construction and output weight estimation.

4.2 Inverse Problems: Results and Analysis

Table 2 reports parameter identification error, training time, and state RMSE.

The inverse results in Table 2 show the same trend as in the forward experiments. PI-SC-I trains fastest, PI-SC-II balances speed and accuracy, and PI-SC-III generally achieves the highest accuracy.

Across all three inverse benchmarks, PI-SC-III achieves lower state RMSE than the deep PINN, while providing parameter identification accuracy comparable to or better than that of the deep PINN at substantially lower training cost. These results demonstrate that PI-SCM retains high predictive accuracy and computational efficiency in inverse problems.

5 Conclusion

This paper introduced PI-SCM, a backpropagation-free framework for forward and inverse problems involving nonlinear differential equations. Physical, boundary, and observational constraints guide both hidden node construction and output weight estimation, embedding physics-informed learning throughout the constructive training process. By coupling residual linearization with progressive node construction, PI-SCM removes the need for an inner nonlinear optimization loop, therefore provides the computational foundation for its efficiency.

Within the PI-SCM framework, PI-SC-I provided the fastest training but was limited in accuracy, PI-SC-III achieved the highest accuracy at greater computational cost, and PI-SC-II balanced accuracy and efficiency. In particular, PI-SC-III achieved errors comparable to or lower than those of gradient-based PINNs while reducing training time by orders of magnitude, and attained substantially higher accuracy compare to the PIELM baselines. These results highlight the potential of PI-SCM to serve as a powerful and scalable foundation for real-time scientific machine learning applications.

Future work should examine scalability to higher dimensional geometries, more severe multiscale stiffness, noisy parameter identification settings, and adaptive selection of the candidate range and update scope. The modular design of PI-SCM provides an extensible algorithmic foundation for incorporating complementary advances from the randomized network, PINN, and SCN communities. Promising directions include adaptive sampling (Wu et al., 2023), domain decomposition (Dong & Li, 2021), the constrained expressions of boundary conditions (Schiassi et al., 2021), the enforcement of temporal causality (Datar et al., 2026), and deep or adaptive SCN constructions (Wang & Li, 2018; Felicetti & Wang, 2022a;b). These integrations could extend the current architecture while preserving analytical training and numerical stability.

Reproducibility statement

The PI-SCM source code and instructions for reproducing the experimental results are provided in the supplementary material and in the anonymous repository at `https://anonymous.4open.science/r/PI-SCM-B256`. The experimental settings are detailed in Appendix D.

Ethics statement

Neural networks and differential equation solvers can support both beneficial and harmful applications. PI-SCM combines physics-informed learning with analytical Jacobians and least-squares estimation, making the governing constraints and training procedures explicit. This transparency supports analysis of numerical robustness, failure modes, and reproducibility. The experiments use synthetic benchmarks without human participants or personal data. We expect the framework to benefit scientific computing, while applications with safety implications require validation under their intended operating conditions.

LLM Usage

Large language models (LLMs) were used solely to aid and polish manuscript writing, including language refinement, grammar checking, and improvements to clarity and readability.

Research ideas, methods, mathematical derivations and experiments were developed by the authors without LLM assistance. The authors reviewed all assisted text and take full responsibility for the manuscript's accuracy, originality, and integrity.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015.

Joel Andersson, Johan Åkesson, and Moritz Diehl. Dynamic optimization with casadi. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pp. 681–686. IEEE, 2012.

Nathan Baker, Frank Alexander, Timo Bremer, Aric Hagberg, Yannis Kevrekidis, Habib Najm, Manish Parashar, Abani Patra, James Sethian, Stefan Wild, et al. Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence. Technical report, USDOE Office of Science (SC), Washington, DC (United States), 2019.

Sarah Berkhahn and Matthias Ehrhardt. A physics-informed neural network to model covid-19 infection and hospitalization scenarios. *Advances in continuous and discrete models*, 2022(1):61, 2022.

Keqiong Chen, Jiaxi An, Yu Fang, and Tianrui Bu. Research on solid waste plastic bottle cognitive based on yolov5s and deep stochastic configuration network. In *2022 7th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pp. 275–280. IEEE, 2022.

Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of scientific computing*, 92(3):88, 2022.

Chinmay Datar, Taniya Kapoor, Abhishek Chandra, Qing Sun, Erik L Bolager, Iryna Burak, Anna Veselovska, Massimo Fornasier, and Felix Dietrich. Fast training of accurate physics-informed neural networks without gradient descent. In *International Conference on Learning Representations*, volume 2026, pp. 104770–104829, 2026.

Yu Diao, Jianchuan Yang, Ying Zhang, Dawei Zhang, and Yiming Du. Solving multi-material problems in solid mechanics using physics-informed neural networks based on domain decomposition technology. *Computer Methods in Applied Mechanics and Engineering*, 413:116120, 2023.

Suchuan Dong and Zongwei Li. Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 387:114129, 2021.

Vikas Dwivedi and Balaji Srinivasan. Physics informed extreme learning machine (pielm)–a rapid method for the numerical solution of partial differential equations. *Neurocomputing*, 391:96–118, 2020.

Stanley J Farlow. *Partial differential equations for scientists and engineers*. Courier Corporation, 1993.

Matthew J Felicetti and Dianhui Wang. Deep stochastic configuration networks with different random sampling strategies. *Information Sciences*, 607:819–830, 2022a.

Matthew J Felicetti and Dianhui Wang. Deep stochastic configuration networks with optimised model and hyper-parameters. *Information Sciences*, 600:431–441, 2022b.

Hafeez Y Hafeez, Chifu E Ndikilar, Sabo Isyaku, et al. Analytical study of the van der pol equation in the autonomous regime. *Progress*, 11:252, 2015.

Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1-3):489–501, 2006.

Jianguo Huang and Haohao Wu. The neural network basis method for nonlinear partial differential equations and its gauss–newton optimizer. *Communications in Nonlinear Science and Numerical Simulation*, 143:108608, 2025.

Ameya D Jagtap, Kenji Kawaguchi, and George Em Karniadakis. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404:109136, 2020.

Seungchan Ko and Sanghyeon Park. Vs-pinn: A fast and efficient training of physics-informed neural networks using variable-scaling methods for solving pdes with stiff behavior. *Journal of Computational Physics*, 529:113860, 2025.

Kang Li, Wei Wang, and Shaohan Lin. Soft measurement of ammonia nitrogen concentration based on ga-scn. In *2018 IEEE Symposium on Product Compliance Engineering-Asia (ISPCE-CN)*, pp. 1–4. IEEE, 2018.

Weitao Li, Hai Tao, Hua Li, Keqiong Chen, and Jianping Wang. Greengage grading using stochastic configuration networks and a semi-supervised feedback mechanism. *Information Sciences*, 488: 1–12, 2019.

Weitao Li, Qian Zhang, Dianhui Wang, Wei Sun, and Qiyue Li. Stochastic configuration networks for self-blast state recognition of glass insulators with adaptive depth and multi-scale representation. *Information Sciences*, 604:61–79, 2022.

Derong Liu, Tsu-Shuan Chang, and Yi Zhang. A constructive algorithm for feedforward neural networks with incremental training. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 49(12):1876–1879, 2002.

Wei Liu, Chao Ren, and Yan Xu. Pv generation forecasting with missing input data: A super-resolution perception approach. *IEEE Transactions on Sustainable Energy*, 12(2):1493–1496, 2020.

Jiaqi Pan, Feng Luan, Yunqi Gao, and Yangjie Wei. Fpga-based implementation of stochastic configuration network for robotic grasping recognition. *IEEE Access*, 8:139966–139973, 2020.

Y-H Pao and Yoshiyasu Takefuji. Functional-link net computing: theory, system architecture, and functionalities. *Computer*, 25(5):76–79, 1992.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, Adam Lerer, et al. Automatic differentiation in pytorch. 2017.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training pinns: A loss landscape perspective. *arXiv preprint arXiv:2402.01868*, 2024.

Enrico Schiassi, Roberto Furfaro, Carl Leake, Mario De Florio, Hunter Johnston, and Daniele Mortari. Extreme theory of functional connections: A fast physics-informed neural network method for solving ordinary and partial differential equations. *Neurocomputing*, 457:334–356, 2021.

Enrico Schiassi, Andrea D'Ambrosio, Kristofer Drozd, Fabio Curti, and Roberto Furfaro. Physics-informed neural networks for optimal planar orbit transfers. *Journal of Spacecraft and Rockets*, 59(3):834–849, 2022.

Wouter F Schmidt, Martin A Kraaijveld, Robert PW Duin, et al. Feed forward neural networks with random weights. In *International conference on pattern recognition*, pp. 1–1. IEEE Computer Society Press, 1992.

Khemraj Shukla, Patricio Clark Di Leoni, James Blackshire, Daniel Sparkman, and George Em Karniadakis. Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks. *Journal of Nondestructive Evaluation*, 39(3):61, 2020.

Dianhui Wang and Ming Li. Stochastic configuration networks: Fundamentals and algorithms. *IEEE transactions on cybernetics*, 47(10):3466–3479, 2017.

Dianhui Wang and Ming Li. Deep stochastic configuration networks with universal approximation property. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2018.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

Sifan Wang, Shyam Sankaran, Hanwen Wang, and Paris Perdikaris. An expert's guide to training physics-informed neural networks. *arXiv preprint arXiv:2308.08468*, 2023.

Sifan Wang, Bowen Li, Paris Perdikaris, et al. Gradient alignment in physics-informed neural networks: A second-order optimization perspective. *Advances in Neural Information Processing Systems*, 38:168482–168532, 2026.

Chenxi Wu, Min Zhu, Qinyang Tan, Yadhu Kartha, and Lu Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 403:115671, 2023.

Liang Wu, Wallace Gian Yion Tan, Leqi Zhou, Richard D Braatz, and Jan Drgona. Least-squares multi-step koopman operator learning for model predictive control. *arXiv preprint arXiv:2601.11901*, 2026.

Zixue Xiang, Wei Peng, Xu Liu, and Wen Yao. Self-adaptive loss balanced physics-informed neural networks. *Neurocomputing*, 496:11–34, 2022.

Yang Xiao, LM Yang, C Shu, H Dong, YJ Du, and YX Song. Least-square finite difference-based physics-informed neural network for steady incompressible flows. *Computers & Mathematics with Applications*, 175:33–48, 2024.

Lei Xu, Chunhua Yang, Xiaodong Xu, Biao Luo, and Tingwen Huang. Physics-informed stochastic configuration network promoted model predictive control with multi-objective optimization. *Artificial Intelligence Review*, 58(9):281, 2025.

APPENDIX

A Extended Background and Related Work

This appendix expands the background summarized in the Introduction, covering applications of physics-informed learning, the development of randomized networks and SCNs, and a related least-squares approach to dynamical-system learning.

Physics-Informed Learning and Applications. Scientific machine learning integrates physical knowledge with data-driven approximation (Baker et al., 2019). PINNs impose governing equations through the training loss (Raissi et al., 2019) and have been applied across several scientific and engineering domains. Schiassi et al. (2022) used PINNs to learn state–costate dynamics satisfying optimality conditions for orbital transfers. Shukla et al. (2020) studied surface-breaking cracks in metal plates, while Berkhahn & Ehrhardt (2022) modeled COVID-19 infection and hospitalization scenarios. Diao et al. (2023) addressed boundary conditions in multi-material solid-mechanics problems. These applications demonstrate the broad applicability of PINNs across scientific and engineering domains.

Beyond these applications, Adaptive activation functions (Jagtap et al., 2020), adaptive loss balancing (Wang et al., 2021; Xiang et al., 2022), and second-order optimization for gradient alignment (Wang et al., 2026) have been developed to improve PINN convergence and training accuracy. Least-square finite difference approximations (Xiao et al., 2024) replace AD for spatial derivatives to reduce computational costs.

Randomized Networks and Stochastic Configuration Networks. Randomized networks separate hidden-parameter assignment from output-weight estimation. Early examples include Random Vector Functional Link Networks (RVFLNs) (Pao & Takefuji, 1992) and random-weight feedforward networks (Schmidt et al., 1992). Extreme Learning Machines (ELMs) (Huang et al., 2006) further developed this approach, combining randomly assigned hidden parameters with analytical output-weight computation.

Incremental Random Weight Neural Networks (IRWNNs) (Liu et al., 2002) construct networks node by node to address network-size selection. SCNs (Wang & Li, 2017) introduce a supervisory inequality for configuring hidden parameters within adaptive ranges. This mechanism links incremental construction to residual reduction and a universal approximation property, while retaining linear least-squares estimation of output weights.

Subsequent work includes DeepSCN (Wang & Li, 2018), genetic-algorithm-based optimization (Li et al., 2018), and Monte Carlo tree search strategies (Felicetti & Wang, 2022a;b). Applications include robotic grasping recognition (Pan et al., 2020), greengage grading (Li et al., 2019), glass-insulator state recognition (Li et al., 2022), plastic-bottle recognition (Chen et al., 2022), and photovoltaic generation forecasting with missing inputs (Liu et al., 2020). These developments demonstrate the widespread application of SCNs.

Related Least-Squares Learning Approaches. Operator-theoretic approaches provide another route to efficient dynamical-system learning. Wu et al. (2026) proposed a parallelizable multi-step extended dynamic mode decomposition (EDMD) method that decomposes least-squares identification across prediction horizons and state coordinates. Although this approach enables efficient offline modeling, the cited formulation remains data-driven and does not impose physical constraints.

B Additional Algorithmic Details

B.1 PI-SC-II Derivations for Forward Problems

PI-SC-II follows the correction principle of PI-SC-III while restricting the active scope to a window of W hidden nodes. When $L \leq W$, it uses the global correction. The following formulation considers $W < L$, for which the first $L - W$ nodes are frozen and only the most recent W nodes are corrected. We partition the hidden layer matrix and output weights as $\boldsymbol{H}_L(\boldsymbol{x}) = [\boldsymbol{H}_{pre}(\boldsymbol{x}), \boldsymbol{H}_{win}(\boldsymbol{x})]$ and $\boldsymbol{\beta}_{glob} = [\boldsymbol{\beta}_{pre}^\top, \boldsymbol{\beta}_{win}^\top]^\top$, respectively. The frozen state is $\boldsymbol{u}_{L-W}(\boldsymbol{x}) = \boldsymbol{H}_{pre}(\boldsymbol{x})\boldsymbol{\beta}_{pre}$.

The active window fits the discrepancies left by the frozen state. The residual targets and design matrices are defined below.

$$\begin{aligned}
\boldsymbol{r}^p_{L-W}(\boldsymbol{x}) &= \boldsymbol{b}(\boldsymbol{x}) - \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{loc}) D^{\boldsymbol{\gamma}}\boldsymbol{u}_{L-W}(\boldsymbol{x}), \quad & \boldsymbol{A}_W(\boldsymbol{x}) &= \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{loc}) D^{\boldsymbol{\gamma}}\boldsymbol{H}_{win}(\boldsymbol{x}),\\
\boldsymbol{r}^b_{L-W}(\boldsymbol{x}) &= \boldsymbol{g}_{bc}(\boldsymbol{x}) - \sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}}\boldsymbol{u}_{L-W}(\boldsymbol{x}), & \boldsymbol{C}_W(\boldsymbol{x}) &= \sum_{\boldsymbol{\gamma}\in\mathbb{I}_{q_b}} \boldsymbol{P}_{\boldsymbol{\gamma}}(\boldsymbol{x}) D^{\boldsymbol{\gamma}}\boldsymbol{H}_{win}(\boldsymbol{x}),\\
\boldsymbol{r}^d_{L-W}(\boldsymbol{x}) &= \boldsymbol{u}_{data}(\boldsymbol{x}) - \boldsymbol{u}_{L-W}(\boldsymbol{x}), & \boldsymbol{D}_W(\boldsymbol{x}) &= \boldsymbol{H}_{win}(\boldsymbol{x}).
\end{aligned} \tag{21}$$

The window correction is formulated as the linear least squares objective

$$\begin{aligned}
\mathcal{L}_{win}(\boldsymbol{\beta}_{win}) = \frac{1}{N_p}\left\|\boldsymbol{A}_W(\boldsymbol{x})\boldsymbol{\beta}_{win} - \boldsymbol{r}^p_{L-W}(\boldsymbol{x})\right\|^2_{\Omega_p} + \frac{\omega_{bc}}{N_b}\left\|\boldsymbol{C}_W(\boldsymbol{x})\boldsymbol{\beta}_{win} - \boldsymbol{r}^b_{L-W}(\boldsymbol{x})\right\|^2_{\Omega_b}\\
+ \frac{\omega_{data}}{N_d}\left\|\boldsymbol{D}_W(\boldsymbol{x})\boldsymbol{\beta}_{win} - \boldsymbol{r}^d_{L-W}(\boldsymbol{x})\right\|^2_{\Omega_d}.
\end{aligned} \tag{22}$$

The three terms in (22) give the weighted design matrix $\boldsymbol{M}_W$ and target vector $\boldsymbol{v}_W$:

$$\boldsymbol{M}_W = \begin{bmatrix} \frac{1}{\sqrt{N_p}} \underset{i=1,\dots,N_p}{\operatorname{col}} \left(\boldsymbol{A}_W(\boldsymbol{x}^p_i)\right) \\ \sqrt{\frac{\omega_{bc}}{N_b}} \underset{i=1,\dots,N_b}{\operatorname{col}} \left(\boldsymbol{C}_W(\boldsymbol{x}^b_i)\right) \\ \sqrt{\frac{\omega_{data}}{N_d}} \underset{i=1,\dots,N_d}{\operatorname{col}} \left(\boldsymbol{D}_W(\boldsymbol{x}^d_i)\right) \end{bmatrix}, \boldsymbol{v}_W = \begin{bmatrix} \frac{1}{\sqrt{N_p}} \underset{i=1,\dots,N_p}{\operatorname{col}} \left(\boldsymbol{r}^p_{L-W}(\boldsymbol{x}^p_i)\right) \\ \sqrt{\frac{\omega_{bc}}{N_b}} \underset{i=1,\dots,N_b}{\operatorname{col}} \left(\boldsymbol{r}^b_{L-W}(\boldsymbol{x}^b_i)\right) \\ \sqrt{\frac{\omega_{data}}{N_d}} \underset{i=1,\dots,N_d}{\operatorname{col}} \left(\boldsymbol{r}^d_{L-W}(\boldsymbol{x}^d_i)\right) \end{bmatrix}. \tag{23}$$

Thus, $\mathcal{L}_{win}(\boldsymbol{\beta}_{win}) = \|\boldsymbol{M}_W\boldsymbol{\beta}_{win} - \boldsymbol{v}_W\|^2_2$. The active weights are computed using pseudo inverse and reassembled with the frozen weights as

$$\boldsymbol{\beta}^*_{win} = \boldsymbol{M}^\dagger_W \boldsymbol{v}_W, \qquad \tilde{\boldsymbol{\beta}}^*_{win} = \begin{bmatrix} \boldsymbol{\beta}_{pre} \\ \boldsymbol{\beta}^*_{win} \end{bmatrix}. \tag{24}$$

The matrix $\boldsymbol{M}_W$ contains at most mW columns, independent of the total network width L. To control the Taylor truncation error, PI-SC-II applies the same discrete line search as PI-SC-III, with $\Delta\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}^*_{win} - \boldsymbol{\beta}^*_{loc}$ and $\boldsymbol{\beta}(\alpha) = \boldsymbol{\beta}^*_{loc} + \alpha\Delta\boldsymbol{\beta}$. The selected step size α^* gives the final weights $\boldsymbol{\beta}^* = \boldsymbol{\beta}(\alpha^*)$.

B.2 Inverse Problem Derivations

Inverse PI-SC-I. At step L, PI-SC-I linearizes the physical residual simultaneously with respect to the generalized state and the physical parameters around $(\mathcal{D}^{(q)}\boldsymbol{u}_{L-1}, \boldsymbol{\lambda}_{L-1})$:

$$\boldsymbol{e}^p_L \approx \boldsymbol{e}^p_{L-1} + \sum_{\boldsymbol{\gamma}\in\mathbb{I}_q} \boldsymbol{J}_{D^{\boldsymbol{\gamma}}\boldsymbol{u}}(\boldsymbol{u}_{L-1})\left(D^{\boldsymbol{\gamma}}\boldsymbol{u}_L - D^{\boldsymbol{\gamma}}\boldsymbol{u}_{L-1}\right) + \boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{L-1}, \boldsymbol{\lambda}_{L-1})\left(\boldsymbol{\lambda}_L - \boldsymbol{\lambda}_{L-1}\right). \tag{25}$$

Using $D^{\boldsymbol{\gamma}}\boldsymbol{u}_L - D^{\boldsymbol{\gamma}}\boldsymbol{u}_{L-1} = \boldsymbol{\beta}_L D^{\boldsymbol{\gamma}} h_L(\boldsymbol{x})$ and $\Delta\boldsymbol{\lambda}_L = \boldsymbol{\lambda}_L - \boldsymbol{\lambda}_{L-1}$ reduces (25) to the augmented physical system in (15). The corresponding augmented design matrices are $\boldsymbol{A}_{L,aug}(\boldsymbol{x}) = [\boldsymbol{A}_L(\boldsymbol{x}), \boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{L-1}, \boldsymbol{\lambda}_{L-1})]$, $\boldsymbol{C}_{L,aug} = [\boldsymbol{C}_L, \boldsymbol{0}]$, and $\boldsymbol{D}_{L,aug} = [\boldsymbol{D}_L, \boldsymbol{0}]$. The state and parameters candidate is then obtained from

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}_L) = \frac{1}{N_p}\|\boldsymbol{A}_{L,aug}(\boldsymbol{x})\boldsymbol{\theta}_L + \boldsymbol{e}^p_{L-1}\|^2_{\Omega_p} + \frac{\omega_{bc}}{N_b}\|\boldsymbol{C}_{L,aug}(\boldsymbol{x})\boldsymbol{\theta}_L + \boldsymbol{e}^b_{L-1}\|^2_{\Omega_b}\\
+ \frac{\omega_{data}}{N_d}\|\boldsymbol{D}_{L,aug}(\boldsymbol{x})\boldsymbol{\theta}_L + \boldsymbol{e}^d_{L-1}\|^2_{\Omega_d}.
\end{aligned} \tag{26}$$

Solving this linear least-squares objective yields the candidate $\widehat{\boldsymbol{\theta}}_L$. As in forward PI-SC-I, the candidate is damped as $\boldsymbol{\theta}^*_L = \alpha\widehat{\boldsymbol{\theta}}_L$ and accepted through the supervisory node selection mechanism. The accepted local configuration is $\boldsymbol{u}_{loc} = \boldsymbol{u}_{L-1} + \boldsymbol{\beta}^*_L h_L$ and $\boldsymbol{\lambda}_{loc} = \boldsymbol{\lambda}_{L-1} + \Delta\boldsymbol{\lambda}^*_L$.

Inverse PI-SC-III. Updating $\boldsymbol{\lambda}$ strictly through localized node additions can restrict parameter-identification accuracy. PI-SC-III therefore jointly refines all output weights and physical parameters around the local prediction $(\mathcal{D}^{(q)}\boldsymbol{u}_{loc}, \boldsymbol{\lambda}_{loc})$. Let $\Delta\boldsymbol{\lambda}_{glob} = \boldsymbol{\lambda} - \boldsymbol{\lambda}_{loc}$ and define the global augmented vector $\boldsymbol{\theta}_{glob} = [\boldsymbol{\beta}_{glob}^\top, \Delta\boldsymbol{\lambda}_{glob}^\top]^\top$. Using the forward PI-SC-III formulation, the global linearized physical equation becomes

$$\boldsymbol{A}_{glob}(\boldsymbol{x})\boldsymbol{\beta}_{glob} + \boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{loc}, \boldsymbol{\lambda}_{loc})\Delta\boldsymbol{\lambda}_{glob} \approx \boldsymbol{b}(\boldsymbol{x}), \tag{27}$$

where $\boldsymbol{A}_{glob}$ and $\boldsymbol{b}$ are defined in Section 3.1, and $\boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{loc}, \boldsymbol{\lambda}_{loc}) = \frac{\partial \mathbf{F}}{\partial \boldsymbol{\lambda}}\big|_{\mathcal{D}^{(q)}\boldsymbol{u}=\mathcal{D}^{(q)}\boldsymbol{u}_{loc}, \boldsymbol{\lambda}=\boldsymbol{\lambda}_{loc}}$.

Let $\boldsymbol{J}_{\lambda,loc}(\boldsymbol{x}) = \boldsymbol{J}_\lambda(\boldsymbol{x}; \boldsymbol{u}_{loc}, \boldsymbol{\lambda}_{loc})$. The augmented matrices are $\boldsymbol{A}_{glob,aug}(\boldsymbol{x}) = [\boldsymbol{A}_{glob}(\boldsymbol{x}), \boldsymbol{J}_{\lambda,loc}(\boldsymbol{x})]$, $\boldsymbol{C}_{glob,aug}(\boldsymbol{x}) = [\boldsymbol{C}_{glob}(\boldsymbol{x}), \mathbf{0}]$, and $\boldsymbol{D}_{glob,aug}(\boldsymbol{x}) = [\boldsymbol{D}_{glob}(\boldsymbol{x}), \mathbf{0}]$. The global joint update is obtained from

$$\begin{aligned}\mathcal{L}_{glob}(\boldsymbol{\theta}_{glob}) = \frac{1}{N_p}\|\boldsymbol{A}_{glob,aug}(\boldsymbol{x})\boldsymbol{\theta}_{glob} - \boldsymbol{b}(\boldsymbol{x})\|_{\Omega_p}^2 + \frac{\omega_{bc}}{N_b}\|\boldsymbol{C}_{glob,aug}(\boldsymbol{x})\boldsymbol{\theta}_{glob} - \boldsymbol{g}_{bc}(\boldsymbol{x})\|_{\Omega_b}^2 \\ + \frac{\omega_{data}}{N_d}\|\boldsymbol{D}_{glob,aug}(\boldsymbol{x})\boldsymbol{\theta}_{glob} - \boldsymbol{u}_{data}(\boldsymbol{x})\|_{\Omega_d}^2.\end{aligned} \tag{28}$$

The corresponding solution is $\tilde{\boldsymbol{\theta}}_{glob}^* = [(\tilde{\boldsymbol{\beta}}_{glob}^*)^\top, (\Delta\tilde{\boldsymbol{\lambda}}_{glob}^*)^\top]^\top$.

Inverse PI-SC-II. PI-SC-II reduces the cost of full-batch joint inversion by restricting the correction to the most recent W hidden nodes. Define $\Delta\boldsymbol{\lambda}_{win} = \boldsymbol{\lambda} - \boldsymbol{\lambda}_{loc}$ and $\boldsymbol{\theta}_{win} = [\boldsymbol{\beta}_{win}^\top, \Delta\boldsymbol{\lambda}_{win}^\top]^\top$. Using $\boldsymbol{A}_W$, $\boldsymbol{C}_W$, and $\boldsymbol{D}_W$ from (21), the corresponding augmented matrices are $\boldsymbol{A}_{W,aug}(\boldsymbol{x}) = [\boldsymbol{A}_W(\boldsymbol{x}), \boldsymbol{J}_{\lambda,loc}(\boldsymbol{x})]$, $\boldsymbol{C}_{W,aug}(\boldsymbol{x}) = [\boldsymbol{C}_W(\boldsymbol{x}), \mathbf{0}]$, and $\boldsymbol{D}_{W,aug}(\boldsymbol{x}) = [\boldsymbol{D}_W(\boldsymbol{x}), \mathbf{0}]$. The sliding-window joint update is therefore formulated as

$$\begin{aligned}\mathcal{L}_{win}(\boldsymbol{\theta}_{win}) = \frac{1}{N_p}\|\boldsymbol{A}_{W,aug}(\boldsymbol{x})\boldsymbol{\theta}_{win} - \boldsymbol{r}_{L-W}^p(\boldsymbol{x})\|_{\Omega_p}^2 + \frac{\omega_{bc}}{N_b}\|\boldsymbol{C}_{W,aug}(\boldsymbol{x})\boldsymbol{\theta}_{win} - \boldsymbol{r}_{L-W}^b(\boldsymbol{x})\|_{\Omega_b}^2 \\ + \frac{\omega_{data}}{N_d}\|\boldsymbol{D}_{W,aug}(\boldsymbol{x})\boldsymbol{\theta}_{win} - \boldsymbol{r}_{L-W}^d(\boldsymbol{x})\|_{\Omega_d}^2.\end{aligned} \tag{29}$$

The corresponding solution is $\boldsymbol{\theta}_{win}^* = [(\boldsymbol{\beta}_{win}^*)^\top, (\Delta\tilde{\boldsymbol{\lambda}}_{win}^*)^\top]^\top$.

Discrete line search. Inverse PI-SC-II and PI-SC-III apply the discrete line search to the absolute joint configuration $\boldsymbol{\Theta} = [\boldsymbol{\beta}^\top, \boldsymbol{\lambda}^\top]^\top$. Let $\boldsymbol{\Theta}_{loc}^* = [\boldsymbol{\beta}_{loc}^{*\top}, \boldsymbol{\lambda}_{loc}^\top]^\top$. The candidate absolute configuration is reassembled as

$$\tilde{\boldsymbol{\Theta}}^* = \begin{cases} \begin{bmatrix} \tilde{\boldsymbol{\beta}}_{glob}^* \\ \boldsymbol{\lambda}_{loc} + \Delta\tilde{\boldsymbol{\lambda}}_{glob}^* \end{bmatrix}, & \text{for PI-SC-III,} \\ \begin{bmatrix} \boldsymbol{\beta}_{pre} \\ \boldsymbol{\beta}_{win}^* \\ \boldsymbol{\lambda}_{loc} + \Delta\tilde{\boldsymbol{\lambda}}_{win}^* \end{bmatrix}, & \text{for PI-SC-II.} \end{cases} \tag{30}$$

The search path and optimal step size are

$$\boldsymbol{\Theta}(\alpha) = \boldsymbol{\Theta}_{loc}^* + \alpha(\tilde{\boldsymbol{\Theta}}^* - \boldsymbol{\Theta}_{loc}^*), \tag{31}$$

and

$$\alpha^* = \arg\min_{\alpha \in \mathbb{A}} \mathcal{E}_L^2(\alpha), \tag{32}$$

respectively. The final joint update is $\boldsymbol{\Theta}^* = \boldsymbol{\Theta}(\alpha^*)$.

B.3 Detailed PI-SCM Training Algorithms

The complete pseudocode for the three PI-SCM algorithms is provided below. Algorithm 1 gives the localized construction used by all variants. Algorithms 2 and 3 augment this construction with sliding-window and global corrections, respectively.

Algorithm 1 Training procedure for PI-SC-I

Input: Given coordinates $\boldsymbol{x} = (x_1, \ldots, x_n)^\top$ partitioned into collocation sets $\Omega_p, \Omega_b, \Omega_d$; Sparse observational target $\boldsymbol{u}_{data}$; The differential operator $\mathbf{F}$ and initial/boundary operator $\mathcal{B}$; Initialized L_{max}, T_{max}; Error tolerance ϵ; Three sets of scalars $\Upsilon = \{\tau_1, \ldots, \tau_{end}\}, \mathbb{S}_r = \{r_1, \ldots, r_{end}\}, \mathbb{A}_1 = \{\alpha_{1,1} : \Delta\alpha_1 : \alpha_{1,end}\}$; Penalty weights $\omega_{bc}, \omega_{data}$.

Output: Hidden parameters $\{\boldsymbol{w}_L^*, b_L^*\}_{L=1}^{L_{end}}$, output weights $\{\boldsymbol{\beta}_L^*\}_{L=1}^{L_{end}}$ (and identified physical parameter $\boldsymbol{\lambda}^*$ for inverse tasks).

Initialize $L = 1, \boldsymbol{u}_0 = \mathbf{0}$. If inverse problem, initialize $\boldsymbol{\lambda}_0$.
Evaluate initial residuals $\boldsymbol{e}_0^p, \boldsymbol{e}_0^b, \boldsymbol{e}_0^d$ and comprehensive norm $\mathcal{E}_0^2$.
while $L \leq L_{max}$ **and** $\mathcal{E}_{L-1}^2 > \epsilon$ **do**
  Initialize two empty sets $\mathbb{P}$ and Ω_1.
  for $\alpha_1 \in \mathbb{A}_1$ **do**
    for $\tau \in \Upsilon$ **do**
      for $k = 1, 2, \ldots, T_{max}$ **do**
        Sample $\mathrm{w}_L \sim U(-\tau, \tau)$ and $\mathrm{b}_L \sim U(-\tau, \tau)$, and denote their realizations by $\boldsymbol{w}_L$ and b_L, respectively.
        For forward tasks, assemble $\boldsymbol{M}_L, \boldsymbol{v}_L$ by (7) and compute $\boldsymbol{\beta}_L^*$ by (8). For inverse tasks, solve (26) to obtain $\widehat{\boldsymbol{\theta}}_L$, and set $\boldsymbol{\theta}_L^* = \alpha_1 \widehat{\boldsymbol{\theta}}_L$.
        Extract $\boldsymbol{\beta}_L^*$ (and $\Delta\boldsymbol{\lambda}^*$ for inverse tasks).
        Update $\boldsymbol{u}_L = \boldsymbol{u}_{L-1} + \boldsymbol{\beta}_L^* h_L$.
        Evaluate $\mathcal{E}_L^2$.
        Calculate $\xi_L = (r + \mu_L)\mathcal{E}_{L-1}^2 - \mathcal{E}_L^2$, where $\mu_L = (1 - r)/(L + 1)$.
        if $\xi_L > 0$ **then**
          Save $\boldsymbol{w}_L, b_L, \boldsymbol{\beta}_L^*$ (and $\Delta\boldsymbol{\lambda}^*$ for inverse tasks) in $\mathbb{P}$, ξ_L in Ω_1.
        end if
      end for
      if $\mathbb{P}$ is not empty **then**
        Find $\boldsymbol{w}_L, b_L, \boldsymbol{\beta}_L^*$ (and $\Delta\boldsymbol{\lambda}^*$ for inverse tasks) that maximize ξ_L in Ω_1.
        Break (go to Step 27).
      else
        Update r to the subsequent value in $\mathbb{S}_r$.
        Continue
      end if
    end for
  end for
  Update $\boldsymbol{u}_L, \boldsymbol{e}_L^p, \boldsymbol{e}_L^b, \boldsymbol{e}_L^d, \mathcal{E}_L^2$ (and set $\boldsymbol{\lambda}_L^* = \boldsymbol{\lambda}_{L-1} + \Delta\boldsymbol{\lambda}_L^*$).
  $L \leftarrow L + 1$.
end while
return $\{\boldsymbol{w}_L^*, b_L^*\}_{L=1}^{L_{end}}$, $\{\boldsymbol{\beta}_L^*\}_{L=1}^{L_{end}}$ (and $\boldsymbol{\lambda}^*$ for inverse tasks).

Algorithm 2 Training procedure for PI-SC-II

Input: The inputs of Algorithm 1; Window size $W < L_{max}$; correction interval $L_g < L_{max}$; $\mathbb{A}_2 = \{\alpha_{2,1} : \Delta\alpha_2 : \alpha_{2,end}\}$.

Output: Hidden parameters $\{\boldsymbol{w}_L^*, b_L^*\}_{L=1}^{L_{end}}$, output weights $\{\boldsymbol{\beta}_L^*\}_{L=1}^{L_{end}}$ (and identified parameter $\boldsymbol{\lambda}^*$ for inverse tasks).

Initialize $L = 1, \boldsymbol{u}_0 = \mathbf{0}$. If inverse problem, initialize $\boldsymbol{\lambda}_0$.
Evaluate initial residuals $\boldsymbol{e}_0^p, \boldsymbol{e}_0^b, \boldsymbol{e}_0^d$ and comprehensive norm $\mathcal{E}_0^2$.
while $L \leq L_{max}$ **and** $\mathcal{E}_{L-1}^2 > \epsilon$ **do**
  Execute Step 4-26 of Algorithm 1 to secure $\boldsymbol{w}_L^*, b_L^*$ and $\boldsymbol{\beta}_{loc}^*$ (or $\boldsymbol{\Theta}_{loc}^*$ for inverse tasks).
  if $L \% L_g = 0$ **then**
    if $L \leq W$ **then**
      For forward tasks, assemble $\boldsymbol{M}_{glob}, \boldsymbol{v}_{glob}$ by (13) and compute $\tilde{\boldsymbol{\beta}}^*$ by (14). For inverse tasks, solve (28) to obtain $\tilde{\boldsymbol{\theta}}_{glob}^*$.
    else
      Retrieve $\boldsymbol{\beta}_{pre} = [\boldsymbol{\beta}_1^{*\top}, \ldots, \boldsymbol{\beta}_{L-W}^{*\top}]^\top$.
      Calculate $\boldsymbol{r}_{L-W}^p, \boldsymbol{r}_{L-W}^b, \boldsymbol{r}_{L-W}^d$ by (21).
      For forward tasks, assemble $\boldsymbol{M}_W, \boldsymbol{v}_W$ by (23) and compute $\boldsymbol{\beta}_{win}^*$ by (24).
      For inverse tasks, solve (29) to obtain $\boldsymbol{\theta}_{win}^*$.
      For forward tasks, let $\tilde{\boldsymbol{\beta}}^* = [\boldsymbol{\beta}_{pre}^\top, (\boldsymbol{\beta}_{win}^*)^\top]^\top$.
    end if
    Initialize an empty set Ω_2.
    Obtain $\Delta\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}^* - \boldsymbol{\beta}_{loc}^*$ (or obtain $\tilde{\boldsymbol{\Theta}}^*$ by (30), $\Delta\boldsymbol{\Theta} = \tilde{\boldsymbol{\Theta}}^* - \boldsymbol{\Theta}_{loc}^*$ for inverse tasks).
    for $\alpha_2 \in \mathbb{A}_2$ **do**
      Calculate $\boldsymbol{\beta}(\alpha_2) = \boldsymbol{\beta}_{loc}^* + \alpha_2 \Delta\boldsymbol{\beta}$ (or $\boldsymbol{\Theta}(\alpha_2)$ for inverse tasks) and corresponding $\mathcal{E}_L^2(\alpha_2)$. Save $\mathcal{E}_L^2(\alpha_2)$ to Ω_2.
    end for
    Find α_2^* that minimizes $\mathcal{E}_L^2(\alpha_2)$ in Ω_2 and adopt $\boldsymbol{\beta}^* = \boldsymbol{\beta}(\alpha_2^*)$ (or $\boldsymbol{\Theta}^* = \boldsymbol{\Theta}(\alpha_2^*)$ for inverse tasks).
  end if
  Extract $\boldsymbol{\beta}^*$ (and $\boldsymbol{\lambda}^*$ for inverse tasks).
  Update $\boldsymbol{u}_L, \boldsymbol{e}_L^p, \boldsymbol{e}_L^b, \boldsymbol{e}_L^d, \mathcal{E}_L^2$ (and set $\boldsymbol{\lambda}_L^* = \boldsymbol{\lambda}^*$).
  $L \leftarrow L + 1$.
end while
return $\{\boldsymbol{w}_L^*, b_L^*\}_{L=1}^{L_{end}}$, $\{\boldsymbol{\beta}_L^*\}_{L=1}^{L_{end}}$ (and $\boldsymbol{\lambda}^*$ for inverse tasks).

Algorithm 3 Training procedure for PI-SC-III

Input: The inputs of Algorithm 1; correction interval $L_g < L_{max}$; $\mathbb{A}_2 = \{\alpha_{2,min} : \Delta\alpha_2 : \alpha_{2,max}\}$.

Output: Hidden parameters $\{\boldsymbol{w}_L^*, b_L^*\}_{L=1}^{L_{end}}$, output weights $\{\boldsymbol{\beta}_L^*\}_{L=1}^{L_{end}}$ (and identified parameter $\boldsymbol{\lambda}^*$ for inverse tasks).

Initialize $L = 1, \boldsymbol{u}_0 = \mathbf{0}$. If inverse problem, initialize $\boldsymbol{\lambda}_0$.
Evaluate initial residuals $\boldsymbol{e}_0^p, \boldsymbol{e}_0^b, \boldsymbol{e}_0^d$ and comprehensive norm $\mathcal{E}_0^2$.
while $L \leq L_{max}$ **and** $\mathcal{E}_{L-1}^2 > \epsilon$ **do**
  Execute Step 4-26 of Algorithm 1 to secure $\boldsymbol{w}_L^*, b_L^*$ and $\boldsymbol{\beta}_{loc}^*$ (or $\boldsymbol{\Theta}_{loc}^*$ for inverse tasks).
  if $L \% L_g = 0$ **then**
    For forward tasks, assemble $\boldsymbol{M}_{glob}, \boldsymbol{v}_{glob}$ by (13) and compute $\tilde{\boldsymbol{\beta}}_{glob}^*$ by (14). For inverse tasks, solve (28) to obtain $\tilde{\boldsymbol{\theta}}_{glob}^*$.
    Initialize an empty set Ω_2.
    Obtain $\Delta\boldsymbol{\beta} = \tilde{\boldsymbol{\beta}}_{glob}^* - \boldsymbol{\beta}_{loc}^*$ (or obtain $\tilde{\boldsymbol{\Theta}}^*$ by (30), $\Delta\boldsymbol{\Theta} = \tilde{\boldsymbol{\Theta}}^* - \boldsymbol{\Theta}_{loc}^*$ for inverse tasks).
    for $\alpha_2 \in \mathbb{A}_2$ **do**
      Calculate $\boldsymbol{\beta}(\alpha_2) = \boldsymbol{\beta}_{loc}^* + \alpha_2\Delta\boldsymbol{\beta}$ (or $\boldsymbol{\Theta}(\alpha_2)$ for inverse tasks) and corresponding $\mathcal{E}_L^2(\alpha_2)$. Save $\mathcal{E}_L^2(\alpha_2)$ to Ω_2.
    end for
    Find α_2^* that minimizes $\mathcal{E}_L^2(\alpha_2)$ in Ω_2 and adopt $\boldsymbol{\beta}^* = \boldsymbol{\beta}(\alpha_2^*)$ (or $\boldsymbol{\Theta}^* = \boldsymbol{\Theta}(\alpha_2^*)$ for inverse tasks).
  end if
  Extract $\boldsymbol{\beta}^*$ (and $\boldsymbol{\lambda}^*$ for inverse tasks).
  Update $\boldsymbol{u}_L, \boldsymbol{e}_L^p, \boldsymbol{e}_L^b, \boldsymbol{e}_L^d, \mathcal{E}_L^2$ (and set $\boldsymbol{\lambda}_L^* = \boldsymbol{\lambda}^*$).
  $L \leftarrow L + 1$.
end while
return $\{\boldsymbol{w}_L^*, b_L^*\}_{L=1}^{L_{end}}$, $\{\boldsymbol{\beta}_L^*\}_{L=1}^{L_{end}}$ (and $\boldsymbol{\lambda}^*$ for inverse tasks).

C Proofs of the Universal Approximation Property

This appendix provides the detailed arguments for the theorems stated in Section 3.4.

The proofs below focus on forward problems. For inverse problems, the joint state and parameter increment $\boldsymbol{\theta}_L$ defined in Section 3.2 yields an augmented linearized system of the same form, allowing the convergence argument to carry over when the corresponding conditions hold.

As the convergence of the PI-SCM framework relies on the stability of incremental node additions, we initially prove the universal approximation property for the PI-SC-I algorithm, which serves as the convergent baseline for the subsequent variants. We recall $\tilde{\boldsymbol{e}}_L$, $\mathcal{E}_L^2$, $\boldsymbol{M}_L$, and $\boldsymbol{v}_L = -\tilde{\boldsymbol{e}}_{L-1}$ established in Section 3.1.

By applying the first-order Taylor expansion to $\tilde{\boldsymbol{e}}_L$, we have $\tilde{\boldsymbol{e}}_L = \tilde{\boldsymbol{e}}_{L-1} + \boldsymbol{M}_L\boldsymbol{\beta}_L + \boldsymbol{r}(\boldsymbol{\beta}_L) = \boldsymbol{M}_L\boldsymbol{\beta}_L - \boldsymbol{v}_L + \boldsymbol{r}(\boldsymbol{\beta}_L)$. The Taylor truncation error $\boldsymbol{r}(\boldsymbol{\beta}_L)$ originates exclusively from the nonlinear physical residual. The true nonlinear residual norm at the L-th step is thus given by:

$$\mathcal{E}_L^2 = \|\boldsymbol{M}_L\boldsymbol{\beta}_L - \boldsymbol{v}_L\|^2 + \mathcal{R}_{Taylor}(\boldsymbol{\beta}_L), \tag{33}$$

where $\mathcal{R}_{Taylor}(\boldsymbol{\beta}_L)$ is defined as:

$$\mathcal{R}_{Taylor}(\boldsymbol{\beta}_L) = 2(\boldsymbol{M}_L\boldsymbol{\beta}_L - \boldsymbol{v}_L)^\top \boldsymbol{r}(\boldsymbol{\beta}_L) + \|\boldsymbol{r}(\boldsymbol{\beta}_L)\|^2. \tag{34}$$

As shown in (34), $\mathcal{R}_{Taylor}(\boldsymbol{\beta}_L)$ represents the omitted second-order Taylor remainder.

Under the assumptions stated in Section 3.4, we establish the following lemma.

Lemma 1. *Suppose Assumptions 1 and 2 hold. There exist constants $\kappa \in (0, 1]$ and $D > 0$ such that whenever $\boldsymbol{v}_L \neq \mathbf{0}$, there exists a candidate node $h_L \in \Gamma$ that satisfies*

$$\|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 \geq \kappa\|\boldsymbol{v}_L\|^2, \tag{35}$$

and

$$\|\boldsymbol{M}_L(h_L)^\dagger\boldsymbol{v}_L\|^2 \leq D\|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2, \tag{36}$$

where

$$\boldsymbol{P}_L(h) = \boldsymbol{M}_L(h)\boldsymbol{M}_L(h)^\dagger. \tag{37}$$

Proof. Since the collocation state has fixed finite dimension, the uniform boundedness in Assumption 1 makes $\mathbb{K}$ compact. The operator regularity also implies that $\boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h_j)$ is continuous in $\boldsymbol{z}_{\mathrm{col}}$ for each fixed reference node.

The full row rank condition in Assumption 2, together with continuity and compactness, yields

$$a_0 := \min_{\boldsymbol{z}_{\mathrm{col}} \in \mathbb{K}} \lambda_{\min}\left(\boldsymbol{M}_{\mathrm{ref}}(\boldsymbol{z}_{\mathrm{col}})\boldsymbol{M}_{\mathrm{ref}}(\boldsymbol{z}_{\mathrm{col}})^\top\right) > 0, \tag{38}$$

$$B_0 := \max_{\boldsymbol{z}_{\mathrm{col}} \in \mathbb{K},\, 1 \le j \le J} \|\boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h_j)\|_2^2 < \infty. \tag{39}$$

For each nonzero reference block, constant rank ensures that its smallest positive singular value is continuous and positive on $\mathbb{K}$, and therefore has a positive minimum. Zero-rank blocks have zero pseudoinverses. Consequently,

$$D := \max_{\boldsymbol{z}_{\mathrm{col}} \in \mathbb{K},\, 1 \le j \le J} \|\boldsymbol{M}(\boldsymbol{z}_{\mathrm{col}}, h_j)^\dagger\|_2^2 < \infty. \tag{40}$$

Fix a reachable state at step L with $\boldsymbol{v}_L \neq \boldsymbol{0}$. By the definition of a_0,

$$\sum_{j=1}^{J} \|\boldsymbol{M}_L(h_j)^\top \boldsymbol{v}_L\|^2 \ge a_0 \|\boldsymbol{v}_L\|^2. \tag{41}$$

Hence there exists an index j_* such that

$$\|\boldsymbol{M}_L(h_{j_*})^\top \boldsymbol{v}_L\|^2 \ge \frac{a_0}{J} \|\boldsymbol{v}_L\|^2. \tag{42}$$

Choose $h_L = h_{j_*}$. Since $\boldsymbol{P}_L(h_L)$ is the orthogonal projector onto the column space of $\boldsymbol{M}_L(h_L)$, we have

$$\frac{a_0}{J}\|\boldsymbol{v}_L\|^2 \le \|\boldsymbol{M}_L(h_L)^\top \boldsymbol{v}_L\|^2 = \|\boldsymbol{M}_L(h_L)^\top \boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 \le B_0 \|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2. \tag{43}$$

Thus the first inequality holds with $\kappa = a_0/(JB_0)$. Since $\sum_{j=1}^{J} \|\boldsymbol{M}_L(h_j)^\top \boldsymbol{v}_L\|^2 \le JB_0 \|\boldsymbol{v}_L\|^2$, Eq. (41) gives $a_0 \le JB_0$ and hence $\kappa \in (0, 1]$.

For the same candidate node, $\boldsymbol{M}_L(h_L)^\dagger \boldsymbol{P}_L(h_L) = \boldsymbol{M}_L(h_L)^\dagger$ gives

$$\|\boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L\|^2 = \|\boldsymbol{M}_L(h_L)^\dagger \boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 \le D \|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2. \tag{44}$$

This completes the proof. □

We now prove Theorem 1.

Proof. If $\boldsymbol{v}_L = \boldsymbol{0}$, residual convergence has already been achieved. Otherwise, by Lemma 1, choose a node h_L satisfying Eqs. (35) and (36).

Recalling $\boldsymbol{P}_L(h_L)$ from Eq. 37, it satisfies $\boldsymbol{P}_L(h_L)^\top = \boldsymbol{P}_L(h_L)$ and $\boldsymbol{P}_L(h_L)^2 = \boldsymbol{P}_L(h_L)$. Hence, $\boldsymbol{M}_L(h_L)\boldsymbol{\beta}_L(\alpha) - \boldsymbol{v}_L = (\alpha \boldsymbol{P}_L(h_L) - \boldsymbol{I})\boldsymbol{v}_L$. Substituting the relation into Eq. (33), we define $\Delta\mathcal{E}^2(\alpha)$ as the reduction in the squared norm of the true residual:

$$\Delta\mathcal{E}^2(\alpha) = \mathcal{E}_{L-1}^2 - \mathcal{E}_L^2 = \alpha(2-\alpha)\|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 - \mathcal{R}_{Taylor}\left(\alpha \boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L\right). \tag{45}$$

The values and derivatives up to order q of h_L are uniformly bounded. Eq. (36) and the boundedness of the generalized residuals ensure that $\boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L$ is uniformly bounded. By compactness of $\mathbb{K}$, there exists $\bar{\alpha} \in (0, 1]$ such that all trial segments with $0 \le \alpha \le \bar{\alpha}$ lie in a common compact subset of the C^2 neighborhood in Assumption 1. Taylor's theorem and Eq. (34), together with the

preceding bounds, therefore yield a uniform constant $\bar{C} > 0$ such that $|\mathcal{R}_{Taylor}(\alpha \boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L)| \leq \alpha^2 \bar{C} \|\boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L\|^2$ for $0 \leq \alpha \leq \bar{\alpha}$. Applying this bound to Eq. (45) gives

$$\Delta\mathcal{E}^2(\alpha) \geq \alpha(2-\alpha)\|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 - \alpha^2\bar{C}\|\boldsymbol{M}_L(h_L)^\dagger \boldsymbol{v}_L\|^2. \tag{46}$$

Let D be the constant in Lemma 1. We reduce $\bar{\alpha}$ so that $\bar{\alpha} \leq (1+\bar{C}D)^{-1}$. Using Eqs. (36) and (35), together with $\|\boldsymbol{v}_L\|^2 = \mathcal{E}_{L-1}^2$, for any $\alpha \in ((1-r)/\kappa, \bar{\alpha}]$, we obtain

$$\begin{aligned} \Delta\mathcal{E}^2(\alpha) &\geq \alpha[2-\alpha(1+\bar{C}D)]\|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 \geq \alpha\|\boldsymbol{P}_L(h_L)\boldsymbol{v}_L\|^2 \\ &\geq \alpha\kappa\mathcal{E}_{L-1}^2 > (1-r)\mathcal{E}_{L-1}^2 \geq \delta_L. \end{aligned} \tag{47}$$

Consequently, we obtain

$$\mathcal{E}_L^2 \leq \mathcal{E}_{L-1}^2 - \delta_L = (r+\mu_L)\mathcal{E}_{L-1}^2. \tag{48}$$

Note that $\lim_{L\to+\infty} \mu_L = 0$ and $r < 1$. By utilizing (48), it follows that $\lim_{L\to+\infty} \mathcal{E}_L^2 = 0$. □

We next prove Theorem 2.

Recalling the optimal localized configuration $\boldsymbol{\beta}_{loc}^*$ and the global configuration $\tilde{\boldsymbol{\beta}}_{glob}^*$ established in Section 3.1, let $\mathcal{E}_L^2(\boldsymbol{\beta}) = \|\tilde{\boldsymbol{e}}_L(\boldsymbol{\beta})\|^2$ denote the comprehensive true residual norm evaluated at any candidate configuration $\boldsymbol{\beta}$. Consequently, the local residual norm is explicitly defined as $\mathcal{E}_{L,loc}^2 = \mathcal{E}_L^2(\boldsymbol{\beta}_{loc}^*)$.

Proof. If $\boldsymbol{M}_{glob}\tilde{\boldsymbol{\beta}}_{glob}^* \neq \boldsymbol{M}_{glob}\boldsymbol{\beta}_{loc}^*$, $\Delta\boldsymbol{\beta}$ constitutes a strict descent direction for the linearized least-squares objective. Analogous to the proof of Theorem 1, the first-order linear descent dominates the bounded second-order remainder for sufficiently small $\alpha > 0$. This guarantees the existence of a positive upper bound $\bar{\alpha} \in (0, 1]$ such that for all $\alpha \in (0, \bar{\alpha}]$, we have $\mathcal{E}_L^2(\boldsymbol{\beta}) < \mathcal{E}_{L,loc}^2$.

If $\boldsymbol{M}_{glob}\tilde{\boldsymbol{\beta}}_{glob}^* = \boldsymbol{M}_{glob}\boldsymbol{\beta}_{loc}^*$, taking $\bar{\alpha} = 0$ satisfies the acceptance criterion. Since $\alpha = 0$ inherently satisfies the equality $\mathcal{E}_L^2(\boldsymbol{\beta}) = \mathcal{E}_{L,loc}^2$, the inequality described in (19) holds for all $\alpha \in [0, \bar{\alpha}]$.

By (48), $\mathcal{E}_{L,loc}^2 \leq (r+\mu_L)\mathcal{E}_{L-1}^2$. Thus, we have:

$$\mathcal{E}_L^2 \leq \mathcal{E}_{L,loc}^2 \leq (r+\mu_L)\mathcal{E}_{L-1}^2. \tag{49}$$

Using the same arguments established in the proof of Theorem 1, we conclude that $\lim_{L\to+\infty} \mathcal{E}_L^2 = 0$. □

Since PI-SC-II algorithm employs an identical discrete line search mechanism as PI-SC-III, for brevity, we omit the proof of Theorem 3.

D Supplementary Details on Experiments

This appendix summarizes the common implementation and evaluation details used across all experiments, followed by the governing equations and problem-specific configurations of the benchmark systems, the complete activation-function study, and additional ablation studies.

Baseline Selection. We evaluate PI-SCM alongside PINNs (Raissi et al., 2019) and PIELM (Dwivedi & Srinivasan, 2020) as representative frameworks for physics-informed learning, spanning gradient optimization and randomized networks with analytical output weight estimation. PIELM is used for the linear Helmholtz equation, while NNBM (Huang & Wu, 2025) extends the randomized network comparison to the nonlinear systems. We additionally include VS-PINN (Ko & Park, 2025) for Allen–Cahn and PISCN (Xu et al., 2025) for semi-supervised Van der Pol. The inverse experiments retain PINNs as baselines for joint state and parameter estimation.

Hardware Details. All methods are implemented in Python and evaluated on a workstation equipped with an NVIDIA RTX 4060 Ti GPU, an Intel Core i5-14600KF CPU, and 16 GB of

RAM. PI-SCM, PIELM, and PIELM(NNBM) were run on the CPU, whereas PINN, VS-PINN, and PISCN were run on the GPU.

Evaluation Metric. For each system, state accuracy is measured by the root mean square error (RMSE) over N_{test} test inputs:

$$\text{RMSE} = \sqrt{\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \|\boldsymbol{u}_{\text{pred}}(\boldsymbol{\zeta}_i) - \boldsymbol{u}_{\text{true}}(\boldsymbol{\zeta}_i)\|_2^2}, \tag{50}$$

where $\boldsymbol{\zeta}_i$ denotes the ith test input, and $\boldsymbol{u}_{\text{pred}}$ and $\boldsymbol{u}_{\text{true}}$ denote the predicted and reference states, respectively.

D.1 Van der Pol Oscillator

The Van der Pol oscillator (Hafeez et al., 2015) is a nonlinear, non-conservative dynamical system used in circuit modeling, seismology, biological neuron modeling, and optimal control (Andersson et al., 2012). We consider the controlled second-order equation

$$u_{tt} - \mu(1 - u^2)u_t + u - U = 0, \qquad (t, U) \in [0, 5] \times [0, 1], \tag{51}$$

where μ is the nonlinear damping coefficient and U is the exogenous input. The initial conditions are $u(0, U) = -0.25$ and $u_t(0, U) = -2.0$. The value $\mu = 1$ is prescribed in the forward task, whereas μ is identified in the inverse task.

D.1.1 Forward Problems: Experimental Setup

We consider both unsupervised and semi-supervised settings. For the Van der Pol system, we sample $N_p = 1000$ interior points and $N_b = 100$ initial points, with an additional $N_d = 100$ observations in the semi-supervised setting. The penalty weights are set to $\omega_{bc} = 1$ and $\omega_{data} = 1$. The constructive parameters are $\Upsilon = 15 \times \{1, 2, 4, 8, 16, 32, 64\}$, $T_{max} = 50$, $L_{max} = 300$, $W = 200$, and the correction interval is evaluated at $L_g \in \{1, 50\}$, with $\mathbb{A}_1 = \{1.0, 0.9, \ldots, 0.1\}$, $\mathbb{A}_2 = \{1.0, 0.9, \ldots, 0\}$, and $\mathbb{S}_r = \{1 - 10^{-k} \mid k = 1, \ldots, 7\}$. PI-SCM uses the sine activation.

The PINN baselines use shallow and deep architectures of 1×300 and 4×64, respectively. Each baseline uses the same physics collocation, initial, and observational points as PI-SCM. The PINN baselines employ the hyperbolic tangent activation as recommended by Wang et al. (2023), and are trained with Adam for 5,000 epochs using an initial learning rate of 10^{-2} and a staircase exponential decay schedule with a decay factor of 0.9 applied every 100 epochs.

The NNBM baseline uses the same 1×300 architecture and the same physics collocation, initial, and observational points as PI-SCM. It employs the sine activation, and its hidden weights and biases are sampled as $\mathbf{w} \sim U(-15, 15)$ and $\mathrm{b} \sim U(-15, 15)$, respectively. The output weights are optimized using Gauss–Newton iterations. Following the setting in Huang & Wu (2025), the iterations terminate when the relative change in the linearized mean squared error falls below 10^{-2}.

For the Van der Pol equation, we additionally compare with PISCN, which adopts the same constructive settings as PI-SCM and is subsequently optimized with Adam for 100 epochs using a learning rate of 5×10^{-3} and a staircase exponential decay schedule with a decay factor of 0.9 applied every 100 epochs. Since PISCN requires labeled data during construction, it is evaluated only in the semi-supervised setting. PISCN uses the sine activation.

D.1.2 Forward Problems: Solution Visualization

Figure 1 shows the predicted state, absolute error, and representative cross-sections for the Van der Pol forward problem.

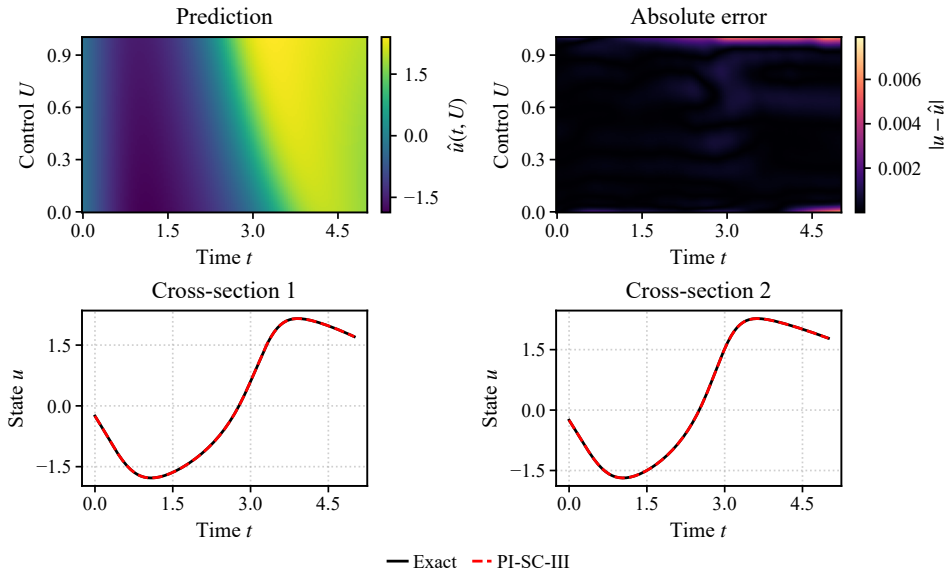

Figure 1: Semi-supervised PI-SC-III solution for the Van der Pol forward problem. The panels show the prediction, absolute pointwise error, and representative cross-sections at $U = 0.3$ and $U = 0.6$.

D.1.3 Inverse Problems: Experimental Setup

This inverse experiment jointly recovers the state and the scalar physical coefficient μ, and compares the two PINN architectures with PI-SC-I, PI-SC-II, and PI-SC-III. The unknown physical parameter has a ground-truth value of 1.0 and is initialized at 0.1, with $\mu_{true} = 1.0$ and $\mu^{(0)} = 0.1$. The experiment uses $N_d = 100$ observations, while all other sampling, construction, and baseline settings match the corresponding forward experiment.

D.2 Two-dimensional Helmholtz Equation

The Helmholtz equation is a canonical elliptic model of time-harmonic wave propagation. The benchmark is

$$u_{xx} + u_{yy} + k^2 u - q(x, y) = 0, \qquad (x, y) \in [-1, 1] \times [-1, 1], \tag{52}$$

with homogeneous Dirichlet conditions

$$u(-1, y) = u(1, y) = u(x, -1) = u(x, 1) = 0. \tag{53}$$

A manufactured source is used:

$$q(x, y) = -(a_1\pi)^2 \sin(a_1\pi x) \sin(a_2\pi y) - (a_2\pi)^2 \sin(a_1\pi x) \sin(a_2\pi y) + \sin(a_1\pi x) \sin(a_2\pi y), \tag{54}$$

where $a_1 = 1$ and $a_2 = 4$. The value $k^2 = 1$ is prescribed in the forward task, whereas k^2 is identified in the inverse task.

The corresponding analytical solution is

$$u(x, y) = \sin(a_1\pi x) \sin(a_2\pi y). \tag{55}$$

D.2.1 Forward Problems: Experimental Setup

For the two-dimensional Helmholtz equation, we use $N_p = 2000$ interior collocation points, $N_b = 1000$ boundary points, $L_{max} = 400$, and the correction interval is evaluated at $L_g \in \{1, 10\}$. The remaining PI-SCM settings follow those of the Van der Pol experiment.

The PINN baselines use shallow and deep architectures of 1×400 and 4×64, respectively. Each baseline uses the same physics collocation, boundary, and observational points as PI-SCM. The

PINN baselines employ the hyperbolic tangent activation and are trained with Adam for 5,000 epochs using an initial learning rate of 10^{-2} and a staircase exponential decay schedule with a decay factor of 0.9 applied every 100 epochs.

The PIELM baseline uses the same 1×400 architecture and the same physics collocation, boundary, and observational points as PI-SCM. It employs the sine activation, and its hidden weights and biases are sampled as $\mathbf{w} \sim U(-15, 15)$ and $\mathrm{b} \sim U(-15, 15)$, respectively.

D.2.2 Forward Problems: Solution Visualization

Figure 2 shows the predicted state, absolute error, and representative cross-sections for the Helmholtz forward problem.

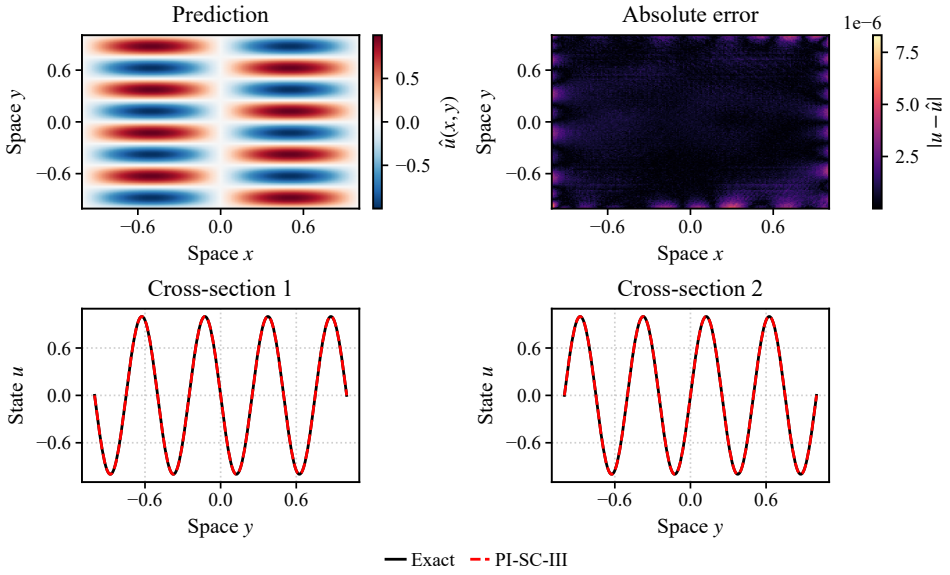

Figure 2: Semi-supervised PI-SC-III solution for the Helmholtz forward problem. The panels show the prediction, absolute pointwise error, and representative cross-sections at $x = -0.5$ and $x = 0.5$.

D.2.3 Inverse Problems: Experimental Setup

This inverse experiment jointly recovers the state and the scalar physical parameter k^2, and compares the two PINN architectures with PI-SC-I, PI-SC-II, and PI-SC-III. The unknown physical parameter has a ground-truth value of 1.0 and is initialized at 0.1, with $k^2_{true} = 1.0$ and $\left(k^2\right)^{(0)} = 0.1$. The experiment uses $N_d = 100$ observations, while all other sampling, construction, and baseline settings match the corresponding forward experiment.

D.3 Allen–Cahn Equation

The Allen–Cahn equation models phase separation and presents a challenging combination of a stiff reaction term and sharp moving interfaces. We consider

$$u_t - \epsilon u_{xx} + 5u^3 - 5u = 0, \qquad (x,t) \in [-1,1] \times [0,0.5], \tag{56}$$

where $\epsilon = 10^{-4}$, $u(x,0) = x^2 \cos(\pi x)$, and the periodic boundary constraints are $u(-1,t) = u(1,t)$ and $u_x(-1,t) = u_x(1,t)$. Because the small diffusion coefficient produces a large scale disparity between the derivative and reaction terms, we define $X = \omega_n x$ with $\omega_n = 1/\sqrt{\epsilon} = 100$. The transformed equation is

$$\frac{\partial u}{\partial t} - \epsilon \omega_n^2 \frac{\partial^2 u}{\partial X^2} + 5u^3 - 5u = 0. \tag{57}$$

Since $\epsilon\omega_n^2 = 1$, the coordinate stretching balances the state- and derivative-dependent Jacobian contributions and improves the conditioning of $\boldsymbol{M}_L$ without changing the underlying physical solution.

D.3.1 Forward Problems: Experimental Setup

For the Allen–Cahn equation, we use $N_b = 200$, $L_{max} = 500$, and the correction interval is evaluated at $L_g \in \{1, 10\}$, with $\Upsilon = 50 \times \{1, 2, 4, 8, 16, 32, 64\}$. The remaining PI-SCM settings follow those of the Van der Pol experiment.

The corresponding variable-scaling physics-informed neural network (VS-PINN) baselines use shallow and deep architectures of 1×500 and 4×64, respectively. Each baseline uses the same physics collocation, initial/boundary, and observational points as PI-SCM. The PINN-based baselines employ the hyperbolic tangent activation and are trained with Adam for 5,000 epochs using an initial learning rate of 10^{-3} and a staircase exponential decay schedule with a decay factor of 0.9 applied every 100 epochs.

The NNBM baseline uses the same 1×500 architecture and the same physics collocation, initial/boundary, and observational points as PI-SCM. It employs the sine activation, and its hidden weights and biases are sampled as $\mathbf{w} \sim U(-50, 50)$ and $\mathbf{b} \sim U(-50, 50)$, respectively. The Gauss–Newton settings follow those of the Van der Pol experiment.

D.3.2 Forward Problems: Solution Visualization

Figure 3 shows the predicted state, absolute error, and representative cross-sections for the Allen–Cahn forward problem.

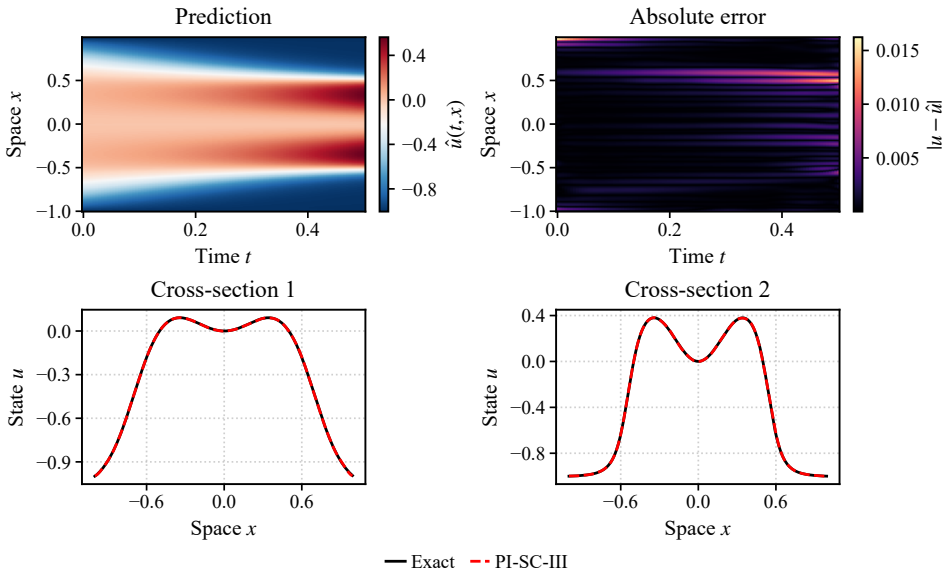

Figure 3: Semi-supervised PI-SC-III solution for the Allen–Cahn forward problem. The panels show the prediction, absolute pointwise error, and representative cross-sections at $t = 0.1$ and $t = 0.4$.

D.4 Five-dimensional Reaction–Diffusion Equation

The reaction–diffusion equation models spatial pattern formation arising from the interaction between diffusion and nonlinear reaction processes. We consider the five-dimensional nonlinear equation

$$u_t - \Delta u - u^2 = f(\boldsymbol{x}, t), \qquad (\boldsymbol{x}, t) \in [-1, 1]^5 \times [0, 1], \tag{58}$$

where $\boldsymbol{x} = (x_1, \ldots, x_5)^\top$ and $\Delta = \sum_{j=1}^{5} \partial^2/\partial x_j^2$ denotes the spatial Laplacian. A manufactured source is used:

$$f(\boldsymbol{x}, t) = (\pi^2 - 2)\sin\Big(\frac{\pi}{2}x_1\Big)\cos\Big(\frac{\pi}{2}x_2\Big)\,e^{-t} - 4\sin^2\Big(\frac{\pi}{2}x_1\Big)\cos^2\Big(\frac{\pi}{2}x_2\Big)\,e^{-2t}. \tag{59}$$

The corresponding analytical solution is

$$u(\boldsymbol{x}, t) = 2\sin\Big(\frac{\pi}{2}x_1\Big)\cos\Big(\frac{\pi}{2}x_2\Big)\,e^{-t}. \tag{60}$$

The initial condition and time-dependent Dirichlet boundary conditions are obtained from Eq. (60).

D.4.1 Forward Problems: Experimental Setup

For the five-dimensional reaction–diffusion equation, we use $N_p = 1000$ physical collocation points, $N_b = 2750$ boundary points, set $L_{max} = 1000$, and the correction interval is evaluated at $L_g \in \{1, 10\}$, with $\Upsilon = 1.5 \times \{1, 2, 4, 8, 16, 32, 64\}$. The remaining PI-SCM settings follow those of the Van der Pol experiment.

The PINN baselines use shallow and deep architectures of 1×1000 and 4×64, respectively. Each baseline uses the same physics collocation, initial/boundary, and observational points as PI-SCM. The PINN-based baselines employ the hyperbolic tangent activation and are trained with Adam for 5,000 epochs using an initial learning rate of 10^{-3} and a staircase exponential decay schedule with a decay factor of 0.9 applied every 100 epochs.

The NNBM baseline uses the same 1×1000 architecture and the same physics collocation, initial/boundary, and observational points as PI-SCM. It employs the sine activation, and its hidden weights and biases are sampled as $\mathbf{w} \sim U(-1.5, 1.5)$ and $\mathbf{b} \sim U(-1.5, 1.5)$, respectively. The Gauss–Newton settings follow those of the Van der Pol experiment.

D.4.2 Forward Problems: Solution Visualization

Figure 4 shows the predicted state, absolute error, and representative cross-sections for the reaction–diffusion forward problem.

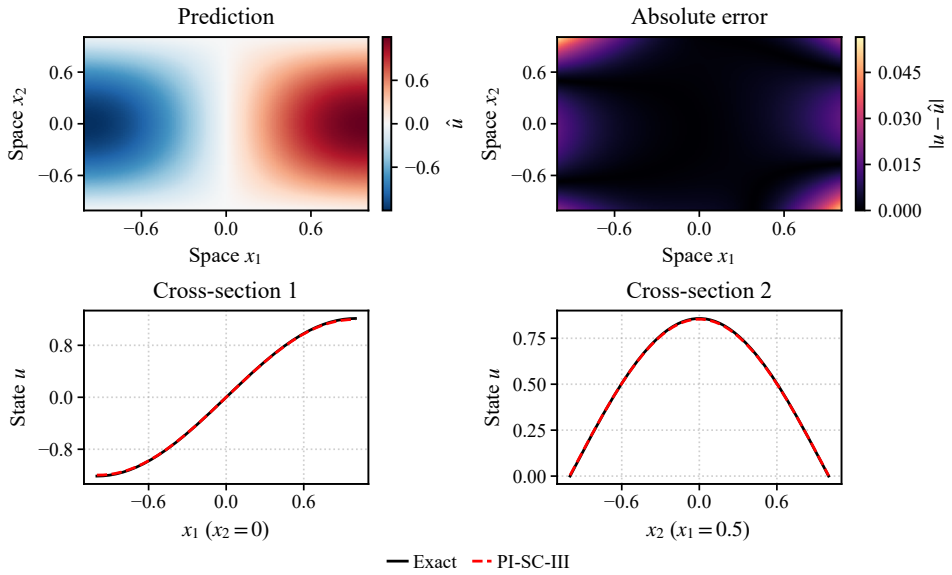

Figure 4: Semi-supervised PI-SC-III solution for the reaction–diffusion forward problem. The panels show the prediction and absolute pointwise error over the displayed (x_1, x_2) slice, together with representative cross-sections at $x_2 = 0$ and $x_1 = 0.5$.

D.5 Two-dimensional Bratu Equation

The Bratu equation is a nonlinear elliptic boundary-value problem with an exponential reaction term. We consider

$$u_{xx} + u_{yy} + \lambda e^u = 0, \qquad (x, y) \in [0, 1] \times [0, 1], \tag{61}$$

with homogeneous Dirichlet conditions

$$u(0, y) = u(1, y) = u(x, 0) = u(x, 1) = 0. \tag{62}$$

The value $\lambda = 1$ is prescribed in the forward task, whereas λ is identified in the inverse task.

D.5.1 Forward Problems: Experimental Setup

For the two-dimensional Bratu equation, we use $N_p = 2000$ interior collocation points, $N_b = 1000$ boundary points, and $T_{max} = 100$, and the correction interval is evaluated at $L_g \in \{1, 30\}$. The remaining PI-SCM settings follow those of the Van der Pol experiment.

The PINN baselines use shallow and deep architectures of 1×300 and 4×64, respectively. Each baseline uses the same physics collocation, boundary, and observational points as PI-SCM. They employ the hyperbolic tangent activation and are trained with Adam for 5,000 epochs using an initial learning rate of 10^{-3} and a staircase exponential decay schedule with a decay factor of 0.9 applied every 100 epochs.

The NNBM baseline uses the same 1×300 architecture and the same physics collocation, boundary, and observational points as PI-SCM. It employs the sine activation, and its hidden weights and biases are sampled as $\mathbf{w} \sim U(-15, 15)$ and $\mathbf{b} \sim U(-15, 15)$, respectively. The Gauss–Newton settings follow those of the Van der Pol experiment.

D.5.2 Forward Problems: Solution Visualization

Figure 5 shows the predicted state, absolute error, and centerline comparison for the Bratu forward problem.

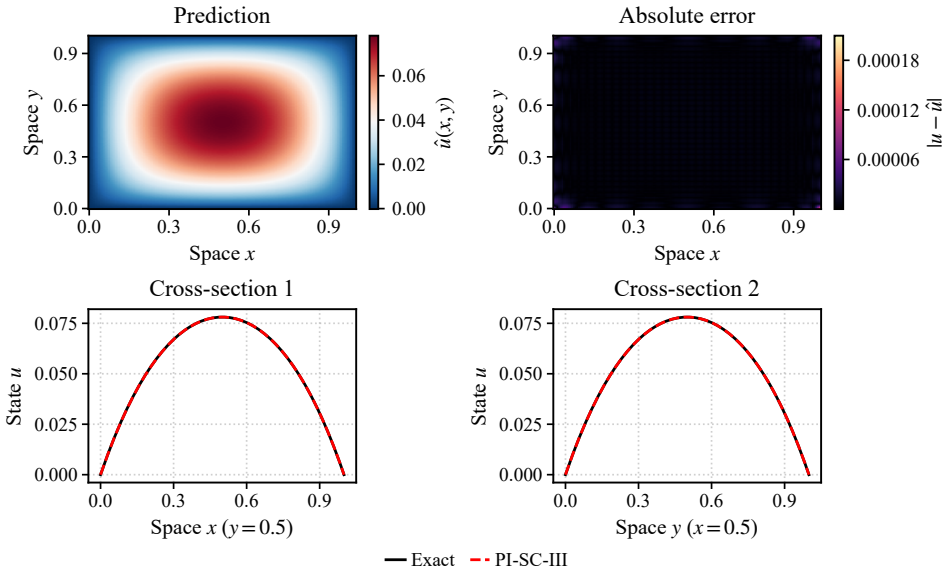

Figure 5: Semi-supervised PI-SC-III solution for the Bratu forward problem. The panels show the prediction and absolute pointwise error over the displayed (x, y) slice, together with representative cross-sections at $y = 0.5$ and $x = 0.5$.

D.5.3 Inverse Problems: Experimental Setup

This inverse experiment jointly recovers the state and the physical parameter λ, and compares the two PINN architectures with PI-SC-I, PI-SC-II, and PI-SC-III. The unknown physical parameter has a ground-truth value of 1.0 and is initialized at 0.1. The experiment uses $N_d = 100$ observations, while all other sampling, construction, and baseline settings match the corresponding forward experiment.

D.6 Activation-Function Study

D.6.1 Experimental Setup

We investigate the influence of the activation function on PI-SCM using the semi-supervised Van der Pol, Helmholtz, Allen–Cahn, reaction–diffusion, and Bratu forward problems. PI-SC-III is evaluated with sine, hyperbolic tangent, and sigmoid activations. For these five systems, L_{max} is set to 1,400 and the correction interval is evaluated at $L_g = 1$ while all other settings are kept identical to those of the corresponding forward problems described in Appendix D.1.1–D.5.1.

D.6.2 Results

All tabulated values are reported as mean $\pm$ standard deviation, and state predictive accuracy is evaluated on 10,000 randomly sampled test coordinates. Each activation is evaluated over 100 independent trials. Table 3 summarizes the state RMSE across the five benchmark systems, while Fig. 6 shows the evolution of the average true residual $\mathcal{E}_L^2$ with the number of accepted hidden nodes.

Table 3: Semi-supervised PI-SC-III results with different activation functions.

System	Activation	RMSE
Van der Pol	Sine	$5.66e{-4} \pm 3.39e{-4}$
	Tanh	$3.26e{-3} \pm 2.74e{-3}$
	Sigmoid	$4.93e{-3} \pm 3.74e{-3}$
Helmholtz	Sine	$6.09e{-7} \pm 1.30e{-7}$
	Tanh	$1.45e{-2} \pm 2.34e{-3}$
	Sigmoid	$8.11e{-4} \pm 5.12e{-4}$
Allen–Cahn	Sine	$1.31e{-3} \pm 7.95e{-4}$
	Tanh	$1.36e{-2} \pm 5.36e{-3}$
	Sigmoid	$2.04e{-3} \pm 1.26e{-3}$
Reaction–diffusion	Sine	$4.42e{-3} \pm 3.74e{-4}$
	Tanh	$2.52e{-2} \pm 1.41e{-3}$
	Sigmoid	$1.41e{-2} \pm 7.56e{-4}$
Bratu	Sine	$4.87e{-6} \pm 5.68e{-8}$
	Tanh	$6.14e{-6} \pm 4.86e{-7}$
	Sigmoid	$4.00e{-6} \pm 3.88e{-8}$

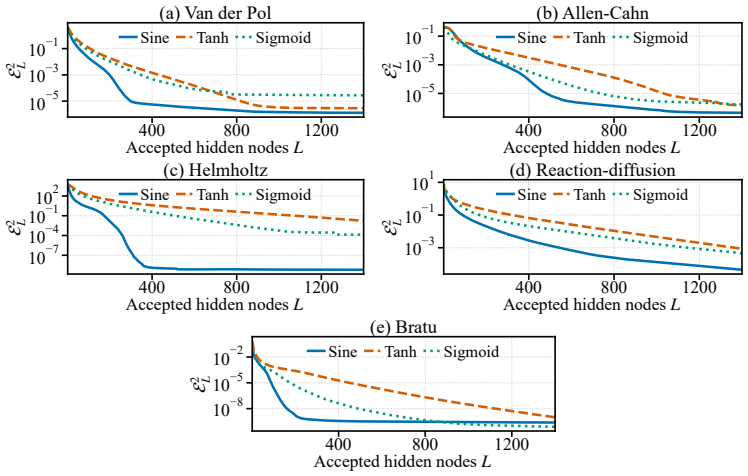

Figure 6: Evolution of $\mathcal{E}_L^2$ with the number of accepted hidden nodes for three activation functions.

D.6.3 Result Analysis

In PI-SCM, sine achieves the lowest predictive RMSE on four of the five benchmark systems and generally provides the fastest residual convergence. As shown in Fig. 6, it reduces the residual most rapidly and reaches the lowest residual level for Van der Pol, Helmholtz, Allen–Cahn, and reaction–diffusion. For Bratu, sigmoid yields the lowest RMSE of 4.00×10^{-6}, compared with 4.87×10^{-6} for sine. Overall, sine remains the most reliable default in PI-SCM across the evaluated systems.

D.7 Ablation Study

D.7.1 Experimental Setup

We examine the sensitivity of semi-supervised PI-SC-III to the number of interior collocation points N_p, the maximum network width L_{max}, and the correction interval L_g across the benchmark systems. All three studies cover the five systems. The studies of N_p and L_{max} use $L_g = 1$, whereas the study of L_g varies the correction interval as specified below. In each study, only the parameter under investigation is varied, and all other settings follow those of the corresponding forward problem in Appendix D.1.1–D.5.1.

For the collocation-point ablation, $N_p \in \{1000, 2000, 4000\}$ for Van der Pol, Helmholtz, reaction–diffusion, and Bratu, whereas $N_p \in \{500, 1000, 2000\}$ for Allen–Cahn, while all other settings are held fixed.

For the network-width ablation, $L_{max} \in \{150, 300, 600\}$ for Van der Pol and Bratu, $L_{max} \in \{200, 400, 800\}$ for Helmholtz, $L_{max} \in \{250, 500, 1000\}$ for Allen–Cahn, and $L_{max} \in \{500, 1000, 2000\}$ for reaction–diffusion, while all other settings are held fixed.

For the correction interval ablation, $L_g \in \{1, 5, 10, 20, 30, 40, 50\}$ for all five systems, while all other settings are held fixed.

D.7.2 Results

All tabulated values are reported as mean $\pm$ standard deviation, and state predictive accuracy is evaluated on the corresponding test set. Each setting is evaluated over 100 independent trials.

Number of interior collocation points. Table 4 reports the training time and state RMSE across the tested values of N_p.

Table 4: Sensitivity of semi-supervised PI-SC-III to the number of interior collocation points.

System	N_p	Time (s)	RMSE
Van der Pol oscillator	1000	1.61 ± 0.04	$6.05e{-4} \pm 1.66e{-4}$
	2000	1.82 ± 0.05	$4.96e{-4} \pm 1.96e{-4}$
	4000	2.33 ± 0.03	$4.87e{-4} \pm 2.18e{-4}$
Helmholtz equation	1000	2.26 ± 0.05	$1.06e{-6} \pm 5.22e{-7}$
	2000	2.62 ± 0.04	$1.04e{-6} \pm 7.75e{-7}$
	4000	3.53 ± 0.05	$1.12e{-6} \pm 2.03e{-6}$
Allen–Cahn equation	500	3.21 ± 0.08	$1.78e{-3} \pm 7.03e{-4}$
	1000	3.54 ± 0.06	$1.52e{-3} \pm 2.66e{-4}$
	2000	4.31 ± 0.04	$2.16e{-3} \pm 2.56e{-4}$
Reaction–diffusion equation	1000	17.1 ± 0.12	$6.08e{-3} \pm 4.72e{-4}$
	2000	19.7 ± 0.09	$5.97e{-3} \pm 4.25e{-4}$
	4000	28.1 ± 0.17	$5.70e{-3} \pm 4.01e{-4}$
Bratu equation	1000	2.48 ± 0.09	$9.25e{-6} \pm 5.15e{-6}$
	2000	3.01 ± 0.05	$7.01e{-6} \pm 1.39e{-6}$
	4000	4.53 ± 0.09	$6.61e{-6} \pm 1.07e{-6}$

Maximum network width. Table 5 reports the training time and state RMSE across the tested values of L_{max}.

Table 5: Sensitivity of semi-supervised PI-SC-III to the maximum network width.

System	L_{max}	Time (s)	RMSE
Van der Pol oscillator	150	0.66 ± 0.05	$2.18e{-2} \pm 9.21e{-3}$
	300	1.67 ± 0.05	$6.04e{-4} \pm 2.43e{-4}$
	600	4.02 ± 0.11	$7.20e{-4} \pm 2.67e{-4}$
Helmholtz equation	200	0.92 ± 0.02	$1.75e{-2} \pm 6.96e{-3}$
	400	2.67 ± 0.09	$9.81e{-7} \pm 5.07e{-7}$
	800	8.29 ± 0.22	$5.94e{-7} \pm 1.39e{-7}$
Allen–Cahn equation	250	1.36 ± 0.05	$3.35e{-2} \pm 1.59e{-2}$
	500	3.60 ± 0.07	$1.62e{-3} \pm 3.25e{-4}$
	1000	10.2 ± 0.11	$1.25e{-3} \pm 8.51e{-4}$
Reaction–diffusion equation	500	4.74 ± 0.06	$1.57e{-2} \pm 1.07e{-3}$
	1000	17.2 ± 0.40	$6.14e{-3} \pm 4.54e{-4}$
	2000	76.1 ± 0.30	$2.50e{-3} \pm 2.33e{-4}$
Bratu equation	150	1.19 ± 0.03	$9.65e{-6} \pm 2.80e{-6}$
	300	2.93 ± 0.04	$7.15e{-6} \pm 1.80e{-6}$
	600	9.50 ± 0.12	$6.65e{-6} \pm 1.46e{-6}$

Correction interval. Table 6 reports the training time and state RMSE across the tested values of L_g.

Table 6: Sensitivity of semi-supervised PI-SC-III to the correction interval.

System	L_g	Time (s)	RMSE
Van der Pol oscillator	1	1.61 ± 0.05	$5.61e{-4} \pm 1.61e{-4}$
	5	1.03 ± 0.06	$6.11e{-4} \pm 2.17e{-4}$
	10	0.93 ± 0.05	$6.49e{-4} \pm 2.74e{-4}$
	20	0.91 ± 0.05	$5.18e{-4} \pm 1.68e{-4}$
	30	0.88 ± 0.02	$4.53e{-4} \pm 1.47e{-4}$
	40	0.87 ± 0.04	$4.72e{-4} \pm 1.38e{-4}$
	50	0.87 ± 0.04	$4.64e{-4} \pm 1.25e{-4}$
Helmholtz equation	1	2.67 ± 0.08	$1.19e{-6} \pm 1.17e{-6}$
	5	1.52 ± 0.05	$1.51e{-6} \pm 4.22e{-6}$
	10	1.35 ± 0.03	$1.16e{-6} \pm 5.59e{-7}$
	20	1.28 ± 0.06	$1.68e{-6} \pm 1.28e{-6}$
	30	1.26 ± 0.06	$2.10e{-6} \pm 1.34e{-6}$
	40	1.22 ± 0.05	$2.59e{-6} \pm 3.42e{-6}$
	50	1.22 ± 0.06	$4.17e{-6} \pm 5.36e{-6}$
Allen–Cahn equation	1	3.58 ± 0.08	$1.65e{-3} \pm 3.10e{-4}$
	5	2.23 ± 0.04	$1.60e{-3} \pm 3.10e{-4}$
	10	2.06 ± 0.08	$1.58e{-3} \pm 2.80e{-4}$
	20	1.93 ± 0.07	$1.72e{-3} \pm 3.33e{-4}$
	30	1.92 ± 0.08	$2.08e{-3} \pm 4.01e{-4}$
	40	1.91 ± 0.08	$2.40e{-3} \pm 5.64e{-4}$
	50	1.91 ± 0.07	$2.54e{-3} \pm 6.81e{-4}$
Reaction–diffusion equation	1	17.5 ± 0.15	$6.13e{-3} \pm 4.42e{-4}$
	5	7.50 ± 0.07	$6.31e{-3} \pm 5.15e{-4}$
	10	6.33 ± 0.12	$6.38e{-3} \pm 4.18e{-4}$
	20	5.89 ± 0.12	$6.56e{-3} \pm 5.47e{-4}$
	30	5.65 ± 0.10	$6.64e{-3} \pm 6.02e{-4}$
	40	5.57 ± 0.09	$6.56e{-3} \pm 5.02e{-4}$
	50	5.60 ± 0.12	$6.54e{-3} \pm 5.30e{-4}$
Bratu equation	1	2.98 ± 0.10	$6.67e{-6} \pm 1.54e{-7}$
	5	1.67 ± 0.09	$6.72e{-6} \pm 1.68e{-7}$
	10	1.50 ± 0.08	$6.77e{-6} \pm 1.77e{-7}$
	20	1.47 ± 0.06	$6.70e{-6} \pm 1.77e{-7}$
	30	1.42 ± 0.07	$6.69e{-6} \pm 1.95e{-7}$
	40	1.42 ± 0.07	$6.98e{-6} \pm 1.72e{-7}$
	50	1.40 ± 0.07	$6.82e{-6} \pm 2.01e{-7}$

D.7.3 Result Analysis

Van der Pol and Bratu improve as N_p increases, whereas the gains saturate after $N_p = 2000$. Reaction–diffusion improves gradually, Helmholtz remains nearly unchanged, and Allen–Cahn attains its lowest RMSE at $N_p = 1000$. These trends show that the benefit of additional collocation points gradually diminishes and the predictive performance tends to converge once the physical domain is adequately sampled.

Increasing L_{max} from $1\times$ to $2\times$ reduces the RMSE by factors of approximately 36, 1.8×10^4, 21, and 2.6 for Van der Pol, Helmholtz, Allen–Cahn, and reaction–diffusion, respectively, while Bratu improves more moderately. Increasing the width from $2\times$ to $4\times$ yields further gains for four systems, but the corresponding training time increases by factors of 2.4–4.4, and the Van der Pol RMSE increases. Thus, the accuracy gains from increasing network width also tend to converge as the model reaches sufficient capacity. Further expansion mainly increases the computational cost and may raise the risk of overfitting, as suggested by the higher Van der Pol RMSE at the $4\times$ width.

Increasing L_g reduces the correction frequency and yields speedups of approximately $1.85\times$ to $3.12\times$ when increased from 1 to 50. From $L_g = 1$ to $L_g = 50$, the RMSE increases for all systems except Van der Pol. Thus, increasing L_g reduces computational cost, while predictive accuracy generally tends to decrease.