LOCALIZE, EDIT, AND LEARN: STRUCTURE-AWARE SYMBOLIC REGRESSION FOR NETWORK DYNAMICS

Anonymous authors
Paper under double-blind review

ABSTRACT

Describing network dynamics through the discovery of symbolic equations is critical for understanding complex systems. However, as a network's dynamics are jointly governed by its nodes' local processes and mutual interactions, where both encapsulate diverse symbolic components, existing symbolic regression methods struggle to precisely localize erroneous components and translate diagnostic evidence into controlled structural edits. Furthermore, despite the rise of large language models (LLMs) in evolutionary symbolic regression for network dynamics, a dedicated policy update paradigm is needed to let LLMs learn from historical edits and internalize such capability, instead of merely reusing experience through prompts. We introduce Network Symbolic Patch Evolution (NetSPE), a framework that connects localization, editing, and policy learning for equation generation. Guided by network structural priors, NetSPE uses residual contributions to localize a target component to be updated, and then instructs an LLM to generate constrained local replacements. We further propose Structural Patch Preference Optimization (SPPO), which adapts the editing policy using preferences among alternative edits generated under the same editing state and evaluated against their parent. Experiments on two real-world and one synthetic benchmarks show NetSPE's efficacy in its prediction accuracy, equation quality, and generalizability. Code is available at `https://anonymous.4open.science/r/NetSPE-35F4`.

1 INTRODUCTION

Many complex real-world systems are represented as networks, e.g., transportation, epidemic, and protein (Lin & Ban, 2013; Pastor-Satorras et al., 2015; Tucker et al., 2001). Symbolic regression (SR) for network dynamics therefore seeks analytical equations that characterize such systems. Beyond the functional relationships considered in conventional SR designed for non-networked data, in network dynamics, a node's states evolve through both self-dynamics and connectivity-based interactions (Gao et al., 2016), necessitating dedicated approaches. An existing approach combines neural models of node and edge dynamics with coordinated genetic search (Qiu et al., 2024), while others use network-aware symbolic operators and pretrained neural models to guide formula discovery (Yu et al., 2026; Hu et al., 2025). A major sticking point lies in navigating the space of candidate equation compositions, especially given the need to distinguish self-dynamics and interaction laws from their combined effects in the observation. As candidate expressions become more complex and observations provide limited signals, numerical fit offers an incomplete basis for identifying such laws. This motivates SR methods that combine numerical evidence with contextual knowledge about the system to guide the exploration of plausible expressions.

Large language models (LLMs) offer a promising basis for this combination through their mathematical knowledge, contextual reasoning, and generation capability (Shojaee et al., 2025). Recent approaches extend equation generation into structural refinement through constrained mutations, diagnostic feedback, and memory (Xia et al., 2026b; Pang et al., 2026; Li & Pan, 2026). However, with a focus on conventional SR, existing solutions inevitably face two intertwined challenges when tackling network dynamics. The first one is *translating diagnostic guidance into localized, controlled edits*. In SR, prediction error reflects the outcome-level diagnostic feedback of a searched equation. Yet, in network dynamics, prediction errors are combined effects of self-dynamics and aggregated interactions. Though they may suggest a missing relationship or a promising functional

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.

form, mapping such diagnostic signals to a particular symbolic component, including attributing to the self-dynamics or interaction branch and identifying erroneous subexpressions inside, requires finer-grained diagnostic feedback. As such, effective refinement requires steering the structural edits with a concrete editing target, which is facilitated by nuanced diagnostic feedback that isolates each symbolic component's contribution. After locating the components to edit, the second challenge concerns *internalizing editing experience into an adaptive generation policy*. Most existing refinement frameworks reuse experience through prompts, structure libraries, or operator selection, while typically keeping the underlying LLM parameters fixed (Xia et al., 2026b; Pang et al., 2026; Li & Pan, 2026). In-search policy optimization enables parameter updates using equation quality and constraint-based feedback (Wang et al., 2026), but adapting an equation generator does not by itself define how to reward a specific local replacement conditioned on the context. Thus, a dedicated policy update mechanism that learns from historical edits the optimal way to revise an arbitrary symbolic component is desirable.

To address these challenges, we introduce Network Symbolic Patch Evolution (NetSPE), a framework that localizes editable components, generates promising edits, and adapts its editing policy by comparing alternative edits under the same editing state. The key insight is to combine network structural priors with numerical diagnostic feedback to guide a hierarchical editing procedure, then learn from alternative replacements evaluated within that task. NetSPE represents candidate equations as typed expression trees within domain-grounded skeletons, preserving prescribed composition and aggregation operations. By using residual patterns and subtree-based ablations to quantify component-level contributions, NetSPE first localizes promising editing targets worth extending, replacing, or simplifying. It then guides the LLM to generate local edits at the selected location while preserving the surrounding symbolic structure. This results in a state that contains the parent equation, branch, target term, and intended edits. For policy optimization, we propose Structural Patch Preference Optimization (SPPO) to construct comparative patches by sampling alternative edits for this state, so as to concentrate feedback on the components being edited alone. It constructs positive-negative preferences based on improvements over the parent equation and adapts the patch generation policy through a regularized pairwise objective. By doing so, NetSPE connects context about "where" to edit with internalized knowledge of "how" to edit in SR tasks for network dynamics. Our contributions are threefold:

- We introduce NetSPE, a localize-edit-learn framework that bridges numerical diagnostic feedback and policy learning to facilitate SR for network dynamics.
- We develop a hierarchical editing target localization method and a policy update mechanism SPPO in NetSPE. The former combines network structural priors with residual diagnostics and refitted term-level contributions to guide edits, while the latter learns from alternative patches to internalize component-wise editing capability.
- We evaluate NetSPE on one synthetic and two real-world network dynamic benchmarks, and analyze discovered expressions and search trajectories to examine its predictive behavior and structural refinement process.

2 Related Work

LLM-based Symbolic Regression Traditional symbolic regression (SR) methods typically rely on genetic programming, neural symbolic models, or reinforcement learning to search over a vast symbolic space. Recent studies have instead begun to leverage the scientific priors and program-generation capability of large language models (LLMs) to propose more meaningful equation hypotheses. LLM-SR (Shojaee et al., 2025) first introduces a programming-based paradigm that provides the LLM with contextual information and observed data, represents candidate equations as program skeletons, and exploits the LLM's code-generation capability to construct symbolic expressions. DrSR (Wang et al., 2025) further incorporates explicit reasoning over data characteristics, such as monotonicity and nonlinearity, together with reflection over historical search experience to guide equation generation. SR-Scientist (Xia et al., 2026a) extends the role of the LLM from a passive equation proposer to an autonomous agent capable of analyzing data, executing code, evaluating candidate equations, and conducting multi-step scientific exploration. PiT-PO (Wang et al., 2026) does not treat the LLM as a fixed generator, instead, it updates the generation policy during equation search through reinforcement learning while incorporating physics-informed constraints and

token-level regularization. Despite these advances, most existing LLM-based SR methods target tabular data or low-dimensional dynamical systems without explicitly accounting for the node–edge relational structure of complex network dynamics in their symbolic representations.

Symbolic Regression for Network Dynamics Unlike general symbolic regression, network dynamics involve relational dependencies: the evolution of a node is determined not only by its own state but also by aggregated interactions with its neighbors. Treating all node and neighbor states as ordinary input features therefore leads to a rapidly expanding search space and fails to preserve the underlying relational structure of the system. PI-NDSR (Qiu et al., 2024) models noisy network trajectories with neural dynamics and subsequently combines them with symbolic search to recover interpretable node dynamics. LLC (Hu et al., 2025) uses neural networks to decouple network dynamics into self and pairwise interaction components and applies pretrained symbolic regression to recover their governing equations. ND^2 (Yu et al., 2026) reformulates high-dimensional network dynamics using a structured symbolic representation with node-level and edge-level variables, together with graph-aware operators, thereby substantially constraining the search space of candidate network equations. Existing network symbolic regression methods introduce useful relational inductive biases, but their search procedures largely rely on genetic search or neural models pretrained on large-scale data. As a result, they do not fully exploit the iterative reasoning, feedback-driven refinement, and local structural editing capability offered by modern LLM-based equation search.

LLM-Based Equation Evolution and Structural Refinement LLM-SR demonstrated that LLMs need not generate equations in a one-shot manner, instead, they can iteratively refine symbolic programs by leveraging high-performing candidates from previous search iterations. Building on this paradigm, subsequent work has gradually shifted the focus from LLM-based equation generation toward LLM-based equation editing and evolution. PROAUG (Liu et al., 2026) enables LLMs to actively analyze datasets and acquire richer search context beyond a single scalar fitness score. IGSR (Saveliev et al., 2026) further estimates the marginal contribution of individual terms to generalization performance. Deliberate Evolution (Pang et al., 2026) incorporates structural diagnosis and search memory, separating the generation of equations from the reasoning process that determines how they should be modified. Another line of work focuses more directly on structure-aware equation editing. FunctionEvolve (Xia et al., 2026b) explicitly represents equation structure using abstract syntax trees (ASTs) and performs localized tree mutations rather than repeatedly regenerating the entire equation. RESTART (Li & Pan, 2026) identifies potentially incorrect equation components from residual and unexplained signals and performs targeted structural corrections. However, these methods treat equation structure primarily as a generic symbolic tree, without exploiting the hierarchical semantics specific to network dynamics, such as the distinction between self-dynamics and interaction dynamics.

3 PRELIMINARIES

Symbolic regression for network dynamics. Consider a directed network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where at snapshot t, nodes $i, j \in \mathcal{V}$ have states $\mathbf{x}_i(t)$, $\mathbf{x}_j(t)$, and edge $(j, i) \in \mathcal{E}$ has attributes $\mathbf{e}_{ji}(t)$. Depending on the task, $y_i(t)$ represents a derivative, a state increment, or a time-sensitive attribute of node i. Given training observations $\mathcal{D}_{\text{tr}} = \{\mathcal{G}, \mathbf{X}(t), \mathbf{E}(t), \mathbf{Y}(t)\}_{t\in\mathcal{T}_{\text{tr}}}$, our goal is to discover a symbolic structure q and continuous coefficients $\boldsymbol{\theta}$ that approximate the target $y_i(t)$ through:

$$\hat{y}_i(t) = F_q\Big(\mathbf{x}_i(t), \{\mathbf{x}_j(t), \mathbf{e}_{ji}(t)\}_{j\in\mathcal{N}_i}; \boldsymbol{\theta}\Big), \tag{1}$$

where $\mathcal{N}_i = \{j : (j, i) \in \mathcal{E}\}$ denotes the incoming neighbors of node i. We adopt the standard snapshot-wise prediction: candidate equations receive observed states and attributes at every t and are evaluated against the corresponding targets, without a recursive rollout (Brunton et al., 2016).

Structural priors of network dynamics. Without loss of generality, the dynamics of a network system (Gao et al., 2016; Barzel et al., 2015) commonly consist of two factors: intrinsic node dynamics f_{self} describing the intrinsic evolution of node i; and aggregated pairwise interactions $f_{\text{interaction}}$ measuring the influence of neighbor j on node i, conditioned on their states and edge attributes.

Following ND^2 and LLC (Yu et al., 2026; Hu et al., 2025) in adopting this structural prior for network dynamics, we further abstract the structure of equations to be discovered into two editable

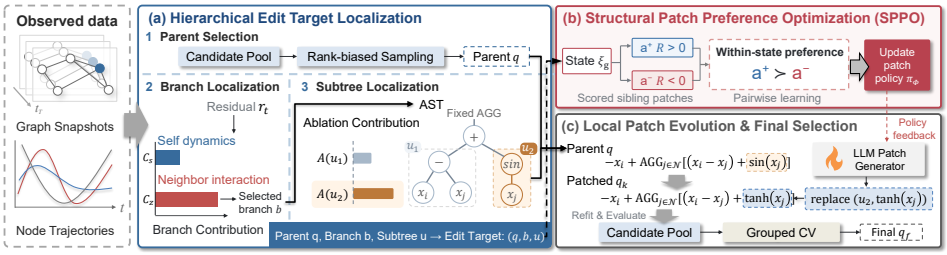


Figure 1: Overview of the NetSPE framework. (a) Hierarchical localization selects a parent expression, branch, and target subtree. (b) SPPO learns within-state patch preferences to improve the patch policy. (c) LLM-generated patches are applied and evaluated to expand the candidate pool, followed by grouped cross-validation and final structure selection.

branches: a self-dynamics branch s_i for intrinsic node dynamics, and a neighbor-interaction branch z_i for aggregated pairwise interactions. Omitting the snapshot index t and coefficients $\boldsymbol{\theta}$ for clarity, their outputs are:

$$s_i = f_{\text{self}}(\mathbf{x}_i), \quad z_i = \text{AGG}_{j\in\mathcal{N}_i} f_{\text{interaction}}(\mathbf{x}_j, \mathbf{e}_{ji}), \tag{2}$$

and the prediction is expressed as $\hat{y}_i = \rho(s_i, z_i)$, where AGG denotes the neighborhood aggregation specified by the structural prior, and ρ denotes the system-specific composition of the branch outputs. The structural priors used are provided in Appendix B.

Typed AST edits. Expression trees are widely used in evolutionary symbolic regression to support component-/subtree-level modifications (McKay et al., 2010; McPhee et al., 2008). For a candidate equation q, let $T_b(q)$ denote the AST of branch $b \in \{s, z\}$. Each open slot in the AST is prescribed with variable scopes and expression types. An editable component, i.e., subtree $u = T_b(q)[v]$ consists of a node v and all its descendants and must lie entirely within a single open slot. Here, a node v represents an operator, a variable, or a constant; the children of an operator node represent the expressions on which it operates. A local edit replaces u with a compatible expression $\tilde{u}$, yielding an updated equation $q' = q[u \leftarrow \tilde{u}]$. Within the selected subtree, operators, constants, variables, and tree structure may change, provided that the replacement uses only variables permitted in that slot, satisfies the typing rules for its operators, and returns the type required at the replacement location. All nodes outside the selected subtree remain unchanged. In particular, edits cannot alter the fixed two-branch skeleton or the neighborhood aggregation. These admissible local edits provide the action space for the localization and policy optimization mechanisms below. Detailed typing rules and validity checks are provided in Appendix C.

4 Methodology

Based on the two-branch representation and typed AST edits defined above, we introduce Network Symbolic Patch Evolution (NetSPE), illustrated in Figure 1. NetSPE addresses two coupled questions in the equation search process through its main components:

- **Where to edit precisely:** Hierarchical Edit Target Localization selects a branch and target component within a parent equation, after which the LLM generates a local patch based on the selected target and editing intent.
- **How to learn better edits:** Structural Patch Preference Optimization (SPPO) improves the patch-generation policy by comparing alternative edits under the same editing state.

These components form an iterative search loop: localization defines the editing state for local patch generation (Sections 4.1 and 4.2), and SPPO uses the resulting feedback to guide subsequent generation (Section 4.3). After search, conservative cross-validation selects the final equation (Section 4.4).

4.1 Hierarchical Edit Target Localization

NetSPE follows an evolutionary search paradigm that iteratively refines candidate equations through local structural edits, requiring a decision about which part of each candidate to revise. The equation-

level loss measures overall predictive error but does not indicate whether editing the self-dynamics branch s_i or the neighbor-interaction branch z_i is more promising, nor which expression within that branch should change. We address this through hierarchical edit target localization: after selecting a parent candidate q, we use residual diagnostics to select a branch b, then assess component-level contributions by refitting ablated versions of q to choose a target symbolic component u for editing.

Parent selection. A promising equation may require several edits before improving, so restricting search to the current best candidate can discard useful intermediate structures. Following LLM-SR and its extensions (Shojaee et al., 2025; Wang et al., 2026), NetSPE maintains multiple candidate islands and samples parents according to their ranks within a selected island. Each candidate's continuous coefficients are fitted by minimizing the mean squared error across snapshots. We then rank candidates using:

$$J(q) = \mathcal{L}_{\text{sel}}(q, \boldsymbol{\theta}_q^*) + \lambda_c |q|_{\text{AST}}, \tag{3}$$

where $\boldsymbol{\theta}_q^*$ denotes the fitted coefficients, $\mathcal{L}_{\text{sel}}$ is the mean squared error on separate selection snapshots t, and $\lambda_c |q|_{\text{AST}}$ penalizes equation complexity measured by AST size (total node count of AST). λ_c controls the strength of the complexity penalty. Both subsets are drawn from $\mathcal{D}_{\text{tr}}$. In short, smaller $J(q)$ scores indicate better candidates.

Residual-guided branch localization. Firstly, NetSPE needs to determine in which branch $b \in \{s, z\}$ a symbolic edit is needed. To facilitate this, we examine whether these errors are more strongly associated with a node's own features or with features describing its interactions with neighbors. These associations help identify whether to edit the self-dynamics branch or the neighbor-interaction branch. In snapshot t, let $r_{it} = y_i(t) - \hat{y}_i(t; q, \boldsymbol{\theta}_q^*)$.

We define two sets of diagnostic features, $\mathcal{H}_s$ and $\mathcal{H}_z$, to help decide which branch to edit. The self-dynamics set $\mathcal{H}_s$ contains the receiver node's own features, while the neighbor-interaction set $\mathcal{H}_z$ contains graph-derived features allowed by the task schema, such as neighbor means, node–neighbor differences, and edge-weighted summaries. We measure how strongly features in each set are associated with the prediction residuals to guide branch selection.

We pool node–snapshot observations from the selection set, denoting r_{it} by r_n with n indexing (i, t) pairs; the aggregate statistics therefore omit node and snapshot indices. For each channel h, we partition its values into empirical quantile bins and measure the residual variance explained by the bin-wise conditional means:

$$V(h) = \frac{\sum_{c \in \mathcal{B}(h)} n_c (\bar{r}_c - \bar{r})^2}{\sum_n (r_n - \bar{r})^2 + \varepsilon}, \tag{4}$$

where n indexes node–snapshot observations, n_c and $\bar{r}_c$ are the size and mean residual of bin c, and $\bar{r}$ is the overall residual mean. This captures nonlinear associations without requiring a parametric residual model.

We select the highest-scoring channel from each set and recompute their marginal scores V_s, V_z on common valid observations. Since self-dynamics and neighborhood features may explain overlapping residual variation, we also compute V_{sz} (see detailed definition in Appendix D.1) using the joint partition of their marginal bins and allocate the explanatory signal symmetrically:

$$C_s = \frac{1}{2}\left[V_s + (V_{sz} - V_z)\right], \quad C_z = \frac{1}{2}\left[V_z + (V_{sz} - V_s)\right]. \tag{5}$$

Each contribution averages its standalone explanatory value and its incremental value given the other channel. As such, these scores guide editing opportunities in the current candidate q by identifying the underperforming branch.

Branch selection initially follows the residual contributions. As search proceeds, we combine them with historical editing utility while retaining exploration of each legal branch. This allows observed edit outcomes to complement residual associations. The entire branch routing procedure is specified in Appendix D.2.

Subtree-based component localization. Residual diagnostics indicate which branch to revise, but not which symbolic component within that branch should change. We assess each legal subtree $u \in \mathcal{U}(q, b)$ by removing it from the equation and measuring the resulting prediction error. We

assess subtree contributions using a removal-based criterion (Saveliev et al., 2026), measuring the change in selection loss after neutral replacement and coefficient refitting. Crucially, after removing a component, retaining the original coefficients of remaining components for downstream testing can lead to biased evaluation of a symbolic component's structural importance, as the current coefficients are not optimized for this ablated equation. Let $q \setminus u$ denote the candidate obtained by removing the symbolic component u. As u is essentially a subtree in the AST expression of the equation, this ensures the removal lead to a legal ablated equation. By refitting the coefficients of the ablated equation, we define the ablation contribution as:

$$A(u) = \frac{\mathcal{L}_{\text{sel}}(q \setminus u, \boldsymbol{\theta}^*_{-u}) - \mathcal{L}_{\text{sel}}(q, \boldsymbol{\theta}^*_q)}{\mathcal{L}_{\text{sel}}(q, \boldsymbol{\theta}^*_q) + \varepsilon}, \quad \boldsymbol{\theta}^*_{-u} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{fit}}(q \setminus u, \boldsymbol{\theta}). \tag{6}$$

A positive $A(u)$ indicates that component u contributes predictive information that the refitted remainder does not recover. A non-positive value identifies a potential replacement or simplification target, since its removal does not increase selection error.

We first sample an edit intent m, exploring available intents during warmup and favoring those with more frequent and larger past improvements afterward. The selected intent determines how we choose the target subtree u: for **grow**, we choose the eligible subtree with the highest $A(u)$; for **replace**, the one with the smallest $|A(u)|$; for **simplify**, the one with the lowest $A(u)$; and for **motif**, we prioritize an editable slot root. The resulting state $\xi = (q, b, m, u)$ defines a localized generation task while preserving the composition ρ and outer aggregation. Generating alternative patches from the same state makes their outcomes comparable for downstream SPPO.

4.2 Local Structural Patch Evolution

Given the editing state $\xi = (q, b, m, u)$, the remaining task is to propose a useful replacement of the selected component. Local generation preserves the surrounding structure and associates each proposal with a specific structural change, providing the basis for subsequent patch-based comparison.

We construct a prompt x_ξ containing the current equation, task schema, structural constraints, residual diagnostics, and the selected component with its ablation contribution. The edit intent m specifies whether to grow, replace, simplify, or rewrite a local motif, while the available variables and operators constrain the replacement. The LLM policy generates an action a by proposing a patch $\tilde{u}(a)$ that replaces u: $a \sim \pi_\phi(\cdot \mid x_\xi)$. The parent q is then updated into a child $q' = q[u \leftarrow \tilde{u}(a)]$. All four intents are implemented through the same typed AST edits defined in the preliminaries.

Each child undergoes structural and numerical validity checks, followed by coefficient refitting and evaluation using $J(q')$ in (Eq. 3). Valid, distinct children are eligible for population insertion even when they do not immediately improve on the parent, so as to balance exploration and exploitation. Their outcomes inform subsequent localization and form the feedback of SPPO. Patch serialization, parameter canonicalization, and auxiliary proposals based on atomic actions and deterministic motifs are described in Appendix D.3; these auxiliary proposals are excluded from policy training.

4.3 Structural Patch Preference Optimization

Localization identifies where to edit, while improving subsequent edits requires learning from their outcomes. Since rewards from different parents or target components reflect different editing difficulties, SPPO compares alternative patches generated under the same editing state and updates the policy to favor successful ones.

State patch preferences. Inspired by the grouped sampling design of GRPO (Shao et al., 2024), we organize candidate patches into groups sharing the same editing state. For each editing state ξ_g, we fix the complete prompt x_g and sample G sibling patches from the rollout policy (exact grouping and sample handling are detailed in Appendix D.4): $a_{g,1}, \ldots, a_{g,G} \sim \pi_{\text{old}}(\cdot \mid x_g)$.

All siblings modify the same target component of parent q_g and are evaluated under shared search and coefficient-fitting conditions. For each valid child $q_{g,k}$, we define:

$$R_{g,k} = \text{clip}_{[-1,1]} \left(\frac{J(q_g) - J(q_{g,k})}{|J(q_g)| + |J(q_{g,k})|} \right), \tag{7}$$

where J as per Eq. 3 balances prediction error and structural complexity. Accepted patches with $R_{g,k} > 0$ form the positive set $\mathcal{P}_g$, while proposals with $R_{g,k} < 0$ form the negative set $\mathcal{N}_g$. Only groups g containing both sets, which we term the informative groups $\mathcal{I}$, contribute to training.

Pairwise policy optimization. Editing difficulty varies across parents and target components, making reward magnitudes difficult to compare across states. We therefore learn from positive–negative patch preferences within the same editing state. While GRPO derives advantages from rewards normalized within each group (Shao et al., 2024), SPPO constructs positive-negative patch pairs according to improvement over the parent equation and optimizes their relative policy scores. Inspired by the pairwise log-ratio objective of DPO (Rafailov et al., 2023) and the length normalization used in SimPO (Meng et al., 2024), we measure the length-normalized change in completion log-likelihood relative to the rollout policy:

$$\Delta_\phi(a \mid x) = \frac{1}{|a|} \sum_{j=1}^{|a|} \log \frac{\pi_\phi(a_j \mid x, a_{<j})}{\pi_{\text{old}}(a_j \mid x, a_{<j})}. \tag{8}$$

Unlike standard DPO, SPPO defines its pairwise score relative to the rollout policy that generated the candidate patches. Length normalization reduces bias from differences in completion length. For each group $g \in \mathcal{I}$, the pairwise objective is:

$$\mathcal{L}_{\text{pair}}^{(g)} = \frac{1}{|\mathcal{P}_g||\mathcal{N}_g|} \sum_{p \in \mathcal{P}_g} \sum_{n \in \mathcal{N}_g} \text{softplus}\left(-\left[\Delta_\phi(a_{g,p} \mid x_g) - \Delta_\phi(a_{g,n} \mid x_g)\right]\right). \tag{9}$$

This objective encourages a larger increase in the normalized policy score of a positive patch than of a negative sibling. Sharing the editing state ensures that each preference compares alternative replacements for the same structural location. We average the pairwise objective over informative group $\mathcal{I}$: $\mathcal{L}_{\text{pair}} = \frac{1}{|\mathcal{I}|} \sum_{g \in \mathcal{I}} \mathcal{L}_{\text{pair}}^{(g)}$. During optimization, we additionally apply KL divergence regularization toward the fixed base policy.

4.4 Final Structure Selection

To reduce sensitivity to the fixed search split, we reassess a small, diverse candidate pool, including the best searched structure q_s, using repeated snapshot-grouped cross-validation within $\mathcal{D}_{\text{tr}}$. Each fold holds out complete snapshots and refits candidate coefficients on the remaining folds. Candidates are ranked by pooled out-of-fold NMSE with a penalty for variability across repeats. The top-ranked candidate replaces q_s only when its paired improvement exceeds a variability-based threshold and remains positive under snapshot-normalized error; otherwise, q_s is retained, with the resulting structure denoted by q_f. Candidate-pool construction, scoring, and replacement criteria are detailed in Appendix D.5.

5 Experiments

We evaluate NetSPE through three main experiments: (1) quantitative comparisons with baselines on one synthetic and two real-world benchmarks of network dynamics to assess its overall effectiveness; (2) an ablation study to examine the contribution of each key component; and (3) qualitative analyses through case studies, comparing discovered equations with the ground-truth formulas and visualizing the search trajectories. Detailed experimental setups are provided in Appendix E, while additional experiments and supplementary analyses are presented in Appendix F.

Main Results. Table 1 reports the overall performance on three benchmarks. Each benchmark problem is evaluated over three independent runs, and we report the mean results across all problems within each benchmark and standard deviation. These quantitative results show that NetSPE achieves the best or second-best performance in nearly all settings, with the majority being the best, demonstrating consistently strong performance across different network dynamics. Beyond prediction accuracy, NetSPE also shows strong structural recovery, achieving SA scores of 56.7% and 50.0% with Llama3.1-8B-Instruct and Qwen3-4B-Instruct, substantially outperforming other LLM-based baselines.

Table 1: Performance comparison on three datasets. Bold and underlined values indicate the best and second-best mean results within each group, respectively.

Methods	Synthetic			Bacteria Growth		Epidemic Transmission	
	NMSE ↓	$ACC_{0.1}$ ↑	SA (%) ↑	NMSE ↓	$ACC_{0.1}$ ↑	NMSE ↓	$ACC_{0.1}$ ↑
Llama3.1-8B-Instruct							
$ND^{2\dagger}$	$\underline{0.0437 \pm 0.0565}$	$\underline{69.25 \pm 36.09}$	$\underline{53.3}$	1.0672 ± 0.1981	28.78 ± 18.35	0.3819 ± 0.2710	21.52 ± 8.25
LLC†	0.4246 ± 0.3585	19.98 ± 21.90	3.3	1.0695 ± 0.0615	5.43 ± 6.73	0.3640 ± 0.3623	17.89 ± 10.23
LLM-SR	0.3346 ± 0.2912	11.42 ± 4.52	0.0	$\underline{0.9909 \pm 0.1043}$	2.54 ± 3.42	0.3991 ± 0.2953	15.02 ± 9.08
FunctionEvolve	0.2918 ± 0.2963	15.17 ± 8.17	0.0	1.2825 ± 0.8389	$\mathbf{30.99 \pm 19.96}$	0.3820 ± 0.2680	$\underline{23.27 \pm 5.71}$
PiT-PO	0.3598 ± 0.2858	11.78 ± 5.28	0.0	0.9988 ± 0.0712	2.51 ± 1.05	0.3710 ± 0.2858	15.63 ± 7.10
IGSR	0.5130 ± 0.2066	9.26 ± 4.78	0.0	0.9938 ± 0.0554	1.28 ± 1.07	$\underline{0.3278 \pm 0.2650}$	14.53 ± 7.52
RESTART	0.3314 ± 0.2861	12.15 ± 4.42	0.0	1.0094 ± 0.0634	1.25 ± 0.62	0.3848 ± 0.3043	14.89 ± 7.02
NetSPE	$\mathbf{0.0084 \pm 0.0186}$	$\mathbf{83.69 \pm 31.05}$	**56.7**	$\mathbf{0.8256 \pm 0.0834}$	$\underline{29.24 \pm 18.69}$	$\mathbf{0.2395 \pm 0.1451}$	$\mathbf{25.56 \pm 6.39}$
Qwen3-4B-Instruct							
$ND^{2\dagger}$	$\underline{0.0437 \pm 0.0565}$	$\underline{69.25 \pm 36.09}$	**53.3**	1.0672 ± 0.1981	$\underline{28.78 \pm 18.35}$	0.3819 ± 0.2710	21.52 ± 8.25
LLC†	0.4246 ± 0.3585	19.98 ± 21.90	3.3	1.0695 ± 0.0615	5.43 ± 6.73	0.3640 ± 0.3623	17.89 ± 10.23
LLM-SR	0.5771 ± 0.3340	8.56 ± 5.71	0.0	1.0081 ± 0.0626	1.93 ± 2.00	0.3896 ± 0.3514	10.18 ± 5.00
FunctionEvolve	0.2854 ± 0.2744	14.99 ± 7.20	0.0	1.6913 ± 1.2896	20.40 ± 12.91	$\underline{0.2644 \pm 0.1601}$	$\underline{22.21 \pm 4.71}$
PiT-PO	0.3403 ± 0.2826	12.40 ± 5.25	0.0	1.0207 ± 0.0541	6.59 ± 14.35	0.3893 ± 0.3023	17.59 ± 5.30
IGSR	0.5039 ± 0.2347	8.62 ± 5.49	0.0	0.9966 ± 0.0567	1.18 ± 0.79	0.3526 ± 0.2853	14.24 ± 7.49
RESTART	0.4427 ± 0.2871	10.05 ± 4.87	0.0	$\underline{0.9862 \pm 0.0770}$	11.48 ± 10.03	0.3967 ± 0.3130	13.60 ± 6.14
NetSPE	$\mathbf{0.0092 \pm 0.0163}$	$\mathbf{80.37 \pm 30.49}$	$\underline{50.0}$	$\mathbf{0.8759 \pm 0.1421}$	$\mathbf{28.92 \pm 19.11}$	$\mathbf{0.2360 \pm 0.1608}$	$\mathbf{24.28 \pm 8.35}$

† ND^2 and LLC do not use an LLM, their results are therefore identical across backbone groups.

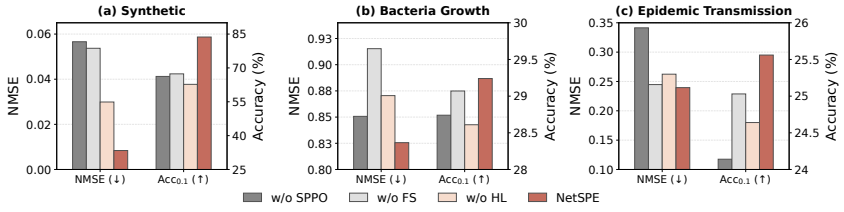


Figure 2: Ablation study on three benchmarks. We report NMSE (left axis) and $Acc_{0.1}$ (right axis).

Moreover, NetSPE maintains consistently strong performance when switching the LLM backbone, suggesting that its effectiveness is not tied to a specific backbone. These results demonstrate the robustness and general applicability of our framework across different LLMs. Complete problem-level results are provided in Appendix F.1, with additional out-of-distribution experiments in Appendix F.3 and test-error threshold coverage results in Appendix F.4.

Ablation Study. We compare NetSPE with three variants: without SPPO updates (w/o SPPO), without final structure selection (w/o FS), and with random rather than hierarchical edit target localization (w/o HL). As shown in Figure 2, the full method achieves the lowest NMSE and the highest $Acc_{0.1}$ across all three datasets. The largest differences appear on Synthetic, where all ablated variants exhibit higher prediction error and lower accuracy than the full method. The relative impact of the ablated configurations varies across datasets: w/o FS has the highest NMSE on Bacteria Growth, whereas w/o SPPO shows the largest degradation in both metrics on Epidemic Transmission. These results favor the complete configuration, with more pronounced differences on Synthetic and smaller

Table 2: Representative equations for Synthetic FHN, WC, and GR.

Method	Synthetic FHN	Synthetic WC	Synthetic GR
Ground Truth	$x_i - x_i^3 - y_i - \frac{1}{d_i}\sum_j (x_j - x_i)$	$-x_i + \sum_j \sigma(a_1(x_j - a_2))$	$a_1 - a_2 x_i + a_3 \sum_j \rho(x_j/a_4, 2)$
LLM-SR	$a_1 x_i^2 y_i - a_2 x_i^2 + a_3 x_i y_i^2$ $- a_4 x_i + a_5 y_i^2 - a_6 y_i + a_7 \sin x_i$ $- a_8 \sin y_i - a_9 \cos x_i + a_{10} \cos y_i$	$- a_1 x_i^5 + a_2 x_i^4 - a_3 x_i^3 - a_4 x_i^2$ $+ a_5 x_i + a_6 e^{a_7 x_i} + a_8 \sin(a_9 x_i) + a_{10}$	$a_1 x_i^2 - a_2 x_i + a_3 \sin(a_4 x_i - a_5)$ $+ a_6 - \frac{a_7}{a_8 x_i^2 + a_9}$
ND^2	$x_i - y_i - x_i^3 - a_1 \sum_j [x_j - (x_i + x_i(y_j - y_j))]$	$-x_i + \sum_j \sigma(a_1(x_j - a_2))$	$x_i - a_1(x_i + x_i) + a_2 \sum_j \rho((a_3 x_j)^2, a_4)$
NetSPE	$x_i - x_i^3 - y_i - a_1 \sum_j (x_j - x_i)$	$-x_i + \sum_j \sigma(a_1(x_j - a_2))$	$a_1 - a_2 x_i + a_3 \sum_j \tanh(a_4 \rho(a_5 x_j, 2))$ $+ a_6 \sum_j \cos(a_7(x_j - x_i))$

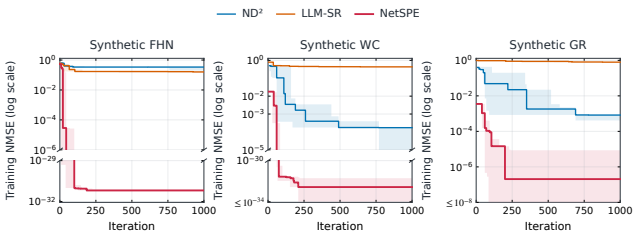


Figure 3: Training NMSE on FHN, WC, and GR. Lines show medians over three runs; shading indicates min–max ranges.

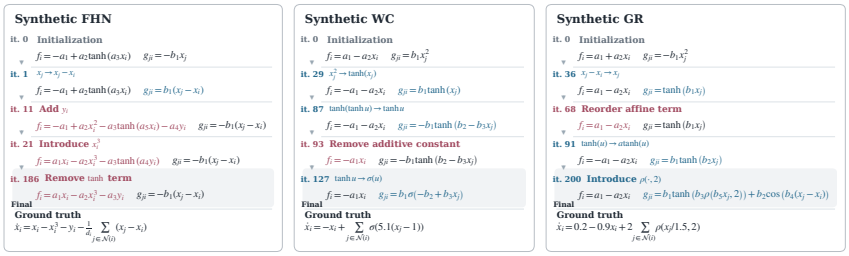


Figure 4: NetSPE search trajectories on FHN, WC, and GR. Red and blue denote self-dynamics and interaction partitions edits, respectively.

accuracy gains on the two real-world datasets. An additional ablation study comparing NetSPE with LLM-based baselines augmented with structural priors is provided in Appendix F.2.

Case Study. We qualitatively analyze NetSPE through three case studies: FHN and WC, where it successfully recovers the governing equations, and GR, where exact recovery fails. Table 2 compares representative equations from NetSPE, LLM-SR, and ND^2 against the ground truth, with fitted coefficients represented symbolically. Figure 3 shows that NetSPE reaches lower median training NMSE than both baselines on all three problems, including GR, demonstrating its ability to find accurate models even when exact structural recovery is not achieved. Figure 4 shows selected representative stages of NetSPE's search, including initialization, key edits to self-dynamics and interactions, and the final equations. On FHN, it introduces the cubic self term and simplifies the self-dynamics while recovering diffusive coupling; on WC, it removes an unnecessary local constant and identifies sigmoidal coupling. These examples highlight the benefit of explicitly searching both components of network dynamics. On GR, NetSPE captures the correct affine self-dynamics and a Hill-based interaction with a linearly scaled input, closer to the ground-truth form than the squared input used by ND^2. Although an additional $\tanh$ transformation and a small cosine residual prevent exact recovery, the resulting model achieves lower NMSE. Appendix F.5 provides the complete generated equation structures and numerical coefficients for all ten synthetic problems, covering all methods.

6 Conclusion

In this work, we introduce NetSPE, a framework for LLM-based symbolic regression of network dynamics that connects structural diagnosis, local equation editing, and policy learning. Rather than repeatedly regenerating entire equations, NetSPE leverages network structural priors to hierarchically localize erroneous components and performs constrained structural patches on targeted components. We further propose Structural Patch Preference Optimization (SPPO), which learns from relative preferences among alternative patches under the same editing state, enabling the LLM to progressively internalize effective editing strategies during equation search. Experiments across synthetic and real-world network dynamics benchmarks demonstrate that NetSPE consistently achieves strong predictive accuracy and structural recovery across different LLM backbones. These results suggest that explicitly coupling where to edit, how to edit, and how to learn from previous edits provides an effective paradigm for LLM-driven equation discovery in network dynamics systems.

REFERENCES

Baruch Barzel, Yang-Yu Liu, and Albert-László Barabási. Constructing minimal models for complex system dynamics. *Nature communications*, 6(1):7186, 2015.

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

Miles Cranmer. Interpretable machine learning for science with pysr and symbolicregression. jl. *arXiv preprint arXiv:2305.01582*, 2023.

Jianxi Gao, Baruch Barzel, and Albert-László Barabási. Universal resilience patterns in complex networks. *Nature*, 530(7590):307–312, 2016.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jiao Hu, Jiaxu Cui, and Bo Yang. Learning interpretable network dynamics via universal neural symbolic regression. *Nature Communications*, 16(1):6226, 2025.

Jiliang Hu, Daniel R Amor, Matthieu Barbier, Guy Bunin, and Jeff Gore. Emergent phases of ecological diversity and dynamics mapped in microcosms. *Science*, 378(6615):85–89, 2022.

Yunlun Li and Sinno Jialin Pan. Robust equation structure learning with adaptive refinement. In *International Conference on Learning Representations*, volume 2026, pp. 70029–70053, 2026.

Jingyi Lin and Yifang Ban. Complex network topology of transportation systems. *Transport reviews*, 33(6):658–685, 2013.

Hao Liu, Xiao-Wen Yang, Atharva Sehgal, Yixin Wang, Lan-Zhe Guo, Yu-Feng Li, and Yisong Yue. Programmatic context augmentation for llm-based symbolic regression. *arXiv preprint arXiv:2605.03101*, 2026.

Robert I McKay, Nguyen Xuan Hoai, Peter Alexander Whigham, Yin Shan, and Michael O'neill. Grammar-based genetic programming: a survey. *Genetic Programming and Evolvable Machines*, 11(3):365–396, 2010.

Nicholas Freitag McPhee, Brian Ohs, and Tyler Hutchison. Semantic building blocks in genetic programming. In *European Conference on Genetic Programming*, pp. 134–145. Springer, 2008.

Yu Meng, Mengzhou Xia, and Danqi Chen. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024.

Xinyu Pang, Zhanke Zhou, Xuan Li, Fangrui Lv, Shanshan Wei, Sen Cui, Bo Han, and Changshui Zhang. Deliberate evolution: Agentic reasoning for sample-efficient symbolic regression with llms. *arXiv preprint arXiv:2606.04360*, 2026.

Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925–979, 2015.

Haiquan Qiu, Shuzhi Liu, and Quanming Yao. Neural symbolic regression of complex network dynamics. *arXiv preprint arXiv:2410.11185*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Evgeny S Saveliev, Samuel Holt, Nabeel Seedat, David L Bentley, Jim Weatherall, and Mihaela van der Schaar. Influence-guided symbolic regression: Scientific discovery via llm-driven equation search with granular feedback. *arXiv preprint arXiv:2605.29184*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan Reddy. Llm-sr: Scientific equation discovery via programming with large language models. In *International Conference on Learning Representations*, volume 2025, pp. 16054–16085, 2025.

Chandra L Tucker, Joseph F Gera, and Peter Uetz. Towards an understanding of complex protein networks. *Trends in cell biology*, 11(3):102–106, 2001.

Boxiao Wang, Kai Li, Tianyi Liu, Chen Li, Junzhe Wang, Yifan Zhang, and Jian Cheng. Llm-based scientific equation discovery via physics-informed token-regularized policy optimization. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 12195–12206, 2026.

Runxiang Wang, Boxiao Wang, Kai Li, Yifan Zhang, and Jian Cheng. Drsr: Llm based scientific equation discovery with dual reasoning from data and experience. *arXiv preprint arXiv:2506.04282*, 2025.

Shijie Xia, Yuhan Sun, and Pengfei Liu. Sr-scientist: Scientific equation discovery with agentic ai. In *International Conference on Learning Representations*, volume 2026, pp. 75787–75811, 2026a.

Zeyu Xia, Jun Zhu, and Dong Yan. Functionevolve: Structure-guided symbolic regression with llms. *arXiv preprint arXiv:2606.07704*, 2026b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zihan Yu, Jingtao Ding, and Yong Li. Discovering network dynamics with neural symbolic regression. *Nature Computational Science*, 6(2):156–168, 2026.

A LLM USAGE

We used Large Language Model to improve writing and clarity, discover relevant literature and technical documentation, and assist with method implementation and result analysis. The authors remain fully responsible for all content, including AI-assisted text and artifacts.

B STRUCTURAL PRIORS FOR NETWORK DYNAMICS

Following the modeling principle adopted in ND^2 and LLC (Yu et al., 2026; Hu et al., 2025), we instantiate three structural priors for network dynamics. Each prior is included in the LLM's patch-generation prompt as an equation skeleton with explicitly marked editable slots, specifying where local replacements are permitted.

B.1 SYNTHETIC NETWORK DYNAMICS

For synthetic dynamics, we use an additive composition:

$$\begin{aligned} s_i &= f_{\text{self}}(\mathbf{x}_i), \qquad z_i = \text{AGG}_{j\in\mathcal{N}_i}\, f_{\text{interaction}}(\mathbf{x}_j, \mathbf{e}_{ji}), \\ \hat{y}_i &= \rho(s_i, z_i, \mathbf{x}_i) = s_i + z_i. \end{aligned} \tag{10}$$

Here, f_{self} describes intrinsic node dynamics, and $f_{\text{interaction}}$ describes the pairwise influence of neighbor j on node i. Both expressions are discovered through local edits, while the additive composition and outer aggregation remain fixed.

Table 3: Variable scopes and output types of editable slots. Scalar coefficients are available in all slots. Scalar outputs are broadcast to the indicated output domain.

Editable slot	Permitted inputs	Output type
Self / correction	Receiver-node features $\mathbf{x}_i$	V
Interaction	Edge-wise receiver and sender features $\mathbf{x}_i, \mathbf{x}_j$, and edge attributes $\mathbf{e}_{ji}$	E
Readout, if open	Local features $\mathbf{x}_i$, aggregated message z_i, and permitted node-level graph summaries	V

B.2 Bacterial dynamics

For bacterial dynamics, we combine a generalized Lotka–Volterra (GLV)-style self term with aggregated interactions:

$$\begin{aligned} s_i &= f_{\mathrm{GLV}}(\mathbf{x}_i), \qquad z_i = \mathrm{AGG}_{j\in\mathcal{N}_i}\, f_{\mathrm{interaction}}(\mathbf{x}_j, \mathbf{e}_{ji}), \\ \widehat{\Delta x_i} &= \rho(s_i, z_i, \mathbf{x}_i) = x_i(s_i + z_i). \end{aligned} \tag{11}$$

Here, $f_{\mathrm{GLV}}(\mathbf{x}_i) = c_0 + g_i + c_1 x_i + f_{\mathrm{corr}}(\mathbf{x}_i)$, where $c_0 + g_i$ represents node-specific intrinsic growth, $c_1 x_i$ captures self-regulation, and f_{corr} is an editable local correction. The interaction expression is also editable, while ρ preserves the outer factor x_i, enforcing zero predicted change at zero abundance. Continuous coefficients and node-specific growth offsets are estimated during coefficient fitting.

B.3 Epidemic dynamics

For epidemic dynamics, aggregated neighbor information is combined with the current node state through a readout:

$$\begin{aligned} z_i(t) &= \mathrm{AGG}_{j\in\mathcal{N}_i}\, f_{\mathrm{interaction}}(\mathbf{x}_j(t), \mathbf{e}_{ji}(t)), \\ \hat{x}_i(t+1) &= \rho(z_i(t), \mathbf{x}_i(t)) = f_{\mathrm{readout}}(\mathbf{x}_i(t), z_i(t)). \end{aligned} \tag{12}$$

Local-state dependence is represented within the readout, so this structural prior does not require a separate self-dynamics expression s_i. The interaction and readout expressions are editable, while the outer aggregation remains fixed. The readout may additionally access schema-permitted graph summaries. A local variant omits neighbor information and predicts the next state using $f_{\mathrm{local}}(\mathbf{x}_i(t))$.

C Typed AST Representation and Patch Validation

Following the node–edge distinction in ND^2 (Yu et al., 2026), we represent each editable expression as a typed AST. Let V, E, and S denote node-level, edge-level, and scalar expressions, respectively. Pointwise unary and binary operators preserve the expression type:

$$f : \tau \to \tau, \qquad \circ : \tau \times \tau \to \tau, \qquad \tau \in \{\mathsf{V}, \mathsf{E}, \mathsf{S}\}, \tag{13}$$

where f and $\circ$ are drawn from the permitted operator set. Scalar constants and coefficients can be broadcast to either the node or edge domain. Receiver/sender mappings convert node features to edge-level values, while neighborhood aggregation converts edge-level expressions back to node-level values:

$$\mathrm{recv}, \mathrm{send} : \mathsf{V} \to \mathsf{E}, \qquad \mathrm{AGG} : \mathsf{E} \to \mathsf{V}. \tag{14}$$

These mappings describe fixed operations supplied by the evaluation framework, rather than operators that can be inserted into an editable slot. Interaction inputs are already mapped to the edge domain, and the fixed outer aggregation is applied after the interaction expression is evaluated. Direct pointwise arithmetic between node-level and edge-level expressions is not permitted.

For a local patch $q' = q[u \leftarrow \tilde{u}]$, let q'_s denote the resulting expression in the edited slot s, whose required output domain is $\tau_s \in \{\mathsf{V}, \mathsf{E}\}$. The patch must satisfy:

$$\mathrm{type}(q'_s) \in \{\tau_s, \mathsf{S}\}, \qquad \mathrm{Vars}(\tilde{u}) \subseteq \mathcal{V}_{\mathrm{allowed}}(s), \tag{15}$$

where scalar outputs are broadcast to the slot's required domain, and $\mathcal{V}_{\text{allowed}}(s)$ is the data-variable scope of the enclosing slot. Type compatibility is checked on the resulting slot expression; the replacement subtree need not have exactly the same type as the original subtree. The replacement must lie within one open slot and respect the permitted operators and structural-size limits. The fixed equation skeleton and outer aggregation remain unchanged. A graph readout must retain dependence on z_i or a permitted graph summary.

Each child undergoes syntax, type, structural, and numerical checks. Parameters are canonicalized and renumbered before coefficient fitting, and canonical structure hashes are used for duplicate detection. Invalid or numerically non-finite candidates are rejected.

D Implementation Details of NetSPE

D.1 Joint Residual Partition

For the selected self and neighbor channels, let a_n and d_n denote their marginal quantile-bin assignments on common valid observations. We form joint cells (a, d) and compute:

$$V_{sz} = \frac{\sum_{a,d} n_{ad}(\bar{r}_{ad} - \bar{r})^2}{\sum_n (r_n - \bar{r})^2 + \varepsilon}, \tag{16}$$

where n_{ad} and $\bar{r}_{ad}$ are the number of observations and mean residual in joint cell (a, d), and $\bar{r}$ is the overall residual mean. The joint partition uses the same marginal bin assignments as V_s and V_z. We reduce bin resolution until occupancy, coverage, and refinement checks are satisfied. If no reliable joint partition is available, branch routing falls back to the best marginal channel scores.

D.2 Branch Routing

Branch routing chooses between self-dynamics and interaction edits. For a parent q, let $\mathcal{B}_q$ denote its nonempty set of legal editable branches. Each branch b has a residual contribution C_b and a historical utility U_b. The former provides a residual-based editing signal; the latter summarizes the success rate and mean relative objective improvement of previous successful edits. Success requires both improving the parent and ranking among the leading candidates in the corresponding island.

Let N_b denote the number of recorded editing attempts for branch b, including failed and duplicate proposals. We set $N_{\text{warm}} = 8$, so historical utility is used only after both branches have accumulated at least eight attempts. During this initial stage, residual contributions are considered nearly tied when $|C_s - C_z| < \tau_{\text{tie}}$. These fixed settings control the history requirement and tie-breaking tolerance of the routing procedure.

Initially, routing follows residual contributions, alternating between branches when their scores are nearly tied. Once both branches have sufficient editing history, it samples from a mixture of residual contributions and historical utilities, with uniform exploration. Coverage and stagnation safeguards can override this choice, with coverage taking priority, as summarized in Algorithm 1.

Coverage is measured over recent branch selections and takes priority over stagnation avoidance. Editing utility and stagnation are measured over proposal outcomes, including unsuccessful attempts. Histories accumulate across parents within a search. For SPPO, each sibling group shares one branch choice; its editing outcomes update the histories after group evaluation.

D.3 Atomic actions and deterministic motifs

Let $\mathcal{A}_\sigma(q, u)$ and $\mathcal{M}_\sigma(u)$ denote atomic and motif replacements instantiated from schema σ at an editable subtree u. Both produce local patches $q' = q[u \leftarrow \tilde{u}], \tilde{u} \in \mathcal{A}_\sigma(q, u) \cup \mathcal{M}_\sigma(u)$, subject to the same patch validation and coefficient fitting as LLM proposals. The template families encode structural priors using elementary operators and node–edge constructions common in symbolic regression and network-dynamics discovery (Cranmer, 2023; Yu et al., 2026). They are instantiated from the available variables, feature domains, and permitted operators without accessing benchmark ground-truth equations. The templates provide candidate structures, whose coefficients and predictive performance are evaluated using training data.

Algorithm 1 Residual- and history-guided branch routing

Require: Legal branches $\mathcal{B}_q$, scores and counts $\{C_b, U_b, N_b\}_{b\in\mathcal{B}_q}$, previous choice b_{prev}, and branch histories
Ensure: Selected branch b

if $|\mathcal{B}_q| = 1$ **then**
  return the sole legal branch
end if
if $\min_{k\in\mathcal{B}_q} N_k < N_{\text{warm}}$ **then**
  $b \leftarrow \arg\max_{k\in\mathcal{B}_q} C_k$
  if $|C_s - C_z| < \tau_{\text{tie}}$ and $b_{\text{prev}} \in \mathcal{B}_q$ **then**
    Select $b \in \mathcal{B}_q \setminus \{b_{\text{prev}}\}$
  end if
else
  Clip negative residual contributions to zero
  Normalize $\{C_b\}$ and $\{U_b\}$ separately
  Use uniform weights for any zero-sum signal
  Mix both normalized signals with uniform exploration to obtain $\mathbf{p}$
  Sample $b \sim \text{Categorical}(\mathbf{p})$
end if
if $\mathcal{B}_{\text{cov}} \neq \emptyset$ **then**
  Select $b \in \mathcal{B}_{\text{cov}}$
else if recent edits to b show no parent improvement and high duplication **then**
  Set b to the other legal branch
end if
return b

Atomic actions. Atomic actions construct small typed edits through additive updates $\tilde{u} = u + t$ or nonlinear replacements: $t = cx_i$ or $c(x_j - x_i), \tilde{u} = \phi(c_1 v + c_2)$, where the additive terms are representative node- and edge-level examples, respectively. Here v is a schema-compatible feature, c, c_1, c_2 are fitted coefficients, and $\phi \in \{\sin, \tanh, \text{sigmoid}\}$. Other atomic terms include squares and feature products.

Deterministic motifs. Deterministic motifs instantiate predefined, slot-specific patterns from the active schema. The self-dynamics, interaction, and readout templates are $x_i^2, x_j - x_i, x_i\,\text{sigmoid}(cz_i)$, with edge-weighted variants when edge attributes are available. These templates serve as prompt examples and directly generate replacements without an LLM call. Direct motif candidates are retained only if they improve the parent or enter the island's leading ranks.

D.4 Exact grouping and sample handling in SPPO

For each state $\xi_g = (q_g, b_g, m_g, u_g)$, all siblings share the complete prompt x_g, rollout policy π_{old}, coefficient-fitting data, settings, and random seed. Duplicate checks use the same pre-group candidate pool $\mathcal{Q}_g$, which is updated only after all G siblings are evaluated.

Let $h(q_{g,k})$ denote a child's canonical structure hash. Within each group, proposals sharing $h(q_{g,k})$ contribute one training representative. Invalid patches, duplicates of candidates in $\mathcal{Q}_g$, and candidate evaluation failures receive negative rewards; infrastructure failures are excluded. Let $\mathcal{K}_g$ contain the remaining representative indices. The preference sets are:

$$\begin{aligned} \mathcal{P}_g &= \{k \in \mathcal{K}_g : a_{g,k} \text{ is accepted},\ R_{g,k} > \delta\}, \\ \mathcal{N}_g &= \{k \in \mathcal{K}_g : R_{g,k} < -\delta\}, \end{aligned} \tag{17}$$

where $\delta > 0$ is a small numerical tolerance. Only groups satisfying $\mathcal{P}_g \neq \emptyset$ and $\mathcal{N}_g \neq \emptyset$ contribute to training.

D.5 Details of conservative final structure selection

Candidate pool. Let q_s denote the best searched structure. We construct a pool of at most M distinct structures from valid candidates retained during search, combining candidates with the lowest

search objective J, including q_s, with the best representatives of distinct structural signatures. Each signature records the equation schema and each branch's root operation type, operators, functions, and input variables. We fill the remaining capacity by visiting islands in round-robin order and considering candidates within each island by increasing J, skipping duplicate structures throughout. After an initial cross-validation assessment with a reduced coefficient-fitting budget, we form a smaller shortlist $\mathcal{C}$ containing q_s, top-ranked candidates, and representatives of equation schemas and function families. This shortlist is reevaluated with more coefficient-fitting restarts and a larger optimization budget for final selection.

Repeated grouped cross-validation. Let $\mathcal{T}_{\mathrm{tr}}$ denote the training snapshots. We use R repeats of K folds, keeping each snapshot intact and stratifying by mean absolute target activity. All candidates share the same fold assignments. Coefficients are refitted on the complementary folds, yielding out-of-fold predictions $\hat{y}_{ti}^{(r)}(q)$. For repeat r, the pooled OOF NMSE is:

$$L_r(q) = \frac{\sum_{t\in\mathcal{T}_{\mathrm{tr}}}\sum_i \left[y_i(t) - \hat{y}_{ti}^{(r)}(q)\right]^2}{\sum_{t\in\mathcal{T}_{\mathrm{tr}}}\sum_i \left[y_i(t) - \bar{y}_{\mathrm{tr}}\right]^2}, \tag{18}$$

where $\bar{y}_{\mathrm{tr}}$ is the mean target value over all training node–snapshot observations. Candidates are ranked by:

$$\begin{aligned} \bar{L}(q) &= \frac{1}{R}\sum_{r=1}^{R} L_r(q), \\ S(q) &= \bar{L}(q) + \sqrt{\frac{1}{R}\sum_{r=1}^{R}\left[L_r(q) - \bar{L}(q)\right]^2}. \end{aligned} \tag{19}$$

The variability penalty favors candidates whose predictive performance is consistent across repeats.

Conservative replacement. Let $q_c = \arg\min_{q\in\mathcal{C}} S(q)$ be the CV leader. Its paired advantage over q_s is:

$$\begin{aligned} \Delta_r &= L_r(q_s) - L_r(q_c), \\ \bar{\Delta} &= \frac{1}{R}\sum_{r=1}^{R}\Delta_r, \\ \widehat{\mathrm{SE}}(\Delta) &= \sqrt{\frac{\sum_{r=1}^{R}(\Delta_r - \bar{\Delta})^2}{R(R-1)}}. \end{aligned} \tag{20}$$

We additionally require the snapshot-normalized error to support the replacement of q_s by q_c. Using the OOF predictions from repeat r, define:

$$e_t^{(r)}(q) = \frac{\frac{1}{N}\sum_{i=1}^{N}\left[y_i(t) - \hat{y}_i^{(r)}(q)\right]^2}{\max(E_t, E_{\mathrm{floor}})}, \qquad E_t = \frac{1}{N}\sum_{i=1}^{N} y_i(t)^2, \tag{21}$$

where N is the number of nodes. The energy floor is computed from the training snapshots as:

$$E_{\mathrm{floor}} = \max\left(\frac{\alpha}{|\mathcal{T}_{\mathrm{tr}}|}\sum_{t\in\mathcal{T}_{\mathrm{tr}}} E_t,\ \varepsilon\right), \tag{22}$$

where $\varepsilon > 0$ is a small numerical constant. $\alpha = 0.01$ is a fixed energy-floor fraction and $\varepsilon > 0$ is a small numerical constant. This floor prevents snapshots with near-zero target energy from receiving excessively large weights. For validation fold $\mathcal{F}_{r,k}$, define:

$$H_r(q) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{|\mathcal{F}_{r,k}|}\sum_{t\in\mathcal{F}_{r,k}} e_t^{(r)}(q). \tag{23}$$

The final structure is:

$$q_f = \begin{cases} q_c, & \bar{\Delta} > \widehat{\mathrm{SE}}(\Delta) \\ & \text{and } \dfrac{1}{R}\displaystyle\sum_{r=1}^{R}[H_r(q_s) - H_r(q_c)] > 0, \\ q_s, & \text{otherwise.} \end{cases} \tag{24}$$

The standard-error threshold acts as a conservative variability safeguard. If the paired comparison is unavailable, q_s is retained. Finally, the selected structure is refitted on all $\mathcal{D}_{\mathrm{tr}}$.

E Experiment Setup

E.1 Benchmarks

We evaluate NetSPE on three benchmarks derived from the datasets: Synthetic (Yu et al., 2026), Bacteria Growth (Hu et al., 2022), and Epidemic Transmission (Yu et al., 2026). These benchmarks cover synthetic network dynamical systems, microbial community dynamics, and epidemic spreading, respectively. We follow ND^2's benchmark settings and data preprocessing procedures (Yu et al., 2026). Each benchmark comprises multiple tasks with node-level observations, graph structures, and prediction targets. Synthetic provides analytical ground-truth equations for evaluating both predictive performance and structural recovery, while Bacteria and Epidemic assess predictive performance on empirical observations where the governing equations are unknown.

Synthetic. The Synthetic benchmark comprises ten established network dynamical systems spanning synchronization, neuroscience, molecular biology, ecology, and epidemiology. Kuramoto (KUR), coupled Rössler (CR), and homogeneous coupled Rössler (HCR) describe interacting oscillators and collective synchronization, with the Rössler systems also exhibiting chaotic dynamics. FitzHugh–Nagumo (FHN) and Wilson–Cowan (WC) model neuronal excitability and neural population activity, respectively. Gene regulatory (GR) and Michaelis–Menten (MM) dynamics capture nonlinear regulation and saturating biochemical reaction kinetics. Lotka–Volterra (LV) and mutualistic population (MP) dynamics describe ecological interactions, while susceptible–infected–susceptible (SIS) dynamics model epidemic spreading with reinfection. They provide diverse mechanisms for evaluating equation discovery, including polynomial self-dynamics, saturating responses, multiplicative interactions, and oscillatory coupling. We evaluate one target component per system using the corresponding node-state variables and available node attributes. For multivariate systems, the remaining state variables are retained as explanatory inputs.

Bacteria Growth. The Bacteria Growth benchmark contains eight tasks, corresponding to the released high and low experimental conditions with 3, 6, 12, or 24 species. The target is the change in species abundance between consecutive observations. Experimental replicates are represented as disconnected graph components, with interactions permitted among species within the same replicate.

Epidemic Transmission. The Epidemic benchmark contains seven tasks: CHI, CN, ILS, NYC, NYS, US, and World, covering epidemic dynamics at city, state, national, and global scales. This geographic diversity enables evaluation across different spatial resolutions and mobility patterns. The target is the normalized next-step case count. Local inputs describe current and recent case counts, while graph inputs encode connectivity and mobility flows. The released preprocessing uses fixed normalization scales computed over all finite snapshots.

E.2 Baselines

We compare NetSPE with seven baselines, organized into three groups according to their target setting and primary search mechanism: symbolic regression for network dynamics, LLM-based symbolic regression with whole-expression proposals, and LLM-based symbolic regression with structural refinement.

Symbolic regression for network dynamics. These methods explicitly incorporate network structure and structural priors when discovering dynamical equations.

- **ND**^{2} (Yu et al., 2026) combines a pretrained NDformer with Monte Carlo tree search to discover network dynamical equations. NDformer conditions the search on observed network data, guiding the construction of expressions with node- and edge-level operators.
- **LLC** (Hu et al., 2025) first learns neural representations of self-dynamics and pairwise interactions, then extracts symbolic expressions for the two components. This decomposition separates local dynamics from network coupling before symbolic recovery. In our implementation, we retain the neural decomposition and use PySR for symbolic extraction to accommodate tasks exceeding the input-dimensionality limit of the released pretrained symbolic model. We refer to this LLC+GP variant as LLC in the comparison tables.

LLM-based symbolic regression with whole-expression proposals. These methods generate complete candidate equation programs at each proposal step and use numerical evaluation to guide subsequent search.

- **LLM-SR** (Shojaee et al., 2025) uses an LLM to propose executable equation programs, with numerical optimization fitting their free coefficients. An evolutionary search procedure retains promising candidates and uses them to condition subsequent proposals.
- **PiT-PO** (Wang et al., 2026) adapts the LLM proposal policy through online reinforcement learning during symbolic regression. Its physics-informed, token-regularized policy optimization uses numerical feedback to improve the generation of complete candidate equations over the course of search.

LLM-based symbolic regression with structural refinement. These methods emphasize revising existing candidates through structural mutation, term-level feedback, or targeted refinement.

- **FunctionEvolve** (Xia et al., 2026b) organizes evolutionary search around explicit expression trees, combining structure-guided parent selection with local mutations that preserve useful subexpressions. Its structure-aware coefficient optimizer separates linear and nonlinear parameters to improve candidate fitting and evaluation.
- **IGSR** (Saveliev et al., 2026) represents an equation as a weighted combination of LLM-generated nonlinear basis functions. It measures each term's influence through the change in validation error when that term is removed, using this feedback to guide term proposal and pruning within Monte Carlo tree search.
- **RESTART** (Li & Pan, 2026) combines short-term equation refinement with a persistent library of reusable structures. Its refinement stage uses boosting to identify unexplained signals and guide targeted corrections, while the library retains useful structures from successful refinements for subsequent search.

The five general-purpose LLM baselines operate on flattened node-level observations and receive local scalar features. ND^2 and LLC are backbone-independent baselines; the same experimental results are therefore reported across the two LLM backbone groups. We follow the baseline hyperparameter settings and search budgets reported in the original papers.

E.3 METRICS

Following common evaluation practice in symbolic regression benchmarks and recent methods such as PiT-PO and RESTART (Wang et al., 2026; Li & Pan, 2026), we assess predictive performance and equation recovery using normalized mean squared error (NMSE), accuracy to tolerance (ACC_τ), and structural accuracy (SA). NMSE measures prediction error relative to the variance of the test targets: $\mathrm{NMSE} = \frac{\sum_{i=1}^{N_{\text{test}}}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{N_{\text{test}}}(y_i - \bar{y})^2}$, where y_i and $\hat{y}_i$ denote the ground-truth and predicted values, respectively, and $\bar{y}$ is the mean test target. Each observation corresponds to a node in a held-out graph snapshot. Lower NMSE indicates better predictive performance. We use the pointwise form of accuracy to tolerance, which measures the fraction of test predictions satisfying a relative-error threshold: $\mathrm{ACC}_\tau = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \mathbf{1}[|\hat{y}_i - y_i| \leq \tau |y_i|]$. We set $\tau = 0.1$ throughout. Higher ACC_τ

indicates better predictive performance. We report benchmark-level summaries of the metrics in our main results. For benchmark-level summaries, we first average results across the included runs of each problem and then take an equally weighted mean across tasks. Problem-level standard deviations describe variation across runs, whereas benchmark-level standard deviations describe variation across problem means.

For synthetic problems with known ground-truth equations, we additionally report structural accuracy (SA). A recovered expression is considered structurally correct if it can be made algebraically equivalent to the ground-truth equation through a finite reassignment of its numerical constants, without introducing new terms or operators. We compute SA over all problem–run pairs: $\mathrm{SA} = \frac{1}{PR}\sum_{p=1}^{P}\sum_{r=1}^{R}\mathbf{1}\left[\hat{f}_{p,r} \equiv_{\mathrm{struct}} f_p^\star\right]$, where P is the number of problems, R is the number of runs per problem, $\hat{f}_{p,r}$ is the expression recovered in run r for problem p, and $f_p^\star$ is its ground-truth equation. The relation $\equiv_{\mathrm{struct}}$ denotes structural equivalence under the criterion above. SA is not evaluated on Bacteria Growth or Epidemic Transmission, where ground-truth equations are unavailable.

E.4 Implementation Details

All experiments are conducted over three independent runs. We use a 8:2 training and test split. All nodes within a snapshot remain in the same partition. Within NetSPE's training partition, 20% of snapshots are held out for internal candidate selection. We use Llama3.1-8B-Instruct (Grattafiori et al., 2024) and Qwen3-4B-Instruct-2507 (Yang et al., 2025) as the LLM backbones. Unless otherwise specified, all experiments involving LLMs use Llama3.1-8B-Instruct as the backbone. Following PiT-PO (Wang et al., 2026), we load both LLM backbones with 4-bit quantization to reduce GPU memory consumption for all LLM-based methods. NetSPE uses a budget of 1,000 proposal slots, 10 search islands, and four proposals per LLM prompt. The sampling temperature is set to 0.8, and the maximum completion length is 512 tokens. Numerical coefficient fitting uses three initializations, at most 300 function evaluations, and coefficient bounds of $[-20, 20]$. The expression-complexity penalty is set to 10^{-5}.

We adapt the patch-generation policy using 4-bit QLoRA with rank 16, scaling factor 32, and zero dropout. The default learning rate is 5×10^{-7}, and the KL regularization coefficient is 0.02. Each policy group contains four proposals generated from an identical edit prompt. Each update window uses eight informative groups for optimization and two held-out groups for evaluating the candidate update. An updated policy is adopted only after passing the held-out acceptance checks, with at most three accepted updates per run by default. Detailed hyperparameter settings are provided in Table 4.

F Additional Experiments

F.1 Complete Results across All Problems in Three Benchmarks

Tables 5 and 6 report problem-level results with Llama3.1-8B-Instruct and Qwen3-4B-Instruct, respectively, covering all 25 problems across Synthetic, Bacteria Growth, and Epidemic Transmission Benchmarks. We report the mean and standard deviation of NMSE and $\mathrm{ACC}_{0.1}$ across 3 runs, with the best results in bold and NetSPE highlighted in gray.

F.2 Additional Ablation Study

We investigate whether NetSPE's performance gains can be explained simply by providing existing methods with structural priors (SP). We augment LLM-based baselines including LLM-SR, FunctionEvolve, PiT-PO, and RESTART with structural priors derived from NetSPE's priors. These priors describe an additive decomposition of self-dynamics and aggregated interactions for Synthetic, a zero-preserving growth structure for Bacteria Growth, and self or interaction-conditioned readout structures for Epidemic Transmission. We adapt these templates to each baseline's existing input interface: graph-dependent terms are provided as structural context, while candidate equations are instructed to use only the supplied variables, without introducing unavailable graph inputs or aggregating across independent observations. These priors are supplied through prompts. These

Table 4: Default hyperparameter settings of NetSPE.

Parameter	Value
Data splits	
Training / test ratio	80% / 20%
Internal selection fraction of training data	20%
LLM generation	
Sampling temperature	0.8
Maximum completion length	512 tokens
Search and coefficient fitting	
Complexity Penalty λ_c	10^{-5}
Proposal budget	1,000
Number of search islands	10
Proposals per LLM prompt	4
Coefficient-fitting initializations	3
Function evaluations per initialization	300
Coefficient bounds	$[-20, 20]$
Online policy optimization	
LoRA rank / scaling factor	16 / 32
Learning rate	5×10^{-7}
KL regularization coefficient	0.02

instructions are incorporated into LLM-SR's generation prompts, FunctionEvolve's generation and evolution context, PiT-PO's sampling and policy-training prompts, and RESTART's proposal, refinement, and analysis prompts. The underlying search procedures remain unchanged, and the priors serve as prompt guidance without additional hard structural constraints.

Table 7 shows that adding SP does not consistently improve performance across methods and datasets. While incorporating structural priors improves some baselines on certain benchmarks, the gains are inconsistent across datasets and evaluation metrics, with some methods showing limited improvements or even performance degradation. These results suggest that adding structural priors alone does not reliably yield substantial improvements in overall performance. NetSPE achieves the lowest mean NMSE across all three datasets and the highest mean accuracy on Synthetic and Epidemic Transmission, suggesting that its advantages extend beyond access to structural priors alone. These findings support the complementary roles of SP, Hierarchical Edit Target Localization, and SPPO: SP structures the search space, localization identifies where to edit, and SPPO improves how edits are proposed. They enable more effective use of structural knowledge during equation discovery.

F.3 Out-of-Distribution Evaluation of NetSPE

We evaluate whether the discovered equations remain predictive under perturbations to the observed system states. For each test graph snapshot across ten synthetic systems from Synthetic Benchmark, we perturb the dynamic variables while keeping the graph topology and system parameters fixed. The perturbations combine a shared shift across nodes with local node-level variations, contributing 80% and 20% of the perturbation energy, respectively, with a total root mean square magnitude of 10% of each variable's training-set standard deviation. Physical state constraints are respected, and derivative targets are recomputed using the ground-truth dynamics at the perturbed states. All methods are evaluated on the same perturbed data with their discovered equations and fitted coefficients frozen. This setting assesses generalization under local state distribution shifts.

As shown in Table 8, NetSPE achieves the lowest mean NMSE and the highest mean $\mathrm{ACC}_{0.1}$, outperforming ND^2, LLC and the other symbolic regression baselines. Its advantage across both metrics indicates that the discovered equations provide more accurate predictions under the evaluated state perturbations. These results support the ability of NetSPE to capture dynamical relationships that remain predictive in nearby, previously unobserved states.

Table 5: Complete problem-level results with Llama3.1-8B-Instruct.

Problem	Metric	LLM-SR	FunctionEvolve	PiT-PO	ND[2]	IGSR	RESTART	LLC	NetSPE
Synthetic									
CR	NMSE ↓	$0.1227 \pm 2.889\times10^{-10}$	0.1247 ±0.0107	0.1381 ±0.0078	0.1504 ±0.0005	0.3814 ±0.2896	0.1332 ±0.0148	0.4622 ±0.0706	**0.0050 ±0.0087**
	$ACC_{0.1}$ ↑		0.1470 ±0.0147	0.1220 ±0.0069	0.1157 ±0.0038	0.0893 ±0.0433	0.1275 ±0.0000	0.1115 ±0.0061	**0.8425 ±0.2728**
FHN	NMSE ↓	0.1820 ±0.0181	0.1617 ±0.0117	0.2284 ±0.0714	0.0955 ±0.1655	0.5083 ±0.1688	0.2232 ±0.0688	0.7975 ±0.6779	$\mathbf{9.660\times10^{-32}}$ **±2.093×10^{-32}**
	$ACC_{0.1}$ ↑	0.0973 ±0.0381	0.1305 ±0.0246	0.0962 ±0.0144	0.6878 ±0.5407	0.0453 ±0.0075	0.1010 ±0.0120	0.0965 ±0.0939	**1.0000 ±0.0000**
GR	NMSE ↓	0.8709 ±0.0170	0.8303 ±0.0333	0.8967 ±0.0383	$0.0002 \pm 5.218\times10^{-6}$	0.8598 ±0.0021	0.8560 ±0.0029	0.0193 ±0.0044	$\mathbf{7.131\times10^{-6}}$ **±1.230×10^{-5}**
	$ACC_{0.1}$ ↑	0.1767 ±0.0081	0.1815 ±0.0135	0.1907 ±0.0094	0.9985 ±0.0005	0.1855 ±0.0095	0.1777 ±0.0004	0.7778 ±0.0312	**0.9993 ±0.0012**
HCR	NMSE ↓	0.1817 ±0.0010	0.1729 ±0.0131	0.1784 ±0.0021	$2.695\times10^{-17} \pm 2.974\times10^{-18}$	0.2700 ±0.0544	0.1861 ±0.0034	0.3944 ±0.0307	$\mathbf{4.981\times10^{-34}}$ **±0.0000**
	$ACC_{0.1}$ ↑	0.1188 ±0.0107	0.1183 ±0.0015	0.1345 ±0.0036	**1.0000 ±0.0000**	0.1193 ±0.0140	0.1167 ±0.0046	0.0862 ±0.0051	**1.0000 ±0.0000**
LV	NMSE ↓	0.1452 ±0.0049	0.0199 ±0.0045	0.1363 ±0.0359	**0.0140 ±0.0128**	0.4360 ±0.0852	0.1283 ±0.0045	0.2448 ±0.0379	0.0211 ±0.0082
	$ACC_{0.1}$ ↑	0.1417 ±0.0818	0.3567 ±0.1092	0.1783 ±0.0700	**0.5307 ±0.4047**	0.1533 ±0.0166	0.1500 ±0.0035	0.1383 ±0.0406	0.3842 ±0.1332
MM	NMSE ↓	0.8497 ±0.0267	0.8067 ±0.0466	0.8388 ±0.0111	$\mathbf{1.428\times10^{-15}}$ **±1.232×10^{-15}**	0.8757 ±0.0065	0.8356 ±0.0014	0.0540 ±0.0570	2.548×10^{-9} ±4.413×10^{-9}
	$ACC_{0.1}$ ↑	0.0375 ±0.0121	0.0733 ±0.0019	0.1037 ±0.0043	**1.0000 ±0.0000**	0.0517 ±0.0075	0.0663 ±0.0018	0.2696 ±0.0878	**1.0000 ±0.0000**
MP	NMSE ↓	0.1672 ±0.0312	0.0806 ±0.0261	0.3648 ±0.1496	0.0993 ±0.1405	0.5804 ±0.2060	0.1418 ±0.0360	1.1232 ±0.1250	**0.0580 ±0.0359**
	$ACC_{0.1}$ ↑	0.0883 ±0.0232	0.1483 ±0.0191	0.0667 ±0.0123	**0.1758 ±0.1000**	0.0550 ±0.0222	0.1537 ±0.0760	0.0542 ±0.0279	0.1458 ±0.0126
SIS	NMSE ↓	0.2530 ±0.0065	0.1650 ±0.0280	0.2411 ±0.0038	0.0778 ±0.0842	0.3205 ±0.0214	0.2328 ±0.0003	0.6874 ±0.4546	$\mathbf{6.691\times10^{-8}}$ **±1.159×10^{-7}**
	$ACC_{0.1}$ ↑	0.1117 ±0.0318	0.1208 ±0.0267	0.0792 ±0.0208	0.4133 ±0.5081	0.0650 ±0.0152	0.0963 ±0.0088	0.0858 ±0.0795	**0.9967 ±0.0058**
WC	NMSE ↓	0.4342 ±0.0120	0.4208 ±0.0176	0.4353 ±0.0044	1.839×10^{-6} ±3.186×10^{-6}	0.4430 ±0.0129	0.4376 ±0.0023	0.3897 ±0.3478	$\mathbf{3.960\times10^{-33}}$ **±6.858×10^{-33}**
	$ACC_{0.1}$ ↑	0.0638 ±0.0043	0.0633 ±0.0159	0.0304 ±0.0130	0.9975 ±0.0043	0.0567 ±0.0095	0.0488 ±0.0124	0.0821 ±0.0488	**1.0000 ±0.0000**
KUR	NMSE ↓	0.1396 ±0.0005	0.1349 ±0.0008	$0.1404 \pm 2.581\times10^{-5}$	$2.207\times10^{-17} \pm 2.921\times10^{-17}$	0.4544 ±0.4653	0.1393 ±0.0009	0.0737 ±0.0068	**4.336×10^{-34} ±0.0000**
	$ACC_{0.1}$ ↑	0.1787 ±0.0014	0.1774 ±0.0012	0.1765 ±0.0010	**1.0000 ±0.0000**	0.1050 ±0.0886	0.1772 ±0.0007	0.2964 ±0.0006	**1.0000 ±0.0000**
Bacteria Growth									
High-3	NMSE ↓	1.1878 ±0.0075	1.0440 ±0.0586	0.8659 ±0.0156	0.8283 ±0.0002	1.0648 ±0.0388	1.0924 ±0.0824	1.0865 ±0.1080	**0.7845 ±0.0210**
	$ACC_{0.1}$ ↑	0.0139 ±0.0120	**0.2431 ±0.0524**	0.0347 ±0.0434	0.2083 ±0.0000	0.0000 ±0.0000	0.0104 ±0.0147	0.0208 ±0.0000	0.2222 ±0.0318
High-6	NMSE ↓	0.9848 ±0.0240	3.3363 ±2.9094	1.0862 ±0.0061	0.9304 ±0.0529	0.9756 ±0.0176	1.0474 ±0.0178	1.0193 ±0.0777	**0.7698 ±0.0212**
	$ACC_{0.1}$ ↑	0.1076 ±0.1775	**0.4132 ±0.0874**	0.0208 ±0.0104	0.3438 ±0.0000	0.0104 ±0.0104	0.0104 ±0.0147	0.1250 ±0.1715	0.3438 ±0.0104
High-12	NMSE ↓	1.1506 ±0.2905	1.0361 ±0.0649	1.3070 ±0.1179	0.9705 ±0.0294	0.9575 ±0.0069	1.0323 ±0.0413	**0.9332 ±0.0010**	
	$ACC_{0.1}$ ↑	0.0104 ±0.0000	**0.4948 ±0.0701**	0.0139 ±0.0030	0.4323 ±0.0138	0.0052 ±0.0052	0.0026 ±0.0037	0.0139 ±0.0080	0.4167 ±0.0000
High-24	NMSE ↓	1.0416 ±0.0093	**0.7373 ±0.3434**	1.0615 ±0.0524	1.1333 ±0.1176	1.0203 ±0.0181	1.0596 ±0.0150	1.1204 ±0.1037	0.7851 ±0.0408
	$ACC_{0.1}$ ↑	0.0061 ±0.0040	**0.4870 ±0.0231**	0.0069 ±0.0040	0.4627 ±0.0015	0.0087 ±0.0040	0.0156 ±0.0074	0.0122 ±0.0091	0.4835 ±0.0080
Low-3	NMSE ↓	1.0673 ±0.0080	1.0636 ±0.0800	1.0047 ±0.0983	1.1438 ±0.1178	1.0765 ±0.0091	1.0650 ±0.0158	1.0516 ±0.0622	**0.9654 ±0.0002**
	$ACC_{0.1}$ ↑	0.0208 ±0.0000	0.0347 ±0.0318	0.0347 ±0.0434	**0.0764 ±0.0241**	0.0347 ±0.0120	0.0104 ±0.0147	0.0139 ±0.0241	0.0625 ±0.0000
Low-6	NMSE ↓	0.9029 ±0.0008	0.9706 ±0.2002	0.9890 ±0.0489	1.3518 ±0.3833	0.9174 ±0.0193	0.9259 ±0.0006	1.0482 ±0.0713	**0.8498 ±0.0046**
	$ACC_{0.1}$ ↑	0.0174 ±0.0060	0.0972 ±0.0335	0.0347 ±0.0335	0.0868 ±0.0159	0.0104 ±0.0104	0.0208 ±0.0147	0.0243 ±0.0120	**0.1007 ±0.0120**
Low-12	NMSE ↓	1.0382 ±0.1084	0.9320 ±0.0134	0.9845 ±0.1039	0.9526 ±0.0172	0.9518 ±0.0023	1.0060 ±0.0822	**0.7867 ±0.0223**	
	$ACC_{0.1}$ ↑	0.0260 ±0.0138	**0.1632 ±0.0437**	0.0295 ±0.0235	0.1424 ±0.0080	0.0208 ±0.0104	0.0208 ±0.0074	0.0312 ±0.0208	0.1458 ±0.0052
Low-24	NMSE ↓	0.8662 ±0.0024	0.9196 ±0.0515	1.0152 ±0.0518	0.8587 ±0.0017	0.9724 ±0.0294	0.9754 ±0.1257	1.1914 ±0.0588	**0.7305 ±0.0081**
	$ACC_{0.1}$ ↑	0.0009 ±0.0015	0.5460 ±0.0153	0.0252 ±0.0054	0.5495 ±0.0000	0.0122 ±0.0015	0.0091 ±0.0129	0.1927 ±0.3157	**0.5642 ±0.0105**
Epidemic Transmission									
CHI	NMSE ↓	0.3830 ±0.0180	**0.2186 ±0.0969**	0.3588 ±0.0579	0.3319 ±0.0222	0.3663 ±0.0402	0.3317 ±0.0456	0.3128 ±0.0322	0.2915 ±0.0345
	$ACC_{0.1}$ ↑	0.1815 ±0.0093	0.2257 ±0.0060	0.1919 ±0.0152	0.2138 ±0.0085	0.1992 ±0.0194	0.2075 ±0.0120	0.2204 ±0.0110	**0.2275 ±0.0311**
CN	NMSE ↓	0.1579 ±0.0239	0.8607 ±0.7383	0.1005 ±0.0301	0.2273 ±0.1609	0.0687 ±0.0581	0.1571 ±0.0265	0.1269 ±0.0693	**0.0653 ±0.0000**
	$ACC_{0.1}$ ↑	0.0242 ±0.0093	**0.3177 ±0.0540**	0.0420 ±0.0096	0.2037 ±0.1622	0.0407 ±0.0088	0.0230 ±0.0000	0.0300 ±0.0204	0.2950 ±0.0000
ILS	NMSE ↓	0.4028 ±0.1103	0.3701 ±0.3880	0.3800 ±0.0403	0.3274 ±0.0951	0.2398 ±0.0424	0.3635 ±0.0435	0.2274 ±0.0116	**0.2055 ±0.0117**
	$ACC_{0.1}$ ↑	0.1134 ±0.0086	0.1328 ±0.1138	0.1395 ±0.0228	0.1345 ±0.0045	0.1225 ±0.0444	0.1212 ±0.0388	0.1640 ±0.0031	**0.1773 ±0.0050**
NYC	NMSE ↓	0.6875 ±0.0353	0.3177 ±0.1506	0.6050 ±0.0898	0.5615 ±0.0717	0.7132 ±0.0956	0.6324 ±0.1163	0.5423 ±0.0411	**0.3156 ±0.0000**
	$ACC_{0.1}$ ↑	0.1396 ±0.0090	0.1992 ±0.0228	0.1721 ±0.0199	0.1689 ±0.0358	0.1278 ±0.0060	0.1637 ±0.0259	0.1426 ±0.0510	**0.2374 ±0.0000**
NYS	NMSE ↓	0.1792 ±0.0081	0.2009 ±0.0391	0.1935 ±0.0319	0.2283 ±0.0655	0.1368 ±0.0420	0.1718 ±0.0291	**0.1282 ±0.0033**	0.1729 ±0.0275
	$ACC_{0.1}$ ↑	0.2013 ±0.0352	**0.2564 ±0.0285**	0.1211 ±0.0204	0.1361 ±0.0424	0.1833 ±0.0315	0.2097 ±0.0128	0.2562 ±0.0470	0.2240 ±0.0276
US	NMSE ↓	0.8894 ±0.0245	0.6140 ±0.4170	0.8713 ±0.0736	0.9035 ±0.2828	0.6617 ±0.1578	0.9417 ±0.0678	1.1083 ±0.0117	**0.5014 ±0.0201**
	$ACC_{0.1}$ ↑	0.3060 ±0.0186	0.2571 ±0.0957	0.2751 ±0.0761	0.3542 ±0.0074	0.2623 ±0.0469	0.2105 ±0.0091	0.3380 ±0.0153	**0.3768 ±0.0032**
World	NMSE ↓	0.0942 ±0.0027	0.0920 ±0.0186	**0.0881 ±0.0027**	0.0933 ±0.0055	0.1078 ±0.0065	0.0954 ±0.0063	0.1017 ±0.0090	0.1241 ±0.0298
	$ACC_{0.1}$ ↑	0.0852 ±0.0088	0.2400 ±0.0824	0.1526 ±0.1067	**0.2954 ±0.0180**	0.0810 ±0.0264	0.1067 ±0.0291	0.1008 ±0.0146	0.2509 ±0.0216

Table 6: Complete problem-level results with Qwen3-4B-Instruct.

Problem	Metric	LLM-SR	FunctionEvolve	PIT-PO	ND[2]	IGSR	RESTART	LLC	NetSPE
Synthetic									
CR	NMSE ↓	0.1858 ±3.932×10^{-11}	0.1269 ±0.0058	0.1373 ±0.0049	0.1504 ±0.0005	0.2572 ±0.0450	0.1717 ±0.0114	0.4622 ±0.0706	**0.0347 ±0.0301**
	ACC$_{0.1}$ ↑	0.1270 ±0.0000	0.1415 ±0.0109	0.1322 ±0.0047	0.1157 ±0.0038	0.1177 ±0.0174	0.1343 ±0.0025	0.1115 ±0.0061	**0.5203 ±0.4156**
FHN	NMSE ↓	0.6715 ±1.272×10^{-10}	0.1563 ±0.0400	0.1974 ±0.0042	0.0955 ±0.1655	0.5488 ±0.2606	0.6173 ±0.0317	0.7975 ±0.6779	**1.143×10^{-31} ±1.259×10^{-32}**
	ACC$_{0.1}$ ↑	0.0500 ±0.0000	0.0903 ±0.0236	0.1067 ±0.0102	0.6878 ±0.5407	0.0477 ±0.0117	0.0500 ±0.0014	0.0965 ±0.0939	**1.0000 ±0.0000**
GR	NMSE ↓	1.0147 ±0.0000	0.7993 ±0.0109	0.8540 ±0.0011	0.0002 ±5.219×10^{-6}	0.8895 ±0.0223	0.9423 ±0.1024	0.0193 ±0.0044	**1.329×10^{-5} ±2.289×10^{-5}**
	ACC$_{0.1}$ ↑	0.1630 ±0.0000	0.1718 ±0.0074	0.1757 ±0.0058	0.9985 ±0.0005	0.1830 ±0.0109	0.1713 ±0.0117	0.7778 ±0.0312	**0.9997 ±0.0006**
HCR	NMSE ↓	0.1940 ±3.158×10^{-12}	0.1747 ±0.0063	0.1790 ±0.0004	2.695×10^{-17} ±2.974×10^{-18}	0.4274 ±0.1561	0.1913 ±0.0039	0.3944 ±0.0307	**4.981×10^{-54} ±0.0000**
	ACC$_{0.1}$ ↑	0.1190 ±0.0000	0.1167 ±0.0068	0.1218 ±0.0121	**1.0000 ±0.0000**	0.0883 ±0.0350	0.1157 ±0.0053	0.0862 ±0.0051	**1.0000 ±0.0000**
LV	NMSE ↓	0.7825 ±1.290×10^{-10}	0.0253 ±0.0115	0.1275 ±0.0306	**0.0140 ±0.0128**	0.4546 ±0.0976	0.4655 ±0.0540	0.2448 ±0.0379	0.0150 ±0.0111
	ACC$_{0.1}$ ↑	0.0400 ±0.0000	0.3242 ±0.1020	0.2200 ±0.0275	**0.5367 ±0.4047**	0.0600 ±0.0195	0.0750 ±0.0106	0.1383 ±0.0406	0.4433 ±0.2522
MM	NMSE ↓	0.9572 ±0.0000	0.7424 ±0.0139	0.8474 ±0.0201	**1.428×10^{-15} ±1.232×10^{-15}**	0.8817 ±0.0055	0.8766 ±0.0629	0.0540 ±0.0570	2.300×10^{-7} ±3.685×10^{-7}
	ACC$_{0.1}$ ↑	0.0425 ±0.0000	0.0858 ±0.0155	0.0833 ±0.0421	**1.0000 ±0.0000**	0.0525 ±0.0082	0.0525 ±0.0141	0.2696 ±0.0878	0.9988 ±0.0012
MP	NMSE ↓	0.6750 ±4.423×10^{-9}	0.0995 ±0.0699	0.2552 ±0.0601	0.0993 ±0.1405	0.5212 ±0.0518	0.2859 ±0.1337	1.1232 ±0.1250	**0.0424 ±0.0045**
	ACC$_{0.1}$ ↑	0.0375 ±0.0000	0.1775 ±0.1113	0.0908 ±0.0255	0.1758 ±0.1000	0.0508 ±0.0142	0.0788 ±0.0018	0.0542 ±0.0279	**0.1808 ±0.0290**
SIS	NMSE ↓	0.3338 ±0.0000	0.1846 ±0.0524	0.2389 ±0.0028	0.0778 ±0.0842	0.4492 ±0.1531	0.3019 ±0.0048	0.6874 ±0.4546	**8.980×10^{-5} ±0.0002**
	ACC$_{0.1}$ ↑	0.0800 ±0.0000	0.1350 ±0.0238	0.0850 ±0.0164	0.4133 ±0.5081	0.0625 ±0.0278	0.1038 ±0.0159	0.0858 ±0.0795	**0.8942 ±0.1833**
WC	NMSE ↓	0.8165 ±0.0000	0.4107 ±0.0540	0.4255 ±0.0155	1.839×10^{-6} ±3.186×10^{-6}	0.4638 ±0.0033	0.4349 ±0.0047	0.3897 ±0.3478	**5.147×10^{-33} ±7.378×10^{-33}**
	ACC$_{0.1}$ ↑	0.0187 ±0.0000	0.0775 ±0.0479	0.0492 ±0.0188	0.9975 ±0.0043	0.0250 ±0.0119	0.0456 ±0.0274	0.0821 ±0.0488	**1.0000 ±0.0000**
KUR	NMSE ↓	0.1399 ±4.747×10^{-5}	0.1346 ±0.0005	0.1403 ±5.092×10^{-5}	2.207×10^{-17} ±2.921×10^{-17}	0.1458 ±0.0100	0.1395 ±0.0004	0.0737 ±0.0068	**4.336×10^{-34} ±0.0000**
	ACC$_{0.1}$ ↑	0.1778 ±0.0006	0.1782 ±0.0037	0.1754 ±0.0010	**1.0000 ±0.0000**	0.1748 ±0.0063	0.1777 ±0.0011	0.2964 ±0.0006	**1.0000 ±0.0000**
Bacteria Growth									
High-3	NMSE ↓	1.0796 ±0.0704	4.1225 ±4.6772	1.0409 ±0.1480	**0.8283 ±0.0002**	1.0762 ±0.0438	1.0902 ±0.0991	1.0865 ±0.1080	0.8740 ±0.0171
	ACC$_{0.1}$ ↑	0.0069 ±0.0120	0.1806 ±0.1418	0.0000 ±0.0000	**0.2083 ±0.0000**	0.0000 ±0.0000	0.0833 ±0.1102	0.0208 ±0.0000	0.2031 ±0.0104
High-6	NMSE ↓	1.0066 ±0.0248	3.3468 ±2.7413	1.0870 ±0.0126	0.9304 ±0.0529	0.9744 ±0.0142	0.9811 ±0.0209	1.0193 ±0.0777	**0.8050 ±0.0403**
	ACC$_{0.1}$ ↑	0.0000 ±0.0000	0.3056 ±0.2561	0.0278 ±0.0217	**0.3438 ±0.0000**	0.0069 ±0.0060	0.1076 ±0.1775	0.1250 ±0.1715	0.3307 ±0.0131
High-12	NMSE ↓	0.9769 ±0.0274	1.1560 ±0.0368	1.0135 ±0.1235	1.3070 ±0.1179	0.9805 ±0.0185	**0.9498 ±0.0022**	1.0323 ±0.0413	0.9990 ±0.0428
	ACC$_{0.1}$ ↑	0.0156 ±0.0000	0.3542 ±0.3036	0.4201 ±0.0030	**0.4323 ±0.0138**	0.0087 ±0.0080	0.1493 ±0.2362	0.0139 ±0.0080	0.4206 ±0.0050
High-24	NMSE ↓	1.0728 ±0.0310	**0.6576 ±0.2529**	1.0376 ±0.0510	1.1333 ±0.1176	1.0139 ±0.0050	1.0405 ±0.0373	1.1204 ±0.1037	0.7821 ±0.0132
	ACC$_{0.1}$ ↑	0.0052 ±0.0045	0.1615 ±0.2751	0.0061 ±0.0054	0.4627 ±0.0015	0.0052 ±0.0026	0.3151 ±0.2661	0.0122 ±0.0091	**0.4889 ±0.0039**
Low-3	NMSE ↓	1.0556 ±0.0191	1.1833 ±0.1572	1.0907 ±0.0193	1.1438 ±0.1178	1.0798 ±0.0114	1.0820 ±0.0236	**1.0516 ±0.0622**	1.1668 ±0.0177
	ACC$_{0.1}$ ↑	0.0556 ±0.0241	0.0278 ±0.0318	0.0139 ±0.0241	**0.0764 ±0.0241**	0.0208 ±0.0208	0.0208 ±0.0208	0.0139 ±0.0241	0.0573 ±0.0312
Low-6	NMSE ↓	0.9583 ±0.0000	1.1244 ±0.4642	0.9818 ±0.0338	1.3518 ±0.3833	0.9217 ±0.0359	0.9236 ±0.0358	1.0482 ±0.0713	**0.8555 ±0.0015**
	ACC$_{0.1}$ ↑	0.0104 ±0.0000	0.1007 ±0.0301	0.0278 ±0.0159	0.0868 ±0.0159	0.0208 ±0.0104	0.0312 ±0.0361	0.0243 ±0.0120	**0.1016 ±0.0100**
Low-12	NMSE ↓	1.0179 ±0.0125	1.0562 ±0.1059	0.9363 ±0.0328	0.9845 ±0.1039	0.9506 ±0.0170	0.9433 ±0.0135	1.0060 ±0.0822	**0.7904 ±0.0091**
	ACC$_{0.1}$ ↑	0.0451 ±0.0782	0.1181 ±0.0759	0.0156 ±0.0104	**0.1424 ±0.0080**	0.0191 ±0.0120	0.0278 ±0.0080	0.0312 ±0.0208	0.1419 ±0.0078
Low-24	NMSE ↓	0.8972 ±3.593×10^{-9}	0.8834 ±0.0941	0.9783 ±0.0089	0.8587 ±0.0017	0.9756 ±0.0055	0.8790 ±0.0253	1.1914 ±0.0588	**0.7347 ±0.0014**
	ACC$_{0.1}$ ↑	0.0156 ±0.0000	0.3837 ±0.3340	0.0156 ±0.0069	0.5495 ±0.0000	0.0130 ±0.0000	0.1832 ±0.3172	0.1927 ±0.3157	**0.5697 ±0.0013**
Epidemic Transmission									
CHI	NMSE ↓	0.3916 ±1.214×10^{-10}	**0.2299 ±0.1090**	0.3295 ±0.0103	0.3319 ±0.0222	0.3570 ±0.0038	0.3574 ±0.0257	0.3128 ±0.0322	0.3249 ±0.0512
	ACC$_{0.1}$ ↑	0.1782 ±0.0000	**0.2379 ±0.0196**	0.2030 ±0.0046	0.2138 ±0.0085	0.2027 ±0.0080	0.1987 ±0.0136	0.2204 ±0.0110	0.2212 ±0.0185
CN	NMSE ↓	**0.0330 ±6.016×10^{-10}**	0.1727 ±0.1542	0.1222 ±0.0852	0.2273 ±0.1609	0.1121 ±0.0734	0.1524 ±0.0768	0.1269 ±0.0693	0.0438 ±0.0143
	ACC$_{0.1}$ ↑	0.0423 ±0.0000	0.2157 ±0.1797	0.1192 ±0.1341	0.2037 ±0.1622	0.0300 ±0.0167	0.0303 ±0.0078	0.0300 ±0.0204	**0.3093 ±0.0533**
ILS	NMSE ↓	0.3085 ±9.905×10^{-9}	0.2273 ±0.1220	0.3735 ±0.0652	0.3274 ±0.0951	0.3117 ±0.1666	0.3710 ±0.0053	0.2274 ±0.0116	**0.1976 ±0.0022**
	ACC$_{0.1}$ ↑	0.0971 ±0.0000	0.1628 ±0.0320	0.1420 ±0.0153	0.1345 ±0.0045	0.1156 ±0.0198	0.1137 ±0.0061	0.1640 ±0.0031	**0.1801 ±0.0002**
NYC	NMSE ↓	0.7926 ±1.821×10^{-9}	0.3630 ±0.1997	0.7314 ±0.1142	0.5615 ±0.0717	0.5616 ±0.1150	0.6759 ±0.0432	0.5423 ±0.0411	**0.3125 ±0.0000**
	ACC$_{0.1}$ ↑	0.0874 ±0.0000	0.1755 ±0.0327	0.1426 ±0.0080	0.1689 ±0.0358	0.1476 ±0.0036	0.1471 ±0.0037	0.1426 ±0.0510	**0.2562 ±0.0000**
NYS	NMSE ↓	0.1588 ±3.224×10^{-9}	0.1891 ±0.0405	0.2076 ±0.0145	0.2283 ±0.0655	0.1517 ±0.0364	0.1702 ±0.0153	**0.1282 ±0.0033**	0.1570 ±2.778×10^{-10}
	ACC$_{0.1}$ ↑	0.0781 ±0.0000	0.2261 ±0.0465	0.1887 ±0.0510	0.1361 ±0.0424	0.1882 ±0.0594	0.1726 ±0.0923	**0.2562 ±0.0470**	0.2399 ±0.0000
US	NMSE ↓	0.9435 ±7.316×10^{-9}	0.5770 ±0.3861	0.8718 ±0.2391	0.9035 ±0.2828	0.8797 ±0.3338	0.9513 ±0.1648	1.1083 ±0.0117	**0.5159 ±0.0174**
	ACC$_{0.1}$ ↑	0.1624 ±0.0000	0.3072 ±0.0416	0.2768 ±0.0594	0.3542 ±0.0074	0.2408 ±0.0111	0.1967 ±0.0494	0.3380 ±0.0153	**0.3745 ±0.0028**
World	NMSE ↓	0.0988 ±7.883×10^{-10}	0.0919 ±0.0259	**0.0892 ±0.0040**	0.0933 ±0.0055	0.0942 ±0.0162	0.0984 ±0.0082	0.1017 ±0.0090	0.1002 ±0.0010
	ACC$_{0.1}$ ↑	0.0673 ±0.0000	0.2294 ±0.1011	0.1591 ±0.1039	**0.2954 ±0.0180**	0.0723 ±0.0080	0.0930 ±0.0348	0.1008 ±0.0146	0.1183 ±0.0188

Table 7: Performance comparison of original baselines, structural prior-enhanced baselines (LLM-based baselines), and NetSPE using Llama3.1-8B-Instruct. Bold and underlined values indicate the best and second-best mean results across all methods, respectively.

Methods	Synthetic		Bacteria Growth		Epidemic Transmission	
	NMSE ↓	$ACC_{0.1}$ ↑	NMSE ↓	$ACC_{0.1}$ ↑	NMSE ↓	$ACC_{0.1}$ ↑
Original Baselines						
LLM-SR	0.3346 ± 0.2912	11.42 ± 4.52	$\underline{0.9909 \pm 0.1043}$	2.54 ± 3.42	0.3991 ± 0.2953	15.02 ± 9.08
FunctionEvolve	0.2918 ± 0.2963	$\underline{15.17 \pm 8.17}$	1.2825 ± 0.8389	$\mathbf{30.99 \pm 19.96}$	0.3820 ± 0.2680	$\underline{23.27 \pm 5.71}$
PiT-PO	0.3598 ± 0.2858	11.78 ± 5.28	0.9988 ± 0.0712	2.51 ± 1.05	0.3710 ± 0.2858	15.63 ± 7.10
IGSR	0.5130 ± 0.2066	9.26 ± 4.78	0.9938 ± 0.0554	1.28 ± 1.07	0.3278 ± 0.2650	14.53 ± 7.52
RESTART	0.3314 ± 0.2861	12.15 ± 4.42	1.0094 ± 0.0634	1.25 ± 0.62	0.3848 ± 0.3043	14.89 ± 7.02
Baselines with Structural Priors						
LLM-SR w/ SP	0.3294 ± 0.2911	12.35 ± 3.90	0.9918 ± 0.1006	27.90 ± 18.78	0.3762 ± 0.3060	15.05 ± 8.96
FunctionEvolve w/ SP	0.2926 ± 0.2554	13.34 ± 4.69	1.2695 ± 0.5205	14.29 ± 20.70	0.3210 ± 0.3490	19.88 ± 8.44
PiT-PO w/ SP	0.3403 ± 0.2765	12.97 ± 6.73	0.9961 ± 0.0857	25.49 ± 20.78	$\underline{0.2780 \pm 0.1421}$	17.12 ± 7.38
RESTART w/ SP	$\underline{0.2883 \pm 0.2342}$	12.89 ± 5.63	0.9956 ± 0.0737	27.77 ± 19.45	0.4924 ± 0.4352	15.32 ± 9.64
NetSPE	$\mathbf{0.0084 \pm 0.0186}$	$\mathbf{83.69 \pm 31.05}$	$\mathbf{0.8256 \pm 0.0834}$	$\underline{29.24 \pm 18.69}$	$\mathbf{0.2395 \pm 0.1451}$	$\mathbf{25.56 \pm 6.39}$

Table 8: Out-of-distribution performance comparison on Synthetic Benchmark. Bold and underlined values indicate the best and second-best mean results, respectively.

Methods	NMSE ↓ (OOD)	$ACC_{0.1}$ (%) ↑ (OOD)
ND^2	$\underline{0.0281 \pm 0.0504}$	$\underline{73.77 \pm 35.52}$
FunctionEvolve	0.3774 ± 0.3810	14.99 ± 8.76
IGSR	0.5598 ± 0.2190	9.38 ± 4.72
PiT-PO	0.3881 ± 0.2930	12.04 ± 5.22
RESTART	0.3655 ± 0.3159	12.11 ± 5.43
LLM-SR	0.3525 ± 0.2922	13.34 ± 5.06
LLC	0.4268 ± 0.3503	20.87 ± 21.78
NetSPE	$\mathbf{0.0119 \pm 0.0302}$	$\mathbf{85.29 \pm 31.44}$

F.4 Test Error Threshold Coverage

While the main results report macro-averaged NMSE, aggregate statistics can obscure how performance varies across tasks and runs. We therefore evaluate the three methods across a range of error tolerances in Figure 5. The horizontal axis shows the test-NMSE threshold τ on a logarithmic scale, and the vertical axis reports the task balanced success rate, $S(\tau) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{R_t}\sum_{r=1}^{R_t}\mathbf{1}[\mathrm{NMSE}_{t,r}^{\mathrm{test}} \le \tau]$, where T is the number of tasks and R_t is the number of runs for task t. Thus, each task contributes equally regardless of its number of runs, and higher curves indicate better performance at a given tolerance. NetSPE achieves higher success rates over most of the displayed threshold ranges across all three datasets. On Synthetic, it attains approximately 67% success at $\tau = 10^{-8}$, compared with 50% for ND^2 and 0% for LLM-SR. On Bacteria Growth and Epidemic Transmission, its curve is similarly shifted toward lower errors. These results support the main findings by demonstrating that NetSPE's advantage persists across a range of accuracy requirements, rather than being captured solely by a lower mean error.

F.5 Complete Results of Generated Equations on Synthetic Benchmarks

We present complete results of the equations generated by all methods, together with the ground truth, for all ten synthetic problems in Table 9 and 10. Here, $\sum_j$ denotes $\sum_{j\in\mathcal{N}(i)}$, $d_i = |\mathcal{N}(i)|$, $\rho(u,p) = u^p/(1+u^p)$, and $\sigma(u) = (1+e^{-u})^{-1}$. Shaded rows highlight NetSPE. Coefficients are displayed to four decimal places; values already rounded in the source retain their original precision.

Table 9: Generated equations (1st run) on synthetic problems: CR, FHN, GR, HCR, and LV.

Problem	Method	Equation
CR	Ground truth	$-\omega_i y_i - z_i + 0.5000 \sum_j \sin(x_j - x_i)$
	NetSPE	$-z_i + \sum_j [-1.1400 y_i + 0.5000 \sin(x_j - x_i) + y_i \omega_j]$
	LLM-SR	$-0.6700\omega_i y_i - 2.6200\omega_i z_i + 1.4900\omega_i - 0.0400 x_i y_i + 0.5500 x_i z_i + 0.1000 x_i + 0.6300 y_i z_i - 0.4000 y_i - 1.8600 z_i$
	FunctionEvolve	$\frac{-0.0500(1 - 4.9500e^{0.1900x_i})(\omega_i + 0.3000)(-0.4200\omega_i x_i + 1.5000 x_i z_i - y_i - 20.0700 z_i + 6.9700e^{0.1900x_i})e^{-0.1600x_i + 0.1000z_i}}{z_i}$
	PiT-PO	$-0.8900\omega_i y_i - 0.9600\omega_i z_i + 0.7900\omega_i + 0.0100 x_i y_i + 0.1300 x_i z_i + 0.0600 x_i + 0.0200 y_i^2 + 0.1300 y_i - 0.8500 z_i - 0.5000$
	ND^2	$(\omega_i \cdot -0.8396)(z_i + y_i)$
	LLC	$-0.251697 x_i - 0.251697 y_i + \sum_j A_{ji} [-0.102996 y_i]$
	IGSR	$4.5327 + 0.1991 \lvert x_i - y_i \rvert + 0.6543 x_i \sin\omega_i - 8.4806 z_i^{0.5000} \sin\omega_i$
	RESTART	$0.0825 u_1 + 0.0501 x_i - 0.5946 z_i - 1.4513 e^{0.2638u_0 - 0.2571 \tanh(0.1938 + 0.4406 y_i + 0.2369 x_i)} - 0.0758 \log(\lvert 0.3774 + 0.8580 z_i + 0.4612 y_i + 0.2778 x_i \rvert) - 0.4063 y_i - 0.9965\omega_i + (-0.4406 + 0.8580\omega_i)(0.1531 u_1 + 0.0930 x_i - 1.1037 z_i - 0.7540 y_i - 1.8496\omega_i)$, $u_0 = \lvert -0.5136 + \omega_i \rvert$, $u_1 = 25.0194 e^{0.5136u_0 - 0.5006 \tanh(0.1938 + 0.4406\omega_i + 0.2369 z_i + 0.1426 y_i + 0.0389 x_i)}$
FHN	Ground truth	$x_i - x_i^3 - y_i - \frac{1}{d_i} \sum_j (x_j - x_i)$
	NetSPE	$x_i - x_i^3 - y_i - 0.1100 \sum_j (x_j - x_i)$
	LLM-SR	$0.5000 x_i^2 y_i - 0.1000 x_i^2 + 0.2100 x_i y_i^2 - 1.6900 x_i + 0.0300 y_i^2 - 0.7500 y_i + 2.2500 \sin x_i - 0.6500 \sin y_i - 0.0400 \cos x_i + 0.2000 \cos y_i$
	FunctionEvolve	$\frac{3.2500 x_i^3 e^{2.4400y_i} + 0.5400 x_i y_i e^{2.4400y_i} + 3.2700 x_i e^{2.4400y_i} - 5.0000 y_i^2 (x_i + 2.8700)^5 e^{-0.2300x_i} - 2.1900 e^{2.4400y_i}}{1 - 9.6200 e^{2.4400y_i}}$
	PiT-PO	$-9.8000 x_i - 0.7700 y_i - 0.2000 e^{x_i} + 0.0600 e^{y_i} + 8.2500 \sin x_i + 0.2500 \sin(x_i y_i) + 0.7800 \cos y_i - 0.4500 \cos(x_i y_i) + 3.0100 \sinh x_i + 0.0400 \cosh y_i$
	ND^2	$x_i - y_i - x_i^3 - 0.1111 \sum_j [x_j - (x_i + x_i(y_j - y_j))]$
	LLC	$-0.163636 x_i (0.625096 x_i + 0.902429)(x_i - 0.976232) - 0.163636 y_i + \sum_j A_{ji} [-0.0918237 y_i - 0.0918237 x_j]$
	IGSR	$-0.1484 + 0.5788 \sin x_i + 0.3255 \cos y_i + 0.0264 \lvert x_i - y_i \rvert - 0.4920 x_i - 0.4347 y_i$
	RESTART	$-0.0323 + 0.6231 x_i + 0.0939 y_i^2 - 0.7286 x_i^2 - 1.0037 y_i - 0.1740 \log(0.3199 + 0.5840 u_1) + 0.8225 x_i \lvert u_0 + 1.6155 x_i \rvert - 0.0496 u_1 x_i$, $u_0 = 1.6597 + 0.5813 y_i$, $u_1 = \lvert u_0 - 37.1943 x_i \rvert$
GR	Ground truth	$0.2000 - 0.9000 x_i + 2 \sum_j \rho(x_j / 1.5000, 2)$
	NetSPE	$0.2007 - 0.8997 x_i + 16.0674 \sum_j \tanh(0.1251 \rho(0.6641 x_j, 2)) + 2.2021 \times 10^{-4} \sum_j \cos(1.6325(x_j - x_i))$
	LLM-SR	$0.1700 x_i^2 - 3.1700 x_i + 0.0900 \sin(4.6100 x_i - 62.6000) + 27.4000 - \frac{47.0400}{0.5000 x_i^2 + 1.9900}$
	FunctionEvolve	$\frac{\left[-10.2500 x_i^3 \sin(298.5500 x_i - 210.7700) + 0.6600 x_i^3 + 0.8700 x_i^2 + 6.2600 x_i + 26.8700 \sin(298.5500 x_i)\right] e^{1.9300 x_i}}{x_i^3 + 2.3600}$
	PiT-PO	$-0.1600 \sin x_i + 0.0100 \cos x_i - 38.5100 \tanh x_i - 38.3200 \arctan x_i + 103.7700 - 105.5100 e^{-x_i}$
	ND^2	$x_i - 0.9504(x_i + x_i) + 2.0206 \sum_j \rho\big((0.6738 x_j)^2, 0.9880\big)$
	LLC	$x_i (x_i (0.00126805 x_i - 0.0257637) - 0.755585) + \sum_j A_{ji} \left[x_j \left(-4.39628 \cdot 10^{-5} x_i^2 + \frac{2.02009}{x_j + \frac{2.29101}{x_j}} \right) \right]$
	IGSR	$23.2995 + 0.3774 \max(0, x_i) + 88.7947 \log(1 + x_i) + 0.2413 x_i^2 - 0.0045 x_i^3 - 1.1216 \tanh x_i - 78.9532 \lvert x_i \rvert^{0.5000}$
	RESTART	$u_1 + u_4 + 8.6417 \log(1 + 1.5004 u_2) + 3.6710 x_i - 7.4060 u_2 + u_2 (3.6710 + 0.3993 u_3) + u_3 u_4 + u_3 (-7.4060 + u_1)$, $u_0 = \lvert -0.9532 + x_i \rvert$, $u_1 = 0.3993 u_0$, $u_2 = u_0^{1.7517}$, $u_3 = \log(1 + u_2)$, $u_4 = 1.4072 + 41.9092 \log(1 + 0.0049 u_2)$
HCR	Ground truth	$-y_i - z_i + 0.5000 \sum_j \sin(x_j - x_i)$
	NetSPE	$-(y_i + z_i) + 0.5000 \sum_j \sin(x_j - x_i)$
	LLM-SR	$0.0100 x_i^2 + 0.0300 x_i y_i - 0.0200 x_i z_i + 0.0900 x_i + 0.0200 y_i^2 - 0.1300 y_i z_i - 0.7000 y_i + 0.0100 z_i^2 - 1.1000 z_i + 0.3900 \sin(x_i + y_i + z_i)$
	FunctionEvolve	$\frac{1.0000 x_i^2 + 0.0900 x_i z_i - 2.2900 y_i z_i - 0.1600 y_i - z_i^2 - 0.6700(z_i + 0.4400) e^{0.1100x_i} - 0.1200(z_i + 0.4400) e^{0.5600y_i} + 0.0200}{z_i + 0.4400}$
	PiT-PO	$0.1300 x_i - 0.7700 y_i - 0.9300 z_i + \tanh(1.5500 x_i + 0.1000 y_i - 58.3600 z_i + 18.5600) + 0.7100$
	ND^2	$-1.0000 z_i - y_i + 0.5000 \sum_j \sin(x_j - x_i)$
	LLC	$-0.228209 y_i - 0.228209 z_i + 0.415276 + \sum_j A_{ji} [-0.113633 y_i - 0.113633 z_i]$
	IGSR	$-0.6699 + 0.6832 \max(x_i, z_i) + 0.6832 \sin y_i - 0.0159 e^{-x_i} - 0.0159 \tanh z_i - 0.7190 x_i - 0.7190 y_i - 0.7190 z_i - 0.0159 x_i y_i$
	RESTART	$-0.0348 + 0.9539 \tanh(0.9539 x_i + 0.9539 y_i) - 0.9866 y_i - 0.9195 z_i - 0.1667 x_i + 0.0019 x_i y_i + 0.0000 z_i x_i^3 y_i^2 + 0.0034 x_i y_i z_i + 0.0020 y_i z_i x_i^2 + 0.0005 z_i x_i^2 y_i^2$
LV	Ground truth	$x_i(\alpha_i - \theta_i x_i) - x_i \sum_j x_j$
	NetSPE	$-0.0700 + 0.2700 x_i + 0.4500\alpha_i - 0.5700\theta_i x_i - 1.0500 x_i \sum_j x_j$
	LLM-SR	$1.8600\alpha_i^2 + 0.8200\alpha_i\theta_i x_i + 2.3700\alpha_i \sin\theta_i + 2.8200\theta_i^2 - 1.9900\theta_i \cos x_i - 0.0600 x_i^2 - 1.4600 x_i$
	FunctionEvolve	$-[2.8000\theta_i x_i e^{4.1200\alpha_i} + 5x_i e^{4.1200\alpha_i} + 0.3900 \log(0.4800 x_i + 1)] e^{5x_i} \sin(2.8900\alpha_i - 35.8200) \sin(6.2900\theta_i - 44.8400)$
	PiT-PO	$2.0500 \sin(\alpha_i\theta_i) + 0.4800 \sin(\alpha_i x_i) - 1.6100 \sin(\theta_i x_i) - \sin(2.0100\alpha_i + 5.1000\theta_i + 1.0800 x_i) + \cos(-0.3500\alpha_i + 3.4600\theta_i + 1.9700 x_i)$
	ND^2	$-0.1817 \sin\theta_i - x_i \sum_j x_j + 0.3509\alpha_i + 1.4045 \sin(x_i(\alpha_i^3)^3) + 0.0063$
	LLC	$\frac{x_i(\alpha_i - 0.444567) - 0.0573826}{\alpha_i + \theta_i} + \sum_j A_{ji} \left[\frac{\theta_i x_i^2}{-1.170620\theta_i x_j - 1.170620\theta_j} + 0.0203472 \right]$
	IGSR	$-1.4337 + 1.3584 \sin\theta_i + 0.3813 \cos\theta_i + 1.4213 e^{\theta_i} + 0.3890 x_i - 1.3584 e^{\alpha_i} - 1.4213 \sin x_i - 0.0076 \tanh\alpha_i - 2.2458\theta_i - 2.2458 \sin\alpha_i + 7.6948 \cos\theta_i \sin\alpha_i$
	RESTART	$1.0849\alpha_i^3 + 1.0849\theta_i^3 + 1.0849 x_i^3 + 0.3308\alpha_i + 0.0071\theta_i - 1.8100 x_i - 1.2524\alpha_i^2 - 1.2524\theta_i^2 - 1.2524 x_i^2 + 9.7640\theta_i\alpha_i^2 + 0.2919\alpha_i\theta_i + 0.1083\alpha_i x_i + 14.2965 x_i\alpha_i^2 + 14.2965 x_i\theta_i^2 - 1.1511\theta_i x_i - 10.9837\theta_i\alpha_i^3 - 10.9837 x_i\alpha_i^3 - 10.9837 x_i\theta_i^3 + 9.3336\alpha_i\theta_i x_i^2 - 13.0674\alpha_i\theta_i x_i - 2.8402\alpha_i\theta_i x_i^4 - 1.6097\alpha_i\theta_i x_i^3$

Table 10: Generated equations (1st run) on synthetic problems: MM, MP, SIS, WC, and KUR.

Problem	Method	Equation
MM	Ground truth	$-x_i + \sum_j \rho(x_j, 2)$
	NetSPE	$-x_i + \sum_j \rho(x_j, 2)$
	LLM-SR	$3.1700x_i^3 - 52.6900e^{0.4600x_i} + 4.0900\sin(2.3400x_i) + 2.1500\cos(2.8000x_i)$ $+ 91.9800\tanh(0.5300x_i) + 37.2400$
	FunctionEvolve	$\frac{\sqrt[3]{x_i}(1.5200 - 0.3000x_i)(x_i^2 - 2.5000) + \sqrt[3]{x_i}(x_i^2 - 2.5000) - 4.6800\sqrt[3]{x_i}\max(0, x_i - 1.5800) - 0.1900x_i^2 + 0.0600x_i(x_i^2 - 2.5000)\sin(5x_i + 6.1700) + 0.4700}{\sqrt[3]{x_i}(x_i^2 - 2.5000)}$
	PiT-PO	$-1.2300x_i^3 + 30.8100x_i^2 - 149x_i + 8.4500\sin x_i + 126.4900\sin(\cos x_i)$ $- 381.2900\sin(\cos(\sin x_i)) - 184.2300\cos x_i + 16.4700\cos(\sin x_i)$ $+ 273.5600\cos(\cos(\cos x_i)) + 194.5600$
	ND^2	$\frac{x_i}{x_i/(x_i - 1.0000)} + 1.0000\sum_j \rho(x_j^2, 1)$
	LLC	$0.0765322x_i^2(x_i - 6.37859) - 0.65799$ $+ \sum_j A_{ji}\left[-x_i x_j\,(0.00680372x_i - 0.0190487) + \frac{x_j}{x_j + \frac{1.06195}{x_j}}\right]$
	IGSR	$436.2393 + 8.0509\tanh(\sin x_i) + 100.1891\max\left(x_i, e^{-x_i}\right) - 524.1899\log(1 + x_i)$ $- 425.8621e^{-x_i} - 12.8030\cos x_i - 15.3024\sin x_i$
	RESTART	$-122.1029 + 0.5000\tanh(-3.4313 + 0.4752x_i) + \frac{123.9801}{(1+\lvert-2.4649+x_i\rvert)^{0.0023}}$ $- 0.8402\tanh(-1830.3317 + 1155.8298x_i)$
MP	Ground truth	$x_i(\alpha_i - \theta_i x_i) + x_i\sum_j \rho(x_j, 2)$
	NetSPE	$0.6400x_i - 0.8800\theta_i x_i^2 + \sum_j[-0.7800(x_j - x_i) + 0.8100x_j - 0.6900\alpha_j]$
	LLM-SR	$3.3600\alpha_i^2 + 32.1300\alpha_i + 30.8200\theta_i^2 - 36.8700\theta_i - 0.1400x_i^2 + 4.8500x_i$ $+ 5.3900e^{\alpha_i\theta_i} + 6.1000\sin(\alpha_i x_i) - 3.6000\cos(\theta_i x_i) - 44.4200\tanh(\alpha_i\theta_i x_i)$
	FunctionEvolve	$(1 - 1.4600\alpha_i)[50.3400\alpha_i\theta_i x_i - 0.0800\theta_i x_i^3 - 35.3100\theta_i e^{4.3500\alpha_i} + 15.7500\theta_i + 0.6300x_i^2 e^{-1.8200\theta_i} - 1.7600x_i e^{4.3500\alpha_i} + 18.2900e^{4.3500\alpha_i}]$
	PiT-PO	$2.8900\alpha_i^2 + 51.7700\alpha_i\theta_i - 44.4000\alpha_i + 83.4200\theta_i^2 - 140.4100\theta_i - 0.1000x_i^2$ $+ 0.5000x_i + 59.9500$
	ND^2	$-0.8630\theta_i x_i^2 + 0.8507\sum_j x_i + 0.8498\alpha_i x_i$
	LLC	$-0.519433\theta_i\,(6.91835 - x_i) + 1.11538 + \sum_j A_{ji}\left[\frac{0.710591}{\theta_i}\right]$
	IGSR	$25.9657 + 96.6137\alpha_i - 0.4452\lvert x_i\rvert - 119.2042\sin\alpha_i - 22.1454\theta_i$
	RESTART	$-u_3 + u_0 u_3 + u_0\tanh(-14831.2182 + u_1) + e^{-0.8221\lvert-0.4600+\theta_i\rvert}$, $u_0 = \tanh(-4.2519 + 9.0536\theta_i + 11.7607\alpha_i - 0.0704x_i),\ u_1 = 1.1515x_i$ $+ 31441.7672\theta_i + 33519.4984\alpha_i,\ u_2 = -14833.7182 + u_1,\ u_3 = u_2 - 1.0839u_0 u_2$
SIS	Ground truth	$-\delta_i x_i + \sum_j(1 - x_i)x_j$
	NetSPE	$-\delta_i x_i + \sum_j(x_j - x_i x_j)$
	LLM-SR	$-0.0700\delta_i^2 - 0.0300\delta_i x_i - 0.6300\delta_i - 57.8100x_i^2 + 149.8700x_i + 0.0200e^{\delta_i}$ $+ 1.8300e^{x_i} + 83.2300\sin x_i - 0.4100\cos\delta_i - 219.1300\tanh x_i$
	FunctionEvolve	$\frac{(1.5800 - \delta_i)(\delta_i - 1.8000)(\delta_i^2 + 1.1300) - 3.9300(-0.1600\delta_i^2 + \delta_i x_i - 0.2700\delta_i - 1.7800x_i + 0.9900)e^{0.6800\delta_i}}{(\delta_i - 1.8000)(\delta_i^2 + 1.1300)}$
	PiT-PO	$[0.2700\delta_i^2 - 6.7600x_i^2 - 9.5300\sin(0.1800\delta_i - 1.1000x_i) + 2.8700\cos(0.2000\delta_i + 2.2500x_i)]e^{0.3500\delta_i - 2.0300x_i}$
	ND^2	$-1.0000x_i\delta_i - 1.0000\sum_j x_j\,[x_i - \rho(x_i - x_i, -0.0175)]$
	LLC	$(x_i - 0.0724137\delta_i)\left(\delta_i + \frac{-x_i - 0.344387}{\frac{x_i}{\delta_i} + 0.0481326} - 0.133258\right)$ $+ \sum_j A_{ji}\,[-0.614529x_i - 0.0605977\delta_i + 0.307264x_j + 0.428326]$
	IGSR	$-4.1961 + 5.0915\tanh(\min(\delta_i, \log(1 + \lvert\delta_i\rvert))) + 11.0280e^{-\max(\delta_i, x_i)^{0.5000}}$ $- 3.6891\sin(\min(\delta_i, x_i))$
	RESTART	$22.4955 - u_0 + 0.1146\,(-6.4211 + 0.7165x_i)^3 + 4.1797\delta_i + 0.2679u_1^2$ $+ 0.0000e^{-u_0 + 4.1797\delta_i} - 121.0339\tanh(0.0229\delta_i + 0.0536x_i) + 0.3000\delta_i u_1$, $u_0 = 4.4264x_i,\ u_1 = -6.4211 + 0.4486x_i$
WC	Ground truth	$-x_i + \sum_j \sigma(5.1000(x_j - 1))$
	NetSPE	$-x_i + \sum_j \sigma(5.1000(x_j - 1))$
	LLM-SR	$-0.0100x_i^5 + 0.1000x_i^4 - 0.0100x_i^3 - 1.2500x_i^2 + 1.4900x_i + 0.0100e^{1.3000x_i}$ $+ 0.1500\sin(6.0000x_i) + 2.1200$
	FunctionEvolve	$[6.2000x_i^2 + 1.5600x_i - 0.1600\sin(1.1000x_i)\sin(0.1100x_i + 0.2600) - 0.1500\sin(0.1100x_i + 0.2600) + 2.7600]e^{4.8600x_i}$
	PiT-PO	$-0.0100e^{x_i} - 19.4900\sin^3 x_i + 127.8200\sin x_i - 128.0200\sin(\sin x_i)$ $- 36.1500\sin(\cos x_i) - 0.6600\cos^4 x_i - 5.5200\cos^3 x_i - 0.4800\cos^2 x_i$ $+ 37.0700\cos x_i + 2.4800\cos(\sin x_i)$
	ND^2	$-x_i + \sum_j \sigma(5.1000(x_j - 1.0000))$
	LLC	$(0.288571x_i - 2.36968)\,(x_i - 1.39606) + \sum_j A_{ji}\left[\frac{x_i\left(-0.859136 + \frac{9.76407}{x_i + \frac{5.38817}{x_j}}\right)}{x_j + 1.1008}\right]$
	IGSR	$3.9647 + 9.4542\lvert x_i\rvert + 5.6994\sin x_i + 0.0454x_i^3 - 0.0562e^{x_i} - 23.6820\log(1 + x_i)$ $+ 1.7248\cos x_i\sin x_i$
	RESTART	$10.0334x_i + 5.0267\cos x_i + 3.7607e^{-(-2.1118 + 1.1611x_i)^2} - 22.7421\sin x_i$ $- 2.1055x_i^2 + \frac{0.0791 + 7.9099x_i^2}{1 + (0.0100 + x_i^2)^2} + 5.0004x_i\sin x_i + 2.1652x_i\cos x_i$
KUR	Ground truth	$\omega_{0,i} + \sum_j \sin(x_j - x_i)$
	NetSPE	$\omega_{0,i} + \sum_j \sin(x_j - x_i)$
	LLM-SR	$0.0544x_i + 0.8855\omega_{0,i} + 0.0079x_i^2 + 0.0111\omega_{0,i}^2 + 0.0018x_i\omega_{0,i} - 0.0764\sin x_i$ $- 0.0690\cos\omega_{0,i} - 0.0145e^{x_i} - 0.0026e^{\omega_{0,i}} - 0.0547$
	FunctionEvolve	$\frac{-89.3990\omega_{0,i} + 9.7961}{1.8998\omega_{0,i} - 0.2085} + 63.4057e^{0.0137\omega_{0,i}} + \frac{-16.4099x_i + 50.5110}{x_i - 3.0781}$
	PiT-PO	$0.0461x_i + 0.8856\omega_{0,i} - 0.0553 - 0.0064x_i^2 + 0.0032\omega_{0,i}^2 - 0.0850\sin x_i$ $- 0.1385\cos\omega_{0,i} - 0.0001x_i^3 - 0.0008\omega_{0,i}^3$
	ND^2	$1.0000\omega_{0,i} - 1.0000\sum_j \sin(x_i - x_j)$
	LLC	$-0.000409467x_i + \omega_{0,i}$ $+ \sum_j A_{ji}\left[(-x_i + x_j)\,(x_i x_j\,(0.011723x_i x_j + 0.135844) + 0.225217)\right]$
	IGSR	$-0.0063 + 2.7541\tanh\omega_{0,i} - 1.1613\sin\omega_{0,i}$
	RESTART	$0.0636e^{-0.0516x_i} + 0.0004x_i$ $+ 0.1814\tanh\left(-724.3731 + 19.9087e^{-0.0516x_i} + 313.1529\omega_{0,i}\right) + 0.8224\omega_{0,i}$ $+ 0.0067\tanh 0.5000\omega_{0,i}x_i$

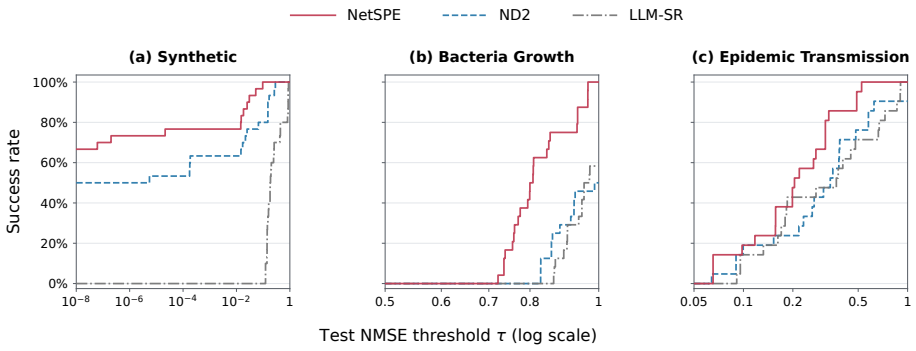


Figure 5: Test-NMSE threshold coverage on (a) Synthetic, (b) Bacteria Growth, and (c) Epidemic Transmission. Success rate is the fraction of runs with test NMSE below or equal to threshold τ, averaged equally across tasks. The horizontal axes use logarithmic scales. Higher curves indicate better performance at a given error tolerance.

G Prompt Templates

Prompt excerpt: Local structure editing

```
You edit one local subtree of a typed symbolic equation for graph
↪  dynamics.

Task: {task.graph_aware_description}
Target: {task.target_name}

Rendered current equation:
  target_i = {parent.canonical_expression}

The root protocol, aggregation cardinality, aggregation operator, and
↪  every
non-selected slot are read-only. Never return the full equation.

Residual diagnosis:
  V_self={branch_report.v_self}
  V_interaction={branch_report.v_interaction}
  V_joint={branch_report.v_joint}
  C_self={branch_report.c_self}
  C_interaction={branch_report.c_interaction}

Selected residual branch: {branch_report.selected_branch}
Selected editable region: {branch}
Selected subtree ID: {target.node_id}
Selected subtree: {target.expression}
Required replacement type: {target.expr_type}
Subtree contribution: {target.contribution}

Assigned edit mode: {edit_mode}
{mode_instruction}
The JSON operation field must still be "replace"; edit mode describes
↪  the required
structural intent for the replacement expression.

Allowed variables in this branch: {', '.join(allowed_variables)}
{allowed_operator_text}

Return exactly one JSON object and no markdown:
```

```
{"target_node_id":"{target.node_id}","operation":"replace","replaceme⌋
↪  nt":"<expression>","share_parameter":false,"rationale":"<short
↪  reason>"}
```

Prompt excerpts: Edit-mode instructions

```
GROW: retain the selected subtree's useful signal and add one
↪  meaningful variable relationship, term, or nonlinear layer. Do not
↪  merely multiply it by another fitted parameter.

REPLACE: change its operator, function, or variable relationship while
↪  keeping comparable scale. A parameter rename or algebraically
↪  equivalent rewrite is not a replacement.

SIMPLIFY: return a lower-complexity expression with fewer AST nodes.
↪  Remove a weak term, redundant operation, or unnecessary nonlinear
↪  layer; do not add structure.

SELF-DYNAMICS MOTIF: propose a concise self-dynamics pattern
↪  appropriate for the selected type and available node features,
↪  inspired by forms such as {', '.join(motif_examples)}. Adapt the
↪  motif instead of copying a parameter-only variant.
```

Prompt excerpt: Structural Prior of Synthetic

```
Root protocol: synthetic_additive
Locked root template: SELF_SLOT + AGG(MESSAGE_SLOT)

The root protocol, aggregation cardinality, aggregation operator, and
↪  every
non-selected slot are read-only. Never return the full equation.
```

Prompt excerpt: Structural Prior of Bacteria Growth

```
Root protocol: bacteria_glv
Locked root template: x_i * (c0 + receiver_growth_i + c1*x_i +
↪  CORRECTION_SLOT + AGG(INTERACTION_MESSAGE_SLOT))

Invariant: the locked outer x_i factor cannot be removed, so x_i=0
↪  must imply target_i=0.

Locked fitted GLV backbone: c0 + c1*x_i and its optional ridge-shrunk
↪  receiving-node growth deviation are not editable. The local slot
↪  is an additive correction only.
```

Prompt excerpts: Structural Prior of Epidemic Transmission

```
Root protocol: epidemic_local
Locked root template: READOUT_SLOT(local node variables)

Root protocol: epidemic_graph_readout
Locked root template: z_i = AGG(MESSAGE_SLOT); target_i =
↪  READOUT_SLOT(local variables, z_i)
```