PINNMORPH: EVOLVING ONLINE ADAPTATION POLICIES FOR PHYSICS-INFORMED NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Physics-informed neural networks (PINNs) provide a learning-based framework for solving partial differential equations (PDEs), yet their training behavior can change substantially throughout optimization. Residual distributions, gradient interactions, regional learning difficulty, and model-capacity requirements may evolve over time, while the network architecture and major training mechanisms are typically determined before training. We propose PINNMorph, an online PINN adaptation framework based on large language model (LLM)-guided policy evolution. PINNMorph maintains a population of state-conditioned adaptation policies that map execution diagnostics to controlled interventions over topology modification, additive representation augmentation, objective balancing, gradient handling, adaptive sampling, and optimizer-phase control. At each intervention opportunity, candidate programs are instantiated from the current policy population, selected according to the observed training state, and applied directly to the PINN under training. The resulting model inherits its existing parameters and training state and continues optimization along the same trajectory. Execution outcomes are subsequently used to evaluate interventions and evolve the policy population. Unlike pre-training architecture search or fixed adaptation rules, PINNMorph jointly adapts the current PINN and the policies governing its interventions using feedback from actual training. Experiments on 13 PDE benchmarks show that PINNMorph achieves lower solution errors than SA-PINN, ConFIG, RoPINN, HARMONIC, and PINNsAgent across all evaluated problems. Ablation studies further examine the effects of online adaptation, state-conditioned intervention selection, and execution-feedback-driven policy evolution. Anonymous Code Repository: `https://anonymous.4open.science/r/PINNMorph/`.

1 INTRODUCTION

Physics-informed neural networks (PINNs) solve partial differential equations (PDEs) by incorporating governing equations and boundary or initial conditions into neural-network training (Raissi et al., 2019). Despite their flexibility, PINNs remain highly sensitive to training dynamics. Imbalanced physical objectives, gradient conflicts, spectral bias, and non-uniform residual distributions can impair convergence and solution accuracy (Wang et al., 2021a; 2022; Wu et al., 2023; Krishnapriyan et al., 2021; Rathore et al., 2024). Their effects also vary across PDEs and training regimes, as demonstrated by large-scale benchmarks such as PINNacle (Hao et al., 2024).

A further difficulty is that the dominant training bottleneck can change *within the same optimization trajectory*. Early training may be limited by imbalance among physical constraints, while later stages can suffer from gradient conflicts, localized residual concentration, insufficient sampling, or limited representational capacity (Daw et al., 2023; Wang et al., 2024b). Hence, the question is not only how a PINN should be adapted, but also what should be adapted as training evolves.

Existing adaptive PINN methods address specific aspects of this problem, including loss weighting, gradient coordination, adaptive sampling, and regional optimization (McClenny & Braga-Neto, 2023; Liu et al., 2025a; Wu et al., 2023; 2024). However, the adaptation family is typically fixed

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.

in advance. Automated PINN design instead searches for suitable architectures or training configurations before training or across independent runs (Wang et al., 2023; Wuwu et al., 2025). Neither paradigm directly addresses state-dependent selection among multiple intervention families within the training trajectory of the same PINN.

We therefore formulate online PINN adaptation as a sequential decision problem. At each diagnosis point, the current physical error, optimization behavior, model state, and remaining budget determine whether intervention is needed and which type of change is appropriate. Unlike fixed adaptation rules or pre-training configuration selection, the intervention space spans heterogeneous and structured changes to model topology, representation, objectives, gradients, sampling, and optimization. This motivates using LLMs to instantiate state-conditioned interventions and evolve the policies that generate them.

We propose PINNMorph, an online PINN adaptation framework based on LLM-guided policy evolution. PINNMorph maintains a population of state-conditioned policies that map training diagnostics to interventions over topology modification, additive representation augmentation, objective balancing, gradient handling, adaptive sampling, and optimizer-phase control. At each diagnosis point, candidate actions are generated from the current state, and the selected intervention is applied directly to the PINN under optimization. Learned parameters and transferable training states are preserved whenever possible, allowing the model to adapt without restarting optimization. PINN design thus becomes a process coupled with training rather than a one-time choice made beforehand.

Evaluating an intervention introduces a second challenge. Continued training may improve the PINN even without intervention, so post-intervention improvement alone does not reveal the effect of the action. PINNMorph therefore compares the realized trajectory with an estimated no-intervention continuation and uses the resulting execution feedback to evolve the policy population. Later decisions can thereby benefit from the observed consequences of earlier interventions along the same training trajectory.

We evaluate PINNMorph on 13 PDE benchmarks spanning nonlinear dynamics, multiscale systems, complex geometries, high-dimensional equations, and long-time problems. The comparison includes SA-PINN, ConFIG, RoPINN, HARMONIC, and PINNsAgent. PINNMorph achieves the lowest mean MSE on all 13 benchmarks, with a Friedman average rank of 1.00. Component ablations further examine the effects of online adaptation, state-conditioned action selection, and execution-feedback-driven policy evolution. Our main contributions are:

- **Cross-family online adaptation.** We formulate PINN adaptation as a within-trajectory sequential decision problem in which the appropriate intervention family can change as training evolves.
- **State-conditioned policy evolution.** We introduce an LLM-guided population of policies that turns multi-view training diagnostics into structured interventions across model, representation, objective, gradient, sampling, and optimizer dimensions.
- **Execution-grounded feedback.** To distinguish intervention effects from improvements due to continued training, we evaluate each intervention relative to an estimated no-intervention continuation and feed the resulting state–action–outcome evidence back into policy evolution, enabling later decisions to learn from earlier interventions.

2 Related Work

2.1 Physics-Informed Neural Networks and Adaptive Training

PINNs formulate PDE solving as neural function approximation constrained by governing equations and boundary or initial conditions (Raissi et al., 2019). Subsequent work has investigated architectures and representations that improve trainability and expressiveness, including Fourier features (Wang et al., 2021b), Transformer-based PINNs (Zhao et al., 2024), adaptive activation functions (Jagtap et al., 2020a; Zhang & Ding, 2025), causal training (Wang et al., 2024b), residual-adaptive networks (Wang et al., 2024a), and domain-decomposed models (Jagtap et al., 2020b; Jagtap & Karniadakis, 2020; Kharazmi et al., 2021). Geometry-aware trial functions can enforce boundary conditions exactly (Sukumar & Srivastava, 2022), while gradient-enhanced residual objectives add derivative information to the PINN loss (Yu et al., 2022). PINNacle provides a broad benchmark

showing that the relative effectiveness of such techniques varies substantially across PDE classes (Hao et al., 2024).

A complementary line of research adapts the training process itself. SA-PINN learns point-wise loss weights (McClenny & Braga-Neto, 2023); other methods dynamically balance physical objectives or transform their residuals (Cao et al., 2025; Gao et al., 2025; Zhang & Deng, 2025). Residual-based and failure-informed methods adapt collocation distributions toward difficult regions (Wu et al., 2023; Daw et al., 2023; Gao et al., 2023); imbalanced-learning, random-walk, and causality-guided schemes offer further sampling strategies (Luo et al., 2025; Wang & Hu, 2025; Lin & Chen, 2025). RoPINN extends optimization from individual points to adaptive local regions (Wu et al., 2024). Gradient-oriented approaches such as ConFIG and HARMONIC instead modify update directions to reduce conflicts among physical objectives (Liu et al., 2025a; Bu et al., 2026). These approaches demonstrate the value of training-time adaptation, but each primarily operates within a predefined adaptation family. PINNMorph instead selects among multiple intervention families according to the current training state.

2.2 Automated PINN Design

Automated PINN design aims to reduce manual configuration by searching over architectures and training choices. Auto-PINN and NAS-PINN explore PDE-specific neural architectures through automated search (Wang et al., 2023; Wang & Zhong, 2024). Evolutionary approaches search PINN architectures and activation functions (Kaplarević-Mališić et al., 2023; Zhang & Yang, 2024), while knowledge distillation has been used to discover network structures from trained models (Liu et al., 2025b). More recently, PINNsAgent uses LLMs together with PDE information and experimental feedback to iteratively generate improved PINN configurations (Wuwu et al., 2025), while Lang-PINN studies multi-agent language-model workflows for constructing physics-informed models from natural-language problem descriptions (You et al., 2026).

These methods generate candidate configurations and evaluate them as separate models or runs. PINNMorph instead operates *inside* one ongoing training process: policies observe the evolving state of the current PINN and directly modify that model without restarting optimization.

2.3 LLM-Driven Algorithm Evolution

LLMs have also been used as generators within iterative algorithm-design loops, as surveyed by Wu et al. (2025); Liu et al. (2026). OPRO uses previously evaluated solutions to guide later language-model proposals (Yang et al., 2024), while FunSearch combines program generation with external execution-based evaluation (Romera-Paredes et al., 2024). EoH evolves heuristic ideas and executable implementations through LLM generation and execution feedback (Liu et al., 2024); ReEvo incorporates reflective feedback during evolution (Ye et al., 2024); and LLaMEA iteratively improves optimization algorithms using runtime performance (van Stein & Bäck, 2025). Subsequent work studies in-loop hyperparameter optimization, controlled mutations, and multiple design objectives within LLM-based heuristic evolution (van Stein et al., 2025; Yin et al., 2025; Yao et al., 2025).

PINNMorph adopts the general principle of execution-feedback-driven evolution but changes the evolutionary object. Rather than evolving a standalone algorithm or independently evaluated model, PINNMorph evolves a state-conditioned policy $\pi : s_t \mapsto a_t$, where s_t is the current PINN training state and a_t is an online adaptation action. Policy evolution and PINN optimization therefore interact within the same continuous training trajectory.

3 Method

3.1 Problem Formulation

Consider a general PDE defined on a computational domain Ω, subject to boundary conditions and, for time-dependent problems, initial conditions:

$$\mathcal{F}[u](\mathbf{x}) = 0, \quad \mathbf{x} \in \Omega, \qquad \mathcal{B}[u](\mathbf{x}) = 0, \quad \mathbf{x} \in \partial\Omega, \qquad \mathcal{I}[u](\mathbf{x}) = 0, \quad \mathbf{x} \in \Omega_0. \tag{1}$$

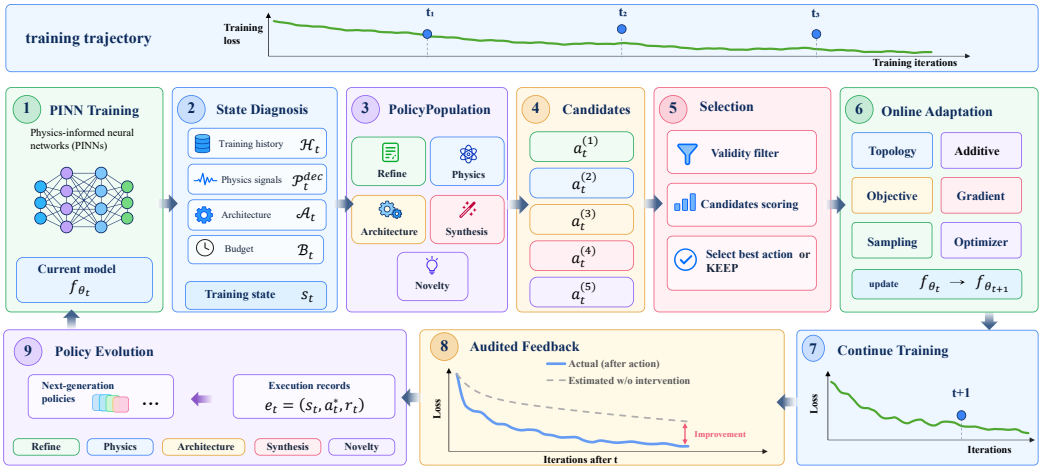

Figure 1: Overview of PINNMorph. At each adaptation point, PINNMorph diagnoses the current training state, uses a policy population to propose candidate actions, selects an executable action based on validity and state-conditioned scoring, and applies it online to the PINN under training. The resulting execution feedback is audited against the estimated no-intervention trajectory and used to evolve the policy population for subsequent adaptations.

Here, $\mathbf{x}$ denotes the spatial or spatiotemporal coordinate, u is the unknown physical field, Ω is the computational domain, $\partial\Omega$ denotes its boundary, and Ω_0 denotes the initial-time domain. The operators $\mathcal{F}$, $\mathcal{B}$, and $\mathcal{I}$ represent the governing equation, boundary condition, and initial condition, respectively. The initial-condition term is omitted when it is not required.

A PINN approximates the solution using a neural network $u_\theta(\mathbf{x})$ with parameters θ and minimizes

$$\mathcal{L}(\theta) = \lambda_{\text{PDE}}\mathcal{L}_{\text{PDE}} + \lambda_{\text{BC}}\mathcal{L}_{\text{BC}} + \lambda_{\text{IC}}\mathcal{L}_{\text{IC}}, \tag{2}$$

where $\mathcal{L}_{\text{PDE}}$, $\mathcal{L}_{\text{BC}}$, and $\mathcal{L}_{\text{IC}}$ denote the governing-equation, boundary-condition, and initial-condition losses, respectively, and λ_{PDE}, λ_{BC}, and λ_{IC} are their corresponding weights.

3.2 Method Overview

PINNMorph maintains a PINN under optimization throughout training. Let $f_{\theta_t}^{A_t}$ denote the PINN at training step t, with parameters θ_t and architecture A_t. At each diagnosis point, PINNMorph constructs a training state

$$s_t = \mathcal{D}(\mathcal{H}_t, \mathcal{P}_t, A_t, B_t), \tag{3}$$

where $\mathcal{H}_t$ summarizes recent optimization and adaptation history, $\mathcal{P}_t$ contains physics diagnostics, and B_t represents the remaining training budget and execution context.

PINNMorph maintains a population of $K = 5$ state-conditioned adaptation policies,

$$\Pi_g = \{\pi_g^{(1)}, \ldots, \pi_g^{(K)}\}, \tag{4}$$

which generate candidate interventions from the current training state. The selected intervention is executed on the current PINN, and its subsequent execution feedback is used to update the policy population.

3.3 Training-State Diagnosis

PINNMorph uses two fixed held-out collocation sets: a decision set $\mathcal{B}_{\text{dec}}$ and an audit set $\mathcal{B}_{\text{aud}}$. Neither participates in gradient updates.

At diagnosis point t, $\mathcal{B}_{\mathrm{dec}}$ is used to estimate physics diagnostics, including PDE, boundary, and initial-condition residuals, residual concentration, dominant constraint violations, gradient statistics, and optimization stagnation. These quantities, together with architecture and execution history, form the state s_t.

The audit set $\mathcal{B}_{\mathrm{aud}}$ is reserved for evaluating the physical effect of an executed intervention. Separating the decision and audit sets reduces direct coupling between action selection and feedback estimation.

3.4 Adaptation Policies

Each policy maps the current training state to an adaptation action, $a_t^{(k)} = \pi_g^{(k)}(s_t)$. The initial policy population is generated from the PDE description, current state, architecture, and controlled action space. Subsequent generations use five complementary modes: Refinement, Physics-Guided, Architecture-Guided, Synthesis, and Novelty.

These modes respectively improve effective policies, respond to dominant physical errors, adjust representational capacity, combine useful mechanisms, and explore new adaptation directions.

3.5 Online Adaptation

The intervention space comprises six action families: topology modification, additive representation, objective balancing, gradient coordination, adaptive sampling, and optimizer-phase control. At each diagnosis point, the policy population generates multiple candidate actions. PINNMorph scores these candidates based on the current training bottleneck and historical execution outcomes, and selects the most appropriate action:

$$a_t^* = \arg\max_{a \in \mathcal{C}_t} \mathcal{S}_\phi \left(s_t, a, \mathcal{H}_t^{\mathrm{exec}}\right). \tag{5}$$

The scoring function favors actions aligned with the current training bottleneck while incorporating evidence from previous interventions. If no action is sufficiently appropriate, training continues without intervention.

The selected action is applied directly to the current PINN. Structural interventions preserve learned parameters and transferable optimizer states whenever possible, while non-structural actions modify the corresponding training mechanism. At most one action is executed at each diagnosis point. Detailed action mechanisms and execution rules are provided in Appendix C.

3.6 Policy Evolution

After executing a_t^*, PINNMorph evaluates its effect from the subsequent training trajectory. Because ordinary optimization would also change the residual without intervention, the observed trajectory is compared with an estimated continuation trend:

$$r_t = \mathcal{U}\left(\mathcal{P}_{>t}, \widehat{\mathcal{P}}_{>t}^{0}\right). \tag{6}$$

Here, $\mathcal{P}_{>t}$ is the observed post-intervention trajectory, $\widehat{\mathcal{P}}_{>t}^{0}$ the estimated no-intervention continuation, and $\mathcal{U}$ the utility function comparing the two. The resulting feedback r_t estimates the incremental effect of the intervention. All intervention outcomes are recorded in the execution history. Numerical policy credit is assigned only when the decision and audit sets meet the forecast-eligibility and cross-set agreement criteria; consistent but low-confidence outcomes may provide bounded qualitative evidence for subsequent policy evolution.

The next policy population is generated as

$$\Pi_{g+1} = \mathcal{E}_\psi\left(\Pi_g, \mathcal{H}_g^{\mathrm{exec}}, s_t\right). \tag{7}$$

Policies associated with beneficial interventions provide positive experience for subsequent generations, whereas ineffective or narrowly applicable actions constrain future updates. In this way,

Table 1: Solution MSE on 13 PDE benchmarks. Results are reported as mean ± standard deviation over five independent runs. Lower is better.

PDE	SA-PINN	ConFIG	RoPINN	HARMONIC	PINNsAgent	PINNMorph
KS 1D	6.40E+0 ± 4.63E-2	1.47E+1 ± 4.60E-1	6.37E+0 ± 5.01E-1	5.55E+1 ± 5.88E+0	1.12E+0 ± 3.27E-2	**1.05E+0 ± 3.44E-1**
Burgers 1D	4.77E-4 ± 3.08E-5	2.47E-4 ± 1.27E-4	6.31E-4 ± 6.35E-5	1.32E-4 ± 2.13E-5	7.24E-5 ± 1.49E-5	**6.48E-5 ± 3.29E-5**
NS 2D CG	7.10E-3 ± 3.11E-4	2.46E-2 ± 7.81E-3	9.91E-3 ± 5.27E-3	5.34E-3 ± 4.84E-4	4.44E-3 ± 1.30E-3	**1.47E-3 ± 4.13E-4**
Heat 2D CG	3.41E-3 ± 6.07E-5	1.60E-2 ± 3.96E-3	4.92E-3 ± 8.93E-4	2.87E-2 ± 4.96E-4	3.61E-3 ± 8.71E-4	**1.81E-3 ± 4.48E-4**
Heat 2D MS	3.03E-4 ± 2.25E-5	3.18E-4 ± 2.99E-5	3.95E-4 ± 8.06E-6	4.20E-4 ± 6.47E-5	3.13E-4 ± 2.59E-5	**2.34E-4 ± 2.21E-5**
Heat 2D VC	1.45E-2 ± 5.17E-3	3.52E-2 ± 4.91E-3	4.70E-1 ± 4.86E-3	9.95E-2 ± 3.86E-2	5.62E-3 ± 3.33E-3	**1.82E-3 ± 2.54E-4**
Burgers 2D	5.13E-1 ± 5.07E-2	4.44E+0 ± 1.05E+0	3.43E-1 ± 3.16E-2	4.48E+0 ± 6.05E-1	2.13E-1 ± 1.46E-2	**1.75E-1 ± 4.14E-2**
Poisson 2D C	4.40E-1 ± 5.45E-2	1.17E+0 ± 4.11E-2	1.46E-1 ± 6.66E-2	3.50E+0 ± 5.72E-1	5.43E-1 ± 1.22E-1	**1.01E-1 ± 3.75E-3**
Poisson 2D CG	1.72E-1 ± 4.53E-2	4.69E-1 ± 1.95E-1	1.65E+0 ± 7.80E-1	1.24E-1 ± 3.99E-2	7.68E-2 ± 8.19E-3	**2.62E-2 ± 1.09E-2**
Poisson 2D MA	6.81E+0 ± 5.82E+0	3.31E+1 ± 9.97E+0	3.18E+0 ± 3.86E-1	5.64E+1 ± 4.99E+1	4.16E+0 ± 8.98E-1	**2.65E+0 ± 7.99E-1**
Gray–Scott 2D	1.81E-2 ± 2.52E-3	1.15E-1 ± 8.65E-3	9.96E-3 ± 4.70E-4	7.43E-2 ± 1.06E-2	9.02E-3 ± 2.84E-4	**3.34E-3 ± 4.74E-4**
Poisson 3D CG	3.05E-2 ± 6.19E-4	1.14E-1 ± 7.10E-2	2.07E-1 ± 1.46E-2	7.54E-2 ± 6.16E-3	1.74E-2 ± 1.17E-2	**6.79E-3 ± 1.23E-3**
Poisson 5D	5.84E-6 ± 3.62E-7	2.13E-6 ± 1.42E-7	7.41E-7 ± 8.02E-8	3.94E-6 ± 2.77E-6	2.32E-6 ± 1.26E-6	**6.26E-7 ± 8.32E-8**
Friedman Avg. Rank	3.69	4.77	4.08	4.92	2.54	**1.00**

training-state diagnosis, online adaptation, and feedback-driven policy evolution form a closed loop. Detailed continuation estimation and execution-feedback rules are provided in Appendix D.

4 EXPERIMENTS

We evaluate PINNMorph on 13 PDEs with diverse training characteristics. The experiments focus on overall solution accuracy, the contribution of online adaptation, the role of state-conditioned action selection, and the effect of execution-feedback-driven policy evolution.

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate PINNMorph on 13 PDEs covering nonlinear transport, multiscale dynamics, complex geometries, high-dimensional equations, and long-time dynamics. For PINNacle tasks, we follow the public problem definitions and reference-solution settings (Hao et al., 2024). Full PDE specifications are provided in Appendix A.

Baselines. We compare with SA-PINN (McClenny & Braga-Neto, 2023), ConFIG (Liu et al., 2025a), RoPINN (Wu et al., 2024), HARMONIC (Bu et al., 2026), and PINNsAgent (Wuwu et al., 2025), covering adaptive weighting, gradient coordination, regional optimization, and LLM-based PINN design.

Metrics. We use mean squared error (MSE) as the primary metric:

$$\mathrm{MSE} = \frac{1}{Nc} \sum_{i=1}^{N} \|\mathbf{u}_\theta(\mathbf{x}_i) - \mathbf{u}^*(\mathbf{x}_i)\|_2^2 . \tag{8}$$

Here, $\mathbf{u}_\theta(\mathbf{x}_i)$ is the solution obtained from the trained PINN, and $\mathbf{u}^*(\mathbf{x}_i)$ is the reference solution at evaluation point $\mathbf{x}_i$. The factor Nc is the total number of scalar solution values in the evaluation set. All methods are evaluated over five random seeds, and we report mean ± standard deviation.

Implementation Details. All methods use a fixed training budget of $20{,}000$ optimizer steps. DeepSeek-V4-Flash initializes and evolves the policy population, while Qwen3.8-27B evaluates candidate actions. PINNsAgent also uses DeepSeek-V4-Flash. The same LLM configuration is used across all PDEs. Additional implementation details are provided in Appendix B.

4.2 MAIN RESULTS

Table 1 compares the final MSE of PINNMorph and the baselines on all 13 PDEs. Each method is independently evaluated with five random seeds, and results are reported as mean ± standard deviation.

PINNMorph achieves the lowest mean MSE on all 13 PDE benchmarks, with a Friedman average rank of 1.00, indicating consistently strong performance across the benchmark suite.

The improvements are substantial on several representative problems. Compared with the best-performing baseline on each PDE, PINNMorph reduces MSE by 67.6% on Heat 2D VC, 66.9%

Table 2: Component configurations used in the ablation study.

Variant	Online Adaptation	State-Conditioned Selection	Policy Evolution	Feedback Evolution
No Morph	×	×	×	×
Random Selection	✓	×	✓	✓
Random Evolution	✓	✓	✓	×
PINNMorph	✓	✓	✓	✓

Table 3: Component ablation results. Results are reported as MSE (mean ± standard deviation) over five independent runs. Lower is better.

PDE	No Morph	Random Selection	Random Evolution	PINNMorph
KS 1D	5.21E+0 ± 1.76E+0	6.24E+0 ± 1.87E+0	5.03E+0 ± 5.74E-1	**1.05E+0 ± 3.44E-1**
Burgers 1D	6.82E-4 ± 5.05E-4	3.62E-4 ± 1.35E-4	7.92E-4 ± 4.96E-4	**6.48E-5 ± 3.29E-5**
NS 2D CG	8.72E-3 ± 3.38E-3	9.46E-2 ± 1.10E-2	4.39E-3 ± 1.86E-3	**1.47E-3 ± 4.13E-4**
Heat 2D CG	5.17E-3 ± 6.43E-4	1.95E-2 ± 4.83E-3	4.01E-2 ± 5.46E-3	**1.81E-3 ± 4.48E-4**
Heat 2D MS	4.26E-3 ± 3.91E-4	3.89E-3 ± 2.72E-4	4.43E-3 ± 8.00E-5	**2.34E-4 ± 2.21E-5**
Heat 2D VC	3.43E-2 ± 1.67E-2	4.20E-2 ± 1.46E-2	4.94E-3 ± 2.07E-3	**1.82E-3 ± 2.54E-4**
Burgers 2D	4.26E+0 ± 5.20E-1	5.51E+1 ± 1.54E+1	2.76E+0 ± 1.62E-1	**1.75E-1 ± 4.14E-2**
Poisson 2D C	6.26E+0 ± 8.75E-1	7.43E+0 ± 4.02E-1	1.04E+0 ± 3.74E-1	**1.01E-1 ± 3.75E-3**
Poisson 2D CG	3.77E-1 ± 1.67E-1	5.08E-1 ± 2.53E-1	7.50E-1 ± 1.02E-1	**2.62E-2 ± 1.09E-2**
Poisson 2D MA	1.30E+1 ± 6.09E+0	1.70E+1 ± 6.12E+0	1.04E+1 ± 2.35E+0	**2.65E+0 ± 7.99E-1**
Gray–Scott 2D	3.31E-2 ± 1.21E-2	2.51E-2 ± 1.14E-2	7.35E-2 ± 3.00E-3	**3.34E-3 ± 4.74E-4**
Poisson 3D CG	8.59E-2 ± 3.32E-3	6.69E-2 ± 1.05E-2	1.56E-2 ± 6.10E-3	**6.79E-3 ± 1.23E-3**
Poisson 5D	9.00E-6 ± 3.72E-6	2.95E-6 ± 8.96E-7	2.80E-6 ± 5.12E-7	**6.26E-7 ± 8.32E-8**

on NS 2D CG, 65.9% on Poisson 2D CG, 63.0% on Gray–Scott 2D, and 61.0% on Poisson 3D CG. These gains span PDEs with different dimensionalities, geometries, coefficients, and nonlinear dynamics.

The advantage also persists on tasks where the performance margins are smaller. PINNMorph achieves the best mean MSE on Burgers 1D, Heat 2D MS, and Poisson 5D, where the strongest baselines already attain relatively low errors. Taken together, the results show that the improvement is consistent across diverse PDE settings rather than driven by a small subset of benchmarks.

The contributions of online adaptation, state-conditioned action selection, and policy evolution are further examined through ablation studies.

4.3 Component Ablation

To analyze the three core mechanisms of PINNMorph, we construct three ablated variants: No Morph, Random Selection, and Random Evolution. All variants use the same base PINN, random seeds, training data, and 20,000-step optimization budget as the complete method. Each variant is independently run five times, and MSE is reported as mean ± standard deviation. No Morph: Online adaptation is removed. The base PINN retains a fixed model structure and fixed training mechanisms throughout optimization. Random Selection: The dependence of final action selection on the training state s_t is removed. At each adaptation point, the final action is sampled randomly from the executable candidate set. Random Evolution: The generation of the next population does not use execution feedback from previous actions; instead, the evolution direction is randomized. Table 2 summarizes the component configurations. Table 3 reports the final MSE of the three ablated variants and the complete PINNMorph on the 13 PDEs.

Because the MSE values span several orders of magnitude across PDEs, we compute the error ratio of each ablated variant relative to the complete PINNMorph:

$$r_p = \frac{\mathrm{MSE}_{\mathrm{ablation},p}}{\mathrm{MSE}_{\mathrm{PINNMorph},p}}. \tag{9}$$

We then summarize performance across PDEs using $r_{\mathrm{median}} = \mathrm{median}_p\,(r_p)$. A value of $r_{\mathrm{median}} > 1$ indicates that the ablated variant has a higher overall solution error than PINNMorph.

Table 4: Representative online adaptations of PINNMorph and their effects under different training states.

Training state	Diagnostic signal	Action	Before $\rightarrow$ After
Localized error	Shock-localized residual	Localized Expert	$1.01 \times 10^{-2} \rightarrow 6.32 \times 10^{-4}$
Representation limit	Non-local residual	Fourier Branch	$3.71 \times 10^{-3} \rightarrow 2.59 \times 10^{-4}$
Spatial imbalance	Residual imbalance	Adaptive Sampling	$3.08 \times 10^{-2} \rightarrow 7.63 \times 10^{-3}$
Gradient conflict	Inter-equation conflict	Topology Split	$1.36 \times 10^{-3} \rightarrow 6.53 \times 10^{-4}$
Spatial imbalance	High-D residual imbalance	Adaptive Sampling	$2.26 \times 10^{-3} \rightarrow 8.08 \times 10^{-4}$
Optimization plateau	Stable gradients	Correction Branch	$4.86 \times 10^{-3} \rightarrow 1.46 \times 10^{-3}$

The full PINNMorph achieves the lowest mean MSE across all 13 PDEs. As shown in Table 5, Removing online adaptation increases the median error ratio to $12.65\times$, demonstrating the importance of adapting the PINN during optimization. Random Selection yields a median error ratio of $10.77\times$ and is inferior to No Morph on 8 of 13 tasks. Although random interventions can occasionally improve performance substantially on individual PDEs, their degradation on the majority of tasks indicates that online adaptation alone does not guarantee consistent improvements when intervention selection is decoupled from the current training state. Random Evolution achieves the lowest median error ratio among the three ablations at $10.30\times$ and is the strongest ablation on 8 of 13 tasks, indicating that state diagnosis and state-conditioned action selection already provide substantial benefits even without execution-feedback-driven evolution. Nevertheless, the complete PINNMorph further improves performance on all 13 PDEs, reducing the median error by approximately one order of magnitude relative to each ablated variant. These results demonstrate the complementary contributions of online adaptation, state-conditioned selection, and execution-feedback-driven policy evolution.

4.4 Analysis of the Online Adaptation Process

To examine the online behavior of PINNMorph, we analyze the evolution of optimization and physics-related signals together with representative adaptations. Figure 2 shows the training loss, Decision physics score, and Audit physics score over four representative runs. The Decision physics score reflects the physical state available to online decision making, while the Audit physics score is computed on the fixed audit set $\mathcal{B}_{\mathrm{aud}}$. Both are residual-based quantities, with lower values indicating better satisfaction of the governing equations.

As shown in Figure 2, these signals evolve non-uniformly and may improve at different rates, indicating that the dominant bottleneck can shift during optimization. Training loss alone therefore cannot distinguish whether poor progress arises from spatial error concentration, limited representation, or interactions among physical objectives. This motivates state-dependent diagnostics for online adaptation.

Table 4 summarizes representative adaptations under different diagnosed training states. The value before intervention is measured at the intervention point, while the value after intervention is recorded 500 iterations later. Different bottlenecks favor different responses: representation-related difficulties can trigger localized specialization or global feature augmentation, spatial imbalance favors adaptive sampling, and conflicts among physical objectives can motivate structural specialization. Similar symptoms, such as slow progress or a plateau, may therefore require different interventions depending on the diagnosed state.

PINNMorph applies the selected action directly to the current PINN and evaluates its effect during continued optimization. The resulting execution feedback is then used for policy evolution, closing the loop between state diagnosis, intervention, execution, and subsequent adaptation.

4.5 Computational Cost

Table 5 shows that PINNMorph introduces only moderate wall-clock overhead under the same optimization budget. This is because it adapts the current PINN in place rather than restarting training with a new model. The additional cost mainly comes from periodic diagnosis, LLM inference,

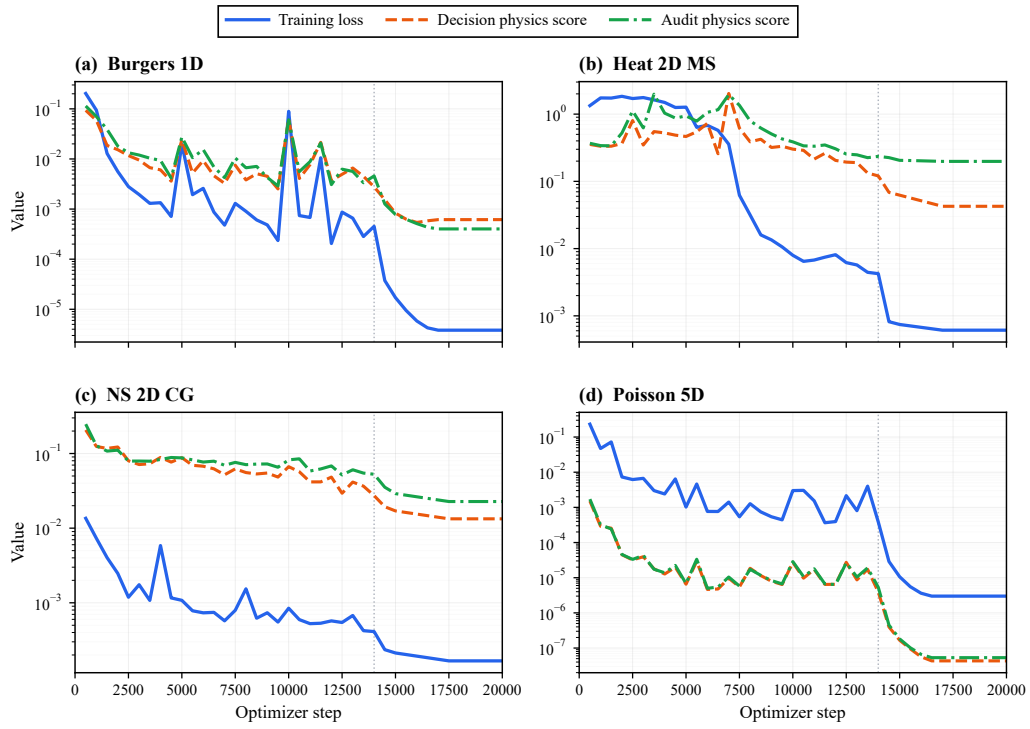

Figure 2: Representative training trajectories of PINNMorph. Training loss and the Decision and Audit physics scores are tracked throughout optimization, with the two physics scores computed on the fixed decision and audit sets, respectively.

Table 5: Computational cost and median MSE ratios of PINNMorph and its ablated variants. All variants use the same 20,000-step optimization budget. Median MSE ratios are computed relative to PINNMorph.

Variant	LLM Calls	LLM Tokens	Wall Time (s)	Median MSE Ratio
No Morph	0	0	1250	12.65×
Random Selection	30	6.52×10^4	1273	10.77×
Random Evolution	33	1.06×10^5	1286	10.30×
PINNMorph	38	4.40×10^5	1308	1.00×

intervention execution, and policy updates. Compared with No Morph, PINNMorph incurs approximately 4.6% additional wall-clock time while achieving substantially lower solution error.

5 Conclusion

We presented PINNMorph, an online adaptation framework for physics-informed neural networks that integrates training-state diagnosis, state-conditioned intervention, and execution-feedback-driven policy evolution within a single training trajectory. Rather than fixing an adaptation mechanism in advance, PINNMorph selects among heterogeneous intervention families according to the evolving training state, applies the selected action directly to the current model, and preserves learned parameters and transferable training state whenever possible. The resulting execution feedback is then used to improve subsequent adaptation policies. Across 13 PDE benchmarks, PINNMorph achieves the lowest mean MSE, with ablations confirming the value of online adaptation, state-conditioned selection, and feedback-driven policy evolution. With the same 20,000-step budget, it incurs only about 4.6% wall-clock overhead over No Morph.

AI USE STATEMENT

Large language models are part of PINNMorph: DeepSeek-V4-Flash initializes and evolves state-conditioned adaptation policies, and Qwen3.8-27B scores candidate actions. Selected interventions undergo deterministic validation before execution, and their outcomes are measured using the fixed decision and audit sets described in the paper.

In addition, ChatGPT was used to assist with language polishing and presentation refinement of the manuscript. All research questions, methodological design, experimental protocols, implementation, result analysis, and scientific conclusions were developed, reviewed, and verified by the authors.

REPRODUCIBILITY STATEMENT

The main paper describes the evaluation protocol, metrics, and baseline comparisons. Appendix A specifies the benchmark PDEs and their constraints; Appendix B reports the model, training schedule, probes, and LLM settings; Appendix C documents the action space and deterministic execution checks; and Appendix D defines the continuation estimator and delayed-feedback rules. Together, these sections describe how interventions are selected, executed, and evaluated. The anonymized supplementary code contains the implementation and experiment configurations used for the reported results.

REFERENCES

Dohyun Bu, Yujung Byun, and Jongseok Lee. Harmonized cone for feasible and non-conflict directions in training physics-informed neural networks. In *International Conference on Learning Representations*, pp. 112050–112093, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/b5ffdfe47e3e68c0d720ae127f276c94-Paper-Conference.pdf.

Fujun Cao, Xiaobin Guo, Xinzheng Dong, and Dongfang Yuan. wbPINN: Weight balanced physics-informed neural networks for multi-objective learning. *Applied Soft Computing*, 170:112632, 2025.

Arka Daw, Jie Bu, Sifan Wang, Paris Perdikaris, and Anuj Karpatne. Mitigating propagation failures in physics-informed neural networks using retain-resample-release (R3) sampling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7264–7302. PMLR, 2023. URL https://proceedings.mlr.press/v202/daw23a.html.

Bo Gao, Ruoxia Yao, and Yan Li. Physics-informed neural networks with adaptive loss weighting algorithm for solving partial differential equations. *Computers & Mathematics with Applications*, 181:216–227, 2025.

Zhiwei Gao, Liang Yan, and Tao Zhou. Failure-informed adaptive sampling for PINNs. *SIAM Journal on Scientific Computing*, 45(4):A1971–A1994, 2023. doi: 10.1137/22M1527763. URL https://doi.org/10.1137/22M1527763.

Zhongkai Hao, Jiachen Yao, Chang Su, Hang Su, Ziao Wang, Fanzhi Lu, Zeyu Xia, Yichi Zhang, Songming Liu, Lu Lu, and Jun Zhu. PINNacle: A comprehensive benchmark of physics-informed neural networks for solving PDEs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 76721–76774. Curran Associates, Inc., 2024. doi: 10.52202/079017-2442. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/8c63299fb2820ef41cb05e2ff11836f5-Paper-Datasets_and_Benchmarks_Track.pdf.

Ameya D. Jagtap and George Em Karniadakis. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020. doi: 10.4208/cicp.OA-2020-0164. URL https://doi.org/10.4208/cicp.OA-2020-0164.

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, 2020a.

Ameya D. Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020b. doi: 10.1016/j.cma.2020.113028. URL https://doi.org/10.1016/j.cma.2020.113028.

Ana Kaplarević-Mališić, Branka Andrijević, Filip Bojović, Srđan Nikolić, Lazar Krstić, Boban Stojanović, and Miloš Ivanović. Identifying optimal architectures of physics-informed neural networks by evolutionary strategy. *Applied Soft Computing*, 146:110646, 2023.

Ehsan Kharazmi, Zhongqiang Zhang, and George E. M. Karniadakis. hp-VPINNs: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021. doi: 10.1016/j.cma.2020.113547. URL https://doi.org/10.1016/j.cma.2020.113547.

Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in neural information processing systems*, volume 34, pp. 26548–26560, 2021.

Shuning Lin and Yong Chen. Causality-guided adaptive sampling method for physics-informed neural networks solving forward problems of partial differential equations. *Physica D: Nonlinear Phenomena*, 481:134878, 2025. doi: 10.1016/j.physd.2025.134878.

Fei Liu, Tong Xialiang, Mingxuan Yuan, Xi Lin, Fu Luo, Zhenkun Wang, Zhichao Lu, and Qingfu Zhang. Evolution of heuristics: Towards efficient automatic algorithm design using large language model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 32201–32223. PMLR, 2024. URL https://proceedings.mlr.press/v235/liu24bs.html.

Fei Liu, Yiming Yao, Ping Guo, Zhiyuan Yang, Xi Lin, Zhe Zhao, Xialiang Tong, Kun Mao, Zhichao Lu, Zhenkun Wang, Mingxuan Yuan, and Qingfu Zhang. A systematic survey on large language models for algorithm design. *ACM Computing Surveys*, 58(8):218, 2026. doi: 10.1145/3787585.

Qiang Liu, Mengyu Chu, and Nils Thuerey. ConFIG: Towards conflict-free training of physics informed neural networks. In *International Conference on Learning Representations*, pp. 59531–59566, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/94e85561a342de88b559b72c9b29f638-Paper-Conference.pdf.

Ziti Liu, Yang Liu, Xunshi Yan, Wen Liu, Han Nie, Shuaiqi Guo, and Chen-an Zhang. Automatic network structure discovery of physics informed neural networks via knowledge distillation. *Nature Communications*, 16(1):9558, 2025b.

Jiaqi Luo, Yahong Yang, Yuan Yuan, Shixin Xu, and Wenrui Hao. An imbalanced learning-based sampling method for physics-informed neural networks. *Journal of Computational Physics*, 534: 114010, 2025.

Levi D. McClenny and Ulisses M. Braga-Neto. Self-adaptive physics-informed neural networks. *Journal of Computational Physics*, 474:111722, 2023. doi: 10.1016/j.jcp.2022.111722. URL https://doi.org/10.1016/j.jcp.2022.111722.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045. URL https://doi.org/10.1016/j.jcp.2018.10.045.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training PINNs: A loss landscape perspective. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42159–42191. PMLR, 21–27 Jul 2024.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024. doi: 10.1038/s41586-023-06924-6. URL https://doi.org/10.1038/s41586-023-06924-6.

Natarajan Sukumar and Ankit Srivastava. Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks. *Computer Methods in Applied Mechanics and Engineering*, 389:114333, 2022.

Niki van Stein and Thomas Bäck. LLaMEA: A large language model evolutionary algorithm for automatically generating metaheuristics. *IEEE Transactions on Evolutionary Computation*, 29(2): 331–345, 2025. doi: 10.1109/TEVC.2024.3497793. URL https://doi.org/10.1109/TEVC.2024.3497793.

Niki van Stein, Diederick Vermetten, and Thomas Bäck. In-the-loop hyper-parameter optimization for LLM-based automated design of heuristics. *ACM Transactions on Evolutionary Learning and Optimization*, 2025. doi: 10.1145/3731567. URL https://doi.org/10.1145/3731567.

Rouhan Wang and Dan Hu. Non-uniform random walk for adaptive sampling. *Journal of Computational Physics*, 538:114160, 2025.

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, 2021a. doi: 10.1137/20M1318043. URL https://doi.org/10.1137/20M1318043.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of Fourier feature networks: From regression to solving multi-scale PDEs with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021b. doi: 10.1016/j.cma.2021.113938. URL https://doi.org/10.1016/j.cma.2021.113938.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022. doi: 10.1016/j.jcp.2021.110768. URL https://doi.org/10.1016/j.jcp.2021.110768.

Sifan Wang, Bowen Li, Yuhan Chen, and Paris Perdikaris. PirateNets: Physics-informed deep learning with residual adaptive networks. *Journal of Machine Learning Research*, 25(402):1–51, 2024a. URL https://jmlr.org/papers/v25/24-0313.html.

Sifan Wang, Shyam Sankaran, and Paris Perdikaris. Respecting causality for training physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 421: 116813, 2024b. doi: 10.1016/j.cma.2024.116813. URL https://doi.org/10.1016/j.cma.2024.116813.

Yicheng Wang, Xiaotian Han, Chia-Yuan Chang, Daochen Zha, Ulisses Braga-Neto, and Xia Hu. Auto-PINN: Understanding and optimizing physics-informed neural architecture. In *NeurIPS 2023 Workshop on AI for Science*, 2023. URL https://openreview.net/pdf/0aa9eb84e1bcdd8fc45ac1c2da549b55421dbaf3.pdf.

Yifan Wang and Linlin Zhong. NAS-PINN: Neural architecture search-guided physics-informed neural network for solving PDEs. *Journal of Computational Physics*, 496:112603, 2024. doi: 10.1016/j.jcp.2023.112603. URL https://doi.org/10.1016/j.jcp.2023.112603.

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. doi: 10.1016/j.cma.2022.115671. URL https://doi.org/10.1016/j.cma.2022.115671.

Haixu Wu, Huakun Luo, Yuezhou Ma, Jianmin Wang, and Mingsheng Long. RoPINN: Region optimized physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 37, pp. 110494–110532. Curran Associates, Inc., 2024. doi: 10.52202/079017-3508. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/c745bfa5b50544882938ff4f89ff26ac-Paper-Conference.pdf.

Xingyu Wu, Sheng-hao Wu, Jibin Wu, Liang Feng, and Kay Chen Tan. Evolutionary computation in the era of large language model: Survey and roadmap. *IEEE Transactions on Evolutionary Computation*, 29(2):534–554, 2025. doi: 10.1109/TEVC.2024.3506731. URL https://doi.org/10.1109/TEVC.2024.3506731.

Qingpo Wuwu, Chonghan Gao, Tianyu Chen, Yihang Huang, Yuekai Zhang, Jianing Wang, Jianxin Li, Haoyi Zhou, and Shanghang Zhang. PINNsAgent: Automated PDE surrogation with large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 68143–68165. PMLR, 2025. URL https://proceedings.mlr.press/v267/wuwu25a.html.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *International Conference on Learning Representations*, pp. 12028–12068, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/3339f19c5fcee3ad74502947a32be9e6-Paper-Conference.pdf.

Shunyu Yao, Fei Liu, Xi Lin, Zhichao Lu, Zhenkun Wang, and Qingfu Zhang. Multi-objective evolution of heuristic using large language model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(25):27144–27152, 2025.

Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park, and Guojie Song. ReEvo: Large language models as hyper-heuristics with reflective evolution. In *Advances in Neural Information Processing Systems*, volume 37, pp. 43571–43608. Curran Associates, Inc., 2024. doi: 10.52202/079017-1381. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/4ced59d480e07d290b6f29fc8798f195-Paper-Conference.pdf.

Haoran Yin, Anna V Kononova, Thomas Bäck, and Niki van Stein. Controlling the mutation in large language models for the efficient evolution of algorithms. In *Applications of Evolutionary Computation: EvoApplications 2025*, volume 15613 of *Lecture Notes in Computer Science*, pp. 403–417. Springer, 2025. doi: 10.1007/978-3-031-90065-5_25. URL https://link.springer.com/chapter/10.1007/978-3-031-90065-5_25.

Liangliang You, Xin He, Hongduan Tian, Bo Han, Ivor Tsang, and Yew-Soon Ong. LangPINN: From language to physics-informed neural networks via a multi-agent framework. In *ICLR 2026 Workshop on AI with Recursive Self-Improvement*, pp. 1–35, 2026. URL https://openreview.net/forum?id=q5qN3oQ4D1.

Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022.

Bo Zhang and Chao Yang. Discovering physics-informed neural networks model for solving partial differential equations through evolutionary computation. *Swarm and Evolutionary Computation*, 88:101589, 2024.

Jun Zhang and Chensen Ding. Simple yet effective adaptive activation functions for physics-informed neural networks. *Computer Physics Communications*, 307:109428, 2025.

Yabin Zhang and Liang-Jian Deng. Loss-driven dynamic weight and residual transformation in physics-informed neural network. *Neural Networks*, 191:107841, 2025.

Zhiyuan Zhao, Xueying Ding, and B. Aditya Prakash. PINNsFormer: A transformer-based framework for physics-informed neural networks. In *International Conference on Learning Representations*, pp. 38475–38491, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/a6f2763089c0bd8f56006c42f09ee24c-Paper-Conference.pdf.