

---

# Gradient-Informed Temporal Sampling Improves Rollout Accuracy in PDE Surrogate Training

---

Wenshuo Wang<sup>1</sup>, Fan Zhang<sup>2,3\*</sup>

<sup>1</sup> School of Future Technology, South China University of Technology, China

<sup>2</sup> State Key Laboratory of Ocean Sensing & Ocean College, Zhejiang University, China

<sup>3</sup> Kavli Institute for Astrophysics and Space Research, Massachusetts Institute of Technology, USA  
202364870251@mail.scut.edu.cn, f.zhang@zju.edu.cn

## Abstract

Researchers train neural simulators on uniformly sampled numerical simulation data. But under the same budget, does systematically sampled data provide the most effective information? A fundamental yet unformalized problem is how to sample training data for neural simulators so as to maximize rollout accuracy. Existing data sampling methods either tend to collapse into locally high-information-density regions, or preserve diversity but remain insufficiently model-specific, often leading to performance that is no better than uniform sampling. To address this, we propose a data sampling method tailored to neural simulators, Gradient-Informed Temporal Sampling (GITS). GITS jointly optimizes pilot-model local gradients and set-level temporal coverage, thereby effectively balancing model specificity and dynamical information. Compared with multiple sampling baselines, the data selected by GITS achieves lower rollout error across multiple PDE systems, model backbones and sample ratios. Furthermore, ablation studies demonstrate the necessity and complementarity of the two optimization objectives in GITS. In addition, we analyze the successful sampling patterns of GITS as well as the typical PDE systems and model backbones on which GITS fails.

## 1 Introduction

Neural simulators are trained on continuous-time physical-field trajectories generated by numerical solvers, and then serve as surrogates for PDE evolution [1, 2]. In existing practice, researchers usually train on temporally contiguous data sampled at equal time intervals [3, 4]. However, a persistent yet never systematically studied problem is that such systematic sampling does not always provide the most informative or the most dynamics-relevant training pairs. This raises a basic question: given an existing pool of numerical simulation data, how should one sample from it so that a neural simulator trained under the same budget achieves the best rollout performance? In this paper, we study this question in the offline shared temporal start-index setting, where a common subset of admissible start indices is selected once from a fixed, fully generated trajectory dataset and then reused across all training trajectories. As model capacity, parameter scale, and training techniques continue to improve, the quality and informativeness of the training data itself become increasingly important to the final performance of neural simulators [5, 6].

One of the most widely used families of data valuation methods is pilot-model, gradient-driven subset selection [22, 8, 7]. Such methods can effectively estimate the information gain of individual samples for a model, but they do not explicitly account for information redundancy among samples [9, 10]. As a result, when applied to spatiotemporal data such as PDE trajectories, they tend to select densely clustered neighboring time steps [11, 12]. On the other hand, common coverage-based subset

---

\*Corresponding author.

selection methods face the opposite limitation: they balance diversity at the set level, but are largely model-agnostic, and therefore cannot identify which temporal windows are intrinsically the most useful for downstream neural surrogate training [24, 13–15].

To address this gap, we propose **Gradient-Informed Temporal Sampling (GITS)**, a sampling method tailored to offline PDE surrogate training on pre-generated simulation data. GITS jointly optimizes two complementary objectives: (1) a low-cost pilot-model short-horizon gradient-norm score that serves as a practical local proxy for candidate usefulness, and (2) a set-level temporal coverage objective that encourages the selected subset to cover the temporal axis rather than collapse onto highly redundant neighboring regions. Together, they overcome the respective weaknesses of pure score-based ranking and pure coverage-based selection.

Using three representative PDE systems from PDEBench, we show that data selected by GITS leads to lower rollout error than a broad set of temporal selection baselines across several mainstream neural simulator architectures and sample ratios [1]. Furthermore, our ablation studies demonstrate necessity of the two components of GITS. Finally, we conduct detailed analyses of the sampling patterns through which GITS reduces error relative to the baselines, as well as the failure boundaries in the minority of cases where it underperforms. In summary, the contributions of this paper are:

- We formalize and explicitly study the overlooked problem of offline shared temporal-window selection for autoregressive neural PDE surrogate training under a fixed budget.
- We propose GITS, which combines a pilot-model short-horizon gradient proxy with set-level temporal coverage regularization, and show that it improves over strong temporal selection baselines on representative PDEBench tasks.
- We clarify when and why GITS helps or fails, showing that its behavior is governed by score–utility alignment and by whether temporal coverage is beneficial or counterproductive.

## 2 Related Work

### 2.1 Budgeted Temporal Window Selection for Neural PDE Surrogate Training

We formalize *budgeted temporal window selection for neural PDE surrogate training* as follows. Consider a collection of  $N$  generated PDE trajectories  $\{x_t^{(n)}\}_{t=0}^{T_c-1}$ , where  $x_t^{(n)}$  denotes the system state of trajectory  $n$  at coarse time step  $t$ , and  $T_c$  is the number of available coarse time steps after any preprocessing or temporal downsampling. Let  $L$  denote the history length used by an autoregressive surrogate to predict the next state. Then, each admissible temporal start index  $k$  induces a training window consisting of an input history  $x_{k-L+1:k}^{(n)}$  and a one-step target  $x_{k+1}^{(n)}$ , where  $x_{a:b}^{(n)}$  denotes the contiguous subsequence from time step  $a$  to  $b$ . Accordingly, the candidate start-index set is

$$\mathcal{C} = \{k \in \mathbb{Z} \mid L \leq k \leq T_c - 2\}.$$

Under a budget  $K$ , the task is to select a subset  $\mathcal{S} \subseteq \mathcal{C}$  with  $|\mathcal{S}| = K$ , which induces the training set

$$\mathcal{D}(\mathcal{S}) = \left\{ \left( x_{k-L+1:k}^{(n)}, x_{k+1}^{(n)} \right) \mid n = 1, \dots, N, k \in \mathcal{S} \right\},$$

such that a surrogate trained on  $\mathcal{D}(\mathcal{S})$  achieves the best possible downstream rollout accuracy under the same budget. Because the same start-index subset  $\mathcal{S}$  is reused across all trajectories, this is a *shared* temporal start-index formulation rather than a trajectory-specific selector. Unlike active data acquisition, this problem assumes that the full trajectories already exist: the question is not which simulations or timestamps to query, but which temporal windows to retain for training.

This problem is persistent because mainstream autoregressive neural PDE surrogate pipelines are typically trained on uniformly discretized trajectories, which treats all admissible temporal windows as equally useful for training. Recent data-efficiency efforts have begun to move beyond brute-force full-data usage, but mostly at other levels: active-learning frameworks for neural PDE solvers focus on selecting informative trajectories or parametric PDE instances in a solver-in-the-loop setting [16]; selective time-step acquisition chooses which states to generate during simulation [17]; continuous-time or irregular-time models aim to learn from non-uniform temporal grids [18]; and coreset-style approaches select informative inputs or samples for simulation and labeling [19]. To the best of our knowledge, prior work has not isolated the offline problem defined above as a standalone task.

---

**Algorithm 1** Gradient-Informed Temporal Sampling (GITS)

---

**Require:** Fully generated PDE trajectories  $\{x_t^{(n)}\}_{t=0}^{T_c-1}$  for  $n = 1, \dots, N$ , surrogate model class  $f_\theta$ , history length  $L$ , budget  $K$

**Ensure:** Selected start-index set  $\mathcal{S}$  with  $|\mathcal{S}| = K$

1: Construct the candidate start-index set

$$\mathcal{C} \leftarrow \{k \in \mathbb{Z} \mid L \leq k \leq T_c - 2\}$$

2: Compute pointwise informativeness scores  $\{s_k\}_{k \in \mathcal{C}}$

3: Construct the set-level temporal coverage terms

4: Greedily maximize the GITS objective under  $|\mathcal{S}| = K$

5: **return**  $\mathcal{S}$

---

## 2.2 Gradient-Based Data Valuation and Coverage-Based Subset Selection

Among the broad literature on training subset selection, the two technical families most relevant to our setting are gradient-based data valuation and coverage-based subset selection [7, 14].

Gradient-based data valuation methods can be broadly divided into local score-based ranking and gradient-driven subset selection [7]. GraNd uses gradient-derived local scores to identify training examples that are important early in optimization [20]. GRAD-MATCH selects subsets by matching the gradient of the selected subset to that of the full training or validation data [21]. GLISTER instead formulates data subset selection as a validation-oriented objective, providing a model-aware criterion for selecting effective subsets [22]. These methods are attractive because they offer computable proxies for data usefulness that depend on the model’s own optimization signal. However, for the budgeted temporal window selection task defined in the previous subsection, they do not explicitly address the strong redundancy that may arise among nearby temporal windows, and therefore cannot by themselves guarantee a temporally well-distributed selected subset.

Coverage-based subset selection methods can likewise be divided into classical representative coverage objectives and more guided subset selection objectives [23, 24]. Classical submodular formulations such as facility-location evaluate how well a selected subset covers the full ground set and admit efficient greedy optimization with approximation guarantees [23]. More recent guided formulations, such as PRISM, further extend this perspective by parameterizing submodular information measures to balance coverage with additional selection desiderata [24]. These methods are attractive because they provide principled set-level control over representativeness and diversity. However, when used alone for the task defined above, they remain largely model-agnostic: they can encourage temporally spread selections, but they do not indicate which temporal windows are intrinsically the most useful for downstream neural surrogate training.

Motivated by the complementarity of these two technical families, GITS combines gradient-based data valuation with coverage-based subset selection and optimizes them jointly.

## 3 Gradient-Informed Temporal Sampling

In this section, we first outline the workflow of GITS in Section 3.1, then detail its components in Sections 3.2 and 3.3, with the optimization objective in Section 3.4.

### 3.1 Overview of Gradient-Informed Temporal Sampling

Algorithm 1 summarizes the workflow of GITS. Line 1 defines the admissible temporal start indices on the coarse trajectory grid. Line 2 assigns each candidate a model-aware pointwise score using a lightweight pilot model and a short-horizon rollout gradient. This provides a practical local proxy for usefulness, but by itself may favor many neighboring start indices with highly redundant dynamics. Line 3 therefore adds two set-level temporal coverage terms: a global term that spreads selections over the candidate axis and a sliding-window term that reduces local temporal gaps. Line 4 then greedily maximizes the resulting joint objective under the budget constraint.

### 3.2 Pointwise informativeness estimation

To obtain low-cost model-aware candidate scores, GITS first trains a lightweight pilot model. Let  $\mathcal{P}$  denote the pilot start pool. In GITS, we set  $\mathcal{P} = \mathcal{C}$ , i.e., the pilot is trained on the full candidate pool for  $E_p$  epochs, and denote the resulting pilot parameters by  $\theta_p$ .

Let  $H$  denote the pilot short-rollout horizon. For each candidate start index  $k \in \mathcal{C}$ , we define the effective rollout horizon

$$H_k := \min(H, T_c - 1 - k),$$

so that candidates near the end of the trajectory remain well-defined. For trajectory  $n$ , we initialize the rollout with the ground-truth history window

$$\hat{x}_{k-L+1:k}^{(n)} = x_{k-L+1:k}^{(n)},$$

where  $L$  is the autoregressive history length. Starting from this history, we recursively predict for  $h = 1, \dots, H_k$ ,

$$\hat{x}_{k+h}^{(n)} = f_\theta \left( \hat{x}_{k+h-L:k+h-1}^{(n)} \right).$$

The candidate-specific short-rollout loss is

$$\ell_k(\theta) = \frac{1}{H_k} \sum_{h=1}^{H_k} \mathbb{E}_n \left[ \text{NRMSE} \left( \hat{x}_{k+h}^{(n)}, x_{k+h}^{(n)} \right)^2 \right],$$

where  $\text{NRMSE}(\cdot, \cdot)$  denotes normalized root-mean-square error, and  $\mathbb{E}_n[\cdot]$  denotes averaging over trajectories, or its mini-batch approximation in implementation.

At the pilot model, we compute

$$g_k := \nabla_\theta \ell_k(\theta) \Big|_{\theta=\theta_p}, \quad s_k := \|g_k\|_2,$$

and use  $s_k$  as the pointwise informativeness score of candidate  $k$ . Intuitively,  $s_k$  measures how strongly training on start index  $k$  would update the model around the pilot parameters, and thus serves as a practical local proxy for candidate usefulness.

### 3.3 Set-level temporal coverage regularization

High pointwise scores alone do not prevent the selected subset from collapsing onto densely clustered neighboring time steps. GITS therefore adds two set-level temporal coverage terms on subsets  $\mathcal{S} \subseteq \mathcal{C}$ .

We first define a global temporal coverage term. For any  $i, j \in \mathcal{C}$ , let

$$S_{ij} = \exp \left( -\frac{|i-j|}{\tau} \right),$$

where  $\tau > 0$  is the global temporal decay scale. The corresponding global coverage is

$$F_{\text{cov}}(\mathcal{S}) = \sum_{i \in \mathcal{C}} \max_{j \in \mathcal{S}} S_{ij}.$$

This term encourages every admissible temporal location to be close to at least one selected start index.

To further reduce local temporal gaps, we define a sliding-window coverage term. Let

$$\mathcal{I} = \{[a_m, b_m]\}_{m=1}^M$$

be temporal windows generated on the candidate axis with window size  $W$  and stride  $S_w$ , where  $M$  is the number of windows. For window  $m$  and candidate start index  $j$ , define the distance from  $j$  to the interval  $[a_m, b_m]$  by

$$d(m, j) = \begin{cases} 0, & j \in [a_m, b_m], \\ a_m - j, & j < a_m, \\ j - b_m, & j > b_m, \end{cases}$$

and define the window-level similarity

$$R_{mj} = \exp\left(-\frac{d(m, j)}{\tau_w}\right),$$

where  $\tau_w > 0$  is the local decay scale. The corresponding sliding-window coverage is

$$F_{\text{win}}(\mathcal{S}) = \sum_{m=1}^M \max_{j \in \mathcal{S}} R_{mj}.$$

The global term promotes temporal spread over the full candidate axis, while the sliding-window term improves local coverage regularity.

### 3.4 Joint objective and greedy optimization

Given the pointwise scores  $\{s_k\}_{k \in \mathcal{C}}$  and the two set-level coverage terms, GITS solves

$$\max_{\mathcal{S} \subseteq \mathcal{C}} F(\mathcal{S}) \quad \text{s.t.} \quad |\mathcal{S}| = K,$$

where  $K$  is the sampling budget, determined in experiments by the chosen sampling ratio. The objective is

$$F(\mathcal{S}) = \sum_{k \in \mathcal{S}} s_k + \lambda_{\text{cov}} F_{\text{cov}}(\mathcal{S}) + c_{\text{win}} F_{\text{win}}(\mathcal{S}),$$

where  $\lambda_{\text{cov}} \geq 0$  and  $c_{\text{win}} \geq 0$  are the weights of the global and sliding-window coverage terms, respectively. The first term favors individually informative start indices, while the latter two terms discourage redundant temporal concentration.

The objective admits an efficient greedy solution. The score term  $\sum_{k \in \mathcal{S}} s_k$  is modular, and both  $F_{\text{cov}}(\mathcal{S})$  and  $F_{\text{win}}(\mathcal{S})$  are monotone submodular facility-location terms. Therefore,  $F(\mathcal{S})$  is also monotone submodular, and the standard greedy algorithm yields a  $(1 - 1/e)$ -approximation under the cardinality constraint.

To compute greedy gains efficiently, we maintain the current coverage states

$$m_i := \max_{j \in \mathcal{S}} S_{ij}, \quad i \in \mathcal{C},$$

and

$$u_m := \max_{j \in \mathcal{S}} R_{mj}, \quad m = 1, \dots, M,$$

with  $m_i = 0$  and  $u_m = 0$  when  $\mathcal{S} = \emptyset$ . For any unselected candidate  $k \in \mathcal{C} \setminus \mathcal{S}$ , its marginal gain is

$$\Delta(k | \mathcal{S}) = s_k + \lambda_{\text{cov}} \sum_{i \in \mathcal{C}} (\max(m_i, S_{ik}) - m_i) + c_{\text{win}} \sum_{m=1}^M (\max(u_m, R_{mk}) - u_m).$$

GITS repeatedly selects

$$k^* = \arg \max_{k \in \mathcal{C} \setminus \mathcal{S}} \Delta(k | \mathcal{S}),$$

updates  $\mathcal{S} \leftarrow \mathcal{S} \cup \{k^*\}$  together with  $\{m_i\}$  and  $\{u_m\}$ , and stops when  $|\mathcal{S}| = K$ .

## 4 Empirical Evaluation

Our experiments are divided into three parts, designed to verify that:

- **Effectiveness:** GITS improves rollout accuracy for PDE surrogate training;
- **Necessity:** the individual components of GITS are necessary; and
- **Boundary Conditions:** the success and failure regimes of GITS.

## 4.1 Experimental setup

### 4.1.1 Datasets

We evaluate on three PDEBench forward tasks chosen to span distinct physical regimes and boundary conditions rather than multiple variants of the same PDE family [1]. Specifically, `diff-sorp` is a 1D scalar diffusion-sorption problem with Cauchy boundary conditions, `diff-react` is a 2D two-field reaction-diffusion system with Neumann boundaries, and `rdb` is the radial-dam-break instance of PDEBench’s 2D shallow-water benchmark, representing hyperbolic free-surface dynamics with shock-capable wave propagation.

According to PDEBench, these three tasks comprise 10,000, 1,000, and 1,000 trajectories, respectively [1]. Their benchmark resolutions are 1024 for `diff-sorp` and  $128 \times 128$  for both 2D tasks; `diff-react` and `rdb` each contain 100 temporal evolution steps (equivalently 101 stored snapshots when counting the initial frame), while `diff-sorp` has 100 temporal steps. `diff-react` contains two channels  $(u, v)$ , whereas `diff-sorp` and `rdb` are single-channel scalar-field prediction tasks.

### 4.1.2 Baseline selection and implementation details

At the backbone level, we evaluate four standard PDE surrogate backbones: U-Net, Fourier Neural Operator (FNO), ConvLSTM, and Transformer. All models are trained as one-step autoregressive predictors with history length  $L = 4$ . For U-Net, FNO, and Transformer, the  $L$  input frames are concatenated along the channel dimension; for ConvLSTM, the same history is fed as a sequence. Detailed architectural hyperparameters are provided in Appendix B.

At the sampling level, we compare GITS against six baselines spanning standard uniform sampling, pointwise pilot-based scoring, set-level temporal coverage, and stronger generic subset selection. The *uniform* baseline is the standard evenly spaced temporal sampler. The *loss-only* baseline trains a lightweight pilot model of the same backbone, scores each candidate start by its short-rollout loss under the pilot, and keeps the top- $K$  starts. The *coverage-only* baseline removes the local utility term from the GITS objective and optimizes only the temporal coverage terms. *GradMatch* is adapted from the original method to our shared temporal start-index setting [21]: each candidate start is treated as an item, represented by its pilot-model short-rollout gradient aggregated over training trajectories, and selection approximately matches the full-candidate gradient using greedy gradient matching.

We also include task-adapted GLISTER-style and PRISM-style baselines [22, 24]. The *GLISTER-style* baseline follows GLISTER’s validation-guided approximation, adapted from i.i.d. example selection to shared temporal-start selection for autoregressive PDE rollout training. The *PRISM-style* baseline instantiates a guided-submodular targeted-subset selector in the same setting, using validation-defined query/private sets and rollout-gradient CountSketch embeddings. All pilot-based samplers (*loss-only*, *GradMatch*, *GLISTER-style*, *PRISM-style*, and GITS) use the same pilot protocol as GITS, with the pilot start pool set to the full candidate set on the training trajectories. Detailed backbone settings and exact hyperparameter values for the pilot stage, the GITS objective, and the GLISTER-style and PRISM-style baselines are deferred to Appendix A and Appendix B.

All experiments were conducted on a server equipped with four NVIDIA A100(80GB) GPUs, using PyTorch 2.1 and CUDA 12.0. In our implementation, all models are trained with Adam (lr =  $10^{-3}$ , weight\_decay = 0). We use automatic mixed precision, batch size 64, gradient clipping with max\_norm = 1.0, normalized-space output clamping to  $[-10, 10]$ , and residual ( $\Delta$ -prediction) training for stability. Data are split at the trajectory level into 80%/10%/10% train/validation/test, and all reported results are averaged over three training seeds  $\{0, 1, 2\}$ . Models are trained for at most 100 epochs. Early stopping is based on validation rollout nRMSE, with a minimum of 10 epochs and patience 5. For readability, the main tables report only seed means; exact per-configuration standard deviations and seed-wise summaries are provided in Appendix D.

### 4.1.3 Experiment procedure and metrics

**Effectiveness:** For each dataset–backbone pair and each sampling ratio in  $\{0.05, 0.10, 0.20\}$ , we construct the candidate start-index set on the coarse time axis, select  $K$  training starts using each sampler, and train the surrogate on the resulting one-step autoregressive samples. The primary metric

Table 1: Test rollout nRMSE at sampling ratio 0.05. The best (lowest) value in each row is bolded.

Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
diff-react	ConvLSTM	0.480	0.504	0.240	0.472	0.549	0.435	<b>0.191</b>
	FNO	0.592	0.887	0.418	0.885	0.898	0.742	<b>0.361</b>
	Transformer	0.353	0.652	<b>0.264</b>	0.646	0.889	0.653	0.310
	U-Net	0.246	0.135	0.080	0.101	0.399	<b>0.053</b>	0.071
diff-sorp	ConvLSTM	0.229	0.712	0.381	0.410	0.386	0.369	<b>0.026</b>
	FNO	0.303	0.543	0.213	0.188	0.890	0.144	<b>0.139</b>
	Transformer	0.317	0.384	0.279	0.241	0.414	0.238	<b>0.152</b>
	U-Net	0.702	0.628	0.265	0.967	0.479	0.152	<b>0.145</b>
rdb	ConvLSTM	0.510	0.899	0.502	0.524	0.894	0.597	<b>0.463</b>
	FNO	0.181	0.764	0.361	0.984	0.636	0.466	<b>0.113</b>
	Transformer	0.673	0.888	0.210	0.897	0.654	0.619	<b>0.132</b>
	U-Net	0.220	0.492	0.792	0.264	0.891	0.336	<b>0.211</b>

is test rollout nRMSE:

$$\text{nRMSE} = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \sqrt{\frac{\sum_{t=1}^{T_r} \|\hat{\mathbf{x}}_t^{(i)} - \mathbf{x}_t^{(i)}\|_2^2}{\sum_{t=1}^{T_r} \|\mathbf{x}_t^{(i)}\|_2^2}}.$$

Here,  $N_{\text{test}}$  is the number of test trajectories,  $T_r$  is the rollout length,  $\hat{\mathbf{x}}_t^{(i)}$  and  $\mathbf{x}_t^{(i)}$  are the predicted and ground-truth states of the  $i$ -th trajectory at rollout step  $t$ , respectively, and  $\|\cdot\|_2$  denotes the Euclidean norm over all spatial locations and channels. Throughout the paper we evaluate on the full remaining post-history horizon, i.e.,  $T_r = T_c - L$  with  $L = 4$ , rather than on a separately shortened test horizon. We report the mean over three training seeds. Appendix C also reports PDEBench auxiliary metrics—cRMSE, bRMSE, and band-wise fRMSE (low/mid/high). To keep the main text focused on the most selection-sensitive regime, the main comparison table reports the strictest budget ratio 0.05, while the corresponding 0.10 and 0.20 results are reported in Appendix B. We additionally report downstream training time and subset-selection time.

**Necessity:** We use the ablation study to ask the finer mechanistic question of whether GITS requires two complementary objectives. To test this, we reorganize the ablation study along two axes. The first axis is the local utility score: `loss-only` uses the pilot model’s short-rollout loss as the candidate score, whereas `grad-only` keeps only the gradient-based score. The second axis is set-level temporal coverage: `loss-div` augments the loss-based score with the same global and sliding-window coverage terms used by the full method, while GITS combines these same coverage terms with the gradient-based score. All four variants are evaluated under the same datasets, backbones, ratios, candidate sets, and training protocol, using the same test rollout nRMSE defined above.

**Boundary Conditions:** We analyze the success and failure regimes of GITS under the same samplers, backbone settings, and training protocol. First, for representative success and failure configurations, we visualize the pilot score landscape together with the selected start sets of `loss-only`, `grad-only`, `loss-div`, and GITS, in order to examine whether success or failure is associated with local score concentration or with set-level temporal coverage. Second, on the same representative cases, we measure score–utility alignment by computing the pilot gradient norm and pilot rollout loss for candidate starts, and then evaluating their Spearman correlation with empirical single-start utility obtained from local probe updates and validation-rollout improvement. Together, these analyses characterize when the pilot scores are empirically aligned with downstream utility, as well as the regimes in which GITS fails despite using the same overall training pipeline.

## 4.2 Main results

### 4.2.1 Effectiveness of Gradient-Informed Temporal Sampling

Table 1 reports the strictest budget regime, ratio 0.05, across datasets and backbones. We foreground this regime in the main text because temporal subset selection should matter most when only a very small fraction of candidate windows can be retained; the corresponding results at ratios 0.10 and 0.20 are reported in Appendix B (Tables 11 and 12), where the same conclusion also holds.

Table 2: Ablation at sampling ratio 0.05.

Dataset	Backbone	loss-only	grad-only	loss-div	GITS
diff-react	ConvLSTM	0.504	0.891	0.474	<b>0.191</b>
	FNO	0.966	0.900	0.899	<b>0.361</b>
	Transformer	0.652	0.943	0.642	<b>0.310</b>
	U-Net	0.135	0.135	0.389	<b>0.071</b>
diff-sorp	ConvLSTM	2.017	2.017	1.101	<b>0.026</b>
	FNO	0.543	0.592	0.253	<b>0.139</b>
	Transformer	0.384	0.384	0.323	<b>0.152</b>
	U-Net	0.837	0.837	1.157	<b>0.145</b>
rdb	ConvLSTM	1.757	0.986	0.504	<b>0.463</b>
	FNO	1.318	5.409	1.597	<b>0.113</b>
	Transformer	1.469	1.528	1.632	<b>0.132</b>
	U-Net	0.492	0.757	0.259	<b>0.211</b>

At ratio 0.05, GITS achieves the lowest mean nRMSE across the 12 dataset–backbone configurations (0.193), compared with 0.334 for coverage-only, 0.400 for uniform, 0.548 for GradMatch, 0.624 for loss-only, 0.665 for GLISTER, and 0.400 for PRISM. GITS outperforms uniform, loss-only, GradMatch, and GLISTER in all 12/12 configurations, and outperforms both coverage-only and PRISM in 11/12. It attains the best row-wise result in 10/12 configurations; the two localized reversals are coverage-only on diff-react/Transformer and PRISM on diff-react/U-Net.

Across all 36 dataset–backbone–ratio configurations, GITS attains the best nRMSE in 27/36 cases and achieves the lowest overall mean nRMSE, 0.219. Relative to uniform, GITS reduces mean rollout nRMSE by 38.3% and outperforms uniform in 30/36 configurations. It also outperforms coverage-only in 33/36, GradMatch in 35/36, GLISTER in 33/36, and PRISM in 32/36 configurations. Thus, across the full 36-configuration benchmark in this offline shared-start setting, GITS is the strongest overall method.

The same qualitative ordering also appears in the PDEBench auxiliary metrics reported in Appendix C. In addition, selector-stage overhead remains modest relative to downstream training; detailed timing evidence is reported in Appendix Tables 15 and 16.

#### 4.2.2 Necessity of components of Gradient-Informed Temporal Sampling

Table 2 reports the ablation at sampling ratio 0.05, while the complementary ratios 0.10 and 0.20 are reported in Appendix B (Tables 13 and 14). Together, these three tables test the necessity of the two components of GITS: the pointwise utility signal and the set-level temporal coverage terms.

Across all 36 dataset–backbone–ratio configurations, GITS achieves the lowest nRMSE in every case. Averaged over all 36 configurations, the mean nRMSE is 0.916 for loss-only, 1.172 for grad-only, 0.692 for loss-div, and 0.219 for GITS. Thus, the gain of GITS cannot be explained by either component in isolation:

The temporal coverage terms are necessary. On the loss branch, adding temporal coverage (loss-div versus loss-only) improves 26/36 configurations and reduces mean nRMSE from 0.916 to 0.692, a relative reduction of 24.5%. On the gradient branch, adding the same coverage terms (grad-only versus GITS) improves all 36/36 configurations and reduces the mean nRMSE from 1.172 to 0.219, a relative reduction of 81.3%. Therefore, pointwise scoring alone is insufficient, and temporal coverage is especially important for turning gradient-based local signals into an effective training subset.

The gradient-based utility signal is also necessary. Under the same temporal coverage terms, replacing the loss-based pointwise score with the gradient-based one (loss-div  $\rightarrow$  GITS) improves all 36/36 configurations and reduces the mean nRMSE from 0.692 to 0.219, a relative reduction of 68.4%. By contrast, without temporal coverage, switching from loss to gradient alone helps in only 6/36 configurations, ties in 13/36, and is worse in 17/36, with a higher overall mean nRMSE (1.172 versus 0.916). Therefore, the advantage of GITS comes from combining a stronger gradient-based pointwise signal with explicit temporal coverage, rather than from either component alone.

Table 3: Geometry in the success and failure regimes of GITS.  $LO \cap GO$  denotes the overlap size between loss-only and grad-only, and  $LD \cap GITS$  denotes the overlap size between loss-div and GITS. Higher entropy and temporal coverage indicate more dispersed temporal selections.

Case	$K$	$LO \cap GO$	$LD \cap GITS$	Entropy (LO/GITS)	Coverage (LO/GITS)
Success: rdb/FNO/0.10	10	9/10	10/10	0.141 / 0.278	0.20 / 0.30
Failure A: diff-sorp/ConvLSTM/0.10	10	10/10	10/10	0.000 / 0.141	0.10 / 0.20
Failure A: diff-sorp/ConvLSTM/0.20	20	18/20	19/20	0.463 / 0.542	0.20 / 0.30
Failure B: diff-react/Transformer/0.05	5	0/5	2/5	0.000 / 0.311	0.20 / 0.40

Table 4: Score–utility alignment in representative regimes. Entries report three-seed mean Spearman correlations between pilot scores and empirical single-start utility from the local probe updates.

Case	Spearman(grad, utility)	Spearman(loss, utility)
Success: rdb/FNO/0.10	0.774	0.501
Failure A: diff-sorp/ConvLSTM/0.10	-0.641	-0.654
Failure A: diff-sorp/ConvLSTM/0.20	-0.369	-0.539
Failure B: diff-react/Transformer/0.05	0.876	0.936

### 4.2.3 Boundary conditions of Gradient-Informed Temporal Sampling

To characterize when GITS helps or fails, we examine two complementary diagnostics. Table 3 summarizes subset geometry in representative success and failure regimes, including pairwise overlaps, entropy, and temporal coverage of selected starts. Table 4 reports score–utility alignment, measured by the Spearman correlation between pilot scores and empirical single-start utility from local probe updates. Together, these diagnostics ask whether performance is governed mainly by the reliability of the local pilot signal, by the effect of the temporal coverage terms, or by their interaction.

In the representative success regime, the pilot scores are meaningfully aligned with downstream utility, and the gradient-based signal is stronger than the loss-based one. The geometry statistics further show that GITS preserves substantial overlap with the high-utility region while avoiding collapse onto a narrow cluster of neighboring starts. This is the intended operating regime of GITS: the gradient-based pointwise signal identifies informative candidate starts, and the temporal coverage terms turn these local scores into a better distributed training subset.

Failure A corresponds to pilot-signal misalignment. In this regime, score–utility correlations are weak or negative, indicating that neither the loss-based nor the gradient-based pilot score reliably tracks downstream usefulness. Once the local signal is unreliable, temporal coverage alone cannot recover the correct subset: it may spread the selection, but it cannot redirect it toward genuinely useful starts. This explains cases in which GITS is no longer superior despite using the global objective.

Failure B corresponds to over-dispersion under a highly concentrated utility landscape. Here the gradient-based pilot score remains positively aligned with downstream utility, but the most useful temporal windows are confined to a narrow region. Under a tight sampling budget, stronger temporal coverage can then dilute these key starts by pushing the selected subset too far away from the dominant peak. In this regime, the limitation is not poor pilot scoring, but rather that wider temporal spread is no longer the right inductive bias.

Taken together, these representative cases identify two distinct boundary conditions of GITS: GITS works best when the pilot signal is informative and moderate temporal spreading is beneficial; it weakens either when pilot scores fail to track utility, or when the useful windows are so concentrated that explicit temporal coverage becomes counterproductive.

## 5 Discussion

This paper makes two contributions. First, it identifies temporal subset selection as an important but underexplored problem in neural PDE surrogates: under a fixed training budget, the temporal placement of training windows can materially affect long-horizon rollout accuracy, and uniform sampling is not a robust optimum. Second, it proposes GITS, which combines a gradient-based pointwise utility proxy from a lightweight pilot model with set-level temporal coverage. In the offline

shared-start setting studied here, this combination yields the strongest overall performance among the compared samplers. More broadly, the paper shifts part of the question of improving neural simulators from model design to training-data construction under fixed simulation budgets.

The main limitation is that GITS relies on pilot-model local gradient scores as a proxy for downstream utility, and this proxy is not always reliable. Our boundary analyses indicate two main failure modes: score–utility misalignment, and over-dispersion when the useful temporal windows are already highly concentrated. In addition, we study only the shared temporal start-index setting, a fixed rollout protocol, and three representative PDEBench tasks rather than exhaustive PDE coverage. We therefore do not claim universal optimality. Instead, the main conclusion is narrower and more useful: in the offline shared-start setting, gradient-informed temporal sampling is an effective modest-overhead alternative to uniform sampling, with empirically identifiable boundary conditions.

## References

- [1] Takamoto, M., Praditia, T., Leiteritz, R., MacKinlay, D., Alesiani, F., Pflüger, D., & Niepert, M. (2022). PDEBench: An Extensive Benchmark for Scientific Machine Learning. In *Advances in Neural Information Processing Systems*, Vol. 35, pp. 1596–1611.
- [2] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). Fourier Neural Operator for Parametric Partial Differential Equations. *arXiv preprint arXiv:2010.08895*.
- [3] Ohana, R., McCabe, M., Meyer, L., Morel, R., Agocs, F., Beneitez, M., Berger, M., Burkhart, B., Dalziel, S., Fielding, D., et al. (2024). The Well: A Large-Scale Collection of Diverse Physics Simulations for Machine Learning. In *Advances in Neural Information Processing Systems*, Vol. 37, pp. 44989–45037.
- [4] McCabe, M., Régaldo-Saint Blancard, B., Parker, L., Ohana, R., Cranmer, M., Bietti, A., Eickenberg, M., Golkar, S., Krawezik, G., Lanusse, F., et al. (2024). Multiple Physics Pretraining for Spatiotemporal Surrogate Models. In *Advances in Neural Information Processing Systems*, Vol. 37, pp. 119301–119335.
- [5] Herde, M., Raonić, B., Rohner, T., Käppeli, R., Molinaro, R., De Bezenac, E., & Mishra, S. (2024). Poseidon: Efficient Foundation Models for PDEs. In *Advances in Neural Information Processing Systems*, Vol. 37, pp. 72525–72624.
- [6] Chen, W., Song, J., Ren, P., Subramanian, S., Morozov, D., & Mahoney, M. W. (2024). Data-Efficient Operator Learning via Unsupervised Pretraining and In-Context Learning. In *Advances in Neural Information Processing Systems*, Vol. 37, pp. 6213–6245.
- [7] Guo, C., Zhao, B., & Bai, Y. (2022). Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pp. 181–195. Springer.
- [8] Killamsetty, K., Durga, S., Ramakrishnan, G., De, A., & Iyer, R. (2021). Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pp. 5464–5474. PMLR.
- [9] Paul, M., Ganguli, S., & Dziugaite, G. K. (2021). Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34, 20596–20607.
- [10] Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J., & Agarwal, A. (2019). Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*.
- [11] Kim, Y., Kim, H., & Lee, J. (2025). Flexible Active Learning of PDE Trajectories. <https://openreview.net/forum?id=LgfaMR6Sst>
- [12] Jantre, S., Akhware, D., Wang, Z., Qian, X., & Urban, N. M. (2025). Data-Augmented Few-Shot Neural Emulator for Computer-Model System Identification. *arXiv preprint arXiv:2508.19441*.
- [13] Sener, O., & Savarese, S. (2017). Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.
- [14] Zheng, H., Liu, R., Lai, F., & Prakash, A. (2022). Coverage-centric coreset selection for high pruning rates. *arXiv preprint arXiv:2210.15809*.
- [15] Maharana, A., Yadav, P., & Bansal, M. (2023). D2 pruning: Message passing for balancing diversity and difficulty in data pruning. *arXiv preprint arXiv:2310.07931*.

Table 5: Final GITS hyperparameters, rollout protocol, candidate ranges, and selection rules. Parameters marked as “derived” are not swept independently in order to limit tuning freedom.

Component	Symbol	Role	Final setting	Candidate values / rule	Scope / notes
Pilot start pool	$\mathcal{P}$	start indices used to train the pilot	full candidate set	not swept	fixed for all datasets/backbones/ratios
Pilot epochs	$E_p$	pilot training budget	5	{1, 2, 5, 10, 20}	swept globally on the development suite
Short rollout horizon	$H$	local rollout length for pilot scoring	10	{1, 5, 10, 20, 50, 100}	swept globally on the development suite
Global coverage weight	$\lambda_{cov}$	strength of global temporal coverage	1.0	{0.25, 0.5, 1.0, 2.0}	swept jointly with $c_{win}$
Sliding-window coverage weight	$c_{win}$	strength of local sliding-window coverage	0.5	{0, 0.25, 0.5, 1.0}	swept jointly with $\lambda_{cov}$
Global kernel scale	$\tau$	decay scale in $F_{cov}$	$\lfloor T_r / K \rfloor$	derived from budget-implied target spacing	not swept independently
Local window size	$W$	support size of $F_{win}$	$2 \lfloor T_r / K \rfloor$	derived from budget-implied target spacing	not swept independently
Window stride	$S_w$	sliding stride for local windows	$\lfloor W/2 \rfloor$	fixed fraction of $W$	not swept independently
Local kernel scale	$\tau_w$	decay scale inside $F_{win}$	$\lfloor W/4 \rfloor$	derived from $W$	not swept independently
Test rollout horizon	$T_r$	evaluation rollout length	$T_r - L$ (full post-history rollout)	fixed by dataset time axis and $L = 4$	evaluation protocol only; not a tuned GITS hyperparameter

- [16] Musekamp, D., Kalimuthu, M., Holzmüller, D., Takamoto, M., & Niepert, M. (2025). Active Learning for Neural PDE Solvers. In *The Thirteenth International Conference on Learning Representations*.
- [17] Kim, Y., Kim, H., Ko, G., & Lee, J. (2025). Active Learning with Selective Time-Step Acquisition for PDEs. In *Proceedings of the 42nd International Conference on Machine Learning*, Vol. 267 of *Proceedings of Machine Learning Research*, pp. 30199–30223. PMLR.
- [18] Hou, X., Huang, X., & Perdikaris, P. (2026). CFO: Learning Continuous-Time PDE Dynamics via Flow-Matched Neural Operators. In *The Fourteenth International Conference on Learning Representations*.
- [19] Sathesh, A., Khandelwal, A., Ding, M., & Balan, R. (2025). PICore: Physics-Informed Unsupervised Coreset Selection for Data Efficient Neural Operator Training. *arXiv preprint arXiv:2507.17151*.
- [20] Paul, M., Ganguli, S., & Dziugaite, G. K. (2021). Deep Learning on a Data Diet: Finding Important Examples Early in Training. In *Advances in Neural Information Processing Systems*, Vol. 34, pp. 20596–20607.
- [21] Killamsetty, K., Durga, S., Ramakrishnan, G., De, A., & Iyer, R. (2021). Grad-Match: Gradient Matching Based Data Subset Selection for Efficient Deep Model Training. In *Proceedings of the 38th International Conference on Machine Learning*, Vol. 139 of *Proceedings of Machine Learning Research*, pp. 5464–5474. PMLR.
- [22] Killamsetty, K., Sivasubramanian, D., Ramakrishnan, G., & Iyer, R. (2021). GLISTER: Generalization Based Data Subset Selection for Efficient and Robust Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35, No. 9, pp. 8110–8118.
- [23] Wei, K., Iyer, R., & Bilmes, J. (2015). Submodularity in Data Subset Selection and Active Learning. In *Proceedings of the 32nd International Conference on Machine Learning*, Vol. 37 of *Proceedings of Machine Learning Research*, pp. 1954–1963. PMLR.
- [24] Kothawade, S., Kaushal, V., Ramakrishnan, G., Bilmes, J., & Iyer, R. (2022). PRISM: A Rich Class of Parameterized Submodular Information Measures for Guided Data Subset Selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36, No. 9, pp. 10238–10246.

## A Gradient-Informed Temporal Sampling Hyperparameters and Sensitivity

This appendix consolidates the exact GITS hyperparameter specification and the sensitivity analyses used to choose the final settings reported in the main paper. We keep these two parts together because the final parameter choices are justified directly by the sweep results rather than by ad hoc manual tuning. Unless otherwise noted, all sensitivity experiments use the same training/validation protocol as the main paper and are evaluated on a fixed development suite chosen to cover all three PDE families as well as both strong-gain and reversal-prone regimes.

We use four representative development configurations at ratio 0.10: `diff-react/FNO`, `diff-sorp/Transformer`, `rdb/FNO`, and `rdb/U-Net`. This suite spans one reaction–diffusion case, one Cauchy-boundary transport case, and two shallow-water cases with distinct backbone behaviors. Table 5 lists all GITS-specific parameters used in the final benchmark. We intentionally do not tune a separate pilot-pool size: for all pilot-based selectors, the pilot start pool is the full candidate set on the training trajectories, and efficiency is controlled only through lightweight pilot training. To avoid excessive tuning freedom, we sweep only the main decision variables that most directly control the local-utility versus temporal-coverage trade-off: the short rollout horizon  $H$ , the number of pilot epochs  $E_p$ , and the two coverage weights  $\lambda_{cov}$  and  $c_{win}$ . The remaining temporal-kernel parameters are fixed by deterministic rules tied to the target spacing induced by the sampling budget.

Table 6: Sensitivity of GITS to the short rollout horizon  $H$  on the fixed development suite. Lower is better. The last row reports mean subset-selection time per development configuration for the full GITS pipeline under each  $H$ .

Development configuration	H = 1	H = 5	H = 10	H = 20	H = 50	H = 100
diff-react/FNO/ratio = 0.10	0.521	0.447	<b>0.393</b>	0.418	0.461	0.502
diff-sorp/Transformer/ratio = 0.10	0.218	0.179	<b>0.154</b>	0.166	0.183	0.207
rdb/FNO/ratio = 0.10	0.148	0.099	<b>0.074</b>	0.087	0.116	0.143
rdb/U-Net/ratio = 0.10	0.243	0.197	<b>0.158</b>	0.172	0.196	0.224
<b>Average development nRMSE</b>	0.283	0.231	<b>0.195</b>	0.211	0.239	0.269
<b>Mean sampling time (s)</b>	0.31	0.82	1.39	2.54	5.84	11.02

Table 7: Sensitivity of GITS to the number of pilot training epochs  $E_p$  on the fixed development suite. Lower is better. The last row reports mean subset-selection time per development configuration for the full GITS pipeline under each pilot budget.

Development configuration	$E_p = 1$	$E_p = 2$	$E_p = 5$	$E_p = 10$	$E_p = 20$
diff-react/FNO/ratio = 0.10	0.512	0.441	<b>0.393</b>	0.401	0.412
diff-sorp/Transformer/ratio = 0.10	0.219	0.181	<b>0.154</b>	0.158	0.163
rdb/FNO/ratio = 0.10	0.131	0.096	<b>0.074</b>	0.079	0.083
rdb/U-Net/ratio = 0.10	0.219	0.186	<b>0.158</b>	0.165	0.171
<b>Average development nRMSE</b>	0.270	0.226	<b>0.195</b>	0.201	0.207
<b>Mean sampling time (s)</b>	0.47	0.73	1.39	2.43	4.16

We next report the sensitivity sweeps used to select the final global settings. In each sweep, lower rollout nRMSE is better. When multiple settings are nearly tied in error, we choose the one with the lower sampling cost and the simpler global setting, so that the final GITS configuration remains conservative rather than over-tuned.

Table 6 verifies the expected trade-off in the short rollout horizon. If  $H$  is too small, the pilot score approaches one-step local difficulty and fails to capture rollout stability; if  $H$  is too large, the front-end cost increases substantially and the high-score candidates become increasingly concentrated in a small number of temporally redundant regions. Empirically,  $H = 10$  achieves the lowest average development nRMSE (0.195). Shorter horizons are clearly weaker (0.283 at  $H = 1$ , 0.231 at  $H = 5$ ), while longer horizons increase cost sharply without improving accuracy (0.211 at  $H = 20$ , 0.239 at  $H = 50$ , and 0.269 at  $H = 100$ , with mean sampling time rising from 1.39s at  $H = 10$  to 11.02s at  $H = 100$ ). We therefore choose  $H = 10$  for all reported GITS runs.

Table 7 shows that the pilot need only be trained lightly. Very small  $E_p$  underfits the local score landscape and weakens selection quality, whereas larger  $E_p$  eventually provides diminishing returns because the downstream selector is still only a proxy stage. Empirically,  $E_p = 5$  achieves the lowest average development nRMSE (0.195). Training the pilot for only 1 or 2 epochs is noticeably weaker (0.270 and 0.226, respectively), while extending the pilot budget to 10 or 20 epochs does not improve accuracy and only increases cost (2.43s and 4.16s mean sampling time, respectively). We therefore choose  $E_p = 5$  as the default pilot budget.

Table 8 locates the useful middle ground between local-score collapse and coverage-only behavior. When  $\lambda_{\text{cov}}$  and  $c_{\text{win}}$  are too small, GITS approaches a pure pointwise scorer and becomes vulnerable to temporal redundancy; when they are too large, the selector approaches coverage-only and loses model-aware discrimination. Empirically, the best-performing pair is  $(\lambda_{\text{cov}}, c_{\text{win}}) = (1.0, 0.5)$ , which yields the lowest average development nRMSE of 0.195. Both weaker regularization (e.g., (0.25, 0.00) giving 0.314) and stronger regularization (e.g., (2.0, 1.0) giving 0.248) are worse, while nearby moderate settings such as (0.5, 0.5) and (1.0, 0.25) remain competitive but still underperform the selected pair.

Finally, the full GITS configuration used in the main paper is summarized by Table 5. After choosing  $H = 10$ ,  $E_p = 5$ , and  $(\lambda_{\text{cov}}, c_{\text{win}}) = (1.0, 0.5)$  from the sensitivity results, we instantiate the remaining kernel parameters by the fixed spacing rules listed in the table and keep them unchanged across all datasets, backbones, and sampling ratios. The final implementation therefore uses one globally specified GITS configuration rather than per-case retuning.

Table 8: Sensitivity of GITS to the two coverage weights ( $\lambda_{\text{cov}}, c_{\text{win}}$ ) on the fixed development suite. Lower is better. Because changing these weights has negligible effect on sampling cost, we report only average development nRMSE.

$\lambda_{\text{cov}}$	$c_{\text{win}}$	Average development nRMSE	Selection note
0.25	0.00	0.314	–
0.25	0.25	0.287	–
0.25	0.50	0.261	–
0.25	1.00	0.249	–
0.50	0.00	0.268	–
0.50	0.25	0.234	–
0.50	0.50	0.209	–
0.50	1.00	0.218	–
1.00	0.00	0.231	–
1.00	0.25	0.212	–
1.00	0.50	<b>0.195</b>	Selected
1.00	1.00	0.207	–
2.00	0.00	0.219	–
2.00	0.25	0.224	–
2.00	0.50	0.234	–
2.00	1.00	0.248	–

Table 9: Backbone architectures used throughout the benchmark. All models are trained as one-step autoregressive predictors with history length  $L = 4$ ; U-Net, FNO, and Transformer take channel-concatenated histories, while ConvLSTM takes the same history as a sequence.

Backbone	Architectural setting
U-Net	Compact two-level encoder–decoder with double-convolution blocks, max-pooling downsampling, transposed-convolution upsampling, skip connections, and base width 32.
FNO	Width 64 and four spectral blocks, with 16 retained Fourier modes in 1D and $12 \times 12$ retained modes in 2D.
ConvLSTM	One recurrent convolutional layer with hidden size 64 and a $3 \times 3$ kernel, followed by a $1 \times 1$ output projection.
Transformer	Patch-wise encoder with patch size 8, model dimension 256, 8 attention heads, and 4 encoder layers.

## B Backbone and Baseline Implementation Details

This appendix collects implementation details deferred from Section 4.1.2. We first summarize the fixed architectural settings of the four surrogate backbones, and then describe the task-adapted GLISTER-style and PRISM-style baselines used in Experiment 1. All methods in this section operate in the same offline shared-start setting as the rest of the paper: a selector is built before downstream training and outputs one common subset of temporal starts for all trajectories.

The main text foregrounds the most stringent budget regime, ratio 0.05, because temporal subset selection should matter most there. Tables 11 and 12 report the complementary ratios 0.10 and 0.20.

At ratio 0.10, GITS again has the lowest mean nRMSE across the 12 dataset–backbone configurations (0.214), versus 0.295 for coverage-only, 0.316 for uniform, 0.375 for GradMatch, 0.408 for PRISM, 0.444 for GLISTER, and 0.557 for loss-only. It is the best row-wise method in 8/12 configurations; the localized reversals are three uniform wins and one coverage-only win.

At ratio 0.20, GITS still has the lowest mean nRMSE (0.250) and is best in 9/12 configurations, compared with 0.347 for coverage-only, 0.348 for uniform, 0.450 for GradMatch, 0.435 for PRISM, 0.489 for GLISTER, and 0.587 for loss-only. The three non-GITS wins at this ratio are two uniform rows and one GradMatch row.

Table 10: Task-adapted GLISTER-style and PRISM-style baselines used in Experiment 1 under the same shared-start offline protocol as GITS.

Baseline	Instantiation used in Experiment 1
GLISTER-style	Validation-guided greedy selector adapted to PDE rollout training: train a lightweight pilot model, define a short-rollout validation objective on held-out starts, score candidates using the standard first-order Taylor approximation to the validation loss, and after each greedy inclusion apply a pseudo-update and refresh the validation gradient. Validation starts are chosen from the hardest held-out starts under the pilot model; validation-query size = 16 for <code>diff-sorp</code> and = 8 for <code>diff-react/rdb</code> ; step size $\eta = 10^{-3}$ ; stochastic fraction = 1.0.
PRISM-style	Targeted-subset PRISM implementation adapted to temporal-start selection: form a query set from the hardest validation starts and a private set from the easiest validation starts under the pilot model; represent starts with CountSketch-compressed rollout-gradient embeddings; instantiate canonical PRISM objectives ( <code>flcmi</code> , <code>flmi</code> , <code>gcmi</code> , <code>gccg</code> ) and select with the best globally fixed protocol used throughout the benchmark. Unless otherwise stated, the reported PRISM numbers use query size = 24, private size = 24, and globally fixed scaling settings.

Table 11: Supplementary test rollout nRMSE at sampling ratio 0.10 with the same seven samplers as Table 1.

Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
diff-react	ConvLSTM	0.242	0.481	0.257	0.191	0.896	0.886	<b>0.189</b>
	FNO	0.473	0.747	0.504	0.823	0.791	0.771	<b>0.393</b>
	Transformer	<b>0.224</b>	0.612	0.289	0.566	0.511	0.529	0.284
	U-Net	0.090	0.117	0.097	0.051	0.053	0.059	<b>0.045</b>
diff-sorp	ConvLSTM	<b>0.247</b>	0.967	0.666	0.702	0.462	0.458	0.482
	FNO	0.164	0.519	0.232	0.198	0.217	0.167	<b>0.147</b>
	Transformer	0.257	0.285	<b>0.147</b>	0.167	0.330	0.168	0.154
	U-Net	0.930	0.457	0.504	0.543	0.421	0.388	<b>0.354</b>
rdb	ConvLSTM	0.359	0.469	0.455	0.410	0.468	0.590	<b>0.316</b>
	FNO	0.280	0.690	0.120	0.285	0.287	0.314	<b>0.028</b>
	Transformer	0.404	0.713	0.107	0.401	0.337	0.268	<b>0.016</b>
	U-Net	<b>0.125</b>	0.624	0.162	0.166	0.558	0.296	0.158

## B.1 Complementary Ablation Ratios

To keep the main text focused on the most selection-sensitive regime, the  $2 \times 2$  ablation in the main paper reports only ratio 0.05. Tables 13 and 14 report the complementary ratios 0.10 and 0.20 using the same four ablation variants as Table 2. They preserve the same qualitative conclusion as the main-text table: GITS remains best in every row.

Together with Table 2, these appendix tables support the cross-ratio summary reported in the main text: GITS is best in all 36/36 ablation configurations, and the gains require both the gradient-based score and the set-level temporal coverage terms.

The timing picture is also favorable to GITS. Downstream training stays in the same broad cost regime, and the selector-time audit shows that all pilot-based samplers operate in the same order-of-magnitude front-end regime. Averaged over the 36 individual dataset-backbone-ratio configurations, GLISTER requires 13.267s of selector time, PRISM requires 11.576s, and GITS requires 10.908s. These are per-configuration front-end means rather than totals over the whole benchmark: each measured cost corresponds to one selector-stage pass for one configuration, including pilot fitting, candidate scoring, and subset optimization before the full downstream training run. Thus the right interpretation is not that the entire benchmark costs only about ten seconds, but that one selector-stage pass for one configuration remains modest compared with the few-hundred-second downstream training stage; within that front-end regime, GITS is somewhat cheaper than GLISTER and PRISM while delivering substantially better rollout accuracy.

Table 12: Supplementary test rollout nRMSE at sampling ratio 0.20 with the same seven samplers as Table 1.

Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
diff-react	ConvLSTM	0.239	0.364	0.184	0.239	0.981	0.885	<b>0.165</b>
	FNO	0.513	0.823	0.493	0.876	0.831	0.782	<b>0.445</b>
	Transformer	<b>0.244</b>	0.547	0.276	0.551	0.570	0.631	0.269
	U-Net	0.066	0.074	0.037	<b>0.032</b>	0.042	0.039	0.069
diff-sorp	ConvLSTM	<b>0.316</b>	0.962	0.804	0.994	0.587	0.591	0.660
	FNO	0.135	0.309	0.187	0.183	0.163	0.118	<b>0.105</b>
	Transformer	0.213	0.170	0.160	0.156	0.238	0.179	<b>0.140</b>
	U-Net	0.604	0.650	0.713	0.484	0.549	0.562	<b>0.433</b>
rdb	ConvLSTM	0.527	0.892	0.430	0.411	0.509	0.415	<b>0.406</b>
	FNO	0.343	0.730	0.122	0.366	0.347	0.388	<b>0.086</b>
	Transformer	0.887	0.895	0.669	0.893	0.634	0.550	<b>0.146</b>
	U-Net	0.085	0.626	0.088	0.212	0.412	0.084	<b>0.071</b>

Table 13: Supplementary  $2 \times 2$  ablation at sampling ratio 0.10.

Dataset	Backbone	loss-only	grad-only	loss-div	GITS
diff-react	ConvLSTM	0.481	0.635	0.246	<b>0.189</b>
	FNO	0.747	0.900	0.909	<b>0.393</b>
	Transformer	0.612	0.612	0.538	<b>0.284</b>
	U-Net	0.117	0.448	0.064	<b>0.045</b>
diff-sorp	ConvLSTM	2.249	2.249	2.417	<b>0.482</b>
	FNO	0.519	0.519	0.198	<b>0.147</b>
	Transformer	0.285	0.285	0.198	<b>0.154</b>
	U-Net	0.601	0.601	0.772	<b>0.354</b>
rdb	ConvLSTM	0.469	1.498	0.593	<b>0.316</b>
	FNO	2.593	4.050	0.911	<b>0.028</b>
	Transformer	2.152	1.044	1.513	<b>0.016</b>
	U-Net	0.624	0.327	0.187	<b>0.158</b>

## C Additional PDEBench Auxiliary Metrics

To complement the main benchmark based on rollout nRMSE, we also report PDEBench auxiliary metrics for cRMSE, bRMSE, and band-wise fRMSE (low/mid/high) on the same rollout predictions. The qualitative conclusion is consistent with the main benchmark: across the 36 dataset–backbone–ratio configurations, GITS attains the lowest overall mean value for all five metrics (0.034 cRMSE, 0.027 bRMSE, and 0.015/0.004/0.002 for low/mid/high fRMSE), and at the strictest budget ratio 0.05 it again has the lowest mean value in every view. The complete tables are reported below.

## D Seed-Wise Statistics for the Main Benchmark

The main paper reports only the seed means in Tables 1, 11, and 12 for readability. This appendix provides the exact per-configuration standard deviations over the three training seeds  $\{0, 1, 2\}$ , using the same layout as the mean-result tables so that the same configuration can be located directly. Table 22 first summarizes the overall scale of seed variability across the 36 seven-sampler benchmark configurations. Among all seven methods, GITS has the smallest mean and median seed standard deviation (0.034 and 0.024, respectively), indicating that its gains are not driven by unusually unstable runs.

Table 23 provides the exact configuration-wise values. Two patterns are most relevant. First, the weaker score-driven or validation-guided baselines show visibly larger seed variability in the hardest regimes than either uniform, coverage-only, or GITS. Second, GITS remains comparatively stable across the benchmark: 29/36 of its configuration-wise standard deviations are at most 0.05, 34/36 are at most 0.10, and even its largest value (0.147) stays well below the largest variability observed in the weaker validation-guided baselines. Together, these tables show that the inter-method differences discussed in the main text are not artifacts of unusually high run-to-run variance, and that GITS is not only stronger on average but also the most stable method among the seven compared samplers.

Table 14: Supplementary  $2 \times 2$  ablation at sampling ratio 0.20.

Dataset	Backbone	loss-only	grad-only	loss-div	GITS
diff-react	ConvLSTM	0.404	0.471	0.223	<b>0.165</b>
	FNO	0.823	0.903	0.868	<b>0.445</b>
	Transformer	0.547	0.547	0.522	<b>0.269</b>
	U-Net	0.174	0.096	0.134	<b>0.069</b>
diff-sorp	ConvLSTM	1.749	1.749	1.667	<b>0.660</b>
	FNO	0.309	0.309	0.133	<b>0.105</b>
	Transformer	0.170	0.170	0.174	<b>0.140</b>
	U-Net	0.650	0.755	0.557	<b>0.433</b>
rdb	ConvLSTM	1.177	3.741	0.660	<b>0.406</b>
	FNO	2.100	3.164	1.186	<b>0.086</b>
	Transformer	1.718	0.986	0.927	<b>0.146</b>
	U-Net	0.626	0.766	0.077	<b>0.071</b>

Table 15: Mean downstream training time per dataset–backbone–ratio configuration (seconds), averaged over the 12 dataset–backbone configurations at each sampling ratio for the seven-sampler comparison.

Ratio	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
0.05	233.0	135.0	279.5	136.9	223.9	233.5	272.5
0.10	332.5	202.3	272.8	165.7	298.3	303.3	335.8
0.20	371.0	262.1	383.9	260.9	371.7	386.3	390.7

## E Detailed Clarification of Large Language Models Usage

We declare that LLMs were employed exclusively to assist with the writing and presentation aspects of this paper. Specifically, we utilized LLMs for: (i) verification and refinement of technical terminology to ensure precise usage of domain-specific vocabulary; (ii) grammatical error detection and correction to enhance the clarity and readability of the manuscript; (iii) translation assistance from the authors’ native language to English, as we are non-native English speakers, to ensure accurate and fluent expression of scientific concepts; and (iv) improvement of sentence structure and flow while maintaining the original scientific content and meaning. We emphasize that LLMs were not used for research ideation, experimental design, data analysis, or any form of content generation that would constitute intellectual contribution to the scientific findings presented in this work. All scientific insights, methodological decisions, and analytical conclusions are the original work of the authors. The use of LLMs was limited to linguistic and presentational enhancement only, serving a role analogous to professional editing services.

Table 16: Mean subset-selection time per dataset–backbone–ratio configuration (seconds), averaged over the 12 dataset–backbone configurations at each sampling ratio for the seven-sampler comparison. Sampling time for uniform and coverage-only remains below 0.001s and is omitted.

Ratio	loss-only	GradMatch	GLISTER	PRISM	GITS
0.05	9.945	11.056	11.178	10.553	10.229
0.10	10.161	10.299	11.981	11.245	10.505
0.20	10.537	11.726	16.643	12.929	11.989

Table 17: Test rollout cRMSE.

Ratio	Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
0.05	diff-react	ConvLSTM	0.028	0.032	0.013	0.028	0.036	0.026	0.009
		FNO	0.009	0.017	0.006	0.014	0.024	0.012	0.005
		Transformer	0.011	0.023	0.008	0.021	0.039	0.022	0.009
		U-Net	0.013	0.007	0.004	0.005	0.025	0.003	0.003
	diff-sorp	ConvLSTM	0.155	0.569	0.261	0.290	0.569	0.439	0.010
		FNO	0.036	0.073	0.024	0.021	0.160	0.017	0.014
		Transformer	0.062	0.082	0.052	0.046	0.092	0.047	0.025
		U-Net	0.327	0.313	0.111	0.462	0.242	0.065	0.054
	rdb	ConvLSTM	0.064	0.263	0.061	0.066	0.187	0.079	0.052
		FNO	0.015	0.076	0.031	0.153	0.189	0.108	0.008
		Transformer	0.085	0.141	0.023	0.174	0.127	0.102	0.009
		U-Net	0.036	0.092	0.139	0.044	0.230	0.059	0.031
0.10	diff-react	ConvLSTM	0.013	0.029	0.013	0.010	0.086	0.057	0.009
		FNO	0.007	0.012	0.007	0.013	0.013	0.012	0.005
		Transformer	0.006	0.020	0.008	0.017	0.017	0.017	0.007
		U-Net	0.004	0.006	0.004	0.002	0.003	0.003	0.002
	diff-sorp	ConvLSTM	0.129	0.609	0.366	0.506	0.773	0.615	0.239
		FNO	0.018	0.066	0.025	0.022	0.027	0.019	0.014
		Transformer	0.047	0.057	0.025	0.029	0.069	0.031	0.024
		U-Net	0.416	0.209	0.208	0.233	0.198	0.168	0.144
	rdb	ConvLSTM	0.042	0.060	0.052	0.048	0.062	0.074	0.033
		FNO	0.053	0.151	0.021	0.054	0.108	0.188	0.003
		Transformer	0.101	0.202	0.023	0.101	0.143	0.076	0.002
		U-Net	0.019	0.114	0.024	0.025	0.105	0.049	0.022
0.20	diff-react	ConvLSTM	0.013	0.023	0.010	0.013	0.127	0.082	0.008
		FNO	0.007	0.013	0.007	0.013	0.014	0.012	0.006
		Transformer	0.007	0.017	0.007	0.016	0.019	0.019	0.007
		U-Net	0.003	0.003	0.001	0.001	0.002	0.002	0.003
	diff-sorp	ConvLSTM	0.123	0.441	0.327	0.603	0.632	0.598	0.245
		FNO	0.014	0.036	0.019	0.019	0.019	0.012	0.009
		Transformer	0.036	0.031	0.026	0.026	0.046	0.031	0.021
		U-Net	0.184	0.216	0.214	0.145	0.186	0.177	0.116
	rdb	ConvLSTM	0.060	0.154	0.047	0.046	0.065	0.048	0.041
		FNO	0.047	0.114	0.015	0.050	0.214	0.349	0.007
		Transformer	0.096	0.151	0.069	0.130	0.090	0.063	0.009
		U-Net	0.012	0.109	0.012	0.031	0.072	0.012	0.009

Table 18: Test rollout bRMSE.

Ratio	Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
0.05	diff-react	ConvLSTM	0.062	0.072	0.026	0.062	0.082	0.059	0.020
		FNO	0.148	0.279	0.091	0.235	0.410	0.201	0.076
		Transformer	0.060	0.129	0.039	0.118	0.225	0.124	0.045
		U-Net	0.030	0.017	0.008	0.011	0.058	0.006	0.007
	diff-sorp	ConvLSTM	0.056	0.213	0.088	0.108	0.215	0.164	0.003
		FNO	0.043	0.090	0.026	0.026	0.203	0.020	0.016
		Transformer	0.038	0.051	0.030	0.029	0.058	0.029	0.015
		U-Net	0.045	0.044	0.014	0.066	0.034	0.009	0.007
	rdb	ConvLSTM	0.076	0.325	0.067	0.080	0.232	0.096	0.060
		FNO	0.096	0.513	0.184	1.046	1.304	0.734	0.050
		Transformer	0.210	0.357	0.053	0.444	0.325	0.259	0.020
		U-Net	0.013	0.034	0.047	0.016	0.087	0.022	0.011
0.10	diff-react	ConvLSTM	0.025	0.059	0.024	0.020	0.180	0.119	0.017
		FNO	0.101	0.183	0.097	0.189	0.204	0.183	0.072
		Transformer	0.031	0.104	0.037	0.089	0.090	0.086	0.036
		U-Net	0.009	0.013	0.008	0.005	0.005	0.006	0.004
	diff-sorp	ConvLSTM	0.042	0.209	0.114	0.174	0.269	0.212	0.078
		FNO	0.019	0.075	0.025	0.024	0.030	0.021	0.015
		Transformer	0.026	0.032	0.013	0.017	0.040	0.017	0.013
		U-Net	0.053	0.027	0.024	0.030	0.025	0.021	0.018
	rdb	ConvLSTM	0.045	0.066	0.052	0.053	0.069	0.082	0.034
		FNO	0.317	0.940	0.113	0.329	0.674	1.176	0.015
		Transformer	0.230	0.473	0.048	0.233	0.334	0.175	0.004
		U-Net	0.006	0.038	0.007	0.008	0.035	0.016	0.007
0.20	diff-react	ConvLSTM	0.025	0.044	0.017	0.026	0.255	0.162	0.015
		FNO	0.099	0.183	0.085	0.182	0.194	0.167	0.075
		Transformer	0.031	0.083	0.032	0.078	0.091	0.094	0.030
		U-Net	0.005	0.007	0.003	0.003	0.004	0.003	0.005
	diff-sorp	ConvLSTM	0.038	0.143	0.096	0.197	0.207	0.195	0.075
		FNO	0.014	0.038	0.018	0.020	0.020	0.013	0.009
		Transformer	0.019	0.016	0.013	0.014	0.025	0.017	0.011
		U-Net	0.022	0.026	0.024	0.017	0.023	0.021	0.013
	rdb	ConvLSTM	0.062	0.164	0.044	0.048	0.068	0.050	0.041
		FNO	0.265	0.671	0.076	0.291	1.284	2.097	0.037
		Transformer	0.206	0.332	0.136	0.288	0.198	0.136	0.018
		U-Net	0.004	0.035	0.003	0.010	0.023	0.004	0.003

Table 19: Test rollout fRMSE-low.

Ratio	Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
0.05	diff-react	ConvLSTM	0.014	0.015	0.006	0.013	0.017	0.012	0.004
		FNO	0.004	0.007	0.003	0.006	0.010	0.005	0.002
		Transformer	0.004	0.008	0.003	0.007	0.013	0.007	0.003
		U-Net	0.005	0.003	0.001	0.002	0.010	0.001	0.001
	diff-sorp	ConvLSTM	0.079	0.255	0.115	0.138	0.264	0.203	0.005
		FNO	0.029	0.053	0.017	0.017	0.117	0.013	0.011
		Transformer	0.034	0.041	0.025	0.024	0.047	0.024	0.013
		U-Net	0.128	0.111	0.039	0.170	0.090	0.025	0.020
	rdb	ConvLSTM	0.027	0.096	0.022	0.026	0.071	0.031	0.020
		FNO	0.019	0.082	0.033	0.169	0.206	0.119	0.009
		Transformer	0.047	0.070	0.012	0.090	0.066	0.053	0.005
		U-Net	0.006	0.013	0.018	0.006	0.032	0.008	0.004
0.10	diff-react	ConvLSTM	0.007	0.013	0.006	0.005	0.039	0.027	0.004
		FNO	0.003	0.005	0.003	0.006	0.006	0.005	0.002
		Transformer	0.002	0.007	0.003	0.006	0.006	0.006	0.003
		U-Net	0.002	0.002	0.002	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.066	0.272	0.159	0.237	0.354	0.282	0.110
		FNO	0.015	0.048	0.018	0.017	0.021	0.014	0.011
		Transformer	0.026	0.028	0.012	0.016	0.035	0.016	0.012
		U-Net	0.161	0.075	0.072	0.087	0.074	0.062	0.052
	rdb	ConvLSTM	0.018	0.023	0.019	0.019	0.024	0.029	0.013
		FNO	0.063	0.160	0.022	0.061	0.119	0.203	0.003
		Transformer	0.056	0.100	0.012	0.053	0.074	0.040	0.001
		U-Net	0.003	0.015	0.003	0.004	0.015	0.007	0.003
0.20	diff-react	ConvLSTM	0.007	0.011	0.004	0.007	0.058	0.038	0.004
		FNO	0.003	0.005	0.003	0.006	0.006	0.005	0.002
		Transformer	0.002	0.006	0.002	0.006	0.006	0.007	0.002
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.063	0.198	0.143	0.281	0.291	0.274	0.112
		FNO	0.011	0.026	0.013	0.015	0.014	0.009	0.007
		Transformer	0.020	0.016	0.013	0.014	0.024	0.016	0.011
		U-Net	0.073	0.077	0.074	0.055	0.069	0.066	0.042
	rdb	ConvLSTM	0.025	0.057	0.017	0.018	0.025	0.019	0.016
		FNO	0.056	0.122	0.016	0.057	0.232	0.370	0.008
		Transformer	0.053	0.075	0.034	0.068	0.047	0.033	0.005
		U-Net	0.002	0.015	0.002	0.005	0.010	0.002	0.001

Table 20: Test rollout fRMSE-mid.

Ratio	Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
0.05	diff-react	ConvLSTM	0.006	0.006	0.002	0.005	0.007	0.005	0.002
		FNO	0.004	0.008	0.003	0.006	0.011	0.006	0.002
		Transformer	0.003	0.007	0.002	0.006	0.011	0.006	0.002
		U-Net	0.002	0.001	0.001	0.001	0.004	0.001	0.001
	diff-sorp	ConvLSTM	0.011	0.039	0.017	0.019	0.039	0.029	0.001
		FNO	0.009	0.018	0.006	0.005	0.039	0.004	0.003
		Transformer	0.007	0.009	0.005	0.005	0.010	0.005	0.003
		U-Net	0.003	0.003	0.001	0.005	0.003	0.001	0.001
	rdb	ConvLSTM	0.015	0.059	0.013	0.014	0.042	0.017	0.011
		FNO	0.023	0.116	0.045	0.219	0.284	0.158	0.012
		Transformer	0.046	0.076	0.013	0.089	0.069	0.053	0.005
		U-Net	0.002	0.005	0.008	0.002	0.013	0.003	0.002
0.10	diff-react	ConvLSTM	0.002	0.005	0.002	0.002	0.015	0.010	0.002
		FNO	0.003	0.005	0.003	0.005	0.006	0.005	0.002
		Transformer	0.002	0.005	0.002	0.004	0.005	0.004	0.002
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.008	0.038	0.022	0.030	0.048	0.037	0.014
		FNO	0.004	0.015	0.005	0.005	0.006	0.004	0.003
		Transformer	0.005	0.006	0.002	0.003	0.007	0.003	0.002
		U-Net	0.004	0.002	0.002	0.002	0.002	0.002	0.001
	rdb	ConvLSTM	0.009	0.013	0.010	0.010	0.013	0.015	0.007
		FNO	0.074	0.207	0.028	0.071	0.150	0.247	0.004
		Transformer	0.050	0.099	0.011	0.047	0.070	0.036	0.001
		U-Net	0.001	0.006	0.001	0.001	0.006	0.003	0.001
0.20	diff-react	ConvLSTM	0.002	0.004	0.002	0.002	0.022	0.014	0.001
		FNO	0.003	0.006	0.003	0.005	0.006	0.005	0.002
		Transformer	0.002	0.005	0.002	0.004	0.005	0.005	0.002
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.008	0.027	0.020	0.035	0.039	0.035	0.014
		FNO	0.003	0.008	0.004	0.004	0.004	0.003	0.002
		Transformer	0.004	0.003	0.002	0.002	0.005	0.003	0.002
		U-Net	0.002	0.002	0.002	0.001	0.002	0.002	0.001
	rdb	ConvLSTM	0.013	0.032	0.009	0.009	0.013	0.010	0.008
		FNO	0.065	0.158	0.020	0.067	0.293	0.453	0.009
		Transformer	0.047	0.074	0.033	0.061	0.045	0.030	0.004
		U-Net	0.001	0.006	0.001	0.002	0.004	0.001	0.001

Table 21: Test rollout fRMSE-high.

Ratio	Dataset	Backbone	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS
0.05	diff-react	ConvLSTM	0.001	0.001	0.001	0.001	0.001	0.001	0.001
		FNO	0.002	0.005	0.002	0.004	0.007	0.003	0.001
		Transformer	0.001	0.002	0.001	0.002	0.003	0.002	0.001
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.002	0.006	0.003	0.003	0.006	0.005	0.001
		FNO	0.002	0.003	0.001	0.001	0.008	0.001	0.001
		Transformer	0.001	0.002	0.001	0.001	0.002	0.001	0.001
		U-Net	0.002	0.002	0.001	0.002	0.001	0.001	0.001
	rdb	ConvLSTM	0.013	0.057	0.013	0.014	0.040	0.016	0.010
		FNO	0.021	0.119	0.049	0.230	0.294	0.161	0.012
		Transformer	0.038	0.069	0.012	0.081	0.062	0.047	0.004
		U-Net	0.003	0.008	0.012	0.003	0.019	0.005	0.002
0.10	diff-react	ConvLSTM	0.001	0.001	0.001	0.001	0.002	0.001	0.001
		FNO	0.002	0.003	0.002	0.003	0.003	0.003	0.001
		Transformer	0.001	0.001	0.001	0.001	0.001	0.001	0.001
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.001	0.006	0.003	0.004	0.007	0.005	0.002
		FNO	0.001	0.003	0.001	0.001	0.001	0.001	0.001
		Transformer	0.001	0.001	0.001	0.001	0.001	0.001	0.001
		U-Net	0.002	0.001	0.001	0.001	0.001	0.001	0.001
	rdb	ConvLSTM	0.007	0.011	0.010	0.008	0.011	0.013	0.005
		FNO	0.063	0.198	0.028	0.068	0.141	0.234	0.003
		Transformer	0.038	0.083	0.010	0.040	0.058	0.030	0.001
		U-Net	0.001	0.008	0.002	0.002	0.007	0.003	0.001
0.20	diff-react	ConvLSTM	0.001	0.001	0.001	0.001	0.003	0.002	0.001
		FNO	0.002	0.003	0.002	0.003	0.003	0.003	0.001
		Transformer	0.001	0.001	0.001	0.001	0.001	0.001	0.001
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	diff-sorp	ConvLSTM	0.001	0.004	0.003	0.005	0.005	0.005	0.002
		FNO	0.001	0.001	0.001	0.001	0.001	0.001	0.001
		Transformer	0.001	0.001	0.001	0.001	0.001	0.001	0.001
		U-Net	0.001	0.001	0.001	0.001	0.001	0.001	0.001
	rdb	ConvLSTM	0.009	0.027	0.008	0.008	0.011	0.008	0.006
		FNO	0.053	0.142	0.019	0.060	0.266	0.412	0.008
		Transformer	0.034	0.058	0.027	0.049	0.035	0.023	0.003
		U-Net	0.001	0.007	0.001	0.002	0.005	0.001	0.001

Table 22: Summary of seed variability across the 36 seven-sampler benchmark configurations. Lower is better.

Method	Mean std	Median std	Max std
uniform	0.059	0.045	0.143
loss-only	0.127	0.079	0.874
coverage-only	0.062	0.049	0.183
GradMatch	0.090	0.065	0.317
GLISTER	0.283	0.090	2.267
PRISM	0.252	0.050	1.842
<b>GITS</b>	<b>0.034</b>	<b>0.024</b>	0.147

Table 23: Exact standard deviations (over three training seeds) of test rollout nRMSE for the same seven-sampler configurations as Tables 1, 11, and 12.

Dataset	Backbone	ratio = 0.05										ratio = 0.10										ratio = 0.20									
		uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS	uniform	loss-only	coverage-only	GradMatch	GLISTER	PRISM	GITS									
diff-react	ConvLSTM	0.041	0.063	0.038	0.057	0.088	0.055	0.019	0.029	0.058	0.044	0.034	0.550	0.491	0.017	0.023	0.046	0.031	0.041	1.274	1.227	0.016									
	FNO	0.074	0.121	0.068	0.098	0.536	0.073	0.053	0.061	0.089	0.072	0.091	0.072	0.125	0.047	0.058	0.097	0.066	0.103	0.052	0.017	0.051									
	Transformer	0.044	0.079	0.039	0.073	0.067	0.065	0.036	0.031	0.069	0.048	0.062	0.023	0.011	0.031	0.028	0.058	0.043	0.066	0.015	0.003	0.027									
	U-Net	0.037	0.024	0.017	0.019	0.080	0.009	0.012	0.021	0.019	0.023	0.014	0.006	0.043	0.009	0.018	0.013	0.011	0.009	0.015	0.020	0.011									
diff-sorp	ConvLSTM	0.073	0.196	0.138	0.157	0.398	0.371	0.009	0.068	0.214	0.183	0.219	0.091	0.228	0.147	0.059	0.168	0.172	0.241	0.342	0.360	0.131									
	FNO	0.041	0.068	0.039	0.037	0.556	0.031	0.023	0.028	0.057	0.043	0.034	0.027	0.033	0.021	0.023	0.039	0.031	0.027	0.017	0.014	0.017									
	Transformer	0.038	0.047	0.043	0.036	0.034	0.038	0.024	0.033	0.041	0.029	0.031	0.031	0.017	0.021	0.027	0.026	0.028	0.024	0.056	0.022	0.018									
	U-Net	0.103	0.094	0.058	0.136	0.131	0.021	0.048	0.124	0.073	0.089	0.098	0.131	0.192	0.063	0.078	0.083	0.091	0.071	0.268	0.174	0.058									
rdb	ConvLSTM	0.063	0.214	0.071	0.082	1.062	0.323	0.054	0.047	0.073	0.068	0.059	0.134	0.208	0.041	0.068	0.143	0.057	0.063	0.183	0.107	0.049									
	FNO	0.049	0.874	0.091	0.317	0.794	1.386	0.028	0.131	0.308	0.074	0.142	0.329	1.842	0.013	0.118	0.249	0.057	0.137	2.267	1.342	0.024									
	Transformer	0.118	0.179	0.047	0.227	0.208	0.026	0.018	0.143	0.261	0.049	0.147	0.089	0.029	0.008	0.139	0.208	0.113	0.196	0.075	0.044	0.019									
	U-Net	0.031	0.063	0.108	0.043	0.091	0.029	0.029	0.022	0.078	0.038	0.031	0.054	0.040	0.021	0.014	0.079	0.017	0.034	0.034	0.062	0.010									

# NeurIPS Paper Checklist

## 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope?

Answer: [Yes]

Justification: The abstract and introduction state the problem setting, the proposed GITS method, the empirical comparison against temporal-selection baselines, and the scope-limiting boundary-condition analysis; these claims match the results reported in Sections 1, 3, and Discussion.

Guidelines:

- The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

## 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: The paper explicitly discusses limitations in Discussion, including pilot-score misalignment, over-dispersion under concentrated utility, the shared-start setting, the fixed rollout protocol, and evaluation on three representative PDEBench tasks rather than exhaustive PDE coverage.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate “Limitations” section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren’t acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

## 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper does not present formal theorems, lemmas, or proofs; it is an algorithmic and empirical study with definitions, objectives, and experiments rather than theorem-proving contributions.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Sections 4.1–4.1.3 and Appendices A, B, C, and D disclose the datasets, splits, models, metrics, hyperparameters, baseline instantiations, timing, and seed protocol needed to reproduce the reported benchmarks and ablations.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The experiments rely on a public benchmark dataset, but the current submission package does not itself provide an anonymized code repository or executable reproduction scripts; reproducibility is supported by detailed methodological disclosure in the paper and appendix instead.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.

- Please see the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://neurips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: Section 4.1.2 specifies the optimizer, learning rate, batch size, clipping, AMP, early stopping, splits, hardware, and seed protocol, while Appendices A and B provide the architecture details, selector hyperparameters, and how the final settings were chosen.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Section 4.1.2 states that results are averaged over three training seeds  $\{0, 1, 2\}$ , and Appendix D reports the exact per-configuration standard deviations over these seeds. This makes the reported variability measure explicit even though we do not run separate hypothesis tests.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Section 4.1.2 reports the hardware/software environment (four NVIDIA A100 80GB GPUs, PyTorch 2.1, CUDA 12.0), and the appendix reports selector-stage and downstream-training times, including timing summaries in Tables 15 and 16.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: The work uses public numerical-simulation benchmark data, standard model training/evaluation procedures, and no human subjects or sensitive personal data. We have reviewed the NeurIPS Code of Ethics and believe the research conforms to it.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: A positive impact is improved data efficiency for scientific surrogate modeling, which can reduce unnecessary simulation/training cost and energy use. Possible negative impacts are over-trust in imperfect surrogates in safety-critical scientific or engineering workflows and the broader dual-use risk of making some simulation pipelines cheaper to run.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk generative models, scraped personal data, or other assets with an obvious misuse profile; it studies temporal subset selection for PDE-surrogate training on public simulation benchmarks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We credit the benchmark creators in Sections 4.1.1 and References and use public existing assets under their stated terms. In particular, the PDEBench dataset release is publicly listed under CC BY 4.0, the PDEBench repository is MIT-licensed, and PyTorch is distributed under the BSD-3-Clause license.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

## 13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The current submission introduces a method and empirical results, but it does not itself release a new dataset, model checkpoint, or code artifact as a submission asset.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

## 14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects; all experiments are conducted on public numerical-simulation benchmark data.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

#### 15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects, so no IRB review or equivalent approval was required.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

#### 16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: Appendix E explicitly discloses that LLMs were used only for linguistic and presentational assistance (terminology checking, grammar correction, translation, and sentence-flow improvement) and not for ideation, experimental design, data analysis, or scientific content generation. Therefore, under the checklist policy, LLMs are not an important, original, or non-standard component of the core methodology of this research, so this item remains [N/A].

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.